Maximum Margin Bayesian Networks

Asymptotic Consistency, Hybrid Learning, and Reduced-Precision Analysis

PhD Thesis

Dipl.-Ing. Sebastian Tschiatschek, BSc

Graz University of Technology

Signal Processing and Speech Communication Laboratory
Head: Univ.-Prof. Dipl.-Ing. Dr.techn. Gernot Kubin

Supervisor: Assoc.Prof. Dipl.-Ing. Dr.mont. Franz Pernkopf

Members of the Examination Committee:
Assoc.Prof. Dipl.-Ing. Dr.mont. Franz Pernkopf
Prof. Jeffrey A. Bilmes, PhD

Graz, September 2014

Affidavit

I declare that I have authored this thesis independently, that I have not used other than declared sources/resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources. The text document uploaded to TUGRAZonline is identical to the present doctoral dissertation.

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe. Das in TUGRAZonline hochgeladene Dokument ist mit der vorliegenden Dissertation identisch.

(date) (signature)

Abstract

We consider Bayesian networks (BNs) with discriminatively optimized parameters and structures, i.e. BNs that are optimized to maximize a kind of probabilistic margin. These maximum margin Bayesian networks (MM BNs) are inspired by support vector machines (SVMs) that aim to separate samples from different classes by a large margin in some feature space. MM BNs achieve classification performance on par with BNs optimized according to other discriminative criteria, e.g. maximum conditional likelihood. Furthermore, in several applications, they achieve classification performance comparable to that of both, linear and kernelized, SVMs. In the literature, two definitions of MM BNs with respect to their parameters are available. We analyze these definitions in terms of asymptotic consistency, extend these definitions by a generative regularizer and analyze properties of MM BNs with respect to reduced-precision implementations.

We start by analyzing the asymptotic consistency of MM BNs. Our analysis reveals a deficiency of MM BNs according to the definitions available in the literature. This deficiency renders MM BNs inconsistent in the multiclass case under certain conditions. However, this deficiency is not eminent in the binary-class case. In experiments, we demonstrate that MM BNs are nevertheless able to compensate for model-mismatch, i.e. the true data distribution and the distributions representable by the learned models do not match.

By slightly altering the definitions existing in the literature, we extend MM BNs to include a generative norm-regularizer. This regularizer consists of normalized frequency counts and is balanced against a margin term. In this novel formulation, MM BNs can be interpreted as linear SVMs with a special regularizer or, alternatively, as BNs that are optimized generatively as well as discriminatively. This interpretation allows one to naturally deal with missing features scenarios, simply by marginalization of the missing features, and to perform semi-supervised learning — the unlabeled data influence the generative regularizer. State-of-the-art performance of the novel MM BN formulation is achieved in a large set of experiments and efficient algorithms for parameter learning in large scale scenarios are presented.

Furthermore, we consider reduced-precision implementations of BNs. We especially focus on BNs optimized for a large margin, either in terms of their structure or their parameters. In preliminary experiments, we investigate the classification performance of BNs with parameters rounded to some specified precision. These experiments extend results from the literature and reveal that BNs used for classification are well suited for reduced-precision implementations. We continue by deriving several types of classification performance bounds for BNs. These bounds can be used to analyze worst-case classification performance upon parameter rounding. In experiments, these bounds are evaluated. Furthermore, BNs optimized for a large margin (in terms of their parameters and their structure) are compared to generatively optimized BNs in terms of robustness to parameter quantization and in terms of absolute classification performance. We extend our reduced-precision considerations by proposing an alternative to determining reduced-precision parameters for MM BNs by rounding. Therefore, we slightly modify our formulation of MM BNs and propose algorithms for maximizing this modified criterion over the search space of reduced-precision parameters. In several experiments, we demonstrate that parameters learned in this way yield better classification performance than parameters obtained by rounding. Finally, we end our investigations by considering parameter learning using reduced-precision computations only. Therefore, we propose specialized algorithms for generative and discriminative parameter learning and demonstrate their effectiveness in experiments.

Acknowledgements

I want to thank my supervisor Professor Franz Pernkopf for advising me, for amusing coffee breaks as well as for philosophic and therapeutic Monday morning meetings. Putting a thesis together is, at some point, a fight against the distraction caused by open questions and interesting new stuff. You made me focus and push forward.

Franz also ensured financing of my work through the Austrian Science Fund. In this regard, I want to thank both Franz and the Austrian Science Fund for supporting me (I was on the payroll of the following projects: P22488-N23, S10608-N13 and S10610-N13).

I also want to thank Professor Jeff Bilmes for hosting me at his lab for a research visit. Your way of thinking and your style of working really impressed and inspired me. Furthermore, you really made me interested in submodular functions (which did not result in too many meaningful results yet but will surely do so in the future). Another big thanks for advising me far beyond my stay.

Thank you to my colleagues Robert, Martin, Michael, Thomas, Christina, Matthias, and Rishabh. By having interesting discussions, you always kept me interested in what I am doing. One of you was always willing to have a coffee break when my spirits were low. Special thanks to Robert, Martin and Rishabh for working on some challenging and interesting problems together.

I thank the coffee machines at our lab (it was two during my four years at the lab), for keeping my spirits up and making long nights before deadlines taste bittersweet.

A big thank you to my big brother Fabian. Before I even knew the meaning of bits and bytes, or understood anything about the geometry behind triangles, you nourished my interest in computers and maths. Without this, I would probably have slept much more but worked less on the very exciting problems I encountered throughout the years.

Many more thanks to my parents and my sisters for supporting me and encouraging me. Beside my family, also my friends kept me working on — thank you so much! Special thanks to Peter and Stephan for bearing my cynical mood once in a while!

Finally, I want to thank my love Julia for encouraging and supporting me through large parts of this thesis. Even though I dedicated most of my spare time to this thesis and even though I was often distracted from what we were up to because my mind was occupied by my work, you never let me down.

List of Acronyms

BB	branch and bound
BN	Bayesian network
BNC	Bayesian network classifier
CR	classification rate
CPD	conditional probability density
CPT	conditional probability table
DAG	directed acyclic graph
DFE	discriminative frequency estimates
FG	factor graph
FPGA	field programmable gate array
HMM	hidden Markov model
KL	Kullback-Leibler
MAP	maximum a-posteriori
MCL	maximum conditional likelihood
ML	maximum likelihood
MM	maximum margin
MM BN	maximum margin Bayesian network
MN	Markov network
NB	naive Bayes
NN	neural network
PGM	probabilistic graphical model
RV	random variable
SPN	sum-product network
SVM	support vector machine
TAN	tree augmented network

Contents

1 Introduction

Intelligent systems are emerging in almost every field of our daily lives. For example, expert systems are used in medicine to aid the diagnosis of diseases [23], urban traffic is controlled automatically to quickly adopt to changing traffic situations [30] and dialog systems are used to provide advanced phone services like automated ticket reservation [29]. All these intelligent systems rely on machine learning and pattern recognition techniques: by analyzing collected data, these systems try to identify relevant factors and generalize them to perform useful actions on new unseen data.

Data analysis is commonly performed using statistical methods and probability calculus [32], e.g. by using probabilistic models. These models are used for modeling relations in the data of interest. Depending on the application, the models are learned either *supervised* or *unsupervised* [20]. In unsupervised learning, the goal is to identify structure in the data, e.g. the distribution of the data or clusters. In supervised learning, the data of interest consists of input-output pairs and the task is to assign some given input to a desired output value, e.g. classification of images into images representing animals and images not representing animals. The output values can be more complex than a single symbol/category, i.e. structured [46, 45]. Structured outputs are for example whole sentences in machine translation.

Once a probabilistic model is available, it can be used to answer queries. Possible queries are for example the computation of the probability of a certain disease given several symptoms, finding the five fastest routes from A to B tomorrow evening, or identifying the most likely written text corresponding to some recorded utterance. Answering such queries requires probabilistic inference, i.e. deriving (probabilistic) conclusions given some evidence [20].

We consider special instances of probabilistic models, i.e. probabilistic graphical models (PGMs) [28, 24]. PGMs combine elements of graph theory and probability theory. They can be classified into undirected graphical models, i.e. Markov networks (MNs), directed graphical models, i.e. Bayesian networks (BNs), and factor graphs (FGs). All these PGMs have in common that they allow for an easy interpretation and enable one to read off conditional independencies. Given an undirected graph $\mathcal{G}$, a set of random variables (RVs) forms a MN with respect to $\mathcal{G}$ if these RVs satisfy the local Markov properties, i.e. specific conditional independence properties. Under certain conditions, MNs can be equivalently defined as probability distributions that factorize into a product of positive potentials associated with the undirected graph (Hammersley-Clifford Theorem) [4]. MNs are, for example, widely used in computer vision, especially in the form of conditional random fields [26]. FGs are defined as bipartite graphs of variable and factors nodes that define the factorization of a function. For suitable factors, FGs represent probability distributions (or can be normalized to do so). FGs are, for example, used in coding [25]. In the following we consider the third type of PGMs, i.e. BNs, in more detail.

BNs [33, 24] consist of a directed acyclic graph (DAG) encoding conditional independencies and a set of conditional probability densities (CPDs) associated with the nodes of the graph. The encoded conditional independencies can be read off using d-separation [33, 28]. BNs are, for example, prominent in the medical domain [23], in acoustics [50], e.g. ASR, in the form of hidden Markov models (HMMs), and in information retrieval [11]. In many cases, BNs allow a compact representation of probability distributions and, thereby, a reduction of the number of parameters needed to specify that distribution (compared to a fully tabular approach in the case of distributions over discrete RVs) [24]. To use a BN as probabilistic model for some task, the BN must be specified, i.e. its graph and its CPDs must be determined. Learning the DAG of a BN is known as *structure learning* and learning the CPDs as *parameter learning*. The structure as well as the parameters can be optimized either generatively or discriminatively [16, 36, 34, 17, 41, 37], cf. Figure 1.1.

Generatively optimized structures are typically learned using Bayesian scoring functions, e.g. K2 or Bayesian Dirichlet score, or by maximizing some form of penalized likelihood, where the penalizer is for example the minimum description-length, the Akaike

Parameter Learning \ Structure Learning	Generative (CMI)	Discriminative (MM, CL, CR)
Generative (ML)	generative parameter and structure learning	generative parameter and discriminative structure learning
Discriminative (MCL, MM, DFE)	discriminative parameter and generative structure learning	discriminative parameter and structure learning

Figure 1.1: Strategies for learning BNs [35]; for parameter learning, for example, the maximum likelihood (ML), maximum conditional likelihood (MCL), maximum margin (MM) and discriminative frequency estimates (DFE) objectives are available; for structure learning, for example, the conditional mutual information (CMI), the explaining away residual (EAR), the maximum margin (MM), the conditional likelihood (CL), and the classification rate (CR) can be used as objective functions.

information criterion or the Bayesian information criterion — a summary of methods and scores can for example be found in [10]. Discriminatively optimized structures are often found by using greedy hill-climbing methods and optimizing some discriminative objective, e.g. conditional-likelihood, margin, or the classification rate [18, 37].

Similarly, parameters can be either learned *generatively*, i.e. unsupervised, or *discriminatively*, i.e. supervised or semi-supervised. According to the generative paradigm, and from a frequentist viewpoint, parameters are typically learned using ML estimation, i.e. the parameters are optimized to maximize the probability of observing a given training set. From a Bayesian viewpoint, the parameters of a BN are RVs. Assuming a Dirichlet prior for these RVs and global parameter independence, the posterior distribution of the parameters follows a Dirichlet distribution as well [21] (the Dirichlet distribution is a conjugate prior of the multinomial distribution). Then, by marginalizing out the parameters, the probability that some node in a BN is in a specific state given the state of its parent nodes, equals to the ML estimate with imaginary sample counts. In contrast to the generative paradigm, the discriminative paradigm dictates to learn parameters such that some task, e.g. classification, can be performed as well as possible. Examples of discriminatively optimized parameters for classification are MCL parameters [17], DFE [44], and MM parameters [19, 38]. MM parameters have only recently been proposed and have not been studied rigorously in various aspects.

We mainly consider BNs with MM parameters, i.e. BNs for classification, which are also referred to as Bayesian network classifiers (BNCs) [16, 3]. Furthermore, we assume a given network structure and discrete valued nodes. In this setting, MM refers to a kind of probabilistic margin [19, 38]. Margin-based methods are very successful in the field of machine learning, especially due to the success of support vector machines (SVMs) [47, 43]. SVMs and their underlying theory is based on empirical risk minimization combined with strong theoretical guarantees [47]. Large margin methods also made their way into structured prediction tasks, e.g. [46, 45].

1.1 Some Unresolved Issues

Theoretical guarantees for SVMs, which are partly accountable for the wide application of these classifiers, include asymptotic consistency guarantees. Asymptotic consistency ensures that in the limit of infinite training data the optimal classifier within the considered function class is found. As recently proven, SVMs, with proper choice of the regularizer and universal kernels, satisfy this appealing property [42]. Asymptotically inconsistent algorithms for learning classifiers are not guaranteed to return the *right thing* and may result in a systematic bias [47]. With regard to BNCs, the following consistency guarantees are available in the literature: Assume a fixed graph for a BNC when learning ML parameters. Under the condition that the conditional independencies implied by the assumed graph match that of the true data distribution, this true data distribution is learned asymptotically almost surely [24]. Consequently, also all conditionals correspond to the true conditionals asymptotically. Therefore, the learned BNCs are in this case optimal, i.e. the Bayes classifier is obtained. Roos et al. [41] extend these arguments to BNCs with MCL parameters. They also argue that in cases in which the considered graph structure cannot represent the conditional independencies of the data, discriminatively optimized parameters can be advantageous over generatively optimized parameters. While asymptotic consistency is an important property, no similar results are available for BNCs with MM parameters.

As already mentioned, BNCs can be optimized either generatively or discriminatively with respect to their parameters, cf. Figure 1.1 (as mentioned, this is also possible in terms of the structure, but this case is not considered here). Furthermore, it is also possible to define models at the intersection of the generative and the discriminative domain, i.e. hybrid models [27]. Such models have, by definition, both generative and discriminative aspects and can, therefore, combine advantages of both types of models [27]. For instance, there is an ongoing discussion in the machine learning community about whether it is better to use generatively optimized or discriminatively optimized models; Ng and Jordan [31] compare BNCs with naive Bayes (NB) structure that are either trained using the ML criterion or the MCL criterion (this corresponds to logistic regression). They identify regimes in which it is better to use generatively optimized parameters and regimes in which it is better to use discriminatively optimized parameters.[1] Hybrid models have also become important in the last few years in the deep learning community, cf. generative pre-training of neural networks [15].[2] Furthermore, as a possible advantage over purely discriminatively optimized models, hybrid models have a generative *component* by definition. This justifies the usage of these types of models in missing feature scenarios and scenarios other than the classification scenario they are optimized for. Another aspect in favor of hybrid models is that they are directly amenable for semi-supervised learning [8, 9, 13]. Semi-supervised learning becomes increasingly important as the amount of available data increases because of, in general, large costs for labeling training data, whereas unlabeled data can be often collected at almost no cost. All these arguments make hybrid models an interesting class of models, e.g. [40, 5]. Hybrid models in the domain of BNs are rarely considered, e.g. [1]. These models typically optimize a blend of a discriminatively optimized BN and a generatively optimized BN. However, no such (principled) hybrid models are available for BNs with MM parameters and typically interesting scenarios requiring a generative perspective, e.g. missing feature scenarios, are not considered. Only, Pernkopf et al. [38] train BNs with MM parameters in a similar spirit. However, no principled approach is taken and the performed optimization is similar to that of generative pre-training in deep networks, i.e. starting from a BNC with generatively optimized parameters, the parameters are fine-

[1] These regimes correspond to smaller and larger training sets in their case; note, however, that other work finds the existence of these regimes not to be reliable; they state that there is no theoretically correct criterion to answer the question of whether a generatively or discriminatively optimized classifier should be used [49].

[2] This pre-training followed by discriminative fine-tuning may be interpreted as a blend between a generative model, i.e. the pre-trained one, and a discriminative model, i.e. the one obtained by fine-tuning; note, however, that there may be more efficient methods for regularization like dropout/DropConnect [22, 48] that do not require generative training at all.

tuned to maximize the MM objective, where early stopping is used to not deviate too far from the generative solution.

As a last issue, let us turn to low-complexity BNCs. While most research in machine learning is performed using high-end hardware and using vast amounts of computational power, these resources are often not available in practical applications. Therefore, there is partially complementary research, investigating resource constrained algorithms for machine learning.[3] For example, Anguita et al. [2] consider implementations of SVMs that use integer parameters only to avoid expensive computations. Similarly, recently, Piatkowski et al. [39] consider an integer approximation to undirected graphical models for the same reason. Similar analyses have been performed for neural networks (NNs) [12], where NNs with limited precision weights are investigated. Although widely used, no such investigations have been performed for BNCs. Most closely related in mind is sensitivity analysis of BNs [6, 7] that investigates changes of marginal and posterior probabilities of BNs with respect to deviations of some of the parameters. An investigation of BNs with computational constraints is especially interesting for graphs in which inference is easy (otherwise inference resorts in general to an iterative procedure requiring demanding optimization algorithms and many iterations until convergence). In BNs, inference is easy only in trees with low tree-width, and chains (a special case of a tree) and for some other special structures (which typically require the development of specialized inference algorithms). Thus, for low-complexity BNCs especially NB and tree augmented network (TAN) [16] structures are interesting. In the literature, only few results for BNCs with reduced-precision are available; Pernkopf et al. [37] considered BNCs with reduced-precision fixed-point parameters but only performed very basic experiments. However, further investigations for other possible parameterizations of BNCs, methods for efficient reduced-precision parameter learning and investigations of the performance limits of BNCs with reduced-precision parameters are not available in the literature to the best of our knowledge.

1.2 Research Questions and Objectives

Because of the motivation provided above, we pose the research questions and goals of this thesis as enumerated in the following:

- *Asymptotic consistency:* Are BNCs with MM parameters asymptotically consistent? In case of inconsistency, can this inconsistency be resolved? How do these classifiers perform in cases in which the true data distribution cannot be represented by the considered BNCs? Starting off from papers by Guo et al. [19] and Pernkopf et al. [38] providing two different formulations of BNCs with MM parameters, we investigate these questions. We identify conditions for consistency/inconsistency. In experiments, we verify our findings.

- *Generative-Discriminative MM BNCs:* We aim to adopt the MM criterion for parameter learning to resemble a discriminative-generative hybrid objective. Given the hybrid criterion, how can it be optimized? How does the proposed criterion compare to state-of-the-art? As motivated above, hybrid models should enable a generative interpretation of the considered classifiers; how good is this interpretation? Can we efficiently solve scenarios that require a generative interpretation, e.g. missing feature cases?

- *Reduced-precision analysis of BNCs:* We aim for low-complexity BNCs. One key issue in this regard is the usage of reduced-precision parameters. This raises the question, whether BNCs with reduced-precision parameters are feasible or not. How much does performance degrade for a given precision? Can we bound the performance reduction?

[3] Related research investigates possibilities for physical implementation of learning algorithms; see for example the work of Dumoulin et al. [14] that considers aspects relevant for the physical implementation of restricted Boltzmann machines in hardware.

Having answered the above questions, it is intriguing to strive for parameters that are optimized by taking reduced-precision constraints into account. Here we can identify two interesting tasks, i.e. determining well-performing reduced-precision parameters using full-precision arithmetic (training on a full-precision platform, testing on reduced-precision platforms) and determining reduced-precision parameters using reduced-precision computations only (training and testing on reduced-precision platforms). We aim to propose algorithms for both scenarios.

1.3 Organization and Contributions

This thesis is structured as detailed below. Our contributions are summarized along the outline.

- *Chapter 2:* We formally introduce BNs and BNCs. Furthermore, we introduce most of the notation used throughout this thesis and provide an overview over the datasets considered in experiments.

- *Chapter 3:* An analysis of the consistency of some special classes of maximum margin Bayesian networks (MM BNs) is performed. This analysis reveals that these networks are not necessarily consistent and that their definition may be deficient. This insight is important because good classifiers should, at least in the limit of infinite training data, be consistent whenever the optimal classifier is within the model class. Nevertheless, the identified deficiency is only severe in regions where the probability mass of incorrect class assignments is in sum larger than that of the correct class assignment (clearly, in these cases there is no classifier achieving zero classification error either). Furthermore, this problem is only eminent in the multiclass case and does not persist in case of binary classification. The identified problem holds for MM BNs according to the definitions of Guo et al. [19] and Pernkopf et al. [38].

- *Chapter 4:* We present a modification of the MM training criterion for BNCs (this is joint work with Robert Peharz). This novel criterion allows the interpretation of MM BNs as linear SVMs with a generative regularizer, or, alternatively as a hybrid generative-discriminative model. Furthermore, the deficiencies identified in Chapter 3 are resolved because of the generative regularizer (when properly setting the trade-off parameter between generative and discriminative objective) and the early stopping heuristic during training becomes superfluous. The interpretation as a BN also enables one to naturally deal with missing features without the need for imputation techniques which is typically necessary in SVMs. Algorithms for large scale optimization are presented and extensive experiments comparing the modified MM training criterion with other parameter learning methods for BNs are provided. Furthermore, we present algorithms for projecting log-probability parameters onto the convex set of sub-normalized parameters. These algorithms can be used whenever gradient descend/ascend must be performed in the space of log-probabilities.

- *Chapter 5:* We consider reduced-precision implementations of BNCs. Our investigations are more in-depth than previous studies and complement investigations in regard to sensitivity analysis. We derive bounds on the classification performance of BNCs with reduced-precision parameters, as well as methods for discriminative reduced-precision parameter learning. Some ingredients of our learning methods, e.g. specialized projection algorithms, can be used beyond the scope of reduced-precision parameter learning (e.g. projections of log-probabilities onto the set of sub-normalized probabilities with box-constraints). We investigate BNCs with reduced-precision parameters in terms of robustness to parameter quantization and compare generative/discriminative parameters/structures in this regard. Furthermore, we provide preliminary results for learning BNCs using reduced-precision computations only and give an outline of potential research questions for the future.

- *Chapter 6:* We conclude this thesis by reiterating our main results in a brief discussion.
- *Chapter 7:* A list of papers published in the course of my PhD studies is provided. *Note that not all results obtained during my PhD studies made it into this thesis. This is unfortunate, because I had to skip some very interesting work, e.g. on image collection summarization, but necessary, because of the central theme of this work. The reader is welcome to have a look at Chapter 7; the papers referenced therein cover most of our results.*

Bibliography

[1] Pedro A. Aguilera, Antonio Fernández, Fernando Reche, and Rafael Rumí. Hybrid Bayesian Network Classifiers: Application to Species Distribution Models. *Environmental Modelling & Software*, 25(12):1630–1639, 2010.

[2] Davide Anguita, Alessandro Ghio, Stefano Pischiutta, and Sandro Ridella. A Support Vector Machine with Integer Parameters. *Neurocomputing*, 72(1-3):480–489, 2008.

[3] Concha Bielza and Pedro Larrañaga. Discrete Bayesian Network Classifiers: A Survey. *ACM Computing Surveys*, 47(1):5:1–5:43, 2014.

[4] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, 2007.

[5] Anna Bosch, Andrew Zisserman, and Xavier Muñoz. Scene Classification Using a Hybrid Generative/Discriminative Approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 30(4):712–727, 2008.

[6] Hei Chan and Adnan Darwiche. When do Numbers Really Matter? *Journal of Artificial Intelligence Research*, 17(1):265–287, 2002.

[7] Hei Chan and Adnan Darwiche. Sensitivity Analysis in Bayesian Networks: From Single to Multiple Parameters. In *Uncertainty in Artificial Intelligence (UAI)*, pages 67–75, 2004.

[8] Olivier Chapelle, Bernhard Schölkopf, and Alexander Zien. *Semi-Supervised Learning*. The MIT Press, 1st edition, 2010.

[9] Fabio Gagliardi Cozman, Ira Cohen, Marcelo Cesar Cirelo, and Escola Politcnica. Semi-Supervised Learning of Mixture Models. In *International Conference on Machine Learning (ICML)*, pages 99–106, 2003.

[10] Luis M. de Campos. A Scoring Function for Learning Bayesian Networks Based on Mutual Information and Conditional Independence Tests. *Journal of Machine Learning Research*, 7:2149–2187, 2006.

[11] Luis M. de Campos, Juan M. Fernández-Luna, and Juan F. Huete. Bayesian Networks and Information Retrieval: An Introduction to the Special Issue. *Information Processing & Management*, 40(5):727–733, 2004. Bayesian Networks and Information Retrieval.

[12] Sorin Draghici. On the Capabilities of Neural Networks Using Limited Precision Weights. *Neural Networks*, 15(3):395–414, 2002.

[13] Gregory Druck, Chris Pal, Andrew McCallum, and Xiaojin Zhu. Semi-supervised Classification with Hybrid Generative/Discriminative Methods. In *International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, pages 280–289. ACM, 2007.

[14] Vincent Dumoulin, Ian J. Goodfellow, Aaron C. Courville, and Yoshua Bengio. On the Challenges of Physical Implementations of RBMs, 2014.

[15] Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio. Why Does Unsupervised Pre-training Help Deep Learning? *Journal of Machine Learning Research*, 11:625–660, 2010.

[16] Nir Friedman, Dan Geiger, and Moises Goldszmidt. Bayesian Network Classifiers. *Machine Learning*, 29:131–163, 1997.

[17] Russell Greiner, Xiaoyuan Su, Bin Shen, and Wei Zhou. Structural Extension to Logistic Regression: Discriminative Parameter Learning of Belief Net Classifiers. *Machine Learning*, 59(3):297–322, 2005.

[18] Daniel Grossman and Pedro Domingos. Learning Bayesian Network Classifiers by Maximizing Conditional Likelihood. In *International Conference on Machine Learning (ICML)*, pages 361–368, 2004.

[19] Yuhong Guo, Dana Wilkinson, and Dale Schuurmans. Maximum Margin Bayesian Networks. In *Uncertainty in Artificial Intelligence (UAI)*, pages 233–242, 2005.

[20] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2003.

[21] David Heckerman, Dan Geiger, and David M. Chickering. Learning Bayesian Networks: The Combination of Knowledge and Statistical Data. *Machine Learning*, 20: 197–243, 1995.

[22] Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Improving Neural Networks by Preventing Co-adaptation of Feature Detectors. *CoRR*, abs/1207.0580, 2012.

[23] Dirk Husmeier, Richard Dybowski, and Stephen Roberts. *Probabilistic Modelling in Bioinformatics and Medical Informatics*. Springer, 2004.

[24] Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

[25] Frank R. Kschischang, Brendan J. Frey, and Hans-Andrea Loeliger. Factor Graphs and the Sum-product Algorithm. *IEEE Transactions on Information Theory*, 47(2):498–519, 2006.

[26] Sanjiv Kumar and Martial Hebert. Discriminative Random Fields. *International Journal of Computer Vision*, 68(2):179–201, 2006.

[27] Julia A. Lasserre, Christopher M. Bishop, and Thomas P. Minka. Principled Hybrids of Generative and Discriminative Models. In *Computer Vision and Pattern Recognition (CVPR)*, pages 87–94. IEEE, 2006.

[28] Steffen L. Lauritzen. *Graphical Models*. Oxford University Press, 1996.

[29] Michael F. McTear. Spoken Dialogue Technology: Enabling the Conversational User Interface. *Computing Surveys*, 34(1):90–169, 2002.

[30] Pitu Mirchandani and Larry Head. A Real-time Traffic Signal Control System: Architecture, Algorithms, and Analysis. *Transportation Research Part C: Emerging Technologies*, 9(6):415 – 432, 2001.

[31] Andrew Y. Ng and Michael I. Jordan. On Discriminative vs. Generative Classifiers: A Comparison of Logistic Regression and Naive Bayes. In *Advances in Neural Information Processing Systems (NIPS)*, 2001.

[32] R. Lyman Ott and Micheal T. Longnecker. *Introduction to Statistical Methods and Data Analysis*. Duxbury Press, 2006.

[33] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc., 1988.

[34] Robert Peharz and Franz Pernkopf. Exact Maximum Margin Structure Learning of Bayesian Networks. In *International Conference on Machine Learning (ICML)*, 2012.

[35] Franz Pernkopf and Jeff Bilmes. Discriminative versus Generative Parameter and Structure Learning of Bayesian Network Classifiers. In *International Conference on Machine Learning (ICML)*, pages 657–664, 2005.

[36] Franz Pernkopf and Jeff A. Bilmes. Efficient Heuristics for Discriminative Structure Learning of Bayesian Network Classifiers. *Journal of Machine Learning Research*, 11: 2323–2360, 2010.

[37] Franz Pernkopf, Michael Wohlmayr, and Manfred Mücke. Maximum Margin Structure Learning of Bayesian Network Classifiers. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2076–2079, 2011.

[38] Franz Pernkopf, Michael Wohlmayr, and Sebastian Tschiatschek. Maximum Margin Bayesian Network Classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 34(3):521–531, 2012.

[39] Nico Piatkowski, Lee Sangkyun, and Katharina Morik. The Integer Approximation of Undirected Graphical Models. In *International Conference on Pattern Recognition Applications and Methods (ICPRAM)*, 2014.

[40] Rajat Raina, Yirong Shen, Andrew Y. Ng, and Andrew McCallum. Classification with Hybrid Generative/Discriminative Models. In *Advances in Neural Information Processing Systems (NIPS)*, 2003.

[41] Teemu Roos, Hannes Wettig, Peter Grünwald, Petri Myllymäki, and Henry Tirri. On Discriminative Bayesian Network Classifiers and Logistic Regression. *Journal of Machine Learning Research*, 59(3):267–296, 2005.

[42] Ingo Steinwart. Support Vector Machines are Universally Consistent. *Journal of Complexity*, 18(3):768–791, 2002.

[43] Ingo Steinwart and Andreas Christmann. *Support Vector Machines*. Springer, 1st edition.

[44] Jiang Su, Harry Zhang, Charles X. Ling, and Stan Matwin. Discriminative Parameter Learning for Bayesian Networks. In *International Conference on Machine Learning (ICML)*, pages 1016–1023. ACM, 2008.

[45] Ben Taskar, Carlos Guestrin, and Daphne Koller. Max-Margin Markov Networks. In S. Thrun, L.K. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems (NIPS)*, pages 25–32. 2004.

[46] Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun. Large Margin Methods for Structured and Interdependent Output Variables. *Journal of Machine Learning Research*, 6:1453–1484, 2005.

[47] Vladimir N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998.

[48] Li Wan, Matthew D. Zeiler, Sixin Zhang, Yann LeCun, and Rob Fergus. Regularization of Neural Networks using DropConnect. In *International Conference on Machine Learning (ICML)*, volume 28, pages 1058–1066, 2013.

[49] Jing-Hao Xue and D. Michael Titterington. Comment on "On Discriminative vs. Generative Classifiers: A Comparison of Logistic Regression and Naive Bayes". *Neural Processing Letters*, 28(3):169–187, 2008.

[50] Geoffrey G. Zweig. *Speech Recognition with Dynamic Bayesian Networks*. PhD thesis, 1998.

2 Background

In this chapter, we introduce most of our notation, provide some background on probabilistic classification, BNs and BNCs, and describe the datasets used in experiments throughout this thesis.

2.1 Probabilistic Classification

Probabilistic classifiers are embedded in the framework of probability theory. One assumes a RV C denoting the class and RVs $X_1, \ldots, X_L$ representing the attributes/features of the classifier. Each X_i can take one value in the set $\mathbf{val}(X_i)$. Similarly, C can assume values in $\mathbf{val}(C)$, i.e. $\mathbf{val}(C)$ is the set of classes. We denote the random vector consisting of $X_1, \ldots, X_L$ as $\mathbf{X} = (X_1, \ldots, X_L)$. Instantiations of RVs are denoted using lower case letters, i.e. $\mathbf{x}$ is an instantiation of $\mathbf{X}$ and c an instantiation of C, respectively. Abusing the standard notation for sets, if $\mathbf{Y}$ is a subset of $\mathbf{X}$, we denote by $\mathbf{x}(\mathbf{Y})$ the instantiation of the RVs $\mathbf{Y}$ according to $\mathbf{x}$. When $\mathbf{A}$ and $\mathbf{B}$ are disjoint sets of RVs, then $[\mathbf{a}, \mathbf{b}]$ denotes an instantiation of the set $\mathbf{A} \cup \mathbf{B}$. Whenever $\mathrm{P}(C, \mathbf{X})$ is a probability distribution over C and $\mathbf{X}$, we write $\mathrm{P}(c, \mathbf{x})$ as an abbreviation for $\mathrm{P}(C = c, \mathbf{X} = \mathbf{x})$. The expectation of a function $f(C, \mathbf{X})$ with respect to a joint distribution $\mathrm{P}(C, \mathbf{X})$ is denoted as $\mathbb{E}_{\mathrm{P}(C,\mathbf{X})}[f(C, \mathbf{X})]$. The RVs $C, X_1, \ldots, X_L$ are assumed to be jointly distributed according to the distribution $\mathrm{P}^*(C, \mathbf{X})$. We refer to $\mathrm{P}^*(C, \mathbf{X})$ as *true distribution*. In typical settings, this true distribution is unknown and a limited number of samples drawn from the true distribution $\mathrm{P}^*(C, \mathbf{X})$, i.e. a training set $\mathcal{D}$, is available. This set $\mathcal{D}$ consists of N i.i.d. labeled samples, i.e. $\mathcal{D} = ((c^{(n)}, \mathbf{x}^{(n)}) \mid 1 \leq n \leq N)$, where $c^{(n)}$ denotes the instantiation of the RV C and $\mathbf{x}^{(n)}$ the instantiation of $\mathbf{X}$ in the n^{th} training sample. The aim is to induce *good* classifiers provided the training set, i.e. classifiers with low generalization error. Formally, a classifier h is a mapping

$$\begin{aligned} h\colon \mathbf{val}(\mathbf{X}) &\to \mathbf{val}(C), \\ \mathbf{x} &\mapsto h(\mathbf{x}), \end{aligned} \tag{2.1}$$

where $\mathbf{val}(\mathbf{X})$ denotes the set of all possible assignments of $\mathbf{X}$, i.e. a classifier maps an instantiation $\mathbf{x}$ of the attributes to class c. The merit of a classifier can be quantified by its generalization error, or equivalently, by its classification rate.

Definition 1 (Generalization Error, Classification Rate)**.** *Let* $h\colon \mathbf{val}(\mathbf{X}) \to \mathbf{val}(C)$ *be a classifier. Its* generalization error $\mathrm{Err}(h)$ *is*

$$\mathrm{Err}(h) = \mathbb{E}_{\mathrm{P}^*(C,\mathbf{X})}\left[\mathbf{1}_{(h(\mathbf{X}) \neq C)}\right], \tag{2.2}$$

where $\mathbf{1}_{(a)}$ *is the indicator function that equals* 1 *if* a *is true and that equals* 0 *otherwise. The* classification rate $\mathrm{CR}(h)$ *is* $\mathrm{CR}(h) = 1 - \mathrm{Err}(h)$.

Typically, the generalization error can not be evaluated because $\mathrm{P}^*(C, \mathbf{X})$ is unknown but is estimated using cross-validation [2]. Any probability distribution $\mathrm{P}(C, \mathbf{X})$ naturally induces a classifier $h_{\mathrm{P}(C,\mathbf{X})}$ according to

$$\begin{aligned} h_{\mathrm{P}(C,\mathbf{X})}\colon \mathbf{val}(\mathbf{X}) &\to \mathbf{val}(C), \\ \mathbf{x} &\mapsto \arg\max_{c' \in \mathbf{val}(C)} \mathrm{P}(c' \mid \mathbf{x}). \end{aligned} \tag{2.3}$$

In this way, each instantiation $\mathbf{x}$ of $\mathbf{X}$ is classified by the maximum a posteriori estimate of C given $\mathbf{x}$ under $\mathrm{P}(C, \mathbf{X})$. Note that $\arg\max_{c' \in \mathbf{val}(C)} \mathrm{P}(c' \mid \mathbf{x}) = \arg\max_{c' \in \mathbf{val}(C)} \mathrm{P}(c', \mathbf{x})$.

2.2 Bayesian Networks and Bayesian Network Classifiers

2.2.1 Definitions and Notation

We consider probability distributions represented by BNs [13, 11]. A BN $\mathcal{B} = (\mathcal{G}, \mathcal{P}_\mathcal{G})$ consists of a DAG $\mathcal{G} = (\mathbf{Z}, \mathbf{E})$ and a collection of conditional probability distributions

$$\mathcal{P}_\mathcal{G} = (\mathrm{P}(X_0|\mathbf{Pa}(X_0)), \ldots, \mathrm{P}(X_L|\mathbf{Pa}(X_L))), \tag{2.4}$$

where 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. The nodes $\mathbf{Z} = (X_0, \ldots, X_L)$ correspond to RVs and the edges $\mathbf{E}$ encode conditional independencies among these RVs. Throughout this thesis, we often denote X_0 as C, i.e. X_0 represents the class. Then, a BN defines the joint distribution

$$\mathrm{P}^\mathcal{B}(C, X_1, \ldots, X_L) = \mathrm{P}(C|\mathbf{Pa}(C)) \prod_{i=1}^{L} \mathrm{P}(X_i|\mathbf{Pa}(X_i)), \tag{2.5}$$

where, for notational ease, we represent (unconditional) distributions as conditional distributions, e.g. $\mathrm{P}(X_0)$ is denoted as $\mathrm{P}(X_0|\emptyset)$ if X_0 has no parents in $\mathcal{G}$. According to the joint distribution, a BN $\mathcal{B}$ induces the classifier $h_\mathcal{B} = h_{\mathrm{P}^\mathcal{B}(C,\mathbf{X})}$.

In this thesis, we consider BNs with discrete valued RVs only. In this case, a general representation of $\mathcal{P}_\mathcal{G}$ is a collection of conditional probability tables (CPTs), i.e. $\mathcal{P}_\mathcal{G} = (\boldsymbol{\theta}^0, \ldots, \boldsymbol{\theta}^L)$, with

$$\boldsymbol{\theta}^i = (\theta^i_{j|\mathbf{h}} \mid j \in \mathbf{val}(X_i), \mathbf{h} \in \mathbf{val}(\mathbf{Pa}(X_i))), \tag{2.6}$$

where

$$\theta^i_{j|\mathbf{h}} = \mathrm{P}(X_i = j|\mathbf{Pa}(X_i) = \mathbf{h}). \tag{2.7}$$

The BN distribution can then be written as

$$\mathrm{P}^\mathcal{B}(C = c, \mathbf{X} = \mathbf{x}) = \prod_{i=0}^{L} \prod_{j \in \mathbf{val}(X_i)} \prod_{\mathbf{h} \in \mathbf{val}(\mathbf{Pa}(X_i))} {\theta^i_{j|\mathbf{h}}}^{\nu^i_{j|\mathbf{h}}}, \tag{2.8}$$

where

$$\nu^i_{j|\mathbf{h}} = \mathbf{1}_{([c,\mathbf{x}](X_i)=j \text{ and } [c,\mathbf{x}](\mathbf{Pa}(X_i))=\mathbf{h})}. \tag{2.9}$$

We often represent the BN parameters in the logarithmic domain, i.e.

$$w^i_{j|\mathbf{h}} = \log \theta^i_{j|\mathbf{h}}, \tag{2.10}$$

$$\mathbf{w}^i = (w^i_{j|\mathbf{h}} \mid j \in \mathbf{val}(X_i), \mathbf{h} \in \mathbf{val}(\mathbf{Pa}(X_i))), \text{ and} \tag{2.11}$$

$$\mathbf{w} = (\mathbf{w}^0, \ldots, \mathbf{w}^L). \tag{2.12}$$

Often, we will interpret $\mathbf{w}$ as a vector, whose elements are addressed as $w^i_{j|\mathbf{h}}$. We define a vector-valued function $\boldsymbol{\phi}(c, \mathbf{x})$ of the same length as $\mathbf{w}$, collecting $\nu^i_{j|\mathbf{h}}$, analog to the entries $w^i_{j|\mathbf{h}}$ in $\mathbf{w}$. In that way, we can express the logarithm of (2.8) as

$$\log \mathrm{P}^\mathcal{B}(C = c, \mathbf{X} = \mathbf{x}) = \boldsymbol{\phi}(c, \mathbf{x})^T \mathbf{w}. \tag{2.13}$$

2.2.2 Learning Bayesian Network Classifiers

BNs for classification can be optimized in two ways: firstly, one can select the graph structure $\mathcal{G}$, and secondly, one can learn the conditional probability distributions $\mathcal{P}_\mathcal{G}$. Selecting the graph structure is known as *structure learning* and selecting $\mathcal{P}_\mathcal{G}$ is known as *parameter learning*.

Structure Learning

The structure of a BNC, i.e. its graph $\mathcal{G}$, encodes conditional independence assumptions. Algorithms for structure learning can be coarsely grouped according to the underlying paradigm, i.e. whether they yield generatively or discriminatively structured BNCs [14].

Structure learning is naturally combinatorial and in general difficult — even in the general case where scores of structures typically decompose according to the network structure [3]. For the generative case, several formal hardness results are available, e.g. learning polytrees [4] or learning general BNs [8] are NP-hard optimization problems. Algorithms for learning generative structures often optimize some kind of penalized likelihood of the training data and try to determine the structure for example by performing independence tests [6]. Discriminative methods often employ local search heuristics [15].

In this thesis, we consider relatively simple (and fixed) structures only, i.e. we assume that structure learning has already been performed. We use NB structures, and different variants of TAN structures [6]. The reason for not using more expressive structures are twofold: first, more complex structures do not necessarily result in significantly better classification performance [15],[1] and second, analysis becomes more difficult. In more detail, we use the following structures:

- *Naive Bayes (NB).* This structure implies conditional independence of the features, given the class, cf. Figure 2.1a. Obviously, this conditional independence assumption is often violated in practice. Still, NB often yields impressively good performance in many applications [20].

- *Tree Augmented Networks based on conditional mutual information (TAN-CMI).* This structure was introduced by Friedman et al. [6] to relax the strong independence assumptions imposed by the NB structure and to enable better classification performance. TAN-CMI structures are learned by computing the conditional mutual information of all pairs of attributes given the class variable, constructing a complete undirected graph using these conditional mutual information as edge weights, determining a maximum weight spanning tree and converting this undirected tree into a directed tree by selecting a root variable and directing all edges outward from it. Finally, the class node is introduced and an edge from the class to every attribute is added. An example of a TAN structure is shown in Figure 2.1b.

- *TAN-CR structures [10, 15].* These structures are learned using a naive greedy heuristic. Starting from a NB structure, edges are greedily added such that the classification rate of the resulting classifier is maximized (to stay within the model class of TANs, only edges that, when added, do not change that model class are considered).

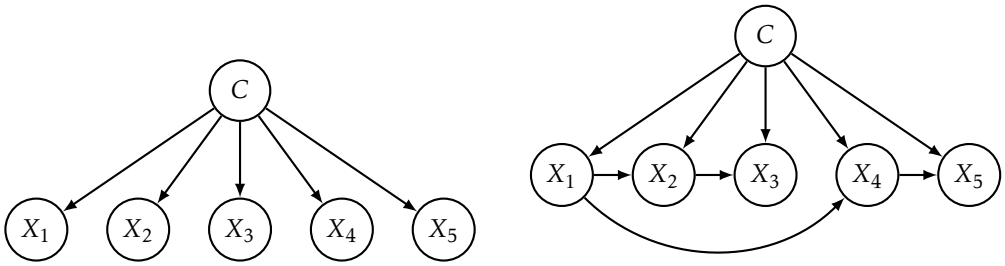
- *TAN-OMI-CR [15].* These structures are learned in a similar fashion as TAN-CR structures, but one computes an ordering of all features based on an information measure first. Based on this ordering, possible edges are greedily added, reducing the number of necessary score evaluations (not all pairs of attributes must be considered, but one progresses according to the pre-computed ordering).

- *TAN-MM structures [16].* These structures are learned similarly as TAN-CR structures, but a margin-objective is maximized instead of the classification rate.

Parameter Learning

Similar to the case of structure learning, the CPDs $\mathcal{P}_\mathcal{G}$ of BNs can be optimized either generatively or discriminatively. We refer to the CPDs as parameters, because CPDs are in general either parametrized (e.g. if a CPD is represented by a multinomial Gaussian distribution, it can be specified by means and covariances), or, in the case of discrete RVs, CPDs

[1] The situation may be different for larger datasets.



(a) Naive Bayes structure (b) TAN structure

Figure 2.1: Exemplary BN structures.

can be specified by CPTs. In this latter case, which we consider throughout this thesis, every entry of the CPTs can be considered as a parameter.[2]

Several approaches for optimizing $\mathcal{P}_\mathcal{G}$ are discussed in the following:

- **Generative Parameters.** In generative parameter learning one aims at identifying parameters modeling the generative process that results in the data of the training set, i.e. generative parameters are based on the idea of *approximating* $\mathrm{P}^*(C, \mathbf{X})$ by a distribution $\mathrm{P}^{\mathcal{B}}(C, \mathbf{X})$. An example of this paradigm is *maximum likelihood (ML)* learning. Its objective is maximization of the likelihood of the training data given the parameters, i.e.

$$\mathcal{P}_\mathcal{G}^{\mathrm{ML}} = \arg\max_{\mathcal{P}_\mathcal{G}} \prod_{n=1}^{N} \mathrm{P}^{\mathcal{B}}(c^{(n)}, \mathbf{x}^{(n)}). \tag{2.14}$$

  Maximum likelihood parameters minimize the Kullback-Leibler (KL)-divergence between $\mathrm{P}^{\mathcal{B}}(C, \mathbf{X})$ and $\mathrm{P}^*(C, \mathbf{X})$ [11].

- **Discriminative Parameters.** In discriminative learning one aims at identifying parameters leading to good classification performance on new samples from $\mathrm{P}^*(C, \mathbf{X})$. This type of learning is for example advantageous in cases where the assumed model distribution $\mathrm{P}^{\mathcal{B}}(C, \mathbf{X})$ cannot approximate $\mathrm{P}^*(C, \mathbf{X})$ well, for example because of a too limited BN structure [18]. Several objectives for discriminative parameter learning are known in the literature. Throughout this thesis, we consider the *maximum conditional likelihood (MCL) [18]* objective and the *maximum margin (MM)* [7, 17] objective.[3]

  MCL parameters $\mathcal{P}_\mathcal{G}^{\mathrm{MCL}}$ are obtained as

$$\mathcal{P}_\mathcal{G}^{\mathrm{MCL}} = \arg\max_{\mathcal{P}_\mathcal{G}} \prod_{n=1}^{N} \mathrm{P}^{\mathcal{B}}(c^{(n)} | \mathbf{x}^{(n)}), \tag{2.15}$$

  where again $\mathrm{P}^{\mathcal{B}}(C, \mathbf{X})$ is the joint distribution induced by the BN $(\mathcal{G}, \mathcal{P}_\mathcal{G})$ and $\mathrm{P}^{\mathcal{B}}(C|\mathbf{X})$ denotes the conditional distribution of C given $\mathbf{X}$ determined from $\mathrm{P}^{\mathcal{B}}(C, \mathbf{X})$ as

$$\mathrm{P}^{\mathcal{B}}(C|\mathbf{X}) = \frac{\mathrm{P}^{\mathcal{B}}(C, \mathbf{X})}{\mathrm{P}^{\mathcal{B}}(\mathbf{X})}. \tag{2.16}$$

  Thus, MCL parameters maximize the conditional likelihood of the class instantiations given the instantiations of the attributes.

[2] Clearly, this resembles an over-specification, because the entries of CPTs must satisfy sum-to-one constraints.

[3] The maximum margin objective is considered in various slightly different formulations — in this introduction a general version is presented and this version is refined as needed in subsequent chapters.

MM parameters $\mathcal{P}_{\mathcal{G}}^{\mathrm{MM}}$ are found as

$$\mathcal{P}_{\mathcal{G}}^{\mathrm{MM}} = \arg\max_{\mathcal{P}_{\mathcal{G}}} \prod_{n=1}^{N} \min\left(\gamma, d^{\mathcal{B}}(c^{(n)}, \mathbf{x}^{(n)})\right), \tag{2.17}$$

where $d^{\mathcal{B}}(c^{(n)}, \mathbf{x}^{(n)})$ is the margin of the n^{th} sample given as

$$d^{\mathcal{B}}(c^{(n)}, \mathbf{x}^{(n)}) = \frac{\mathrm{P}^{\mathcal{B}}(c^{(n)}|\mathbf{x}^{(n)})}{\max_{c \neq c^{(n)}} \mathrm{P}^{\mathcal{B}}(c|\mathbf{x}^{(n)})}, \tag{2.18}$$

and where the hinge loss function is denoted as $\min(\gamma, d^{\mathcal{B}}(c^{(n)}, \mathbf{x}^{(n)}))$. The parameter $\gamma > 1$ controls the margin. In this way, the margin *measures* the ratio of the likelihood of the n^{th} sample belonging to the correct class $c^{(n)}$ to belonging to the most likely competing class. The n^{th} sample is correctly classified iff $d^{\mathcal{B}}(c^{(n)}, \mathbf{x}^{(n)}) > 1$ and vice versa.

An alternative and very simple method for learning discriminative parameters are *discriminative frequency estimates* [19]. According to this method, parameters are estimated using a perceptron-like algorithm, where parameters are updated by the prediction loss, i.e. the difference of the class posterior of the correct class (which is assumed to be 1 for the data in the training set) and the class posterior according to the model using the current parameters. This type of parameter learning yields classification results comparable to those obtained by MCL [19].

2.3 Datasets

Most experiments in the upcoming chapters evaluate the classification performance of BNCs. For these experiments, we use the following datasets:

- **UCI data [1].** This is in fact a large collection of datasets, with small to medium number of samples. Features are discretized as needed using the algorithm proposed in Fayyad and Irani [5]. If not stated otherwise, in case of the datasets *chess*, *letter*, *mofn-3-7-10*, *segment*, *shuttle-small*, *waveform-21*, *abalone*, *adult*, *car*, *mushroom*, *nursery*, *and spambase*, a test set was used to estimate the accuracy of the classifiers. For all other datasets, classification accuracy was estimated by 5-fold cross-validation.

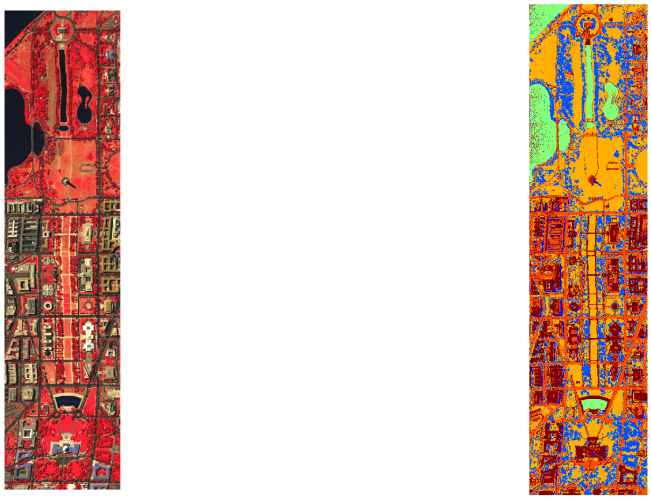
- **satimage/letter** [1]. These two datasets are from the UCI repository. We describe them in more detail because they are used for extensive experiments in Section 5.4. The satimage dataset consists of multi-spectral satellite images. Given a 3×3 multi-spectral pixel image patch, the task is to classify 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-validation. 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.

- **TIMIT-4/6 Data [17].** This dataset is extracted from the TIMIT speech corpus using the dialect speaking region 4. It consists of 320 utterances from 16 male and 16 female speakers. Speech frames are classified into either four or six classes using 110134 and 121629 samples, respectively. Each sample is represented by 20 mel-frequency cepstral coefficients (MFCCs) and wavelet-based features [17]. We perform classification experiments on data of both genders (Ma+Fe). Furthermore, subsets of the data that consist of either male speakers (Ma) or female speakers (Fe) are considered.

- **USPS data [9].** This data set contains 11000 handwritten digit images from zip codes of mail envelopes. The data set is split into 8000 images for training and 3000 for testing. Each digit is represented as a 16×16 greyscale image. These greyscale values are discriminatively quantized [5] and each pixel is considered as feature. Example images from this database (prior to quantization) are shown in Figure 2.2.

Figure 2.2: Random samples from USPS data.

- **MNIST Data [12].** This dataset contains 70000 samples of handwritten digits. In the standard setting, 60000 samples are used for training and 10000 for testing. The digits represented by grey-level images were down-sampled by a factor of two resulting in a resolution of 16×16 pixels, i.e. 196 features.
- **DC-Mall Data [17].** This dataset contains a hyper-spectral remote sensing image of the Washington D.C. Mall area. In total, there are 1280×307 hyper-spectral pixels, each containing 191 spectral bands. From these spectral bands, individual pixels are to be classified to one of 7 classes (roof, road, grass, trees, trail, water, or shadow). For each class, 5000 samples are used for training, i.e. in total 35000 samples. The remaining 357960 samples are used for testing. These hyperspectral images for some frequency bands as well as the reference labeling are shown in Figure 2.3.



(a) Pseudo color image of spectral bands 63, 52, and 36

(b) Reference image

Figure 2.3: Hyperspectral image of the Washington D.C. Mall area.

Bibliography

[1] Kevin Bache and Moshe Lichman. UCI Machine Learning Repository, 2013. URL `http://archive.ics.uci.edu/ml`.

[2] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, 2007.

[3] David Maxwell Chickering, David Heckerman, and Christopher Meek. Large-Sample Learning of Bayesian Networks is NP-Hard. *Journal of Machine Learning Research*, 5: 1287–1330, 2004.

[4] Sanjoy Dasgupta. Learning Polytrees. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 134–141, 1999.

[5] Usama M. Fayyad and Keki B. Irani. Multi-Interval Discretization of Continuous-Valued Attributes for Classification Learning. In *International Conference on Artificial Intelligence (IJCAI)*, pages 1022–1029, 2003.

[6] Nir Friedman, Dan Geiger, and Moises Goldszmidt. Bayesian Network Classifiers. *Machine Learning*, 29:131–163, 1997.

[7] Yuhong Guo, Dana Wilkinson, and Dale Schuurmans. Maximum Margin Bayesian Networks. In *Uncertainty in Artificial Intelligence (UAI)*, pages 233–242, 2005.

[8] David Heckerman, Dan Geiger, and David M. Chickering. Learning Bayesian Networks: The Combination of Knowledge and Statistical Data. *Machine Learning*, 20: 197–243, 1995.

[9] Jonathan J. Hull. A Database for Handwritten Text Recognition Research. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 16(5):550–554, 1994.

[10] Eamonn J. Keogh. Learning Augmented Bayesian Classifiers: A Comparison of Distribution-based and Classification-based Approaches. In *International Workshop on Artificial Intelligence and Statistics*, pages 225–230, 1999.

[11] Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

[12] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[13] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc., 1988.

[14] Franz Pernkopf and Jeff Bilmes. Discriminative versus Generative Parameter and Structure Learning of Bayesian Network Classifiers. In *International Conference on Machine Learning (ICML)*, pages 657–664, 2005.

[15] Franz Pernkopf and Jeff A. Bilmes. Efficient Heuristics for Discriminative Structure Learning of Bayesian Network Classifiers. *Journal of Machine Learning Research*, 11: 2323–2360, 2010.

[16] Franz Pernkopf, Michael Wohlmayr, and Manfred Mücke. Maximum Margin Structure Learning of Bayesian Network Classifiers. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2076–2079, 2011.

[17] Franz Pernkopf, Michael Wohlmayr, and Sebastian Tschiatschek. Maximum Margin Bayesian Network Classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 34(3):521–531, 2012.

[18] Teemu Roos, Hannes Wettig, Peter Grünwald, Petri Myllymäki, and Henry Tirri. On Discriminative Bayesian Network Classifiers and Logistic Regression. *Journal of Machine Learning Research*, 59(3):267–296, 2005.

[19] Jiang Su, Harry Zhang, Charles X. Ling, and Stan Matwin. Discriminative Parameter Learning for Bayesian Networks. In *International Conference on Machine Learning (ICML)*, pages 1016–1023. ACM, 2008.

[20] Harry Zhang. The Optimality of Naive Bayes. In *International Florida Artificial Intelligence Research Society Conference (FLAIRS)*. AAAI Press, 2004.

3 On Asymptotic Consistency

The main parts of this chapter were published in the conference proceedings of the International Conference on Artificial Intelligence and Statistics 2013, as Asymptotic Optimality of Maximum Margin Bayesian Networks [12]. As minor modifications, we added some references, changed some wordings, and extended the discussion.

MM BNs are BNs with discriminatively optimized parameters. They have shown good classification performance in various applications [6]. However, there has not been any theoretic analysis of their asymptotic consistency, i.e. whether they represent the best possible classifier within the considered function class in the limit of infinite training data. For specific classes of MM BNs, i.e. MM BNs with fully connected graphs and discrete-valued nodes, we show consistency for binary-class problems and a sufficient condition for consistency in the multiclass case. We provide simple examples showing that MM BNs in their current formulation are not consistent in general. These examples are especially interesting, as the model used for the MM BNs can represent the assumed true distributions. This indicates that the current formulations of MM BNs may be deficient. Furthermore, experimental results on the generalization performance are presented.

3.1 Introduction

MM BNs were first introduced in [2]. The basic idea is to mimic the concept of the margin known from SVMs in a probabilistic environment. SVMs are one of the best performing classifiers available. In their basic formulation, they separate samples from different classes by a linear hyperplane. While SVMs are theoretically well-understood [13, 7, 10], there exist several issues that are hard to deal with. One example is the treatment of missing features in the data. SVMs usually require imputation techniques to complete the data before further processing [3]. In contrast, BNCs can naturally handle missing features.

Previous results [5] show that logistic regression, i.e. BNCs with NB structure and MCL parameters, exhibit lower asymptotic generalization error than classifiers with ML parameters. For MM BNs, no such results are available. However, comparable performance of these classifiers to SVMs has been reported [6]. This encourages the investigation of MM BNs in more detail. Specifically, in this chapter we address the issue of consistency, i.e. whether classifiers with parameters optimizing the MM objective yield asymptotically almost surely the optimal classifier.

Our findings can be summarized as follows:

1. MM BN classifiers with discrete-valued nodes are in general not consistent.
2. MM BN classifiers with discrete-valued nodes and fully connected graphs are consistent in binary-class classification tasks.
3. A sufficient condition for MM BN classifiers with discrete-valued nodes and fully connected graphs to be consistent in multiclass classification tasks is derived.

The remainder of this chapter is structured as follows: In Section 3.2, we introduce the notion of optimal classifiers and consistency. In Section 3.3, we present definitions of MM BNs from the literature followed by our theoretical results in Section 3.4. In Section 3.5 we illustrate the theoretical insights by empirical results and subsequently discuss some implications of our results in Section 3.6. Finally, we conclude this chapter in Section 3.7.

3.2 Background

We need to extend our notation to capture the necessary details for the study of consistency of MM BNs. Assume a joint probability distribution $P(C, \mathbf{X})$ over a class variable C and a

set of features $\mathbf{X}$. Note that $\arg\max_{c'} \mathrm{P}(c'|\mathbf{x})$ is not necessarily unique, i.e. different classes may achieve $\max_{c'} \mathrm{P}(c'|\mathbf{x})$. These classes are collected in the set $[C|\mathbf{x}]_{\mathrm{P}(C,\mathbf{X})}$, i.e.

$$[C|\mathbf{x}]_{\mathrm{P}(C,\mathbf{X})} = \left\{ c \mid \mathrm{P}(C = c|\mathbf{X} = \mathbf{x}) = \max_{c' \in \mathbf{val}(C)} \mathrm{P}(C = c'|\mathbf{X} = \mathbf{x}) \right\}.$$

Whenever $[C|\mathbf{x}]_{\mathrm{P}(C,\mathbf{X})}$ consists of more than a single class, we assume the classifier $h_{\mathrm{P}(C,\mathbf{X})}$ to return one of these classes uniformly at random[1] — this is termed as *optimally* classified with respect to $\mathrm{P}(C,\mathbf{X})$.

In the following, we consider the hypothesis class $\mathcal{B}(\mathcal{G})$ of BN classifiers with discrete RVs and fixed graph structure $\mathcal{G}$. Optimality of a classifier with respect to this hypothesis class is defined as follows:

Definition 2 (Optimal Classifier)**.** *A classifier h_B, $B \in \mathcal{B}(\mathcal{G})$ is optimal with respect to the hypothesis class $\mathcal{B}(\mathcal{G})$ if it satisfies*

$$\mathrm{Err}\,(h_B) = \inf_{B' \in \mathcal{B}(\mathcal{G})} \mathrm{Err}\,(h_{B'}). \tag{3.1}$$

A classifier from any hypotheses class can not be better than the *Bayes optimal classifier* $h_{\mathrm{P}^*(C,\mathbf{X})}$ [4]. The suboptimality of a classifier $h_B \in \mathcal{B}(\mathcal{G})$ compared to the Bayes optimal classifier can be expressed as

$$\mathrm{Err}\,(h_B) - \mathrm{Err}\left(h_{\mathrm{P}^*(C,\mathbf{X})}\right) = \left(\mathrm{Err}\,(h_B) - \inf_{B' \in \mathcal{B}(\mathcal{G})} \mathrm{Err}\,(h_{B'})\right) + \left(\inf_{B' \in \mathcal{B}(\mathcal{G})} \mathrm{Err}\,(h_{B'}) - \mathrm{Err}\left(h_{\mathrm{P}^*(C,\mathbf{X})}\right)\right), \tag{3.2}$$

where the first term is referred to as *estimation error* and the second term as *approximation error* [9]. The estimation error measures the suboptimality of the classifier h_B with respect to the class $\mathcal{B}(\mathcal{G})$, while the approximation error quantifies how close the best classifier in $\mathcal{B}(\mathcal{G})$ is to the Bayes optimal classifier. When considering limited graph structures $\mathcal{G}$, i.e. $\mathcal{G}$ is not fully connected, the generalization error of the Bayes optimal classifier can not be achieved in general, but there will at least be a bias corresponding to the approximation error. In this chapter, we consider the hypothesis classes of BNs with NB structure and fully connected graphs. As already mentioned, in NB structures the class node has no parents, i.e. $\mathbf{Pa}(C) = \emptyset$, and the only parent of any feature is the class node, i.e. $\mathbf{Pa}(X_i) = \{C\}$. In the fully connected graphs we consider, the class node has no parents, i.e. $\mathbf{Pa}(C) = \emptyset$, and for any feature $\mathbf{Pa}(X_i) = \{C, X_1, \ldots, X_{i-1}\}$, cf. Figure 3.1.

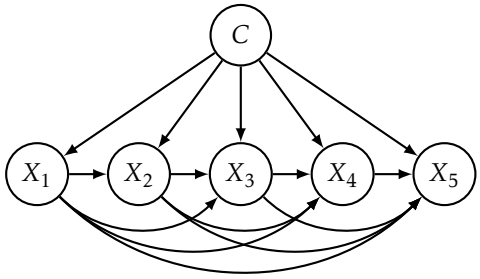


Figure 3.1: Exemplary fully connected graph.

Assume a training set $\mathcal{D} = \{(c^{(1)}, \mathbf{x}^{(1)}), \ldots, (c^{(N)}, \mathbf{x}^{(N)})\}$. Then, a sequence of classifiers $h_B^{A,N}$, where the superscript A, N denotes that the classifier is obtained from a training set of size N using the parameter learning method A (e.g. ML, MCL or MM), is *consistent* with respect to $\mathcal{B}(\mathcal{G})$, iff

$$\mathrm{Err}\left(h_B^{A,N}\right) \to \inf_{B' \in \mathcal{B}(\mathcal{G})} \mathrm{Err}\,(h_{B'}) \text{ a.s. as } N \to \infty. \tag{3.3}$$

[1] Technically, $h_{\mathrm{P}(C,\mathbf{X})}$ is not a mapping, because there is no unique assignment of $\mathbf{x} \in \mathbf{val}(\mathbf{X})$ to some $c \in \mathbf{val}(C)$. For ease of notation, we ignore this fact.

3.3 Maximum Margin Bayesian Networks

Although already introduced, we review MM BNs and specify some more details necessary for our analysis. We want to consider different definitions from the literature. Guo et al. [2] introduced MM BNs as a convex optimization problem for parameter learning. Later, the MM criterion was reformulated and a conjugate gradient based method for parameter learning was provided [6]. In experiments, both formulations have shown similar classification performance while the conjugate gradient optimization is beneficial in terms of computation cost. We briefly review both formulations and provide an example for which neither formulation retrieves a consistent classifier, although the optimal classifier is within the considered hypothesis class $\mathcal{B}(\mathcal{G})$. In the remainder of the chapter we adopt the MM BNs objective of Pernkopf et al. [6].

3.3.1 Formulation by Pernkopf et al.

Assuming a fixed graph $\mathcal{G}$, the objective for learning the joint probability $\mathrm{P}^{\mathcal{B}}(C,\mathbf{X})$ is based on the *margins*

$$d^{\mathcal{B}}(c^{(n)},\mathbf{x}^{(n)}) = \frac{\mathrm{P}^{\mathcal{B}}(c^{(n)},\mathbf{x}^{(n)})}{\max_{c'\neq c^{(n)}} \mathrm{P}^{\mathcal{B}}(c',\mathbf{x}^{(n)})} \tag{3.4}$$

of the training samples. Therefore, the n^{th} sample in the training set is classified correctly iff $d^{\mathcal{B}}(c^{(n)},\mathbf{x}^{(n)}) > 1$. To handle non separable data, a hinge function is used such that

$$\widetilde{d}^{\mathcal{B}}(c^{(n)},\mathbf{x}^{(n)}) = \min\left(\widetilde{\gamma}, d^{\mathcal{B}}(c^{(n)},\mathbf{x}^{(n)})\right), \tag{3.5}$$

where $\widetilde{\gamma} > 1$ is a parameter that controls the influence of the margins. The objective for learning MM BNs is maximization of the product of $\widetilde{d}^{\mathcal{B}}(c^{(n)},\mathbf{x}^{(n)})$ over the samples.

Definition 3 (Maximum Margin Bayesian Network)**.** *A BN $\mathcal{B} = (\mathcal{G},\mathcal{P}_{\mathcal{G}})$ that achieves the optimal value of*

$$\underset{\mathcal{B}'\in\mathcal{B}(\mathcal{G})}{\text{maximize}} \quad \prod_{n=1}^{N} \min\left(\widetilde{\gamma}, d^{\mathcal{B}'}(c^{(n)},\mathbf{x}^{(n)})\right) \tag{3.6}$$

is an MM BN.

This definition can be equivalently stated in the log-domain by requiring an MM BN $\mathcal{B}$ to be an optimal solution of

$$\underset{\mathcal{B}'\in\mathcal{B}(\mathcal{G})}{\text{maximize}} \quad \frac{1}{N}\sum_{n=1}^{N} \min\left(\gamma, \log \mathrm{P}^{\mathcal{B}'}(c^{(n)},\mathbf{x}^{(n)}) - \max_{c'\neq c^{(n)}} \log \mathrm{P}^{\mathcal{B}'}(c',\mathbf{x}^{(n)})\right), \tag{3.7}$$

where $\gamma = \log\widetilde{\gamma}$ and the objective is normalized by the number of training samples N. This allows the introduction of the empirical distribution on the training set $\mathrm{P}^{\mathcal{D}}(C,\mathbf{X})$, i.e.

$$\mathrm{P}^{\mathcal{D}}(c,\mathbf{x}) = \frac{1}{N}\sum_{n=1}^{N} \mathbf{1}_{(c^{(n)}=c,\mathbf{x}^{(n)}=\mathbf{x})} \tag{3.8}$$

to the optimization problem. The objective (3.7) becomes

$$\underset{\mathcal{B}'\in\mathcal{B}(\mathcal{G})}{\text{maximize}} \quad \sum_{c,\mathbf{x}} \mathrm{P}^{\mathcal{D}}(c,\mathbf{x}) \min\left(\gamma, \log \mathrm{P}^{\mathcal{B}'}(c,\mathbf{x}) - \max_{c'\neq c} \log \mathrm{P}^{\mathcal{B}'}(c',\mathbf{x})\right). \tag{3.9}$$

A justification why MM BNs parameters can be advantageous over BNCs with ML or MCL parameters is given in Appendix 3.A.

3.3.2 Formulation by Guo et al.

The formulation by Guo et al. [2] is based on representing the probabilities $\mathrm{P}^{\mathcal{B}}(c, \mathbf{x})$ as

$$\mathrm{P}^{\mathcal{B}}(c, \mathbf{x}) = \exp(\boldsymbol{\phi}(c, \mathbf{x})^T \mathbf{w}), \tag{3.10}$$

cf. Section 2.2.1. This enables to represent the logarithm of the margin (3.4) as

$$\log d^{\mathcal{B}}(c^{(n)}, \mathbf{x}^{(n)}) = \min_{c' \neq c^{(n)}} [\boldsymbol{\phi}(c^{(n)}, \mathbf{x}^{(n)}) - \boldsymbol{\phi}(c', \mathbf{x}^{(n)})]^T \mathbf{w}. \tag{3.11}$$

Learning the parameters of MM BNs is then performed by solving

$$\begin{aligned} \underset{\gamma, \mathbf{w}}{\text{minimize}} \quad & \frac{1}{2\gamma^2} + BN \sum_{c, \mathbf{x}} \mathrm{P}^{\mathcal{D}}(c, \mathbf{x}) \max\left(0, \gamma - \min_{c' \neq c} [\boldsymbol{\phi}(c, \mathbf{x}) - \boldsymbol{\phi}(c', \mathbf{x})]^T \mathbf{w}\right) \\ \text{s.t.} \quad & \sum_j \exp(w^i_{j|\mathbf{h}}) = 1, \forall i, \mathbf{h} \\ & \gamma > 0, \end{aligned} \tag{3.12}$$

where $B \geq 0$ is a trade-off parameter between a large margin and correct classification. To end up with a convex formulation, Guo et al. replace the constraints $\sum_j \exp(w^i_{j|\mathbf{h}}) = 1$ by inequalities, i.e. $\sum_j \exp(w^i_{j|\mathbf{h}}) \leq 1$. Due to this relaxation, the found parameters are in general not normalized. However, as pointed out in [8, 14], for certain network structures renormalization is possible without changing the classifier induced by the unnormalized parameters. The condition is for example satisfied by NB structures and fully connected graphs. The condition for renormalization is as follows:

Condition 1 (Renormalization [14])**.** *For all feature RVs X_j with $C \in \mathbf{Pa}(X_j)$ there exists another RV $X_i \in \mathbf{Pa}(X_j)$ such that $\mathbf{Pa}(X_j) \subseteq \mathbf{Pa}(X_i) \cup \{X_i\}$.*

3.3.3 Inconsistent Multiclass Maximum Margin Bayesian Networks

In this section we present an example for which both definitions of MM BNs result almost surely in inconsistent classifiers. Consider a classifier with no features, i.e. $\mathbf{X} = \emptyset$, in a three-class scenario. Let the true distribution be defined by

$$\begin{aligned} &\mathrm{P}^*(C = 1) = 0.4, \\ &\mathrm{P}^*(C = 2) = 0.3, \text{ and} \\ &\mathrm{P}^*(C = 3) = 0.3. \end{aligned}$$

Hence, the optimal classifier would classify all instances as belonging to class 1. By the strong law of large numbers, the empirical distribution will satisfy asymptotically almost surely $\mathrm{P}^{\mathcal{D}}(C = 1) > \mathrm{P}^{\mathcal{D}}(C = 2)$, $\mathrm{P}^{\mathcal{D}}(C = 1) > \mathrm{P}^{\mathcal{D}}(C = 3)$ and $\mathrm{P}^{\mathcal{D}}(C = 1) < \mathrm{P}^{\mathcal{D}}(C = 2) + \mathrm{P}^{\mathcal{D}}(C = 3)$. In this case, any distribution inducing an optimal classifier has strictly smaller (larger) objective than the uniform distribution according to problem (3.9) (problem (3.12)).[2] Consequently, any MM distribution induces almost surely an inconsistent classifier.

In this example, the optimal classifier can be represented by the assumed model. Nevertheless, an inconsistent classifier is obtained. We can deduce that in the multiclass case we must not hope for consistency of MM BNs in general.

3.4 Theoretical Results

3.4.1 Consistency of Fully Connected Maximum Margin Bayesian Networks

In this section, we show that fully-connected binary-class MM BN classifiers with discrete-valued nodes are consistent. Furthermore, we present a sufficient condition for consistency

[2] The necessary calculations for this result are straightforward, but provided in Appendix 3.E for completeness.

of multiclass MM BN classifiers. The proof consists of two parts. In the first part, we prove optimality with respect to the empirical distribution of the training set. In the second part, we conclude that MM BN classifiers are almost surely consistent.

Lemma 1 (Optimality in the binary-class case)**.** *Let C be a binary class variable, and let $\mathcal{D}$ be a training set with empirical distribution $\mathrm{P}^{\mathcal{D}}(C, \mathbf{X})$ and $\mathcal{G}$ a fully connected graph. Any MM BN classifier on $\mathcal{G}$ is optimal with respect to $\mathrm{P}^{\mathcal{D}}(C, \mathbf{X})$.*

The proof is provided in Appendix 3.B.

Lemma 2 (Optimality in the multiclass case)**.** *Let C be a class variable with $|\mathbf{val}(C)| > 2$, and let $\mathcal{D}$ be a training set with empirical distribution $\mathrm{P}^{\mathcal{D}}(C, \mathbf{X})$ and $\mathcal{G}$ a fully connected graph. Any MM BN classifier on $\mathcal{G}$ is optimal with respect to $\mathrm{P}^{\mathcal{D}}(C, \mathbf{X})$ if*

$$\forall \mathbf{x}\, \exists c:\ \mathrm{P}^{\mathcal{D}}(c, \mathbf{x}) > \sum_{c' \neq c} \mathrm{P}^{\mathcal{D}}(c', \mathbf{x}). \tag{3.13}$$

The proof is similar to that of Lemma 1 (for reference it is provided in Appendix 3.C). Bluntly speaking, condition (3.13) requires that for every instantiation of the features $\mathbf{x}$ there is a dominant class.

Using Lemma 1 and 2 we can derive the following theorem.

Theorem 1. *Any MM BN classifier with a fully connected graph is consistent if*

(a) $|\mathbf{val}(C)| = 2$, *i.e. the class variable is binary, or*

(b) $|\mathbf{val}(C)| > 2$, *i.e. the multiclass case, and additionally the true distribution $\mathrm{P}^*(C, \mathbf{X})$ satisfies*

$$\forall \mathbf{x}\, \exists c:\ \mathrm{P}^*(c, \mathbf{x}) > \sum_{c' \neq c} \mathrm{P}^*(c', \mathbf{x}). \tag{3.14}$$

Proof. We have already established that, given the stated conditions, MM BN classifiers are optimal with respect to the empirical distribution on the training set. With growing sample size *the empirical distribution converges to the true distribution* asymptotically almost surely. Therefore, the MM BN classifier converges asymptotically almost surely to the optimal classifier. □

3.4.2 Maximum Margin Bayesian Networks are not Necessarily Consistent

MM BN classifiers with not fully-connected graphs $\mathcal{G}$ are not consistent in general. This is even true in cases in which the true distribution can be represented by some BN $\mathcal{B} \in \mathcal{B}(\mathcal{G})$.

As an example consider a naive Bayes classifier with two features. Assume that the true data distribution $\mathrm{P}^*(C, \mathbf{X})$ satisfies the independence assumptions of the naive Bayes network and that the conditional probability densities are given according to Table 3.1a. For $\gamma = 1$, there exist MM BN classifiers that are consistent and MM BN classifiers that are inconsistent, i.e. there is no unique optimal (and consistent) solution. Corresponding MM distributions are shown in Table 3.1b, and Table 3.1c, respectively. The inconsistent distribution induces a classifier which has uniform class posterior for the samples ($x_1 = 1, x_2 = 1$) and ($x_1 = 1, x_2 = 2$). This results in a classification rate that is 4.5 percent smaller than the maximum classification rate, i.e.

$$\begin{aligned}
\mathrm{CR}\left(h_{\mathrm{P}^*(C,\mathbf{X})}\right) - \mathrm{CR}\left(h_{\mathrm{P}^{\mathrm{MM}}(C,\mathbf{X})}\right) =& \mathrm{P}^*(c_1, \mathbf{x}_1) - \frac{1}{2}\left(\mathrm{P}^*(c_1, \mathbf{x}_1) + \mathrm{P}^*(c_2, \mathbf{x}_1)\right) + \mathrm{P}^*(c_2, \mathbf{x}_2) - \\
& \frac{1}{2}\left(\mathrm{P}^*(c_1, \mathbf{x}_2) + \mathrm{P}^*(c_2, \mathbf{x}_2)\right) \\
=& 0.14 - \frac{1}{2}(0.14 + 0.12) + 0.28 - \frac{1}{2}(0.28 + 0.21) \\
=& 0.045,
\end{aligned} \tag{3.15}$$

where c_1 is a shorthand for $C = 1$, c_2 for $C = 2$, $\mathbf{x}_1$ for $X_1 = 1, X_2 = 1$, and $\mathbf{x}_2$ for $X_1 = 1, X_2 = 2$, respectively.

Table 3.1: Probability distribution for which MM BN classifiers can be inconsistent.

(a) True distribution $\mathrm{P}^*(C, \mathbf{X})$; Objective (3.9) evaluates to 0.049.

	$c=1$	$c=2$
$\mathrm{P}(C=c)$	0.5	0.5
$\mathrm{P}(X_1=1\|C=c)$	0.7	0.8
$\mathrm{P}(X_1=2\|C=c)$	0.3	0.2
$\mathrm{P}(X_2=1\|C=c)$	0.4	0.3
$\mathrm{P}(X_2=2\|C=c)$	0.6	0.7

C	X_1	X_2	$\mathrm{P}^*(C,\mathbf{X})$
1	1	1	0.14
2	1	1	0.12
1	2	1	0.06
2	2	1	0.03
1	1	2	0.21
2	1	2	0.28
1	2	2	0.09
2	2	2	0.07

(b) Inconsistent MM distribution $\mathrm{P}^{\mathrm{MM}}(C, \mathbf{X})$; Objective (3.9) evaluates to 0.05.

	$c=1$	$c=2$
$\mathrm{P}(C=c)$	0.5938	0.4062
$\mathrm{P}(X_1=1\|C=c)$	0.5	0.7311
$\mathrm{P}(X_1=2\|C=c)$	0.5	0.2689
$\mathrm{P}(X_2=1\|C=c)$	0.5	0.5
$\mathrm{P}(X_2=2\|C=c)$	0.5	0.5

C	X_1	X_2	$\mathrm{P}^{\mathrm{MM}}(C,\mathbf{X})$
1	1	1	0.1485
2	1	1	0.1485
1	2	1	0.1485
2	2	1	0.0546
1	1	2	0.1485
2	1	2	0.1485
1	2	2	0.1485
2	2	2	0.0546

(c) Consistent MM distribution $\mathrm{P}^{\mathrm{MM}}(C, \mathbf{X})$; Objective (3.9) evaluates to 0.05.

	$c=1$	$c=2$
$\mathrm{P}(C=c)$	0.5798	0.4202
$\mathrm{P}(X_1=1\|C=c)$	0.4750	0.5250
$\mathrm{P}(X_1=2\|C=c)$	0.5250	0.4750
$\mathrm{P}(X_2=1\|C=c)$	0.5	0.3100
$\mathrm{P}(X_2=2\|C=c)$	0.5	0.6900

C	X_1	X_2	$\mathrm{P}^{\mathrm{MM}}(C,\mathbf{X})$
1	1	1	0.1377
2	1	1	0.0684
1	2	1	0.1522
2	2	1	0.0619
1	1	2	0.1377
2	1	2	0.1522
1	2	2	0.1522
2	2	2	0.1377

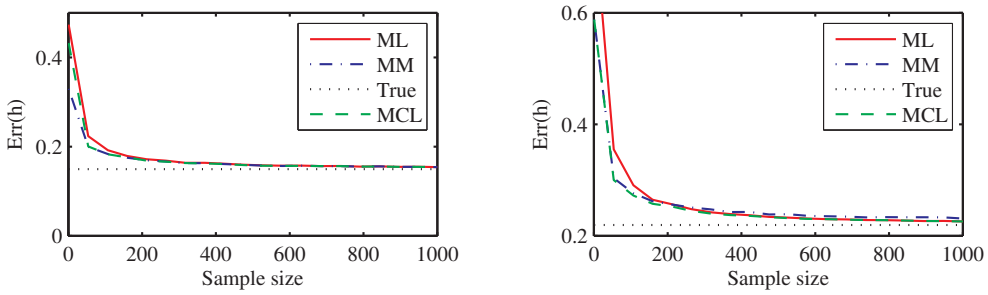
3.5 Experimental Results

We performed two experiments supporting the theoretical results in this chapter. Furthermore, an experiment demonstrating that MM BNs can perform well in the case of model mismatch is presented.

3.5.1 Bayes Consistent Classification Using Fully Connected Graphs

We assumed an arbitrary random distribution $P^*(C, \mathbf{X})$ for a fully connected graph. These distributions were obtained by selecting each entry of the conditional probabilities associated with the nodes of the graph uniformly at random in the range $[0, 1]$. To end up with properly normalized distributions, each conditional probability distribution was normalized subsequently. From the obtained distribution we generated training sets with an increasing number of samples. On these training sets we determined BN classifiers with fully connected graphs and using ML, MCL, and MM parameters. MM parameters are determined using the linear program provided in Appendix 3.D and employing 5-fold cross-validation to select the value of γ. We evaluated the generalization performance of these classifiers with respect to the true distribution.

As pointed out above, classifiers with both ML and MM parameters have to converge to the optimal classifier as the training set size increases. Results are averaged over 100 different training sets for every sample size using 100 different random parameter sets for the true distribution (the selected true distributions satisfy the condition of Theorem 1). Results for using 5 binary features and 2 classes, as well as 5 binary features and 4 classes are shown in Figures 3.2a and 3.2b, respectively. Convergence to the optimal classifier can be observed.



(a) 2 classes (b) 4 classes

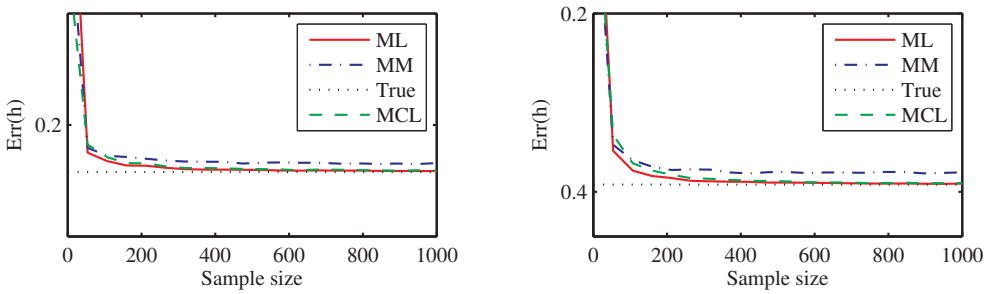
Figure 3.2: Convergence of ML, MCL and MM BN classifiers to the optimal classifier assuming a fully connected graph. The generalization error of the optimal classifier is indicated by the dotted line (= True).

3.5.2 Convergence Experiments Assuming Naive Bayes Structure

We repeated the experiment from above using true distributions satisfying the factorization properties of NB networks. BN classifiers with NB structure and ML, MCL and MM parameters are determined. In hope of obtaining unique MM parameters, we selected MM parameters with minimum ℓ_1-norm. Results for networks with 5 binary features and 2 classes, as well as 5 binary features and 4 classes are shown in Figures 3.3a and 3.3b, respectively. As noticed in Section 3.4, the MM BN classifiers do not converge to the optimal classifier, while ML and MCL classifiers achieve the lowest possible generalization error.

3.5.3 Model Mismatch

The setup for this experiments is similar to Section 3.5.1. For the true distribution an arbitrary distribution over $(C, \mathbf{X})$ is assumed, but the BN classifiers are determined using NB

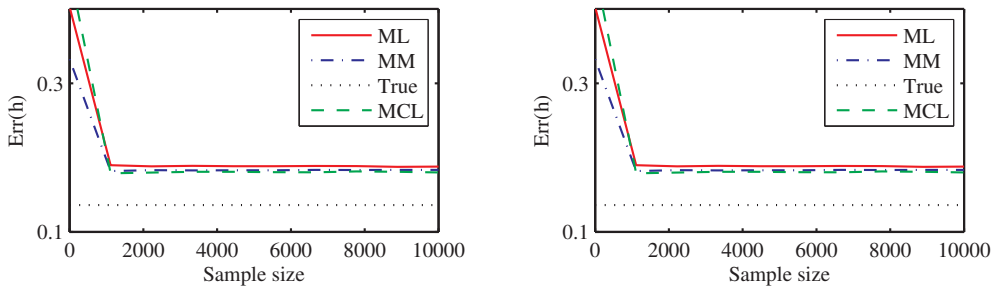


(a) 2 classes (b) 4 classes

Figure 3.3: Convergence of ML, MCL and MM BN classifiers assuming a NB structure. The generalization error of the optimal classifier is indicated by the dotted line (= True).

structures. The results for two-class and four-class classification are shown in Figures 3.4a and 3.4b, respectively.

We observe that classifiers with MCL and MM parameters converge to a lower asymptotic error than classifiers with ML parameters. This is consistent with the observations in [5] where generatively optimized NB classifiers are compared to logistic regression. In cases of model mismatch discriminative learning is usually beneficial. This is also supported by arguments in [8] with respect to MCL parameters: They argue that MCL parameters converge in probability to a distribution $\mathrm{P}^{\mathrm{MCL}}(C,\mathbf{X})$ that is closest in conditional KL-divergence to the true conditional probability $\mathrm{P}^*(C|\mathbf{X})$ (within the class of conditional distributions that can be represented by the fixed BN structure). In contrast, ML parameters converge to a distribution $\mathrm{P}^{\mathrm{ML}}(C,\mathbf{X})$ such that this distribution is closest in KL-divergence to $\mathrm{P}^*(C,\mathbf{X})$. For classification, however, a small conditional KL-divergence seems to be more directly related to the classification error than a small KL-divergence.



(a) 2 classes (b) 4 classes

Figure 3.4: Compensation of model mismatch assuming an arbitrary true distribution and ML, MCL and MM BN classifiers with NB structure. The generalization error of the optimal classifier is indicated by the dotted line (= True).

3.6 Discussion

We presented multiclass examples for which the optimal classifier can be represented by the considered models but is not retrieved by learning BN classifiers with MM parameters, cf. Sections 3.3.3 and 3.4.2. This suggests that the formulations of MM BNs is deficient — reasonable learning algorithms for classification purposes should, asymptotically, result in a consistent classifier in this setup. This result raises the question why good classification results have been reported in the literature, e.g. in [6]. We attribute these results to the model mismatch and the implemented early stopping heuristic: MM parameters are ob-

tained by starting at the maximum likelihood solution and subsequent maximization of the margin objective by gradient ascent. This maximization is stopped after a certain number of steps, where the stopping time is determined using 5-fold cross-validation. Thus, the obtained solution is in general not locally optimal. Consequently, in our earlier work we actually do not compute an MM BN but a blend between a BN with generatively and discriminatively optimized parameters.

Furthermore, we observed that in some binary-class examples for which the true distribution can be represented by the model, MM BNs do not necessarily induce a optimal classifiers, cf. Section 3.4.2. There are consistent and inconsistent parameters that achieve the same margin objective. This suggests that one should exploit the degrees of freedom still remaining after achieving a certain margin to optimize some additional criterion, e.g. maximization of the entropy or of the likelihood of the training data, cf. Chapter 4.

Our results can be aligned with more general results on consistency of multiclass classification methods. In Tewari and Bartlett [11], the authors derive sufficient and necessary conditions for the consistency of multiclass classification methods. They further show, that the margin formulation proposed in [1] (which essentially corresponds to the margin formulation used in (3.9)) does not satisfy these conditions and is, therefore, inconsistent. Nevertheless, this margin formulation is widely used in multiclass SVMs.

3.7 Conclusion and Future Work

In this chapter, we presented results on the consistency of MM BN classifiers with fully connected graphs. We provided examples where MM BN classifiers can be inconsistent and demonstrated experimentally that these classifiers are able to efficiently compensate model mismatch.

In future work, we aim to quantify the asymptotic *suboptimality* of MM BN classifiers in terms of the true distribution. We want to establish rates of convergence to the asymptotic performance. Furthermore, we aim at extending the definition of the margin objective. In particular, consistency shall be achieved whenever the true distribution can be represented by the considered BNs. We see at least three possible remedies in this regard:

1. The objective in (3.9), i.e.

$$\underset{\mathcal{B}'\in\mathcal{B}(\mathcal{G})}{\text{maximize}} \quad \sum_{c,\mathbf{x}} \mathrm{P}^{\mathcal{D}}(c,\mathbf{x}) \min\Big(\gamma, \log \mathrm{P}^{\mathcal{B}'}(c,\mathbf{x}) - \max_{c'\neq c} \log \mathrm{P}^{\mathcal{B}'}(c',\mathbf{x})\Big). \tag{3.16}$$

   can be modified such that for every possible feature instantiation only the margin of the most likely class under the empirical distribution $\mathrm{P}^{\mathcal{D}}(C,\mathbf{X})$ is maximized. This yields the objective

$$\underset{\mathcal{B}'\in\mathcal{B}(\mathcal{G})}{\text{maximize}} \sum_{\mathbf{x}} \mathrm{P}^{\mathcal{D}}(\mathbf{x}) \mathrm{P}^{\mathcal{D}}(h_{\mathrm{P}^{\mathcal{D}}(C,\mathbf{X})}(\mathbf{x})|\mathbf{x}) \min\Big(\gamma, \log \mathrm{P}^{\mathcal{B}'}(h_{\mathrm{P}^{\mathcal{D}}(C,\mathbf{X})}(\mathbf{x}),\mathbf{x}) - \max_{c'\neq \hat{c}(\mathbf{x})} \log \mathrm{P}^{\mathcal{B}'}(c',\mathbf{x})\Big), \tag{3.17}$$

   where $h_{\mathrm{P}^{\mathcal{D}}(C,\mathbf{X})}(\mathbf{x}) = \arg\max_c \mathrm{P}^{\mathcal{D}}(c,\mathbf{x})$. This should, assuming proper selection of γ, resolve the inconsistency at least for fully connected graphs. A rigorous investigation of this approach is subject to future work.

2. The objective in (3.9) can be extended by a generative regularizer, i.e. the data likelihood under the considered BN with ML parameters, cf. Chapter 4. Then, the regularizer weight can be set using cross-validation. In cases, where the margin-objective would yield an inconsistent classifier, this could be compensated by the regularizer. In extreme cases, overwhelming the margin-objective completely.

3. The loss function used in the objective (3.9) can be modified. Clearly, consistent loss functions should be used, cf. examples in [11, Section 5]. However, as a disadvantage, optimization of this modified objectives may be more challenging.

Appendix

3.A Maximum Margin Bayesian Networks Maximize a Lower Bound of the Classification Rate

The ideal approach for learning BN classifiers with fixed structure $\mathcal{G}$ would be solving

$$\underset{\mathcal{B}' \in \mathcal{B}(\mathcal{G})}{\text{maximize}} \quad \mathbb{E}_{\mathrm{P}^*(C,\mathbf{X})}\left[\mathbf{1}_{(h_{\mathcal{B}'}(\mathbf{X})=C)}\right], \tag{3.18}$$

i.e. maximization of the expected classification rate. Directly finding a solution to this problem is difficult as $\mathrm{P}^*(C,\mathbf{X})$ is unknown. Even if $\mathrm{P}^*(C,\mathbf{X})$ would be available, the maximization in general corresponds to a hard nonlinear optimization problem. Therefore, approximations are needed.

Solving (3.18) is equivalent to solving

$$\underset{\mathcal{B}' \in \mathcal{B}(\mathcal{G})}{\text{maximize}} \quad \sum_{c,\mathbf{x}} \mathrm{P}^*(c,\mathbf{x}) \mathbf{1}_{(h_{\mathcal{B}'}(\mathbf{x})=c)}. \tag{3.19}$$

The expression $\mathbf{1}_{(h_{\mathcal{B}'}(\mathbf{x})=c)}$ equals 1 iff $h_{\mathcal{B}'}(\mathbf{x}) = c$, or equivalently if $\mathrm{P}^{\mathcal{B}'}(c,\mathbf{x}) > \mathrm{P}^{\mathcal{B}'}(c',\mathbf{x})$ for all $c' \neq c$ (ignoring the possibility of equally large joint probabilities), otherwise it is zero. In comparison, the corresponding term in (3.9) with $\gamma = 1$ is at most 1 and positive iff $\mathrm{P}^{\mathcal{B}'}(c,\mathbf{x}) > \mathrm{P}^{\mathcal{B}'}(c',\mathbf{x})$ for all $c' \neq c$. Otherwise it is negative. Consequently,

$$\min\left(1, \log \mathrm{P}^{\mathcal{B}'}(c,\mathbf{x}) - \max_{c' \neq c} \log \mathrm{P}^{\mathcal{B}'}(c',\mathbf{x})\right) \leq \mathbf{1}_{(h_{\mathcal{B}'}(\mathbf{x})=c)}. \tag{3.20}$$

This holds for all c and $\mathbf{x}$. Therefore, the MM objective (3.9) asymptotically lower bounds the classification rate (as *the empirical distribution converges to the true distribution* with increasing sample size).

3.B Proof of Lemma 1

Proof. We give a proof by contradiction. Assume that $\mathcal{B}^{\mathrm{MM}} = (\mathcal{G}, \mathrm{P}^{\mathrm{MM}}(C,\mathbf{X}))$ is an MM BN trained on the training set $\mathcal{D}$ with empirical distribution $\mathrm{P}^{\mathcal{D}}(C,\mathbf{X})$. Additionally, assume that the induced classifier $h_{\mathrm{P}^{\mathrm{MM}}(C,\mathbf{X})}$ is not optimal with respect to $\mathrm{P}^{\mathcal{D}}(C,\mathbf{X})$. Thus, there exists an instantiation of the features $\mathbf{x}^f$ that is not optimally classified by $h_{\mathrm{P}^{\mathrm{MM}}(C,\mathbf{X})}$, i.e. for which

$$[C|\mathbf{x}^f]_{\mathrm{P}^{\mathrm{MM}}(C,\mathbf{X})} \setminus [C|\mathbf{x}^f]_{\mathrm{P}^{\mathcal{D}}(C,\mathbf{X})} \neq \emptyset. \tag{3.21}$$

Because of the binary class variable, the set $[C|\mathbf{x}^f]_{\mathrm{P}^{\mathcal{D}}(C,\mathbf{X})}$ consists only of a single element (otherwise deciding for any of the two classes is optimal).

We consider the cases $|[C|\mathbf{x}^f]_{\mathrm{P}^{\mathrm{MM}}(C,\mathbf{X})}| = 1$ and $|[C|\mathbf{x}^f]_{\mathrm{P}^{\mathrm{MM}}(C,\mathbf{X})}| = 2$ separately. Beforehand, note that since $\mathcal{G}$ is fully connected, i.e. $\mathcal{B}(\mathcal{G})$ is the set of *all* possible distributions over $(C,\mathbf{X})$, we can arbitrarily select the probabilities $\mathrm{P}^{\mathrm{MM}}(C = c, \mathbf{X} = \mathbf{x})$, as long as a correctly normalized distribution results. Consequently, we can select $\mathrm{P}^{\mathrm{MM}}(C = c|\mathbf{X})$ without changing $\mathrm{P}^{\mathrm{MM}}(\mathbf{X})$. We use this to show that the MM BN objective (3.9) can be strictly increased.

Case 1. If $[C|\mathbf{x}^f]_{\mathrm{P}^{\mathrm{MM}}(C,\mathbf{X})}$ consists of one element $c^f \in \mathbf{val}(C)$, then there exists a $c^* \in \mathbf{val}(C) \setminus \{c^f\}$ such that $\mathrm{P}^{\mathrm{MM}}(c^f|\mathbf{x}^f) > \mathrm{P}^{\mathrm{MM}}(c^*|\mathbf{x}^f)$ and such that $\mathrm{P}^{\mathcal{D}}(c^*|\mathbf{x}^f) > \mathrm{P}^{\mathcal{D}}(c^f|\mathbf{x}^f)$. We generate a new probability distribution $\tilde{\mathrm{P}}^{\mathrm{MM}}(C,\mathbf{X})$ from $\mathrm{P}^{\mathrm{MM}}(C,\mathbf{X})$ by setting

$$\tilde{\mathrm{P}}^{\mathrm{MM}}(c,\mathbf{x}) = \mathrm{P}^{\mathrm{MM}}(c,\mathbf{x}) \quad \forall\, \mathbf{x} \neq \mathbf{x}^f \;\forall c, \tag{3.22}$$

$$\tilde{\mathrm{P}}^{\mathrm{MM}}(c^f,\mathbf{x}^f) = \mathrm{P}^{\mathrm{MM}}(c^*,\mathbf{x}^f), \text{ and} \tag{3.23}$$

$$\tilde{\mathrm{P}}^{\mathrm{MM}}(c^*,\mathbf{x}^f) = \mathrm{P}^{\mathrm{MM}}(c^f,\mathbf{x}^f). \tag{3.24}$$

The distribution $\tilde{\mathrm{P}}^{\mathrm{MM}}(C,\mathbf{X})$ optimally classifies $\mathbf{x}^f$ and has higher objective (3.9) than $\mathrm{P}^{\mathrm{MM}}(C,\mathbf{X})$. Consequently, $\mathrm{P}^{\mathrm{MM}}(C,\mathbf{X})$ is no MM BN.

Case 2. If $[C|\mathbf{x}^f]_{\mathrm{P}^{\mathrm{MM}}(C,\mathbf{X})}$ consists of two elements, both classes have posterior probabilities of 0.5 according to $\mathrm{P}^{\mathrm{MM}}(C,\mathbf{X})$. Therefore, in the objective (3.9) the sum

$$\sum_c \mathrm{P}^{\mathcal{D}}(c,\mathbf{x}^f)\min\left(\gamma, \log \mathrm{P}^{\mathrm{MM}}(c,\mathbf{x}^f) - \max_{c'\neq c} \log \mathrm{P}^{\mathrm{MM}}(c',\mathbf{x}^f)\right) \tag{3.25}$$

evaluates to zero.

Let $c^*, c^f \in \mathbf{val}(C)$ satisfy $\mathrm{P}^{\mathcal{D}}(c^*|\mathbf{x}^f) > \mathrm{P}^{\mathcal{D}}(c^f|\mathbf{x}^f)$. As above, we generate a new distribution $\tilde{\mathrm{P}}^{\mathrm{MM}}(C,\mathbf{X})$ that classifies $\mathbf{x}^f$ optimally and has higher objective. The distribution $\tilde{\mathrm{P}}^{\mathrm{MM}}(C,\mathbf{X})$ is generated from $\mathrm{P}^{\mathrm{MM}}(C,\mathbf{X})$ by setting

$$\tilde{\mathrm{P}}^{\mathrm{MM}}(c,\mathbf{x}) = \mathrm{P}^{\mathrm{MM}}(c,\mathbf{x}) \quad \forall\, \mathbf{x} \neq \mathbf{x}^f \;\forall c, \tag{3.26}$$

$$\tilde{\mathrm{P}}^{\mathrm{MM}}(c^f,\mathbf{x}^f) = \frac{1}{1+\exp(\gamma)} \cdot \mathrm{P}^{\mathrm{MM}}(\mathbf{x}^f), \text{ and} \tag{3.27}$$

$$\tilde{\mathrm{P}}^{\mathrm{MM}}(c^*,\mathbf{x}^f) = \frac{\exp(\gamma)}{1+\exp(\gamma)} \cdot \mathrm{P}^{\mathrm{MM}}(\mathbf{x}^f). \tag{3.28}$$

The terms in the objective (3.9) that change their value, sum up to

$$\begin{aligned}
&\sum_c \mathrm{P}^{\mathcal{D}}(c,\mathbf{x}^f)\min\left(\gamma, \log \tilde{\mathrm{P}}^{\mathrm{MM}}(c,\mathbf{x}^f)\left(-\max_{c'\neq c} \log \tilde{\mathrm{P}}^{\mathrm{MM}}(c',\mathbf{x}^f)\right)\right) \\
&\quad = \gamma\left(\mathrm{P}^{\mathcal{D}}(c^*,\mathbf{x}^f) - \mathrm{P}^{\mathcal{D}}(c^f,\mathbf{x}^f)\right) \\
&\quad > 0.
\end{aligned}$$

As the objective increases, $\mathrm{P}^{\mathrm{MM}}(C,\mathbf{X})$ is not an MM BN. □

3.C Proof of Lemma 2

Proof. Similar to the proof of Lemma 1, we give a proof by contradiction and make the same assumptions. As the induced classifier $h_{\mathrm{P}^{\mathrm{MM}}(C,\mathbf{X})}$ is assumed not to be optimal with respect to $\mathrm{P}^{\mathcal{D}}(C,\mathbf{X})$, there exists an instantiation of the features $\mathbf{x}^f$ that is not optimally classified by $h_{\mathrm{P}^{\mathrm{MM}}(C,\mathbf{X})}$, i.e. for which

$$[C|\mathbf{x}^f]_{\mathrm{P}^{\mathrm{MM}}(C,\mathbf{X})} \setminus [C|\mathbf{x}^f]_{\mathrm{P}^{\mathcal{D}}(C,\mathbf{X})} \neq \emptyset. \tag{3.29}$$

Because of the assumption of Lemma 2, the set $[C|\mathbf{x}^f]_{\mathrm{P}^{\mathcal{D}}(C,\mathbf{X})}$ consists only of a single element. If $[C|\mathbf{x}^f]_{\mathrm{P}^{\mathrm{MM}}(C,\mathbf{X})}$ consists of a single element, a contradiction can be shown similar as in the binary-class case. If $[C|\mathbf{x}^f]_{\mathrm{P}^{\mathrm{MM}}(C,\mathbf{X})}$ consists of multiple elements, the sum

$$\sum_c \mathrm{P}^{\mathcal{D}}(c,\mathbf{x}^f)\min\left(\gamma, \log \mathrm{P}^{\mathrm{MM}}(c,\mathbf{x}^f) - \max_{c'\neq c} \log \mathrm{P}^{\mathrm{MM}}(c',\mathbf{x}^f)\right) \tag{3.30}$$

in the MM-objective evaluates to at most zero.

Let $\{c^*\} = [C|\mathbf{x}^f]_{\mathrm{P}^{\mathcal{D}}(C,\mathbf{X})}$, i.e. c^* satisfies $\mathrm{P}^{\mathcal{D}}(c^*|\mathbf{x}^f) > \mathrm{P}^{\mathcal{D}}(c'|\mathbf{x}^f)$ for all $c' \neq c^*$. We generate a new distribution $\widetilde{\mathrm{P}}^{\mathrm{MM}}(C,\mathbf{X})$ that classifies $\mathbf{x}^f$ optimally and has higher objective. The distribution $\widetilde{\mathrm{P}}^{\mathrm{MM}}(C,\mathbf{X})$ is constructed from $\mathrm{P}^{\mathrm{MM}}(C,\mathbf{X})$ by setting

$$\widetilde{\mathrm{P}}^{\mathrm{MM}}(c,\mathbf{x}) = \mathrm{P}^{\mathrm{MM}}(c,\mathbf{x}) \qquad \forall \mathbf{x} \neq \mathbf{x}^f \;\forall c, \tag{3.31}$$

$$\widetilde{\mathrm{P}}^{\mathrm{MM}}(c',\mathbf{x}^f) = \frac{1}{|\mathbf{val}(C)|-1+\exp(\gamma)} \cdot \mathrm{P}^{\mathrm{MM}}(\mathbf{x}^f) \quad \forall c' \neq c^*, \text{ and} \tag{3.32}$$

$$\widetilde{\mathrm{P}}^{\mathrm{MM}}(c^*,\mathbf{x}^f) = \frac{\exp(\gamma)}{|\mathbf{val}(C)|-1+\exp(\gamma)} \cdot \mathrm{P}^{\mathrm{MM}}(\mathbf{x}^f). \tag{3.33}$$

The terms in the objective that change their value, sum up to

$$\sum_c \mathrm{P}^{\mathcal{D}}(c,\mathbf{x}^f)\min\left(\gamma,\log\widetilde{\mathrm{P}}^{\mathrm{MM}}(c,\mathbf{x}^f)\ -\max_{c'\neq c}\log\widetilde{\mathrm{P}}^{\mathrm{MM}}(c',\mathbf{x}^f)\right)=\gamma\left(\mathrm{P}^{\mathcal{D}}(c^*,\mathbf{x}^f)-\sum_{c'\neq c}\mathrm{P}^{\mathcal{D}}(c',\mathbf{x}^f)\right)$$
$$>0,$$

where the inequality is due to the assumption of the Lemma. As the objective increases, $\mathrm{P}^{\mathrm{MM}}(C,\mathbf{X})$ is not an MM BN. □

3.D Maximum Margin Parameter Learning by Linear Programming

The convex optimization problem for learning MM parameters [2] is based on a relaxation of the normalization constraints inherent to learning probability distributions.

We exploit the renormalization property and ideas of Guo et al. [2] and Pernkopf et al. [6] to come up with a linear program for optimally learning MM BNs using the objective of Pernkopf et al. [6]. Assume a fixed graph $\mathcal{G}$ of the BNs. Then, we can express the joint probability $\mathrm{P}^{\mathcal{B}}(C,\mathbf{X})$ in (2.8) satisfying the independence properties implied by $\mathcal{G}$ as $\mathrm{P}^{\mathcal{B}}(C,\mathbf{X})=\exp(\boldsymbol{\phi}(C,\mathbf{X})^T\mathbf{w})$. Consequently, optimization problem (3.9), can be rewritten as

$$\begin{aligned}\underset{\mathbf{w}}{\text{maximize}}\quad & \sum_{c,\mathbf{x}}\mathrm{P}^{\mathcal{D}}(c,\mathbf{x})\min\left(\gamma,\boldsymbol{\phi}(c,\mathbf{x})^T\mathbf{w}-\max_{c'\neq c}\boldsymbol{\phi}(c',\mathbf{x})^T\mathbf{w}\right),\\ \text{s.t.}\quad & \sum_j\exp\left(w^i_{j|\mathbf{h}}\right)=1\quad\forall i,\mathbf{h}\in\mathbf{val}(\mathbf{Pa}(X_i)),\end{aligned}\tag{3.34}$$

where optimization is now solved over the log-parameters $\mathbf{w}$. The constraints ensure that $\mathbf{w}$ represents properly normalized conditional probabilities.

Problem (3.34) can readily be expressed as the optimization problem

$$\begin{aligned}\underset{\mathbf{w},\gamma_{(c,\mathbf{x})}}{\text{maximize}}\quad & \sum_{c,\mathbf{x}}\mathrm{P}^{\mathcal{D}}(c,\mathbf{x})\gamma_{(c,\mathbf{x})}\\ \text{s.t.}\quad & \gamma_{(c,\mathbf{x})}\leq\gamma\quad\forall c,\mathbf{x}\\ & \gamma_{(c,\mathbf{x})}\leq\left[\boldsymbol{\phi}(c,\mathbf{x})-\boldsymbol{\phi}(c',\mathbf{x})\right]^T\mathbf{w}\quad\forall c,\mathbf{x}\,\forall c'\neq c\\ & \sum_j\exp\left(w^i_{j|\mathbf{h}}\right)=1\quad\forall i,\mathbf{h}\in\mathbf{val}(\mathbf{Pa}(X_i)).\end{aligned}\tag{3.35}$$

This problem is nonlinear and non-convex. To achieve convexity, Guo et al. [2] relaxed the normalization constraints[3] to

$$\sum_j\exp\left(w^i_{j|\mathbf{h}}\right)\leq 1\quad\forall i,\mathbf{h}\in\mathbf{val}(\mathbf{Pa}(X_i)).\tag{3.36}$$

These relaxed constraints have the disadvantage to cancel the effect of the parameter γ: To see this, consider the problem in (3.35). If the normalization constraints are neglected, a linear program results. The dual of this linear program exhibits that its solutions in terms of $\mathbf{w}$ are independent of γ. However, every solution of the linear program can be transformed to a feasible solution of (3.35) by subtracting a sufficiently large quantity from each component of $\mathbf{w}$. This subtraction does not change the objective and the induced classifier.

[3] Guo et al. [2] used a different objective function. However, the implications are the same.

To achieve the desired effect of γ, we constrain the components of $\mathbf{w}$ to be smaller than 0 and use an ℓ_1-norm constraint on $\mathbf{w}$. The resulting linear program is

$$\begin{aligned} \underset{\mathbf{w},\gamma_{c,\mathbf{x}}}{\text{maximize}} \quad & \sum_{c,\mathbf{x}} \mathrm{P}^{\mathcal{D}}(c,\mathbf{x})\gamma_{(c,\mathbf{x})} \\ \text{s.t.} \quad & \gamma_{(c,\mathbf{x})} \leq \gamma \quad \forall c,\mathbf{x} \\ & \gamma_{(c,\mathbf{x})} \leq \left[\boldsymbol{\phi}(c,\mathbf{x}) - \boldsymbol{\phi}(c',\mathbf{x})\right]^T \mathbf{w} \quad \forall c,\mathbf{x}\ \forall c' \neq c \\ & -\sum_{i,j,\mathbf{h}} w^i_{j|\mathbf{h}} \leq 1, \quad \mathbf{w} \leq 0. \end{aligned} \tag{3.37}$$

A parameter vector $\mathbf{w}^*$ solving (3.37) will in general not represent a properly normalized distribution. Whenever the renormalization Condition 1 is satisfied, normalization is possible without changing $h_{\mathbf{w}^*}$, where $h_{\mathbf{w}^*}$ is the classifier induced by $\mathbf{w}^*$. Roughly speaking, normalization can be achieved as follows (details are provided in [14]): Due to the DAG $\mathcal{G}$ assumed for the BNCs, the nodes of these classifiers can be topologically ordered. The conditional probabilities of the nodes can be sequentially normalized in a bottom up manner starting with the last node in the topological ordering. Multiplicative factors required for normalization are handed to the parent nodes. This does not affect the normalization of previous nodes.

If Condition 1 is not satisfied, the parameters can still be normalized. However, the resulting parameters are not guaranteed to maximize (3.9).

3.E Optimal Maximum Margin Bayesian Networks for the Three-Class Example

3.E.1 Review of the Example

Consider a classifier with no features, i.e. $\mathbf{X} = \emptyset$, in a three-class scenario. Let the true distribution be defined by

$$\begin{aligned} &\mathrm{P}^*(C=1) = 0.4, \\ &\mathrm{P}^*(C=2) = 0.3, \text{and} \\ &\mathrm{P}^*(C=3) = 0.3. \end{aligned}$$

Hence, the Bayes optimal classifier would classify all instances as belonging to class 1. However, in this case, any distribution inducing a Bayes optimal classifier has strictly smaller (larger) objective than the uniform distribution according to problem (3.9) (problem (3.12)). Consequently, any MM distribution induces an inconsistent classifier almost surely. In the remainder of this section, we assume that $\mathrm{P}^{\mathcal{D}}(C=1) > \mathrm{P}^{\mathcal{D}}(C=2)$, $\mathrm{P}^{\mathcal{D}}(C=1) > \mathrm{P}^{\mathcal{D}}(C=3)$ and $\mathrm{P}^{\mathcal{D}}(C=1) < \mathrm{P}^{\mathcal{D}}(C=2) + \mathrm{P}^{\mathcal{D}}(C=3)$ which holds asymptotically a.s.

3.E.2 Optimal Solution According to Guo et al.

For the considered example, MM BNs according to the formulation by Guo et al. can be found by minimizing

$$\begin{aligned} \frac{1}{2\gamma^2} + BN\Big(& \mathrm{P}^{\mathcal{D}}(C=1)\max\{0, \gamma - \min\{w_1 - w_2, w_1 - w_3\}\} \\ & + \mathrm{P}^{\mathcal{D}}(C=2)\max\{0, \gamma - \min\{w_2 - w_1, w_2 - w_3\}\} \\ & + \mathrm{P}^{\mathcal{D}}(C=3)\max\{0, \gamma - \min\{w_3 - w_1, w_3 - w_2\}\}\Big) \end{aligned}$$

with respect to w_1, w_2, w_3 and γ, under the constraints that $\gamma \geq 0$ and $\exp(w_1) + \exp(w_2) + \exp(w_3) \leq 1$.

Assuming a solution corresponding to the uniform distribution, i.e. $w_1 = w_2 = w_3 = \log(\frac{1}{3})$, the minimization problem becomes

$$\frac{1}{2\gamma^2} + BN\gamma, \tag{3.38}$$

again, subject to $\gamma \geq 0$ and $\exp(w_1) + \exp(w_2) + \exp(w_3) \leq 1$. The latter constraint is clearly satisfied. The optimal value of γ can be determined by setting the derivative of (3.38) to zero. This results in an objective value of

$$\frac{1}{2\,(BN)^{-\frac{2}{3}}} + \sqrt[3]{B^2N^2}. \tag{3.39}$$

We now show by lower-bounding the objective that for any parameters w_1, w_2, w_3 corresponding to the Bayes optimal classifier, the objective is strictly larger than (3.39). Assume that $w_1 > w_2$, $w_1 > w_3$ and without loss of generality, that $w_2 \geq w_3$ (this parameters correspond to a Bayes optimal classifier). Then the following chain of inequalities results:

$$\begin{aligned}
&\frac{1}{2\gamma^2} + BN\Big(\mathrm{P}^{\mathcal{D}}(C=1)\max\{0, \gamma - \min\{w_1 - w_2, w_1 - w_3\}\} \\
&\qquad + \mathrm{P}^{\mathcal{D}}(C=2)\max\{0, \gamma - \min\{w_2 - w_1, w_2 - w_3\}\} \\
&\qquad + \mathrm{P}^{\mathcal{D}}(C=3)\max\{0, \gamma - \min\{w_3 - w_1, w_3 - w_2\}\}\Big) \\
&= \frac{1}{2\gamma^2} + BN\Big(\mathrm{P}^{\mathcal{D}}(C=1)\max\{0, \gamma - (w_1 - w_2)\} \\
&\qquad + \mathrm{P}^{\mathcal{D}}(C=2)\max\{0, \gamma - (w_2 - w_1)\} \\
&\qquad + \mathrm{P}^{\mathcal{D}}(C=3)\max\{0, \gamma - (w_3 - w_1)\}\Big) \\
&\overset{(a)}{\geq} \frac{1}{2\gamma^2} + BN\Big(\mathrm{P}^{\mathcal{D}}(C=1)\max\{0, \gamma - (w_1 - w_2)\} \\
&\qquad + \Big(\mathrm{P}^{\mathcal{D}}(C=2) + \mathrm{P}^{\mathcal{D}}(C=3)\Big)\max\{0, \gamma - (w_2 - w_1)\}\Big) \\
&\overset{(b)}{\geq} \frac{1}{2\gamma^2} + BN\Big(\mathrm{P}^{\mathcal{D}}(C=1)(\gamma - (w_1 - w_2)) \\
&\qquad + \Big(\mathrm{P}^{\mathcal{D}}(C=2) + \mathrm{P}^{\mathcal{D}}(C=3)\Big)(\gamma - (w_2 - w_1))\Big) \\
&\overset{(c)}{>} \frac{1}{2\gamma^2} + BN\gamma,
\end{aligned} \tag{3.40}$$

where (a) is because $w_2 - w_1 \leq w_3 - w_1$, (b) by selecting arbitrary elements instead of performing the maximum operations, and (c) because the empiric distribution satisfies $\mathrm{P}^{\mathcal{D}}(C=2) + \mathrm{P}^{\mathcal{D}}(C=3) > \mathrm{P}^{\mathcal{D}}(C=1)$ almost surely as $N \to \infty$. Consequently, an MM BN according to the formulation of Guo et al. must not be Bayes optimal for this example almost surely.

3.E.3 Optimal Solution According to Pernkopf et al.

For the considered example, MM BNs according to the formulation by Pernkopf et al. can be found by maximizing

$$\begin{aligned}
&\mathrm{P}^{\mathcal{D}}(C=1)\min(\gamma, \log\theta_1 - \max\{\log\theta_2, \log\theta_3\}) \\
+&\mathrm{P}^{\mathcal{D}}(C=2)\min(\gamma, \log\theta_2 - \max\{\log\theta_1, \log\theta_3\}) \\
+&\mathrm{P}^{\mathcal{D}}(C=3)\min(\gamma, \log\theta_3 - \max\{\log\theta_1, \log\theta_2\})
\end{aligned} \tag{3.41}$$

with respect to $\theta_1, \theta_2, \theta_3$, where $\mathrm{P}^{\mathrm{MM}}(C = 1) = \theta_1, \ldots, \mathrm{P}^{\mathrm{MM}}(C = 3) = \theta_3$. In the case $\theta_1 = \theta_2 = \theta_3 = \frac{1}{3}$ the objective (3.41) evaluates to zero.

We now show by calculation, that any $(\theta_1, \theta_2, \theta_3)$ that would correspond to a Bayes optimal classifier results in a strictly smaller objective. For this, assume that $\theta_1 > \theta_2$ and $\theta_1 > \theta_3$. Without loss of generality, additionally assume that $\theta_2 \geq \theta_3$. Consequently,

$$
\begin{aligned}
&\mathrm{P}^{\mathcal{D}}(C = 1) \min(\gamma, \log\theta_1 - \max\{\log\theta_2, \log\theta_3\}) \\
&\quad + \mathrm{P}^{\mathcal{D}}(C = 2) \min(\gamma, \log\theta_2 - \max\{\log\theta_1, \log\theta_3\}) \\
&\quad + \mathrm{P}^{\mathcal{D}}(C = 3) \min(\gamma, \log\theta_3 - \max\{\log\theta_1, \log\theta_2\}) \\
&= \mathrm{P}^{\mathcal{D}}(C = 1) \min(\gamma, \log\theta_1 - \log\theta_2) \\
&\quad + \mathrm{P}^{\mathcal{D}}(C = 2) \min(\gamma, \log\theta_2 - \log\theta_1) \\
&\quad + \mathrm{P}^{\mathcal{D}}(C = 3) \min(\gamma, \log\theta_3 - \log\theta_1) \\
&\overset{(a)}{\leq} \mathrm{P}^{\mathcal{D}}(C = 1) \min(\gamma, \log\theta_1 - \log\theta_2) \\
&\quad + (\mathrm{P}^{\mathcal{D}}(C = 2) + \mathrm{P}^{\mathcal{D}}(C = 3)) \min(\gamma, \log\theta_2 - \log\theta_1) \\
&\overset{(b)}{<} (\mathrm{P}^{\mathcal{D}}(C = 2) + \mathrm{P}^{\mathcal{D}}(C = 3)) \min(\gamma, \log\theta_1 - \log\theta_2) \\
&\quad + (\mathrm{P}^{\mathcal{D}}(C = 2) + \mathrm{P}^{\mathcal{D}}(C = 3)) \min(\gamma, \log\theta_2 - \log\theta_1) \\
&= (\mathrm{P}^{\mathcal{D}}(C = 2) + \mathrm{P}^{\mathcal{D}}(C = 3)) \min(\gamma, \log\theta_1 - \log\theta_2) \\
&\quad - (\mathrm{P}^{\mathcal{D}}(C = 2) + \mathrm{P}^{\mathcal{D}}(C = 3)) (\log\theta_1 - \log\theta_2) \\
&\overset{(c)}{\leq} 0,
\end{aligned}
$$

where (a) is because $\theta_2 \geq \theta_3$ by assumption, (b) because $\mathrm{P}^{\mathcal{D}}(C = 1) < \mathrm{P}^{\mathcal{D}}(C = 2) + \mathrm{P}^{\mathcal{D}}(C = 3)$, and (c) because $\log\theta_1 - \log\theta_2$ is bounded by γ. Hence, any MM BN must not be Bayes optimal for this example almost surely.

Bibliography

[1] Koby Crammer and Yoram Singer. On the Algorithmic Implementation of Multiclass Kernel-based Vector Machines. *Journal of Machine Learning Research*, 2:265–292, 2002.

[2] Yuhong Guo, Dana Wilkinson, and Dale Schuurmans. Maximum Margin Bayesian Networks. In *Uncertainty in Artificial Intelligence (UAI)*, pages 233–242, 2005.

[3] Roderick J. A. Little and Donald B. Rubin. *Statistical Analysis with Missing Data*. Wiley-Interscience, 2nd edition, 2002.

[4] Thomas M. Mitchell. *Machine Learning*. McGraw-Hill, Inc., New York, NY, USA, 1st edition, 1997.

[5] Andrew Y. Ng and Michael I. Jordan. On Discriminative vs. Generative Classifiers: A Comparison of Logistic Regression and Naive Bayes. In *Advances in Neural Information Processing Systems (NIPS)*, 2001.

[6] Franz Pernkopf, Michael Wohlmayr, and Sebastian Tschiatschek. Maximum Margin Bayesian Network Classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 34(3):521–531, 2012.

[7] John C. Platt. Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines. *Advances in Kernel Methods-Support Vector Learning*, pages 1–21, 1999.

[8] Teemu Roos, Hannes Wettig, Peter Grünwald, Petri Myllymäki, and Henry Tirri. On Discriminative Bayesian Network Classifiers and Logistic Regression. *Journal of Machine Learning Research*, 59(3):267–296, 2005.

[9] Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014.

[10] Shai Shalev-Shwartz, Yoram Singer, and Nathan Srebro. Pegasos: Primal Estimated sub-GrAdient SOlver for SVM. In *International Conference on Machine Learning*, pages 807–814, 2007.

[11] Ambuj Tewari and Peter L. Bartlett. On the Consistency of Multiclass Classification Methods. *Journal of Machine Learning Research*, 8:1007–1025, 2007.

[12] Sebastian Tschiatschek and Franz Pernkopf. Asymptotic Optimality of Maximum Margin Bayesian Networks. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 590–598, 2013.

[13] Vladimir N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998.

[14] Hannes Wettig, Peter Grünwald, Teemu Roos, Petri Myllymaki, and Henry Tirri. When Discriminative Learning of Bayesian Network Parameters is Easy. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 491–496, 2003.

4 Generative Maximum Margin Classifiers

This chapter was published in the conference proceedings of the International Conference on Machine Learning 2013 under the title of The Most Generative Maximum Margin Bayesian Networks [20]. The paper is joint work with Robert Peharz. In this thesis, the manuscript is presented with minor modifications and restructuring. Furthermore, some additional material is added to the appendix.

Although discriminative learning in graphical models generally improves classification results, the generative semantics of the model are compromised. In this chapter, we introduce a novel approach of hybrid generative-discriminative learning for BNs. 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 perform on par with state-of-the art generative and discriminative learning methods for BNs, 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.

4.1 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.

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 [10, 19, 27, 11, 12, 23, 21]. 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 not considered. 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 [17]. The motivation in this chapter 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) [16] which combines discriminative estimation with generative models in a principled way.

In this chapter, we consider an SVM-type maximum margin approach for BNs [7, 12, 21]. 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 [28, 29] 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. In our approach, the model parameters are automatically normalized for specific network structures, i.e. the parameters describe proper probability distributions. Thus, the weighted ℓ^1-norm can be interpreted as *likelihood-term*. Therefore, we can interpret our model as a *likelihood-aware SVM*. When the trade-off parameter between the weighted ℓ^1-norm and the sample-margins 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 [22, 3, 2], 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 extend our notation in Section 4.2. In Section 4.3, we present our formulation as convex optimization problem and state Theorems 2 and 3 which guarantee correctly normalized BN parameters, permitting the additional likelihood-interpretation. In Section 4.4, we propose a projected gradient method which is scalable to large datasets. In Section 4.5 we report results on benchmark datasets. Section 4.6 concludes the chapter.

4.2 Background and Notation

Let $\mathbf{w}$ be a vector of BN parameters in the logarithmic domain, cf. Section 2.2.1. We say that $\mathbf{w}$ is *sub-normalized*, iff

$$\log \sum_{j \in \mathbf{val}(X_i)} \exp(w^i_{j|\mathbf{h}}) \leq 0, \quad \forall X_i, \forall \mathbf{h} \in \mathbf{val}(\mathbf{Pa}(X_i)), \tag{4.1}$$

and $\mathbf{w}$ is *normalized*, iff (4.1) holds with equality. A vector $\mathbf{w}$ is *strictly* sub-normalized, iff it is sub-normalized, but not normalized. In order to represent valid BN parameters, $\mathbf{w}$ has to be normalized.

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

$$\hat{w}^i_{j|\mathbf{h}} = \log\left(\frac{m^i_{j|\mathbf{h}}}{m^i_{\mathbf{h}}}\right), \tag{4.2}$$

where

$$m^i_{j|\mathbf{h}} = \left(\sum_{n=1}^{N} \nu^{i,n}_{j|\mathbf{h}}\right) + \frac{\alpha}{|\mathbf{val}(X_i)||\mathbf{val}(\mathbf{Pa}(X_i))|}, \tag{4.3}$$

$$m^i_{\mathbf{h}} = \sum_{j \in \mathbf{val}(X_i)} m^i_{j|\mathbf{h}}, \text{ and} \tag{4.4}$$

$$\nu^{i,n}_{j|\mathbf{h}} = \mathbf{1}_{\left([c^{(n)}, \mathbf{x}^{(n)}](X_i) = j \text{ and } [c^{(n)}, \mathbf{x}^{(n)}](\mathbf{Pa}(X_i)) = \mathbf{h}\right)}. \tag{4.5}$$

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 $|\mathbf{val}(X_i)||\mathbf{val}(\mathbf{Pa}(X_i))|$ achieves that the "virtual samples" are distributed consistently among the CPTs. We say that the likelihood-counts are *consistent*, when for all X_k, $j \in \mathbf{val}(X_k)$, $X_i \in \mathbf{Ch}(X_k)$, and $\mathbf{h} \in \mathbf{val}(\mathbf{Pa}(X_k) \cap \mathbf{Pa}(X_i))$ it holds that

$$\sum_{\mathbf{h}' \in \mathbf{val}(\mathbf{A})} m^k_{j|[\mathbf{h},\mathbf{h}']} = \sum_{\mathbf{h}'' \in \mathbf{val}(\mathbf{B})} \sum_{j' \in \mathbf{val}(X_i)} m^i_{j'|[\mathbf{h},j,\mathbf{h}'']} \tag{4.6}$$

where $\mathbf{A} = \mathbf{Pa}(X_k) \backslash \mathbf{Pa}(X_i)$ and $\mathbf{B} = \mathbf{Pa}(X_i) \backslash (\mathbf{Pa}(X_k) \cup \{X_k\})$, and where $\mathbf{Ch}(X_i)$ are the children of X_i according to $\mathcal{G}$. For $\alpha > 0$, Equation (4.2) is also the MAP solution using Dirichlet priors according to [4, 14].

Our discussion will concentrate on structures satisfying the following condition, as identified in [27]:

Condition 2. *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 2 are denoted as *C-structures*. The class of these structures is quite rich, containing, amongst others, the NBs structure, the TANs [10], and diagnostic networks [27]. C-structures facilitate discriminative learning, since for each unnormalized parameter vector there exists also a normalized parameter vector, specifying the *same conditional distribution* $\mathrm{P}_{\mathcal{B}}(C|\mathbf{X})$. Wettig et al. [27] provided a constructive proof, by proposing Algorithm 1 (shown in Appendix 4.A) for normalizing a set of unnormalized BN parameters, while leaving $\mathrm{P}_{\mathcal{B}}(C|\mathbf{X})$ unchanged. Condition 2 allows a convex relaxation of our optimization problem, presented in Section 4.3, 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 [21].

4.3 A "Generative" Maximum Margin Formulation

As defined before, the probabilistic margin $d^{\mathcal{B}}(c^{(n)}, \mathbf{x}^{(n)})$ of the n^{th} sample is defined as [12, 21]

$$d^{\mathcal{B}}(c^{(n)}, \mathbf{x}^{(n)}) = \frac{\mathrm{P}_{\mathcal{B}}(c^{(n)}|\mathbf{x}^{(n)})}{\max\limits_{c \neq c^{(n)}} \mathrm{P}_{\mathcal{B}}(c|\mathbf{x}^{(n)})} = \frac{\mathrm{P}_{\mathcal{B}}(c^{(n)}, \mathbf{x}^{(n)})}{\max\limits_{c \neq c^{(n)}} \mathrm{P}_{\mathcal{B}}(c, \mathbf{x}^{(n)})}. \tag{4.7}$$

By defining $\phi_c(\mathbf{x}) = \phi(c, \mathbf{x})$, we can express the logarithm of (4.7) as

$$\log d^{\mathcal{B}}(c^{(n)}, \mathbf{x}^{(n)}) = \min_{c \neq c^{(n)}} \Big[(\phi_{c^{(n)}}(\mathbf{x}^{(n)}) - \phi_c(\mathbf{x}^{(n)}))^T \mathbf{w} \Big]. \tag{4.8}$$

When we interpret $\phi_c(\mathbf{x}^{(n)})$ as (class-dependent) feature transformation, we can formulate the following multiclass SVM-type training for BNs [7, 8, 12]:

$$\begin{aligned} \underset{\mathbf{w}, \boldsymbol{\xi}}{\text{minimize}} \quad & \|\mathbf{w}\| + \lambda \sum_{n=1}^{N} \xi_n \\ \text{s.t.} \quad & (\phi_{c^{(n)}}(\mathbf{x}^{(n)}) - \phi_c(\mathbf{x}^{(n)}))^T \mathbf{w} + \xi_n \geq 1 \quad \forall n, c \neq c^{(n)} \end{aligned} \tag{4.9}$$

Here, $\|\mathbf{w}\|$ 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 (4.9) the *BN-SVM*. In general, a solution of the BN-SVM is not normalized, i.e. typically $\log \sum_{j'} \exp(w^i_{j'|\mathbf{h}}) \neq 0$, for some i, $\mathbf{h}$. However, since we consider C-structures, we can simply apply Algorithm 1, cf. Appendix 4.A, 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 (4.9)? 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 (4.9) and use the following weighted ℓ^1-norm for the BN-SVM norm term: $\mathrm{L}_{\mathbf{m}}(\mathbf{w}) = \sum_{i,j,\mathbf{h}} |m^i_{j|\mathbf{h}}\, w^i_{j|\mathbf{h}}|$. Here, the weights $m^i_{j|\mathbf{h}}$ are the likelihood-counts according to (4.3), collected in a vector $\mathbf{m}$. Furthermore, we subject the vector $\mathbf{w}$ to sub-normalization constraints (4.1). 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 $\mathbf{w}$, yielding the same classifier. However, for the BN-SVM according to (4.9), 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 (4.1) imply $w^i_{j|\mathbf{h}} \leq 0$, we can re-write $\mathrm{L}_{\mathbf{m}}(\mathbf{w}) = -\sum_{i,j,\mathbf{h}} m^i_{j|\mathbf{h}}\, w^i_{j|\mathbf{h}} = -\mathbf{m}^T\mathbf{w}$. Finally, we get the modified convex problem:

$$\begin{aligned} \underset{\mathbf{w},\boldsymbol{\xi}}{\text{minimize}} \quad & -\mathbf{m}^T\mathbf{w} + \lambda \sum_{n=1}^{N} \xi_n \\ \text{s.t.} \quad & \left(\phi_{c^{(n)}}(\mathbf{x}^{(n)}) - \phi_c(\mathbf{x}^{(n)})\right)^T \mathbf{w} + \xi_n \geq \gamma \quad \forall n, c \neq c^{(n)} \\ & \log \sum_{j'} \exp\left(w^i_{j'|\mathbf{h}}\right) \leq 0 \quad \forall\ 0 \leq i \leq L, \forall\ \mathbf{h} \in \mathbf{val}(\mathbf{Pa}(X_i)) \\ & \xi_n \geq 0 \quad \forall n \end{aligned} \tag{4.10}$$

Our first interpretation of (4.10) is that of a special instance of an BN-SVM, with (exotic) weighted ℓ^1-norm term $\mathrm{L}_{\mathbf{m}}(\mathbf{w})$ and an arbitrary (but not limiting) sub-normalization constraint on the solution vector. On the other hand, $\mathrm{L}_{\mathbf{m}}(\mathbf{w}) = -\mathbf{m}^T\mathbf{w}$ is formally the *negative log-likelihood* of $\mathbf{w}$. Therefore, although (4.10) 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 $\mathbf{w}$ might be *strictly* sub-normalized. In this case, $\mathbf{w}$ does not represent valid BN parameters, and strictly speaking, $\mathrm{L}_{\mathbf{m}}(\mathbf{w})$ 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 3, for C-structures the log-likelihood can *only increase* when Algorithm 1 is applied to sub-normalized parameters. The proofs for Lemma 3 and Theorems 2 and 3 can be found in Appendices 4.B, 4.C and 4.D, respectively.

Lemma 3. *Let $\mathcal{G}$ be a C-structure, $\tilde{\mathbf{w}}$ be a sub-normalized parameter-vector for $\mathcal{G}$, and $\mathbf{m}$ be a nonnegative vector of consistent likelihood-counts. Then the likelihood is non-decreasing under Algorithm 1, i.e. when $\mathbf{w}$ is the output of Algorithm 1 for input $\mathcal{G}$, $\tilde{\mathbf{w}}$, then $\mathrm{L}_{\mathbf{m}}(\mathbf{w}) \leq \mathrm{L}_{\mathbf{m}}(\tilde{\mathbf{w}})$.*

Using Lemma 3, it is easy to show that (4.10) always has a normalized solution, as stated in Theorem 2.

Theorem 2. *Let $\mathcal{G}$ be a C-structure, $((c^{(1)},\mathbf{x}^{(1)}),\ldots,(c^{(N)},\mathbf{x}^{(N)}))$ be an arbitrary data set, and $\mathbf{m}$ be an element-wise nonnegative vector of consistent likelihood-counts. Then problem* (4.10) *(for $\lambda \geq 0$) always has an optimal solution $\mathbf{w}$, $\boldsymbol{\xi}$, such that $\mathbf{w}$ is normalized.*

Furthermore, for positive likelihood-counts, e.g. when $\alpha > 0$ in (4.3), the solution is *unique* and *normalized*.

Theorem 3. *Assume the same conditions as in Theorem 2, but where $\mathbf{m}$ is element-wise positive. Then problem* (4.10) *has a unique, normalized solution.*

Lemma 3 and Theorems 2 and 3 show that for C-structures, we can *always interpret* $\mathrm{L}_{\mathbf{m}}(\mathbf{w})$ as negative log-likelihood. Due to this generative interpretation, we call formulation (4.10) the *maximum-likelihood* BN-SVM (ML-BN-SVM). Problem (4.10) 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.

4.4 Optimization for Large-scale Data

The main limitation in (4.10) is that we have $N(|\mathbf{val}(C)| - 1)$ linear constraints, which restricts application currently to some thousand samples. Therefore, we slightly modify the

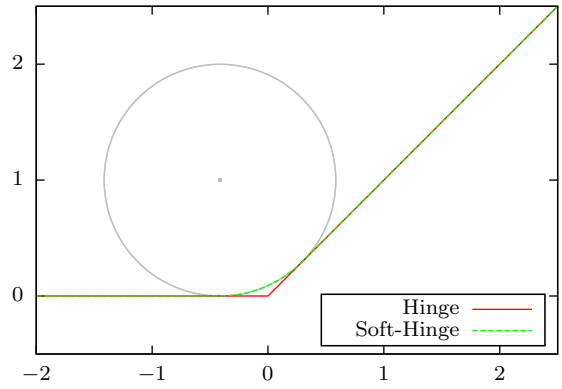


Figure 4.1: Construction of the soft-hinge by fitting a circle segment (here with radius $R = 1$) at the discontinuity of the (hard) hinge function.

problem and propose a scalable gradient-based optimization method. By expressing the slacks as

$$\xi_n = \max\left(\max_{c \neq c^{(n)}}\left[\gamma - (\phi_{c^{(n)}}(\mathbf{x}^{(n)}) - \phi_c(\mathbf{x}^{(n)}))^T \mathbf{w}\right], 0\right), \tag{4.11}$$

we can eliminate these constraints, or in other words, they are absorbed into the objective.

Since the *hinge function* $\max(\cdot, 0)$ and the $\max_{c \neq c^n}[\cdots]$ are not differentiable, we replace them by smooth approximations. The soft-hinge $h_R(\cdot)$ used is defined as

$$h_R(\zeta) = \begin{cases} 0 & \zeta < \mu, \\ \zeta & \zeta > \mu + \frac{R}{\sqrt{2}}, \text{ and} \\ R - \sqrt{R^2 - (\zeta - \mu)^2} & \text{o.w.}, \end{cases} \tag{4.12}$$

where R is the radius of a fitted circle-segment, smoothing the discontinuity of the hinge, and $\mu = R(1 - \sqrt{2})$. The soft-hinge is illustrated in Figure 4.1. In our experiments we set $R = \min(1, \gamma)$. The derivative of $h_R(\cdot)$ is given as

$$\frac{\partial h_R(\zeta)}{\partial \zeta} = \begin{cases} 0 & \zeta < \mu, \\ 1 & \zeta > \mu + \frac{R}{\sqrt{2}}, \text{ and} \\ \frac{\zeta - \mu}{\sqrt{R^2 - (\zeta - \mu)^2}} & \text{o.w.} \end{cases} \tag{4.13}$$

The max function is approximated using the soft-max function

$$\operatorname*{smax}_{\zeta_1, \ldots, \zeta_K} = \frac{1}{\eta} \log \sum_{i=1}^{K} \exp(\eta\, \zeta_i). \tag{4.14}$$

Here η is an approximation parameter, where for $\eta \to \infty$ the soft-max converges to the (hard) max. In our experiments we set $\eta = 10$. The derivative of the soft-max is given as

$$\frac{\partial \operatorname*{smax}_{\zeta_1, \ldots, \zeta_K}}{\partial \zeta_i} = \frac{\exp(\eta \zeta_i)}{\sum_{l=1}^{K} \exp(\eta \zeta_l)}. \tag{4.15}$$

Now, the slack variables are (approximately) expressed as

$$\xi_n = h_R\left(\operatorname*{smax}_{c \neq c^{(n)}}\left[\gamma - (\phi_{c^{(n)}}(\mathbf{x}^{(n)}) - \phi_c(\mathbf{x}^{(n)}))^T \mathbf{w}\right]\right). \tag{4.16}$$

To this end, we obtain the following modified convex problem

$$\begin{aligned} \underset{\mathbf{w}}{\text{minimize}} \quad & -\mathbf{m}^T\mathbf{w} + \lambda \sum_{n=1}^{N} h_R\left(\underset{c\neq c^{(n)}}{\text{smax}}\left[\gamma - (\phi_{c^{(n)}}(\mathbf{x}^{(n)}) - \phi_c(\mathbf{x}^{(n)}))^T\mathbf{w}\right]\right) \\ \text{s.t.} \quad & \log\sum_{j'}\exp\left(w^i_{j'|\mathbf{h}}\right) \leq 0 \quad \forall\ 0\leq i\leq L, \forall\ \mathbf{h}\in\mathbf{val}(\mathbf{Pa}(X_i)) \end{aligned} \tag{4.17}$$

The objective

$$O(\mathbf{w}) = -\mathbf{m}^T\mathbf{w} + \lambda \sum_{n=1}^{N} h_R\left(\underset{c\neq c^{(n)}}{\text{smax}}\left[\gamma - (\phi_{c^{(n)}}(\mathbf{x}^{(n)}) - \phi_c(\mathbf{x}^{(n)}))^T\mathbf{w}\right]\right) \tag{4.18}$$

is continuously differentiable, where the derivative is given as

$$\frac{\partial O(\mathbf{w})}{\partial w^i_{j|\mathbf{h}}} = -m^i_{j|\mathbf{h}} - \lambda \sum_{n}^{N} \frac{\partial h_R}{\partial \text{smax}} \cdot \sum_{c\neq c^{(n)}} \frac{\partial \text{smax}}{\partial \xi^n_c} \cdot (\nu^{i,n}_{j|\mathbf{h}} - \nu^{i,n,c}_{j|\mathbf{h}}), \tag{4.19}$$

where

$$\xi^{(n)}_c = \gamma - (\phi_{c^{(n)}}(\mathbf{x}^{(n)}) - \phi_c(\mathbf{x}^{(n)}))\,\mathbf{w}, \text{ and} \tag{4.20}$$

$$\nu^{i,n,c}_{j|\mathbf{h}} = \mathbf{1}_{\left([c,\mathbf{x}^{(n)}](X_i)=j \text{ and } [c,\mathbf{x}^{(n)}](\mathbf{Pa}(X_i))=\mathbf{h}\right)}. \tag{4.21}$$

For optimization of the objective, we use a projected gradient descent method, i.e. $\mathbf{w}$ is projected onto the set of sub-normalized vectors after each gradient step. This can be done independently for each CPT, i.e. for each combination of $i \in \{0,\ldots,L\}$ and $\mathbf{h}\in\mathbf{val}(\mathbf{Pa}(X_i))$. Projecting an arbitrary vector $\boldsymbol{\zeta}^* = (\zeta^*_1,\ldots,\zeta^*_K)^T$ onto the set of sub-normalized vectors is formulated as

$$\begin{aligned} \underset{\boldsymbol{\zeta}}{\text{minimize}} \quad & \|\boldsymbol{\zeta} - \boldsymbol{\zeta}^*\|_2 \\ \text{s.t.} \quad & \log\sum_{l=1}^{K}\exp(\zeta_l) \leq 0. \end{aligned} \tag{4.22}$$

This problem has no closed-form solution, but can be addressed by the iterative algorithm proposed in [18]. 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 the need 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 $(|\mathbf{val}(C)|-1)NL$. It is also straightforward to implement parallel and stochastic versions of this method. Further details and the pseudo code of the projection algorithm can be found in Appendix 4.E. Furthermore, an alternative algorithm to the iterative algorithm is proposed in Appendix 4.G.

4.5 Experiments

In this section we present experiments for illustrative purposes (Sections 4.5.1 and 4.5.2) and a comparison on real-world datasets (Section 4.5.3). We considered 30 datasets from the UCI repository [1], TIMIT [21] and USPS data [13]. Datasets containing more than 5000 samples were split into training and test set; Otherwise 5-fold cross-validation was used for testing. Dataset details are available in Chapter 2.3. For discretizing continuous attributes, we used the algorithm described in [9]. The smoothing parameter α in (4.3) was constantly set to 1. Although α can have a great impact on classification [10, 24], its evaluation is out of the scope of this thesis.

4.5.1 Generative-Discriminative Trade-off

The parameter λ in problem (4.10) 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 4.2. With increasing λ, the negative log-likelihood increases (i.e. the log-likelihood decreases), while the sum of slacks decreases. Qualitatively, the classification rate increases correspondingly. Similar behavior can be observed on other datasets.

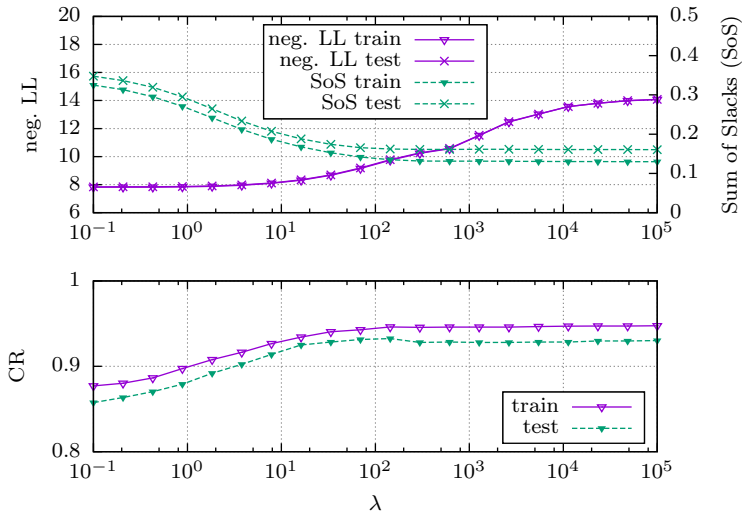


Figure 4.2: Influence of parameter λ using the *car* dataset. (top) Negative log-likelihood (neg. LL) and sum of slacks (SoS), normalized by N. (bottom) Classification rate (CR).

4.5.2 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. Classification results are shown in Figure 4.3, 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.

4.5.3 Benchmark Classification Results

We compared ML-BN-SVMs with ML, maximum conditional likelihood (MCL) and maximum margin (MM) parameters using the algorithm proposed in [21]. In order to enable a

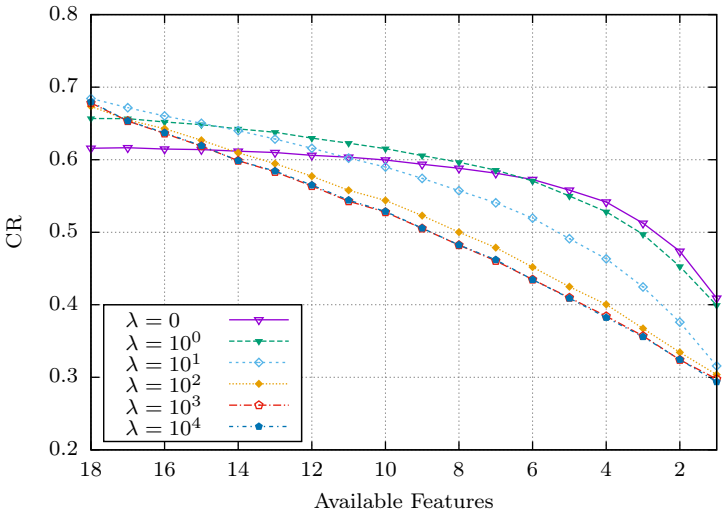


Figure 4.3: Classification rates (CR) for the *vehicle* dataset for varying numbers of missing features and varying λ, using a NB structure.

fair comparison, MM was executed without early stopping. Experiments *with* early stopping are provided in Appendix 4.F. Furthermore, we compared with linear SVMs and SVMs equipped with Gaussian kernels [5]. For ML-BN-SVMs we validated the ranges $\gamma \in \{0.1, 1, 5, 10, 15, 20\}$, and $\lambda \in \{0, 2^{-2}, 2^{-1}, \ldots 2^{10}\}$. 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 [21] for details). For SVMs we validated the trade-off parameter $\lambda \in \{2^{-2}, 2^{-1}, \ldots 2^{10}\}$ and, for kernelized SVMs, the kernel width $\sigma \in \{2^{-5}, 2^{-4}, \ldots, 2^{5}\}$. For the classifiers based on BNs, we used NB and maximum-likelihood TAN structures [10], i.e. TAN-CMI. Classification results for the compared methods are shown in Tables 4.1 and 4.2 for TAN and NB structures, respectively. 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 4.4 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 red line are possible. We observe that the scatter plot for ML-BN-SVM is clearly more concentrated in the vicinity of the red 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 4.5. As ex-

Table 4.1: Detailed classification rates with 95% confidence intervals for BN parameters, using **TAN** structures. ML: maximum likelihood, MCL: maximum condition likelihood, MM: maximum margin BN parameters [21], ML-BN-SVM: proposed method, Linear SVM: support vector machine without kernel, SVM: support vector machine with Gauss kernel.

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
corral	97.53 ± 4.61	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	93.36 ± 4.55	100.00 ± 0.00
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
glass2	81.38 ± 9.20	82.00 ± 8.05	80.75 ± 10.51	80.75 ± 10.51	79.38 ± 4.27	79.96 ± 8.90
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
iris	94.00 ± 1.85	94.00 ± 1.85	92.67 ± 4.53	94.00 ± 1.85	93.33 ± 2.93	93.33 ± 2.93
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
mofn-3-7-10	92.62 ± 1.37	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00
mushroom	100.00 ± 0.07	100.00 ± 0.07	100.00 ± 0.07	100.00 ± 0.07	100.00 ± 0.07	99.82 ± 0.19
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
TIMIT4CF	90.70 ± 0.42	87.25 ± 0.48	91.70 ± 0.40	91.59 ± 0.40	92.05 ± 0.39	92.38 ± 0.39
TIMIT4CM	90.47 ± 0.43	88.57 ± 0.46	85.62 ± 0.51	92.58 ± 0.38	92.88 ± 0.38	93.16 ± 0.37
TIMIT6CF	83.18 ± 0.52	80.92 ± 0.54	84.27 ± 0.50	84.89 ± 0.49	85.57 ± 0.48	85.74 ± 0.48
TIMIT6CM	83.05 ± 0.52	80.98 ± 0.54	85.45 ± 0.49	85.91 ± 0.48	86.66 ± 0.47	86.56 ± 0.47
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

Table 4.2: Detailed classification rates with 95% confidence intervals for BN parameters, using **NB** structures. ML: maximum likelihood, MCL: maximum condition likelihood, MM: maximum margin BN parameters [21], ML-BN-SVM: proposed method, Linear SVM: support vector machine without kernel, SVM: support vector machine with Gauss kernel.

dataset	ML	MCL	MM	ML-BN-SVM	Linear SVM	SVM
abalone	53.64 ± 1.45	59.12 ± 1.71	56.62 ± 0.88	59.12 ± 1.69	58.42 ± 1.77	59.29 ± 1.40
adult	83.37 ± 0.71	86.90 ± 0.64	86.92 ± 0.64	86.94 ± 0.64	86.86 ± 0.64	86.87 ± 0.64
australian	85.92 ± 2.92	84.02 ± 2.76	85.34 ± 2.64	87.24 ± 2.86	85.78 ± 1.69	86.80 ± 2.34
breast	97.63 ± 1.01	95.56 ± 1.45	95.85 ± 2.22	97.04 ± 1.45	96.15 ± 1.51	97.19 ± 0.41
car	85.64 ± 1.59	93.43 ± 1.76	93.78 ± 1.63	92.73 ± 1.14	93.84 ± 0.65	99.65 ± 0.30
chess	87.45 ± 2.57	97.11 ± 1.02	97.58 ± 0.86	97.68 ± 1.21	97.02 ± 0.82	99.50 ± 0.25
cleve	82.87 ± 6.79	82.52 ± 6.36	82.17 ± 6.94	82.53 ± 7.64	83.57 ± 5.29	82.19 ± 6.37
corral	89.16 ± 8.67	93.36 ± 4.55	93.36 ± 4.55	93.36 ± 4.55	93.36 ± 4.55	100.00 ± 0.00
crx	86.84 ± 3.29	85.13 ± 4.10	84.82 ± 3.71	86.06 ± 3.54	85.75 ± 3.20	85.75 ± 2.65
diabetes	73.96 ± 4.17	75.40 ± 5.41	74.61 ± 4.94	74.87 ± 3.47	73.96 ± 4.46	74.48 ± 4.65
flare	76.58 ± 1.04	83.40 ± 1.02	82.63 ± 1.79	83.11 ± 0.82	84.45 ± 0.28	84.45 ± 0.28
german	74.20 ± 3.58	75.10 ± 1.42	76.50 ± 1.52	75.30 ± 3.12	76.10 ± 1.11	75.80 ± 2.80
glass	71.66 ± 3.58	68.05 ± 0.63	68.03 ± 1.91	70.61 ± 3.63	71.61 ± 5.50	73.24 ± 5.33
glass2	81.29 ± 10.50	82.63 ± 8.12	80.09 ± 9.96	82.63 ± 8.12	79.38 ± 4.27	79.96 ± 8.90
heart	81.85 ± 9.40	82.59 ± 5.77	81.85 ± 5.73	83.33 ± 5.14	84.81 ± 4.11	81.85 ± 9.40
hepatitis	88.58 ± 6.57	86.08 ± 3.38	84.92 ± 8.69	92.33 ± 6.75	87.42 ± 10.89	88.67 ± 6.37
iris	93.33 ± 2.93	92.67 ± 3.46	93.33 ± 2.93	93.33 ± 2.93	93.33 ± 2.93	93.33 ± 2.93
letter	74.95 ± 1.05	85.97 ± 0.84	82.53 ± 0.92	85.79 ± 0.85	90.07 ± 0.73	94.07 ± 0.58
lymphography	84.23 ± 5.60	84.23 ± 4.47	82.80 ± 5.54	82.80 ± 4.39	83.57 ± 10.44	86.48 ± 9.99
mofn-3-7-10	87.31 ± 1.94	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00
mushroom	98.04 ± 0.54	100.00 ± 0.07	100.00 ± 0.07	99.78 ± 0.20	100.00 ± 0.07	99.82 ± 0.19
nursery	89.97 ± 0.91	92.38 ± 0.80	92.98 ± 0.77	93.03 ± 0.77	93.31 ± 0.76	100.00 ± 0.04
satimage	81.56 ± 1.80	87.29 ± 1.11	88.82 ± 1.26	88.41 ± 1.33	88.36 ± 1.58	90.59 ± 1.59
segment	92.68 ± 1.78	94.29 ± 0.77	94.98 ± 1.66	95.37 ± 0.86	96.19 ± 0.73	96.84 ± 1.17
shuttle	99.62 ± 0.09	99.91 ± 0.05	99.94 ± 0.04	99.95 ± 0.04	99.96 ± 0.03	99.96 ± 0.03
soybean-large	93.35 ± 1.91	92.98 ± 3.88	92.79 ± 1.59	91.50 ± 3.81	91.15 ± 3.72	93.54 ± 1.19
spambase	90.03 ± 1.11	93.73 ± 0.95	94.01 ± 0.97	94.08 ± 0.75	94.27 ± 0.72	95.04 ± 0.37
TIMIT4CF	87.88 ± 0.47	92.04 ± 0.39	91.90 ± 0.40	91.95 ± 0.39	92.05 ± 0.39	92.38 ± 0.39
TIMIT4CM	88.86 ± 0.46	93.04 ± 0.37	92.88 ± 0.38	92.71 ± 0.38	92.88 ± 0.38	93.16 ± 0.37
TIMIT6CF	82.20 ± 0.53	85.50 ± 0.49	85.20 ± 0.49	85.49 ± 0.49	85.57 ± 0.48	85.74 ± 0.48
TIMIT6CM	82.43 ± 0.53	86.24 ± 0.48	86.04 ± 0.48	86.50 ± 0.47	86.66 ± 0.47	86.56 ± 0.47
USPS	86.89 ± 1.11	94.37 ± 0.76	95.44 ± 0.69	95.08 ± 0.71	95.82 ± 0.66	91.80 ± 0.90
vehicle	61.57 ± 1.44	68.67 ± 3.03	69.76 ± 2.56	67.95 ± 6.00	70.12 ± 1.26	69.76 ± 2.43
vote	90.16 ± 4.70	94.61 ± 2.21	95.78 ± 2.21	94.61 ± 3.19	94.85 ± 2.20	95.54 ± 3.18
waveform-21	81.14 ± 1.05	85.10 ± 1.53	85.43 ± 1.34	85.14 ± 1.52	84.78 ± 1.77	85.16 ± 1.29

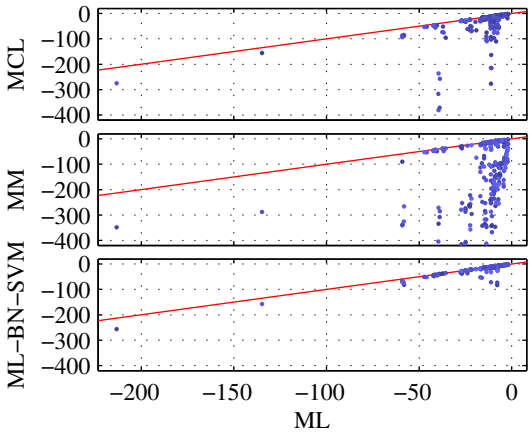


Figure 4.4: Likelihood scatter plot over all data sets. The train likelihoods (normalized by the sample size) of ML parameters are plotted against the train likelihoods of MCL (top), MM (center), and ML-BN-SVM (bottom).

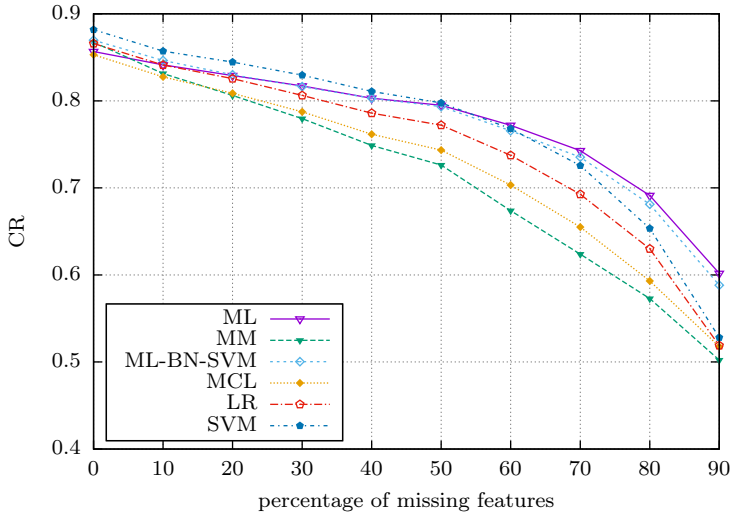


Figure 4.5: Classification results, averaged over UCI datasets, with varying percentage of missing features.

pected, 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.

To summarize our results, in Table 4.3, we provide pair-wise comparisons of all methods conducted on the UCI datasets: Plain numbers denote the number of times where the algorithm in the row outperforms the algorithm in the column at a significance level of 68%. Bold face numbers denote a significance level of 95%. When using 5-fold cross-validation for testing, we used a one-sided t-test, otherwise we used a one-sided binomial test for testing significance. Tables 4.4 and 4.5 show the corresponding results, when 50% and 90% percent of features are missing in the test data, respectively. Similar as before,

these results demonstrate the robustness against missing features of ML and ML-BN-SVM parameters.

Table 4.3: Number of times classifier in row outperforms classifier in column with significance 68 % (plain) and 95 % (bold), when no features are missing.

	ML		MCL		MM		ML-BN-SVM		SVM	
	NB	TAN	NB	TAN	NB	TAN	NB	TAN	Linear	Gauss
ML NB	–	9/**5**	8/4	11/7	9/4	11/4	7/2	9/**5**	5/**1**	5/**0**
ML TAN	20/**18**	–	8/3	14/**8**	8/3	8/**1**	7/3	4/**1**	6/**1**	4/**1**
MCL NB	21/**18**	20/**10**	–	19/**11**	10/2	13/**6**	8/**1**	13/2	5/**1**	5/2
MCL TAN	17/**14**	8/**6**	7/**5**	–	7/4	8/**0**	7/**5**	1/**0**	6/4	3/**1**
MM NB	20/**18**	15/**11**	14/**8**	17/**11**	–	15/**8**	9/3	12/4	8/2	4/2
MM TAN	18/**18**	18/**12**	12/7	19/**12**	10/**6**	–	10/**8**	8/3	8/4	3/2
ML-BN-SVM NB	24/**19**	21/**11**	15/**9**	21/**14**	14/7	20/**8**	–	15/4	9/3	7/**1**
ML-BN-SVM TAN	19/**18**	21/**15**	13/**8**	21/**16**	12/**8**	15/3	12/**6**	–	10/4	3/2
LinSVM	21/**18**	22/**14**	19/7	21/**14**	16/**6**	15/7	15/7	15/**8**	–	6/2
SVM	23/**18**	26/**18**	20/**14**	25/**18**	18/**12**	25/**13**	17/**10**	21/**11**	17/**9**	–

Table 4.4: Number of times classifier in row outperforms classifier in column with significance 68 % (plain) and 95 % (bold), with 50% missing features.

	ML		MCL		MM		ML-BN-SVM		SVM	
	NB	TAN	NB	TAN	NB	TAN	NB	TAN	Linear	Gauss
ML NB	–	8/2	23/**19**	20/**13**	25/**18**	25/**15**	14/7	8/**5**	11/**5**	12/**3**
ML TAN	20/**13**	–	24/**20**	25/**16**	26/**21**	28/**17**	18/**13**	9/**3**	13/**4**	13/**3**
MCL NB	6/2	2/**0**	–	13/**5**	11/7	16/**9**	2/**0**	2/2	4/**3**	4/2
MCL TAN	10/7	4/**1**	15/**9**	–	15/**13**	19/**8**	9/**6**	5/2	5/**3**	5/2
MM NB	5/**3**	5/2	14/**11**	12/7	–	16/**10**	4/4	3/**1**	2/**1**	3/2
MM TAN	5/4	2/**1**	11/**9**	11/**6**	10/**6**	–	6/4	4/2	1/**0**	1/**0**
ML-BN-SVM NB	12/**6**	7/**1**	25/**19**	18/**12**	23/**15**	22/**13**	–	7/2	11/4	11/**6**
ML-BN-SVM TAN	18/**11**	13/**3**	25/**19**	24/**20**	26/22	27/**17**	18/**11**	–	10/4	10/**6**
LinSVM	17/**11**	12/**6**	26/22	25/**18**	26/**23**	27/**17**	18/**11**	14/**7**	–	10/**3**
SVM	16/**12**	13/**9**	25/22	24/**19**	26/**20**	25/**17**	17/**12**	14/**9**	15/**7**	–

Table 4.5: Number of times classifier in row outperforms classifier in column with significance 68 % (plain) and 95 % (bold), with 90% missing features.

	ML		MCL		MM		ML-BN-SVM		SVM	
	NB	TAN	NB	TAN	NB	TAN	NB	TAN	Linear	Gauss
ML NB	–	3/**0**	22/**16**	20/**15**	26/**18**	24/**17**	18/**10**	16/4	23/**12**	24/**13**
ML TAN	8/4	–	22/**16**	20/**14**	26/**18**	24/**17**	19/**11**	18/4	24/**13**	25/**14**
MCL NB	0/**0**	0/**0**	–	8/4	14/7	13/7	6/2	5/**1**	7/4	8/4
MCL TAN	3/**1**	2/**1**	15/**10**	–	13/**8**	15/**8**	7/**5**	2/**1**	13/**5**	11/**6**
MM NB	0/**0**	0/**0**	11/**8**	10/**5**	–	15/**8**	7/3	6/2	7/**5**	8/4
MM TAN	0/**0**	0/**0**	9/**5**	8/**6**	9/**3**	–	4/**3**	3/**1**	9/**6**	10/**6**
ML-BN-SVM NB	1/**0**	2/**0**	18/**11**	14/**7**	19/**9**	20/**13**	–	6/2	16/**6**	14/**7**
ML-BN-SVM TAN	5/**3**	3/**1**	19/**14**	20/**11**	22/**14**	20/**14**	17/**10**	–	23/**11**	23/**11**
LinSVM	2/2	1/**1**	17/**10**	13/**8**	17/**10**	19/**9**	7/**3**	5/**1**	–	7/4
SVM	3/2	2/**1**	15/**11**	14/**7**	17/**9**	19/**9**	8/**3**	6/**1**	11/4	–

4.6 Conclusion

A BN distribution is a log-linear model, enabling SVM-type training for BNs [12, 21], which we call BN-SVM. For a large class of network structures [27], one can always obtain correctly normalized parameters, i.e. a formally valid BN. In this chapter, 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* — a similar extension for MM BNs has been investigated in Tschiatschek et al. [25]. In the BN framework, this naturally includes *learning with missing features* and *semi-supervised learning*.

Appendix

4.A Parameter Renormalization

The parameters of a BNC $\mathcal{B}$ with C-structure [27], cf. Condition 2, can be normalized without changing the class-conditional distribution $\mathrm{P}^{\mathcal{B}}(C|X)$ using Algorithm 1. Roughly speaking, normalization can be achieved as follows: First, the nodes of the BNC are topologically ordered. Second, the conditional probabilities of the nodes can be sequentially normalized in a bottom up manner starting with the last node in the topological ordering. Multiplicative factors required for normalization are handed to the parent nodes. This does not affect the normalization of previous nodes.

Algorithm 1 Renormalization [27]

Require: $\mathcal{G}$, unnormalized parameters $\tilde{\mathbf{w}}$
Ensure: Normalized parameters $\mathbf{w}$, with $\mathrm{P}^{\mathcal{B}}(C|\mathbf{X};\mathbf{w}) = \mathrm{P}^{\mathcal{B}}(C|\mathbf{X};\tilde{\mathbf{w}})$

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

4.B Proof of Lemma 3

Proof of Lemma 3. First note that in Algorithm 1, $\mathbf{w}$ always remains sub-normalized: If $\mathbf{w}$ 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, $\mathbf{w}$ remains sub-normalized and $\beta \le 0$.

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

$$\mathrm{L}_{\mathbf{m}}(\mathbf{w}'') - \mathrm{L}_{\mathbf{m}}(\mathbf{w}') = -\mathbf{m}^T\mathbf{w}'' + \mathbf{m}^T\mathbf{w}' \tag{4.23}$$

$$= \beta \sum_j m^{\pi_i}_{j|\mathbf{h}} \tag{4.24}$$

$$\le 0. \tag{4.25}$$

Therefore, when X_{π_i} is *not* a class-child, $\mathrm{L}_{\mathbf{m}}(\mathbf{w}'') \leq \mathrm{L}_{\mathbf{m}}(\mathbf{w}')$. When X_{π_i} *is* a class-child, we additionally have in step 12 that

$$w''^{k_i}_{\mathbf{h}(X_{k_i})|[\mathbf{h}(\mathbf{I}),\mathbf{a}]} = w'^{k_i}_{\mathbf{h}(X_{k_i})|[\mathbf{h}(\mathbf{I}),\mathbf{a}]} + \beta \quad \forall \mathbf{a} \in \mathbf{val}(\mathbf{A}), \tag{4.26}$$

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 C-structure, it holds that $\mathbf{Pa}(X_{\pi_i}) \setminus \left(\mathbf{Pa}(X_{k_i}) \cup \{X_{k_i}\}\right) = \emptyset$. Therefore, since $\mathbf{n}$ are consistent likelihood-counts, cf. (4.6), we have that $\sum_{\mathbf{a}\in\mathbf{val}(\mathbf{A})} n^{k_i}_{\mathbf{h}(X_{k_i})|[\mathbf{h}(\mathbf{I}),\mathbf{a}]} = \sum_{j'\in\mathbf{val}(X_{\pi_i})} m^{\pi_i}_{j'|\mathbf{h}}$, and thus

$$\begin{aligned}
\mathrm{L}_{\mathbf{m}}(\mathbf{w}'') - \mathrm{L}_{\mathbf{m}}(\mathbf{w}') &= -\mathbf{m}^T \mathbf{w}'' + \mathbf{m}^T \mathbf{w}' && (4.27)\\
&= \beta \sum_{j'} m^{\pi_i}_{j'|\mathbf{h}} - \beta \sum_{\mathbf{a}} m^{k_i}_{\mathbf{h}(X_{k_i})|[\mathbf{h}(\mathbf{I}),\mathbf{a}]} && (4.28)\\
&= 0. && (4.29)
\end{aligned}$$

We see that $\mathrm{L}_{\mathbf{m}}(\mathbf{w}'') \leq \mathrm{L}_{\mathbf{m}}(\mathbf{w}')$, and by induction $\mathrm{L}_{\mathbf{m}}(\mathbf{w}) \leq \mathrm{L}_{\mathbf{m}}(\tilde{\mathbf{w}})$. □

4.C Proof of Theorem 2

Proof of Theorem 2. Let $\mathbf{w}^*, \boldsymbol{\xi}^*$ be an optimal solution of (4.10). When we apply Algorithm 1 to $\mathbf{w}^*$, obtaining the normalized $\mathbf{w}$ as output, we see that $\mathbf{w}$, $\boldsymbol{\xi}$, with $\boldsymbol{\xi} = \boldsymbol{\xi}^*$, is feasible, since the class-conditional distribution is invariant under Algorithm 1. Furthermore, since $\mathbf{w}^*$ is sub-normalized, we have by Lemma 3 that $\mathrm{L}_{\mathbf{m}}(\mathbf{w}) \leq \mathrm{L}_{\mathbf{m}}(\mathbf{w}^*)$. Therefore,

$$\mathrm{L}_{\mathbf{m}}(\mathbf{w}) + C \sum_n \xi_n \leq \mathrm{L}_{\mathbf{m}}(\mathbf{w}^*) + C \sum_n \xi^*_n, \tag{4.30}$$

which implies that $\mathbf{w}$, $\boldsymbol{\xi}$ is optimal. □

4.D Proof of Theorem 3

Proof of Theorem 3. We first prove by contradiction, that under the conditions of Theorem 3, all solutions are normalized. Assume that $\mathbf{w}^*, \boldsymbol{\xi}^*$ are optimal for (4.10), where for some X_{π_i} and $\mathbf{h} \in \mathbf{val}(\mathbf{Pa}(X_{\pi_i}))$, the corresponding CPT in $\mathbf{w}^*$ is *strictly sub-normalized*. Let $\mathbf{w}$ be the output of Algorithm 1 for input $\mathcal{G}$, $\mathbf{w}^*$. Let $\mathbf{w}'$ be the vector before the strictly sub-normalized CPT is processed, and $\mathbf{w}''$ 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.

$$\mathrm{L}_{\mathbf{m}}(\mathbf{w}'') < \mathrm{L}_{\mathbf{m}}(\mathbf{w}').$$

Since the negative log-likelihood is never increased afterwards, $\mathbf{w}^*, \boldsymbol{\xi}^*$ can not be optimal.
When X_{π_i} *is* a class child, this decrease of negative log-likelihood is compensated in step 12 (cf. (4.27)). 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 $\mathbf{w}^*, \boldsymbol{\xi}^*$ are optimal.
Now we show that the solution is unique. Assume two optimal solutions $\mathbf{w}^*$, $\boldsymbol{\xi}^*$ and $\mathbf{w}^{*\prime}$, $\boldsymbol{\xi}^{*\prime}$, $\mathbf{w}^* \neq \mathbf{w}^{*\prime}$. Since (4.10) is a convex problem, the convex combination $\mathbf{w} = 0.5\,\mathbf{w}^* + 0.5\,\mathbf{w}^{*\prime}$, $\boldsymbol{\xi} = 0.5\,\boldsymbol{\xi}^* + 0.5\,\boldsymbol{\xi}^{*\prime}$ is also optimal. Since all solutions are normalized,

$$\log \sum_j \exp(w^{*i}_{j|\mathbf{h}}) = \log \sum_j \exp(w^{*\prime i}_{j|\mathbf{h}}) = 0 \quad \forall i, \mathbf{h}. \tag{4.31}$$

However, since $\log\sum\exp$ is a *strictly* convex function, $\mathbf{w}$ is *strictly* sub-normalized, which contradicts that $\mathbf{w}$, $\boldsymbol{\xi}$ is optimal. □

4.E Projection for Projected Gradient Descent

The gradient (4.19) is used in conjugate gradient descent, where $\mathbf{w}$ is projected onto the set of sub-normalized vectors after each gradient step. This can be done for each CPT individually. For projecting, we use a variant of the algorithm described in [18], which projects an arbitrary vector onto the intersection of strictly convex sets. Here, we have the set $\mathcal{M} = \{\boldsymbol{\zeta} \mid \log \sum_l \exp(\zeta_l) \leq 0\}$, which is only a *single* strictly convex set. The algorithm is depicted in Algorithm 2, where $\boldsymbol{\zeta}^*$ is some arbitrary input vector, i.e. some CPT which has to be projected onto $\mathcal{M}$. The solution vector $\boldsymbol{\zeta}$ is initialized with some arbitrary vector $\boldsymbol{\zeta}_0$, with $\log \sum_l \exp(\zeta_{0,l}) = 0$. Vector $\mathbf{g}$ is the normalized gradient vector of the $\log \sum \exp(\cdot)$ function at the current solution vector $\boldsymbol{\zeta}$, which is the normal vector of $\mathcal{M}$. Vector $\mathbf{d}$ is the normalized residual vector. As easily shown via the KKT conditions, $\boldsymbol{\zeta}$ is optimal when $\mathbf{g} \propto \mathbf{d}$, as checked in step 10. Following [18], in each iteration, $\mathcal{M}$ is locally approximated with a ball of radius ρ and center $\boldsymbol{\mu}$, and the projection $\bar{\boldsymbol{\zeta}}$ onto this ball is calculated. In our experiments we used a radius $\rho = 1$. When $\bar{\boldsymbol{\zeta}}$ is feasible (steps 14-15), this solution is improved by finding the point closest to $\boldsymbol{\zeta}^*$ on the line segment $[\bar{\boldsymbol{\zeta}}, \boldsymbol{\zeta}^*]$. When $\bar{\boldsymbol{\zeta}}$ is infeasible (steps 17-18), a feasibility restoration is performed as depicted in [18]. In both cases, the Newton-Raphson method is used to find scalar κ.

The projection algorithm interacts nicely with the projected gradient method, since we use the solution of the previous gradient step as initialization $\boldsymbol{\zeta}_0$. Therefore, since in each iteration of Algorithm 2 the distance $\|\boldsymbol{\zeta}^* - \boldsymbol{\zeta}\|$ is reduced (see [18]), we do not need to run the projection algorithm until convergence, but only for some few iterations (in fact, a single iteration is sufficient).

Algorithm 2 Projection onto subnormalized set

Input: $\boldsymbol{\zeta}^*$, $\boldsymbol{\zeta}_0$ with $\log \sum_l \exp(\zeta_{0,l}) = 0$, $\rho > 0$
Output: $\boldsymbol{\zeta} = \arg\min_{\boldsymbol{\zeta}} \|\boldsymbol{\zeta}^* - \boldsymbol{\zeta}\|$, s.t. $\log \sum_l \exp \zeta_l \leq 0$

1: **if** $\log \sum_i \exp(\zeta_i^*) \leq 0$ **then**
2: $\quad \boldsymbol{\zeta} \leftarrow \boldsymbol{\zeta}^*$
3: $\quad$ return
4: **end if**
5: $\boldsymbol{\zeta} \leftarrow \boldsymbol{\zeta}_0$
6: $\mathbf{g} \leftarrow \exp(\boldsymbol{\zeta})$
7: $\mathbf{g} \leftarrow \frac{\mathbf{g}}{\|\mathbf{g}\|_2}$
8: $\mathbf{d} \leftarrow \boldsymbol{\zeta}^* - \boldsymbol{\zeta}$
9: $\mathbf{d} \leftarrow \frac{\mathbf{d}}{\|\mathbf{d}\|_2}$
10: **while** $\mathbf{g}^T \mathbf{d} < 1$ **do**
11: $\quad \boldsymbol{\mu} = \boldsymbol{\zeta} - \rho \mathbf{g}$
12: $\quad \bar{\boldsymbol{\zeta}} = \boldsymbol{\mu} + \rho \mathbf{d}$
13: $\quad$ **if** $\log \sum_l \exp(\bar{\zeta}_l) \leq 0$ **then**
14: $\quad\quad$ find κ: $\log \sum_l \exp\left(\bar{\zeta}_l + \kappa\,(\zeta_l^* - \bar{\zeta}_l)\right) = 0$
15: $\quad\quad \boldsymbol{\zeta} \leftarrow \bar{\boldsymbol{\zeta}} + \kappa\,(\boldsymbol{\zeta}^* - \bar{\boldsymbol{\zeta}})$
16: $\quad$ **else**
17: $\quad\quad$ find κ: $\log \sum_l \exp\left(\bar{\zeta}_l + \kappa\,(\zeta_l - \bar{\zeta}_l)\right) = 0$
18: $\quad\quad \boldsymbol{\zeta} \leftarrow \bar{\boldsymbol{\zeta}} + \kappa\,(\boldsymbol{\zeta} - \bar{\boldsymbol{\zeta}})$
19: $\quad$ **end if**
20: $\quad \mathbf{g} \leftarrow \exp(\boldsymbol{\zeta})$
21: $\quad \mathbf{g} \leftarrow \frac{\mathbf{g}}{\|\mathbf{g}\|_2}$
22: $\quad \mathbf{d} \leftarrow \boldsymbol{\zeta}^* - \boldsymbol{\zeta}$
23: $\quad \mathbf{d} \leftarrow \frac{\mathbf{d}}{\|\mathbf{d}\|_2}$
24: **end while**

4.F Effect of Early Stopping

In the main part, we compared our method with state-of-the art maximum margin (MM) training for BNs [21]. In [21], MM training was proposed with early stopping. This makes it hard to assess, to which part the classification performance stems from the problem formulation, and to which part from the early stopping heuristic. Therefore, in the main part, we performed all experiments *without* early stopping. However, early stopping is easy to use, and an effective method to improve classification results. Here we show results for MM and ML-BN-SVM training when using early stopping; for both methods we performed gradient descent until convergence, but maximally for 10000 iterations, recording the performance on the validations set and storing maximizing parameter vectors. Finally, we used those parameters achieving the highest performance over all iterations and hyperparameters (γ and λ in our method, λ and κ for MM, see [21]). Table 4.6 compares results with and without early stopping. We see that for NB, the ML-BN-SVM performs in 25 cases better than MM, while MM performs better in 9 cases. For TAN, the ML-BN-SVM performs in 22 cases better than MM, while MM performs better in 12 cases. We see that also in the case of early stopping the ML-BN-SVM performs favorable in comparison to MM. Furthermore, we see that early stopping tends to improve classification results significantly. In cases where methods with early stopping perform worse than the version without early stopping, the degradation is small.

Table 4.6: Classification results for MM [21] and ML-BN-SVM (this chapter), with and without early stopping.

	without early stopping				with early stopping			
	MM		ML-BN-SVM		MM		ML-BN-SVM	
dataset	NB	TAN	NB	TAN	NB	TAN	NB	TAN
abalone	56.62 ± 0.88	57.78 ± 0.96	59.12 ± 1.69	58.69 ± 1.86	58.16 ± 0.96	58.11 ± 1.65	58.88 ± 1.71	58.90 ± 1.49
adult	86.92 ± 0.64	86.54 ± 0.65	86.94 ± 0.64	86.76 ± 0.64	86.89 ± 0.64	86.38 ± 0.65	86.96 ± 0.64	86.47 ± 0.65
australian	85.34 ± 2.64	85.49 ± 3.40	87.24 ± 2.86	84.76 ± 3.78	85.48 ± 3.57	85.04 ± 2.33	86.80 ± 2.75	85.93 ± 1.95
breast	95.85 ± 2.22	96.59 ± 0.50	97.04 ± 1.45	96.00 ± 2.31	97.04 ± 0.65	96.59 ± 1.05	97.04 ± 0.92	96.74 ± 1.67
car	93.78 ± 1.63	97.79 ± 0.79	92.73 ± 1.14	98.08 ± 1.07	93.84 ± 1.68	98.26 ± 0.92	92.97 ± 1.43	97.85 ± 0.83
chess	97.58 ± 0.86	97.43 ± 0.79	97.68 ± 1.21	97.99 ± 0.92	97.21 ± 0.94	97.40 ± 0.62	97.62 ± 1.33	97.93 ± 0.84
cleve	82.17 ± 6.94	79.09 ± 7.56	82.53 ± 7.64	80.79 ± 7.58	81.51 ± 7.16	83.90 ± 4.95	82.53 ± 7.49	83.55 ± 7.08
corral	93.36 ± 4.55	100.00 ± 0.00	93.36 ± 4.55	100.00 ± 0.00	87.73 ± 10.44	100.00 ± 0.00	93.36 ± 4.55	100.00 ± 0.00
crx	84.82 ± 3.71	83.89 ± 5.89	86.06 ± 3.54	84.20 ± 4.56	86.21 ± 3.96	84.81 ± 5.20	86.37 ± 3.26	84.97 ± 3.64
diabetes	74.61 ± 4.94	73.31 ± 5.71	74.87 ± 3.47	74.35 ± 5.42	74.22 ± 4.01	73.96 ± 4.14	73.44 ± 4.14	74.61 ± 5.09
flare	82.63 ± 1.79	84.45 ± 0.28	83.11 ± 0.82	83.30 ± 1.06	81.09 ± 2.92	84.26 ± 0.73	83.88 ± 0.34	84.17 ± 0.57
german	76.50 ± 1.52	73.20 ± 4.01	75.30 ± 3.12	72.60 ± 2.89	74.10 ± 1.42	72.00 ± 2.15	74.60 ± 2.46	74.70 ± 4.09
glass	68.03 ± 1.91	71.71 ± 10.88	70.61 ± 3.63	72.61 ± 6.35	71.61 ± 6.96	71.13 ± 5.18	72.16 ± 4.60	72.13 ± 6.23
glass2	80.09 ± 9.96	80.75 ± 10.51	82.63 ± 8.12	80.75 ± 10.51	83.98 ± 6.92	83.34 ± 6.52	81.29 ± 10.50	84.00 ± 7.38
heart	81.85 ± 5.73	77.41 ± 9.81	83.33 ± 5.14	81.48 ± 9.34	82.96 ± 6.97	80.74 ± 9.97	81.48 ± 8.13	82.22 ± 10.61
hepatitis	84.92 ± 8.69	86.08 ± 3.38	92.33 ± 6.75	86.17 ± 6.31	89.83 ± 8.95	89.92 ± 6.86	96.17 ± 4.35	87.42 ± 7.63
iris	93.33 ± 2.93	92.67 ± 4.53	93.33 ± 2.93	94.00 ± 1.85	93.33 ± 2.93	94.00 ± 1.85	93.33 ± 2.93	94.67 ± 2.27
letter	82.53 ± 0.92	89.58 ± 0.74	85.79 ± 0.85	88.57 ± 0.77	82.40 ± 0.92	89.55 ± 0.74	86.06 ± 0.84	90.25 ± 0.72
lymphography	82.80 ± 5.54	80.66 ± 11.11	82.80 ± 4.39	76.92 ± 10.54	83.52 ± 11.07	82.91 ± 10.65	86.54 ± 10.49	82.14 ± 5.76
mofn-3-7-10	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	99.90 ± 0.27	100.00 ± 0.00	100.00 ± 0.00
mushroom	100.00 ± 0.07	100.00 ± 0.07	99.78 ± 0.20	100.00 ± 0.07	99.56 ± 0.27	100.00 ± 0.07	99.67 ± 0.24	100.00 ± 0.07
nursery	92.98 ± 0.77	98.84 ± 0.33	93.03 ± 0.77	98.68 ± 0.35	92.66 ± 0.79	98.80 ± 0.34	92.92 ± 0.78	98.38 ± 0.39
satimage	88.82 ± 1.26	86.82 ± 2.66	88.41 ± 1.33	86.98 ± 1.30	89.17 ± 1.39	88.33 ± 1.60	88.61 ± 1.42	87.68 ± 1.47
segment	94.98 ± 1.66	96.02 ± 1.21	95.37 ± 0.86	95.76 ± 0.62	94.94 ± 1.21	95.80 ± 1.15	95.15 ± 0.62	95.54 ± 0.94
shuttle	99.94 ± 0.04	99.91 ± 0.05	99.95 ± 0.04	99.92 ± 0.04	99.94 ± 0.04	99.91 ± 0.05	99.96 ± 0.03	99.91 ± 0.05
soybean-large	92.79 ± 1.59	90.77 ± 2.16	91.50 ± 3.81	91.87 ± 2.26	92.62 ± 1.61	91.32 ± 3.30	92.24 ± 1.80	92.79 ± 1.95
spambase	94.01 ± 0.97	93.62 ± 0.80	94.08 ± 0.75	94.03 ± 0.84	93.99 ± 0.66	94.27 ± 0.59	93.97 ± 0.80	94.06 ± 0.39
TIMIT4CF	91.90 ± 0.40	91.70 ± 0.40	91.95 ± 0.39	91.59 ± 0.40	91.82 ± 0.40	87.46 ± 0.48	91.95 ± 0.39	91.78 ± 0.40
TIMIT4CM	92.88 ± 0.38	85.62 ± 0.51	92.71 ± 0.38	92.58 ± 0.38	92.89 ± 0.38	85.84 ± 0.51	92.88 ± 0.38	92.62 ± 0.38
TIMIT6CF	85.20 ± 0.49	84.27 ± 0.50	85.49 ± 0.49	84.89 ± 0.49	85.20 ± 0.49	83.86 ± 0.51	85.21 ± 0.49	84.99 ± 0.49
TIMIT6CM	86.04 ± 0.48	85.45 ± 0.49	86.50 ± 0.47	85.91 ± 0.48	85.98 ± 0.48	85.68 ± 0.49	86.47 ± 0.47	86.04 ± 0.48
USPS	95.44 ± 0.69	95.98 ± 0.65	95.08 ± 0.71	95.98 ± 0.65	94.89 ± 0.73	95.77 ± 0.67	95.68 ± 0.67	95.44 ± 0.69
vehicle	69.76 ± 2.56	69.04 ± 4.30	67.95 ± 6.00	69.88 ± 2.41	66.99 ± 3.10	70.60 ± 1.93	68.80 ± 4.41	70.72 ± 1.70
vote	95.78 ± 2.21	96.01 ± 2.45	94.61 ± 3.19	95.31 ± 2.74	96.01 ± 3.50	95.32 ± 2.72	95.31 ± 3.86	94.37 ± 2.40
waveform-21	85.43 ± 1.34	82.86 ± 0.51	85.14 ± 1.52	83.48 ± 0.56	85.29 ± 1.26	84.18 ± 0.59	85.55 ± 0.98	84.00 ± 0.90

4.G An Alternative Projection Algorithm[1]

For convenience and reference, we restate the projection that we need to compute:

$$\begin{aligned} \underset{\xi}{\text{minimize}} \quad & \frac{1}{2}\|\xi - \xi^*\|_2^2 \\ \text{s.t.} \quad & \sum_l \exp(\xi_l) \leq 1, \end{aligned} \tag{4.32}$$

where $\xi^* = (\xi_1^*, \ldots, \xi_L^*)$ is the point that is to be projected and where $\xi = (\xi_1, \ldots, \xi_L)$. An optimal solution of the above problem corresponds to the ℓ_2-projection of ξ^* onto the set of sub-normalized probability mass functions over L logarithmic probabilities ξ_l.

Algorithm

The projection according to (4.32) can be computed using Algorithm 3. The algorithm works as follows: In line 1, the algorithm checks if the point ξ^* to project is already feasible. If it is feasible, no projection is necessary and ξ^* is the optimal solution for (4.32). Otherwise, the algorithms computes a lower bound $\alpha_{\min}$ and an upper bound $\alpha_{\max}$ in lines 4 and 5, respectively. Within the interval $[\alpha_{\min}, \alpha_{\max}]$, a root α^+ of the function

$$f(\alpha) = \sum_l \exp(\xi_l^* - W(\alpha \exp(\xi_l^*))) - 1 \tag{4.33}$$

is determined in line 6, where $W(\cdot)$ denotes the real branch of Lambert W function [26]. Exploiting this root, the optimal projection is computed and returned in lines 7 and 8, respectively.

Note that for root-finding a multitude of algorithms can be used [15]. In our experiments, we used the bisection method, that repeatedly bisects the interval $[\alpha_{\min}, \alpha_{\max}]$ and selects the sub-interval in which the root must lie for further processing. A popular alternative to the bisection method is for example Newton's method. This latter method has the disadvantage that it requires the computation of the gradient of $f(\alpha)$ but, as an advantage, converges quadratically [15].

Algorithm 3 Project(ξ^*): Solve problem (4.32)

Require: Point ξ^* to project
1: **if** $\sum_l \exp(\xi_l^*) \leq 1$ **then** ▷ Is ξ^* already feasible?
2: **return** ξ^* ▷ Yes, i.e. no projection necessary
3: **end if**
4: $\alpha_{\min} \leftarrow 0$ ▷ Lower bound on α
5: $\alpha_{\max} \leftarrow \max_l \left((\xi_l^* - \tilde{\xi}_l) \exp(-\tilde{\xi}_l) \right)$ ▷ Upper bound on α
6: $\alpha^+ \leftarrow$ Root of $f(\alpha) = [\sum_l \exp(\xi_l^* - W(\alpha \exp(\xi_l^*))) - 1]$ in the interval $[\alpha_{\min}, \alpha_{\max}]$
7: $\xi^+ \leftarrow \xi^* - W(\alpha^+ \exp(\xi^*))$ ▷ Apply operations element-wise
8: **return** ξ^+

Correctness of the Algorithm

Consider the projection in (4.32). If $\sum_l \exp(\xi_l^*) \leq 1$, then ξ^* optimally solves (4.32). Therefore, assume throughout this section that $\sum_l \exp(\xi_l^*) > 1$. The Lagrangian $\mathcal{L}(\xi, \alpha)$ of (4.32) is

$$\mathcal{L}(\xi, \alpha) = \frac{1}{2}\|\xi - \xi^*\|_2^2 + \alpha \left(\sum_l \exp(\xi_l) - 1 \right), \tag{4.34}$$

[1] The algorithm presented in this section was derived after the original paper [20] was published. It provides advantages with respect to runtime compared to the general purpose projection algorithm proposed by [18].

where $\alpha \geq 0$ is a Lagrange multiplier. According to the KKT conditions, if $\boldsymbol{\xi}^+$ is an optimal solution of (4.32), then there exists $\alpha^+ \geq 0$ such that $(\boldsymbol{\xi}^+, \alpha^+)$ is a stationary point of the Lagrangian and such that

$$\alpha^+ \left(\sum_l \exp(\xi_l^+) - 1 \right) = 0. \tag{4.35}$$

We now exploit the KKT conditions to derive an algorithm for solving (4.32). Differentiating the Lagrangian (4.34) with respect to ξ_l and comparing to zero gives the optimality condition

$$\frac{\partial \mathcal{L}(\boldsymbol{\xi}, \alpha)}{\partial \xi_l} = \xi_l - \xi_l^* + \alpha \exp(\xi_l) = 0. \tag{4.36}$$

Solving for ξ_l yields[2]

$$\xi_l(\alpha) = \xi_l^* - W(\alpha \exp(\xi_l^*)), \tag{4.37}$$

where we wrote $\xi_l = \xi_l(\alpha)$ to explicitly show the dependency of ξ_l on α. Note that $\xi_l(\alpha)$ as a function of α is continuous and strictly monotone decreasing in $[0, \infty)$[3]. Hence, there exists a unique α^+ such that

$$\sum_l \exp\left(\xi_l(\alpha^+)\right) = 1. \tag{4.38}$$

This α^+ can be determined by finding the root of the auxiliary function $f(\alpha)$ given as

$$f(\alpha) = \sum_l \exp\left(\xi_l(\alpha^+)\right) - 1. \tag{4.39}$$

The following Lemma provides bounds on α^+ that can be used for bisection.

Lemma 4. *Assume* $\sum_l \exp(\xi_l^*) > 1$. *Then, there exists* $\alpha^+ \in \left[0, \max_l \left((\xi_l^* - \tilde{\xi}_l) \exp(-\tilde{\xi}_l) \right) \right]$ *such that* $f(\alpha^+) = 0$, *where* $\tilde{\xi}_l = \xi_l^* - \log \sum_j \exp(\xi_j^*)$.

Proof. For brevity, let $\alpha_{\min} = 0$ and $\alpha_{\max} = \max_l \left((\xi_l^* - \tilde{\xi}_l) \exp(-\tilde{\xi}_l) \right)$.

As mentioned above, $\xi_l(\alpha)$ is continuous and strictly monotone decreasing in $[0, \infty)$ as a function of α. Therefore, the auxiliary function $f(\alpha)$ is also continuous and strictly monotone decreasing in α. Hence, it suffices to show that $f(\alpha_{\min}) > 0$ and $f(\alpha_{\max}) \leq 0$ to prove the claim:

- $f(\alpha_{\min}) > 0$. Note that $W(0) = 0$. Therefore, $\xi_l(\alpha_{\min}) = \xi_l(0) = \xi_l^*$. Consequently,

$$f(\alpha_{\min}) = \sum_l \exp(\xi_l^*) - 1 > 0, \tag{4.40}$$

  by assumption that $\boldsymbol{\xi}^*$ is super-normalized.

- $f(\alpha_{\max}) \leq 0$. First, note that

$$\sum_l \exp(\tilde{\xi}_l) = \sum_l \left[\frac{\exp(\xi_l^*)}{\sum_j \exp(\xi_j^*)} \right] = 1. \tag{4.41}$$

[2] By substituting $z = \xi_l^* - \xi_l$ in $\xi_l - \xi_l^* + \alpha \exp(\xi_l) = 0$ we obtain $z = \alpha \exp(\xi_l^*) \exp(-z)$, or equivalently, $z \exp(z) = \alpha \exp(\xi_l^*)$. By the definition of the Lambert W function, $W(\alpha \exp(\xi_l^*)) = z$. Consequently, $\xi_l = \xi_l^* - W(\alpha \exp(\xi_l^*))$.

[3] According to [6], the Lambert W function is differentiable in $[0, \infty)$. Therefore, it is also continuous in that region. Furthermore, the derivative is positive in $[0, \infty)$ and, hence, the function is strictly monotone increasing in $[0, \infty)$.

Furthermore,

$$\xi_l(\alpha_{\max}) = \xi_l^* - W\left(\alpha_{\max}\exp(\xi_l^*)\right) \tag{4.42}$$

$$\overset{(a)}{\leq} \xi_l^* - W\left((\xi_l^* - \tilde{\xi}_l)\exp(-\tilde{\xi}_l)\exp(\xi_l^*)\right) \tag{4.43}$$

$$= \xi_l^* - W\left(\log\left(\sum_j \exp(\xi_j^*)\right)\left(\sum_j \exp(\xi_j^*)\right)\right) \tag{4.44}$$

$$\overset{(b)}{=} \xi_l^* - \log\sum_j \exp(\xi_j^*), \tag{4.45}$$

where (a) is because of the definition of $\alpha_{\max}$ and because $\xi_l(\alpha)$ is monotone decreasing in α, and (b) is because of the definition of the Lambert W function. Consequently,

$$\sum_l \exp(\xi_l(\alpha_{\max})) \leq \sum_l \exp\left(\xi_l^* - \log\sum_j \exp(\xi_j^*)\right) \tag{4.46}$$

$$= \sum_l \exp(\tilde{\xi}_l) = 1. \tag{4.47}$$

□

We immediately obtain the following theorem:

Theorem 4 (Correctness of Algorithm 3)**.** *Algorithm 3 optimally solves* (4.32).

Proof. If $\boldsymbol{\xi}^*$ is sub-normalized, the algorithm is trivially optimal. Therefore, assume $\boldsymbol{\xi}^*$ is super-normalized. Lemma 4 ensures that α^+ as determined by the algorithm is a root of the auxiliary function $f(\alpha)$. Therefore, $\boldsymbol{\xi}^+ = \boldsymbol{\xi}^* - W(\alpha^+\exp(\boldsymbol{\xi}^*))$ satisfies $\sum_l \exp(\xi_l^+) = 1$. It is further a stationary point of the Lagrangian and also satisfies all other KKT conditions. □

Bibliography

[1] Kevin Bache and Moshe Lichman. UCI Machine Learning Repository, 2013. URL http://archive.ics.uci.edu/ml.

[2] Christopher M. Bishop and Julia Lasserre. Generative or Discriminative? Getting the Best of Both Worlds. *Bayesian Statistics*, 8:3–24, 2007.

[3] Guillaume Bouchard and Bill Triggs. The Trade-Off between Generative and Discriminative Classifiers. In *COMPSTAT*, pages 721–728, 2004.

[4] Wray Buntine. Theory Refinement on Bayesian Networks. In *Uncertainty in Artificial Intelligence (UAI)*, pages 52–60, 1991.

[5] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A Library for Support Vector Machines. *ACM TIST*, 2:27:1–27:27, 2011. URL http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[6] Robert M. Corless, Gaston H. Gonnet, D. E. G. Hare, David J. Jeffrey, and Donald E. Knuth. On the Lambert W Function. In *Advances in Computational Mathematics*, pages 329–359, 1996.

[7] Corinna Cortes and Vladimir Vapnik. Support-Vector Networks. *Machine Learning*, 20(3):273–297, 1995.

[8] Koby Crammer and Yoram Singer. On the Algorithmic Implementation of Multiclass Kernel-based Vector Machines. *Journal of Machine Learning Research*, 2:265–292, 2001.

[9] Usama M. Fayyad and Keki B. Irani. Multi-Interval Discretization of Continuous-Valued Attributes for Classification Learning. In *International Conference on Artificial Intelligence (IJCAI)*, pages 1022–1029, 2003.

[10] Nir Friedman, Dan Geiger, and Moises Goldszmidt. Bayesian Network Classifiers. *Machine Learning*, 29:131–163, 1997.

[11] Russell Greiner, Xiaoyuan Su, Bin Shen, and Wei Zhou. Structural Extension to Logistic Regression: Discriminative Parameter Learning of Belief Net Classifiers. *Machine Learning*, 59(3):297–322, 2005.

[12] Yuhong Guo, Dana Wilkinson, and Dale Schuurmans. Maximum Margin Bayesian Networks. In *Uncertainty in Artificial Intelligence (UAI)*, pages 233–242, 2005.

[13] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2003.

[14] David Heckerman, Dan Geiger, and David M. Chickering. Learning Bayesian Networks: The Combination of Knowledge and Statistical Data. *Machine Learning*, 20: 197–243, 1995.

[15] Joe D. Hoffman and Steven Frankel. *Numerical Methods for Engineers and Scientists*. CRC Press, 2nd edition, 2001.

[16] Tony Jebara. *Discriminative, Generative and Imitative learning*. PhD thesis, MIT, 2001.

[17] John Lafferty, Andrew McCallum, and Fernando Pereira. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *International Conference on Machine Learning (ICML)*, pages 282–289, 2001.

[18] Anhua Lin. A Class of Methods for Projection on a Convex Set. *Advanced Modeling and Optimization (AMO)*, 5(3), 2003.

[19] Andrew Y. Ng and Michael I. Jordan. On Discriminative vs. Generative Classifiers: A Comparison of Logistic Regression and Naive Bayes. In *Advances in Neural Information Processing Systems (NIPS)*, 2001.

[20] Robert Peharz, Sebastian Tschiatschek, and Franz Pernkopf. The Most Generative Maximum Margin Bayesian Networks. In *International Conference on Machine Learning (ICML)*, volume 28, pages 235–243, 2013.

[21] Franz Pernkopf, Michael Wohlmayr, and Sebastian Tschiatschek. Maximum Margin Bayesian Network Classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 34(3):521–531, 2012.

[22] Rajat Raina, Yirong Shen, Andrew Y. Ng, and Andrew McCallum. Classification with Hybrid Generative/Discriminative Models. In *Advances in Neural Information Processing Systems (NIPS)*, 2003.

[23] Fei Sha. *Large Margin Training of Acoustic Models for Speech Recognition*. PhD thesis, University of Pennsylvania, 2007.

[24] Tomi Silander, Petri Kontkanen, and Petri Myllymäki. On Sensitivity of the MAP Bayesian Network Structure to the Equivalent Sample Size Parameter. In *Proceedings of UAI*, pages 360–367, 2007.

[25] Sebastian Tschiatschek, Nikolaus Mutsam, and Franz Pernkopf. Handling Missing Features in Maximum Margin Bayesian Network Classifiers. In *International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6, 2012.

[26] Eric W. Weisstein. *CRC Concise Encyclopedia of Mathematics*. Chapman and Hall/CRC, 1999.

[27] Hannes Wettig, Peter Grünwald, Teemu Roos, Petri Myllymaki, and Henry Tirri. When Discriminative Learning of Bayesian Network Parameters is Easy. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 491–496, 2003.

[28] Ji Zhu, Saharon Rosset, Trevor Hastie, and Rob Tibshirani. 1-norm Support Vector Machines. *Advances in Neural Information Processing Systems*, 16:49–56, 2004.

[29] Hui Zou and Ming Yuan. The F_∞-Norm Support Vector Machine. *Statistica Sinia*, 18: 379–398, 2008.

5 Reduced-Precision BNCs

This chapter is mainly a compilation of the following three papers that are either already published or were accepted for publication: The conference paper Bayesian Network Classifiers with Reduced Precision Parameters *[33] which appeared in the proceedings of the European Conference on Machine Learning in 2012, the journal article* On Reduced Precision Bayesian Network Classifiers *[32] which was accepted for publication in IEEE Transactions on Pattern Analysis and Machine Intelligence in 2014, and the conference paper* Integer Bayesian Network Classifiers *[35] which will appear in the proceedings the European Conference on Machine Learning in 2014. As minor modifications, overlapping sections of the papers were merged or removed. Furthermore, some additional, yet unpublished, material was added.*

5.1 Introduction and Overview

Most commonly BNCs are implemented on nowadays desktop computers, where double-precision floating-point numbers are used for parameter representation and arithmetic operations. In these BNCs, inference and classification is typically performed using the same precision for parameters and operations, and the executed computations are considered as exact (this does not mean that inference is exact, but only that computations therefore are considered as exact). However, there is a need for BNCs working with limited computational resources. Such resource-constrained BNCs are important in domains such as ambient computing, on-satellite computations,[1] acoustic environment classification in hearing aids, machine learning for prosthetic control (consider for example a brain implant to control hand movements), etc. In these kinds of applications, a trade-off between accuracy and required computational resources is essential.

In this chapter, we investigate BNCs with limited computational demands by considering BNCs with reduced-precision parameters, e.g. fixed-point/floating-point parameters with limited precision. Using reduced-precision parameters is advantageous in many ways: For example, power consumption compared to full-precision implementations can be reduced [30]. Another advantage is that reduced-precision parameters enable one to implement many BNCs in parallel on field programmable gate arrays (FPGAs), i.e. the circuit area requirements on the FPGA correlate with the parameter precision [15]. Our investigations are similar to those performed in digital signal-processing, where reduced-precision implementations for digital signal processors are of great importance [18].

Even though we already narrowed down the field of interest, a multitude of further design decisions and research questions remains, as well as potential approaches for answering them. Some intriguing questions are:

- **Parameter domain.** To use reduced-precision arithmetic for BNCs, the parameters of the BN have to be represented in a reduced-precision format. These parameters can be either specified in the linear or in the logarithmic domain. Which type of parametrization is more promising/advantageous? Logarithmic representation of the parameters is advantageous as a large range of parameters can be efficiently represented. But, when aiming for BNCs with low computational demands, also the inference scenario of interest should be taken into account:
  - **Classification.** This type of inference can be performed using *max-product* [13] message passing. If parameters are represented in the logarithmic domain, *max-sum* message passing can be performed instead [13]. This is computationally advantageous, especially if used in conjunction with fixed-point representations for which addition of numbers is easy.

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

  - **Marginalization.** This type of inference requires sum-product message passing, i.e. summations and products must be computed. If logarithmic parameters are used, these need to be exponentiated when summing, which is computational expensive. Thus, using a linear representation of the parameters could be advantageous.

- **Number format.** The parameters of BNCs can be either represented using fixed-point or floating-point numbers. Is any of these two representations advantageous compared to the other? A brief description of these representations and some advantages and disadvantageous are:

  - **Fixed-point:** Numbers are essentially integers scaled by a constant factor, i.e. the fractional part has a fixed number of digits. We characterize fixed-point numbers by the number of integer bits b_i and the number of fractional bits b_f. The addition of two fixed-point numbers can be easily and accurately performed, while the multiplication of two fixed-point numbers often leads to overflows and requires truncation to achieve results in the same format.
  - **Floating-point:** Numbers are represented by a mantissa and an exponent, e.g. $a = m \cdot 2^e$, where a is the represented number, m the mantissa and e the exponent. Typically, by convention, $|m| < 1$. A fixed number of bits b_e is used to represent the exponent and a fixed number of bits b_m to represent the mantissa. In floating-point arithmetic, computing the product of two numbers is easy and accurate while computing the sum of two numbers causes larger inaccuracies.

- **Feasibility.** Given choices for the above two design decisions, are reduced-precision BNCs feasible, i.e. does the performance of BNCs degrade with decreasing precision in an acceptable manner? This question is partially answered in Section 5.2.

- **Performance Limits.** Can we provide guarantees or limits on how much performance we lose by using reduced-precision parameters? This issue is discussed in Section 5.3.

- **Reduced-Precision Parameter Learning.** Should reduced-precision parameters be learned in a pre-computation step in which we can exploit the full computational power of nowadays computers? Or is it necessary to learn/adopt parameters using reduced-precision arithmetic only? The answer to this question depends on the application of interest. This is depicted in Figure 5.1 and allows us to identify four potential scenarios:

  - **Training and testing using full-precision arithmetic.** This corresponds to what machine learners typically do, i.e. all computations are performed using full-precision arithmetic.
  - **Training using reduced-precision and testing using full-precision arithmetic.** Not of interest.
  - **Training using full-precision and testing using reduced-precision arithmetic.** This describes an application scenario where BNCs with pre-computed parameters can be used, e.g. hearing-aids for auditory scene classification. This scenario enables one to exploit large computational resources for parameter learning, while limiting computational demands at test time. In Section 5.4, we show that learning parameters considering constraints on available computational resources at test time, can be advantageous.
  - **Training and testing using reduced-precision arithmetic.** This is the most challenging case and briefly considered in Section 5.5. Possible applications include for example hearing-aids that continuously adapt their parameters using reduced-precision computations only. Another example could be a satellite-based system for remote sensing that tunes its parameter according to changing atmospheric conditions.

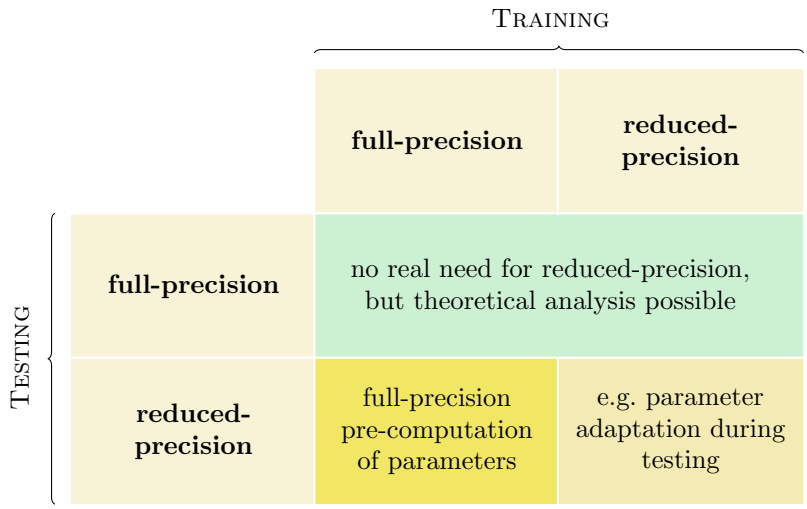


Figure 5.1: Combinations of training/testing using full-precision/reduced-precision arithmetic.

There is clearly a wide range of related questions and issues that we did not mention here. A small collection of such questions as well as possible directions for future research are given in Section 5.6.

Related work deals with sensitivity analysis of BNs, cf. Section 5.2.2, and with credal networks, i.e. generalizations of BNs that associate a whole set of CPDs with every node in the DAG [39], allowing for robust classification and supporting imprecisely specified CPDs. Furthermore, there is related work in terms of undirected graphical networks; recently, a paper on approximating undirected graphical models using integer parameters has been published [27]. 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 the remainder of this chapter, we answer some of the raised questions more thoroughly:

1. Section 5.2 addresses the feasibility of BNCs with reduced-precision floating-point parameters. The feasibility of reduced-precision BNCs with fixed-point parameters has already been investigated briefly in the literature [24]. We extend these investigations by considering the feasibility of reduced-precision BNCs with floating-point parameters, i.e. we analyze the effect of precision-reduction of the parameters represented by floating-point numbers on the classification performance of BNCs. Furthermore, we show how BNCs with reduced-precision parameters can be implemented using integer computations only. The parameters of BNCs are either determined generatively or discriminatively. Discriminatively optimized parameters are typically more extreme, i.e. they take on values close to zero or one. However, our results indicate that BNCs with discriminatively optimized parameters are almost as robust to precision-reduction as BNCs with generatively optimized parameters. Furthermore, even large precision-reduction does not decrease classification performance significantly. This supports application in embedded systems using floating-point numbers with small bit-width.

2. In Section 5.3, we continue our investigations only using fixed-point parameters for the following two reasons: First, because fixed-point parameters can even be used on computing platforms without floating-point processing capabilities. Second, because

summation of fixed-point numbers is exact (neglecting the possibility of overflows), while summation of floating-point numbers is in general not exact.

We investigate the quantization of the parameters of BNCs with discrete valued nodes including the implications on the classification rate (CR). We derive worst-case and probabilistic bounds on the CR for different bit-widths. These bounds are evaluated on several benchmark datasets. Furthermore, we compare the classification performance and the robustness of BNCs with generatively and discriminatively optimized parameters, i.e. parameters optimized for high data likelihood and parameters optimized for classification, with respect to parameter quantization. Generatively optimized parameters are more robust for very low bit-widths, i.e. less classifications change because of quantization. However, classification performance is better for discriminatively optimized parameters for all but very low bit-widths. Additionally, we perform analysis for margin-optimized TAN structures which outperform generatively optimized TAN structures in terms of CR and robustness.

3. Section 5.4 addresses learning of reduced-precision parameters using full-precision computations. An algorithm for the computation of margin maximizing reduced-precision 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 numbers of bits.

4. In Section 5.5, we briefly consider parameter learning using reduced-precision computations only. We start by investigating the effect of approximate computations on online parameter learning. This leads to the observation that the approximate projections needed in the used projected gradient ascent/descent algorithms can severely affect the learning process. We circumvent the need for these projections by proposing special purpose learning algorithms for ML and MM parameters.

5. We end this chapter by briefly considering some open questions and directions for future research in Section 5.6.

5.2 Bayesian Network Classifiers with Reduced-Precision Floating-Point Parameters[2]

As already mentioned, parameters of BNCs are typically represented using high numerical precision, i.e. double-precision floating-point numbers. This high precision representation results in large storage requirements for the parameters, cf. Table 5.1. Furthermore, performing inference using these parameters, requires complex computer architectures (high precision floating-point processing units running at reasonable clock-speeds). Low energy computers or integrated solutions that need to optimize the used hardware resources do not necessarily provide a suitable computing platform. To aid *complexity* reduction, we investigate the performance of BNCs with reduced-precision *floating-point* parameters. Especially, we are interested in comparing the robustness of generatively (ML) and discriminatively (MCL, MM) optimized probability distributions with respect to precision reduction of their parameters using various BN structures.

Some of our findings can be related to results from sensitivity analysis of BNs [4, 5]. Amongst others, the framework of sensitivity analysis describes the dependency of inference queries to variations in the local conditional probability parameters. The precision reduction of the probability parameters resorts to such variations and can, therefore, be interpreted in this framework. However, the focus in this section is different. We are particularly interested in analyzing the classification performance of BNCs when reducing

[2]This section was published in the proceedings of ECML 2012 as *Bayesian Network Classifiers with Reduced Precision Parameters* [33].

Table 5.1: Number of probability parameters (# parameters) and the storage requirements (storage) for these parameters in BNCs with different graph structures and for different datasets. Each parameter is assumed to be stored in double-precision floating-point format, i.e. 64 bits are required for each parameter. Details on the structures and datasets are provided in Chapter 2.

data	structure	# parameters	storage [kB]
USPS	NB	8650	67.6
	TAN-CR	20840	162.8
MNIST	NB	6720	52.5
	TAN-CR	39980	312.3
TIMIT (4 classes)	NB	1320	10.3
TIMIT (6 classes)	NB	1998	15.6

the bit-width of all parameters simultaneously. Additionally, we are interested in comparing the robustness of the classification of BNCs with generatively and discriminatively optimized parameters with respect to this precision reduction. As the local conditional probability parameters of discriminatively optimized BNCs tend to be more extreme, we suspected classification rates of these classifiers to depend stronger on the used precision than the classification rates of BNCs with generatively optimized parameters. Nevertheless, our results demonstrate that this is not true.
Our main findings are:

- The number of extreme conditional probability values, i.e. probabilities close to 0 or 1, in BNCs with discriminatively optimized parameters is larger than in BNCs with generatively optimized parameters, cf. Section 5.2.4. Using results from sensitivity analysis, this suggests that BNCs with discriminatively optimized parameters might be more susceptible to precision reduction than BNCs with generatively optimized parameters. Nevertheless, we observed in experiments that BNCs with both types of parameters can achieve good classification performance using reduced-precision floating-point parameters. In fact, the classification performance is close to BNCs with parameters represented in full double-precision floating-point format, cf. Section 5.2.4.

- The reduction of the precision allows for mapping the classification process of BNCs to the integer domain, cf. Section 5.2.3. Thereby, exact computation in that domain, reduced computational resource requirements and implementation on simple embedded hardware is supported. In fact, some of the considered BNCs can perform classification using integer arithmetic without significant reduction of performance.

The outline of this section is as follows: In Section 5.2.1 we provide a motivating example demonstrating that there is large potential in reducing the precision of the parameters of BNCs. Afterwards, we review some results from sensitivity analysis of BNs in Section 5.2.2. An approach for mapping the parameters of BNCs to the integer domain is presented in Section 5.2.3 and various experiments are provided in Section 5.2.4. Finally, we summarize this section in Section 5.2.5.

5.2.1 Motivating Example

In this section, we provide an example demonstrating that the parameters of BNs employed for classification do not require high precision. They can be approximated coarsely without reducing the classification rate significantly. In some cases, only a few bits for representing each probability parameter of a BNC are necessary to achieve classification rates close to optimal.

The probability parameters of BNCs are typical stored in double-precision floating-point format [19, 16]. We use logarithmic probability parameters $w = \log(\theta)$, with $0 \leq \theta \leq$

1, represented as

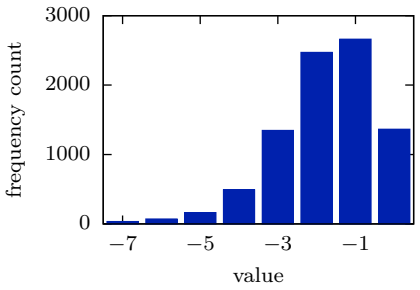
$$w = (-1)^s \left(1 + \sum_{k=1}^{52} b^k 2^{-k}\right) 2^{\left(\sum_{l=0}^{10} e^l 2^l - 1023\right)}, \tag{5.1}$$

where $s \in \{0,1\}$, $b^k \in \{0,1\}$ for all k, and $e^l \in \{0,1\}$ for all l. The term

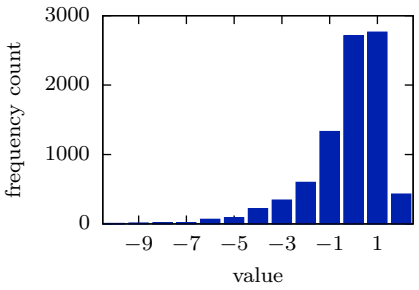
- $(-1)^s$ is the *sign*,
- $(1 + \sum_{k=1}^{52} b^k 2^{-k})$ is the *mantissa*, and
- $(\sum_{l=0}^{10} e^l 2^l - 1023)$ is the *exponent*

of w, respectively. In total 64 bits are used to represent each log-parameter. Processing these parameters on desktop computers does not impose any problems. However, this large bit-width of the parameters can be a limiting factor in embedded systems or applications optimized for low run-times or low energy-consumption.

The range of the parameters using double-precision floating-point format is approximately $\pm 10^{300}$ and by far larger than required; The distribution of the log-parameters of a BNC with ML parameters for handwritten digit data (USPS data, cf. Section 2.3) is shown in Figure 5.2a. Additionally, the distribution of the values of the exponent is shown in Figure 5.2b. All the log-parameters are negative and their range is $[-7, 0]$. The range of the exponent of the log-parameters is $[-10, 2]$.



(a) logarithmic probabilities



(b) exponent of logarithmic conditional probabilities in double-precision

Figure 5.2: Histograms of (a) the log-parameters, and (b) the exponents of the log-parameters of a BNC for handwritten digit data with ML parameters assuming NB structure.

The required bit-width to store the log-parameters in a floating-point format, cf. Equation (5.1), can be reduced in three aspects:

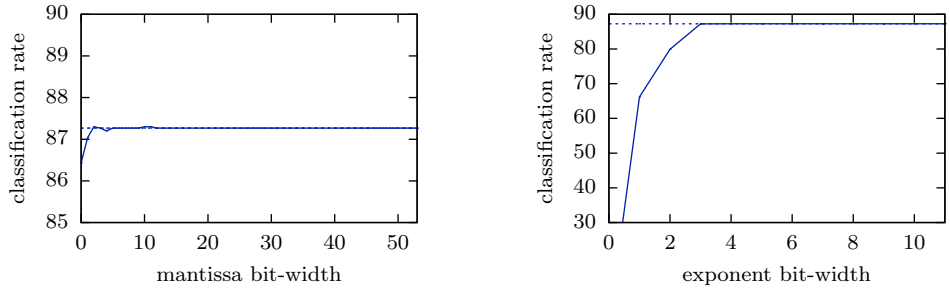
1. **Sign bit.** Every probability θ satisfies $0 \leq \theta \leq 1$. Therefore, its logarithm is in the range $-\infty \leq w \leq 0$. Consequently, the sign bit can be removed without any change in the represented parameters.

2. **Bit-width of the mantissa.** We varied the bit-width of the mantissa of the log-parameters while keeping the exponent unchanged. As a result, we observed that this does not influence the classification rate significantly when using ML parameters, cf. Figure 5.3a. When using 4 or more bits to represent the mantissa, the performance is almost the same as when using the full double-precision floating-point format, i.e. 53 bits for the mantissa.

3. **Bit-width of the exponent.** Changing the bit-width of the exponent has the largest impact on the classification performance. A change of the exponent of a parameter results in a change of the scale of this parameter. The classification rates resulting

from reducing the bit-width of the exponent are shown in Figure 5.3b. Note that we reduced the bit-width starting with the most significant bit (MSB). Only a few bits are necessary for classification rates on par with the rates achieved using full double-precision floating-point parameters.

Based on this motivating example demonstrating the potential of precision reduction we can even map BNCs to the integer domain, cf. Section 5.2.3. Further experimental results are shown in Section 5.2.4.



(a) varying mantissa bit-width, using full bit-width for exponent

(b) varying exponent bit-width, using full bit-width for mantissa

Figure 5.3: Classification rates for varying bit-widths of (a) the mantissa, and (b) the exponent, for handwritten digit data, NB structure, and log ML parameters. The classification rates using full double-precision logarithmic parameters are indicated by the horizontal dotted lines.

5.2.2 Background: Sensitivity of Bayesian Networks

The *sensitivity* of a BN $\mathcal{B} = (\mathcal{G}, \mathcal{P}_\mathcal{G})$ describes the change of a query with respect to changes in the local conditional probabilities in $\mathcal{P}_\mathcal{G}$. For example, a query is the calculation of a posterior probability of the form $\mathrm{P}^\mathcal{B}(\mathbf{X}_q|\mathbf{X}_e)$, with $\mathbf{X}_q, \mathbf{X}_e \subseteq \{C, X_1, \ldots, X_L\}$ and $\mathbf{X}_q \cap \mathbf{X}_e = \emptyset$, where $\mathbf{X}_q$ denotes the query variables and $\mathbf{X}_e$ denotes the evidence (observed) variables. Several results on estimating and bounding this sensitivity exist in the literature, cf. for example [4, 36]. The results therein essentially state that the sensitivity of BNs depends mainly on probability parameters being close to 0 or 1 and queries being close to uniform.

In this context, consider the following theorem:

Theorem 5 (from [4]). *Let X_i be a binary RV in a BN $\mathcal{B} = (\mathcal{G}, \mathcal{P}_\mathcal{G})$, then*

$$\left| \frac{\partial \mathrm{P}^\mathcal{B}(X_i|\mathbf{X}_e)}{\partial \tau_{X_i|\mathbf{Pa}(X_i)}} \right| \leq \frac{\mathrm{P}^\mathcal{B}(X_i|\mathbf{X}_e) \cdot (1 - \mathrm{P}^\mathcal{B}(X_i|\mathbf{X}_e))}{\mathrm{P}^\mathcal{B}(X_i|\mathbf{Pa}(X_i)) \cdot (1 - \mathrm{P}^\mathcal{B}(X_i|\mathbf{Pa}(X_i)))}, \tag{5.2}$$

where $\tau_{X_i|\mathbf{Pa}(X_i)}$ is a meta-parameter such that

$$\mathrm{P}^\mathcal{B}(X_i = 0|\mathbf{Pa}(X_i)) = \tau_{X_i|\mathbf{Pa}(X_i)}, \text{ and} \tag{5.3}$$

$$\mathrm{P}^\mathcal{B}(X_i = 1|\mathbf{Pa}(X_i)) = 1 - \tau_{X_i|\mathbf{Pa}(X_i)}. \tag{5.4}$$

The Theorem states that the magnitude of the partial derivative of $\mathrm{P}^\mathcal{B}(X_i|\mathbf{X}_e)$ with respect to $\tau_{X_i|\mathbf{Pa}(X_i)}$ is bounded above. The bound depends on the query under the current parameters $\mathrm{P}^\mathcal{B}(X_i|\mathbf{X}_e)$ and on the conditional probabilities $\mathrm{P}^\mathcal{B}(X_i|\mathbf{Pa}(X_i))$. The partial derivative is large whenever $\mathrm{P}^\mathcal{B}(X_i|\mathbf{X}_e)$ is close to uniform and whenever $\mathrm{P}^\mathcal{B}(X_i = 0|\mathbf{Pa}(X_i))$ is close to 0 or 1. In classification, the query of interest is the probability of the class variable given the features, i.e. $\mathrm{P}^\mathcal{B}(X_i|\mathbf{X}_e) = \mathrm{P}^\mathcal{B}(C|\mathbf{X})$. Discriminative objectives for parameter learning in BNs aim at good class separation, i.e. $\mathrm{P}^\mathcal{B}(C|\mathbf{X})$ or $1 - \mathrm{P}^\mathcal{B}(C|\mathbf{X})$ is typically large. However, also the parameters tend to be extreme, i.e. $\mathrm{P}^\mathcal{B}(X_i|\mathbf{Pa}(X_i))$ is close to 0 or 1 (some

empirical results supporting this are shown in Section 5.2.4). We expect the bound to be large for discriminatively optimized parameters, as the denominator in the above theorem scales the bound inversely proportional [4]. Hence, either the bound is loose or the partial derivative is actually large resulting in high sensitivity to parameter deviations. This could be the tripping hazard for BNCs with discriminatively optimized parameters. However, experimental observations in Section 5.2.4 show a robust classification behavior using discriminatively optimized small bit-width parameters.

The above Theorem only describes the sensitivity with respect to a single parameter. There are some extensions of sensitivity analysis describing the sensitivity of queries with respect to changes of many parameters [5]. However, to the best of the authors knowledge, these do not extend to changes of all parameters, which is the focus of this analysis. Furthermore, in classification we are not directly interested in the sensitivity of certain queries. The focus is rather on the maximum of a set of queries, i.e. the sensitivity of the MAP classification.

5.2.3 Bayesian Network Classifiers in the Integer Domain

In this section, we present how to cast classification using BNCs to the integer domain. This is possible when using reduced-precision log-parameters for the BNCs. Without reduced-precision, the mapping can not be achieved considering the large range of numbers representable by double-precision floating-point numbers.

Remember, a BNC given by the BN $\mathcal{B} = (\mathcal{G}, \mathcal{P}_\mathcal{G})$ where the class variable has no parents assigns an instantiation $\mathbf{x}$ of the attributes to class

$$c = \arg\max_{c' \in \mathbf{val}(C)} \mathrm{P}^{\mathcal{B}}(c', \mathbf{x}) \tag{5.5}$$

$$= \arg\max_{c' \in \mathbf{val}(C)} \mathrm{P}(C = c') \prod_{i=1}^{L} \mathrm{P}(X_i = \mathbf{x}(X_i) | \mathbf{Pa}(X_i) = \mathbf{x}(\mathbf{Pa}(X_i))), \tag{5.6}$$

where $\mathbf{x}(X_k)$ denotes the entry in $\mathbf{x}$ corresponding to X_k. This classification rule can be equivalently stated in the logarithmic domain, i.e. $\mathbf{x}$ is assigned to class

$$c = \arg\max_{c' \in \mathbf{val}(C)} \left[\log \mathrm{P}(C = c') + \sum_{i=1}^{L} \log \mathrm{P}(X_i = \mathbf{x}(X_i) | \mathbf{Pa}(X_i) = \mathbf{x}(\mathbf{Pa}(X_i))) \right]. \tag{5.7}$$

As shown in Sections 5.2.1 and 5.2.4 the logarithmic probabilities in the above equation can often be represented using only a few bits without reducing the classification rate significantly. In many cases, 2 bits for the mantissa and 4 bits for the exponent are sufficient to achieve good classification rates. Using these 6 bits, the logarithmic probability $w^i_{j|\mathbf{h}} = \log \theta^i_{j|\mathbf{h}}$ is given as

$$w^i_{j|\mathbf{h}} = -(1 + b^{i,1}_{j|\mathbf{h}} \cdot 2^{-1} + b^{i,2}_{j|\mathbf{h}} \cdot 2^{-2}) \cdot 2^{\left(\sum_{k=0}^{3} e^{i,k}_{j|\mathbf{h}} \cdot 2^k - 7\right)}. \tag{5.8}$$

Hence,

$$c = \arg\max_{c' \in \mathbf{val}(C)} \left[w^0_{c'} + \sum_{i=1}^{L} w^i_{\mathbf{x}(X_i)|\mathbf{x}(\mathbf{Pa}(X_i))} \right] \tag{5.9}$$

$$= \arg\min_{c' \in \mathbf{val}(C)} \left[-w^0_{c'} - \sum_{i=1}^{L} w^i_{\mathbf{x}(X_i)|\mathbf{x}(\mathbf{Pa}(X_i))} \right] \tag{5.10}$$

$$= \arg\min_{c' \in \mathbf{val}(C)} \Bigg[(1 + b^{0,1}_{c'} \cdot 2^{-1} + b^{0,2}_{c'} \cdot 2^{-2}) \cdot 2^{\left(\sum_{k=0}^{3} e^{i,k}_{c'} \cdot 2^k - 7\right)} + \tag{5.11}$$

$$\sum_{i=1}^{L} (1 + b^{i,1}_{\mathbf{x}(X_i)|\mathbf{x}(\mathbf{Pa}(X_i))} 2^{-1} + b^{i,2}_{\mathbf{x}(X_i)|\mathbf{x}(\mathbf{Pa}(X_i))} 2^{-2}) \cdot 2^{\left(\sum_{k=0}^{3} e^{i,k}_{\mathbf{x}(X_i)|\mathbf{x}(\mathbf{Pa}(X_i))} 2^k - 7\right)} \Bigg].$$

Multiplying (5.11) by the constant 2^9 does not change the classification. Hence, classification can be performed by

$$c = \arg \min_{c' \in \mathbf{val}(C)} \left[(4 + b^{0,1}_{c'} \cdot 2 + b^{0,2}_{c'}) \cdot 2^{\left(\sum_{k=0}^{3} e^{i,k}_{c'} \cdot 2^k\right)} + \right. \tag{5.12}$$
$$\left. \sum_{i=1}^{L} (4 + b^{i,1}_{\mathbf{x}(X_i)|\mathbf{x}(\mathbf{Pa}(X_i))} \cdot 2 + b^{i,2}_{\mathbf{x}(X_i)|\mathbf{x}(\mathbf{Pa}(X_i))}) \cdot 2^{\left(\sum_{k=0}^{3} e^{i,k}_{\mathbf{x}(X_i)|\mathbf{x}(\mathbf{Pa}(X_i))} \cdot 2^k\right)} \right]$$

which resorts to integer computations only. Furthermore, no floating-point rounding errors of any kind are introduced during computation when working purely in the integer domain. Integer arithmetic is sufficient for implementation.

5.2.4 Experiments

In this section, we present classification performance experiments using reduced-precision log-probability parameters for BNCs. Throughout this section we consider the TIMIT-4/6, USPS and MNIST data, cf. Section 2.3. As BN structures, we considered the naive Bayes (NB) structure, the generative TAN-CMI structure [6] and the discriminative TAN-OMI-CR and TAN-CR structures [23], cf. Section 2.2.2.

Number of Extreme Parameter Values in BNCs

We determined BNCs with ML, MCL and MM parameters. For calculating the MCL and MM parameters we used the conjugate gradient based approaches proposed in [25]. However, we did not use the proposed early-stopping heuristic for determining the number of conjugate gradient iterations but rather performed up to 200 iterations (or until there was no further increase in the objective). We then counted the number of conditional probability parameters with a maximal distance of ϵ to the extreme values 0 and 1, i.e. the count is given as

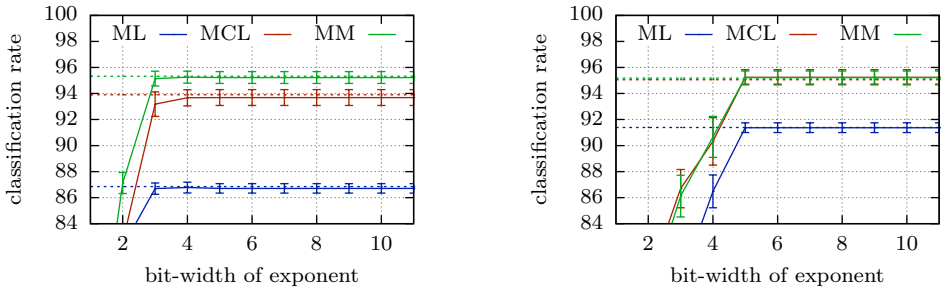
$$M_\epsilon = \sum_{i,j,\mathbf{h}} \mathbf{1}_{\left((1-\theta^i_{j|\mathbf{h}})<\epsilon\right)} + \sum_{i,j,\mathbf{h}} \mathbf{1}_{\left(\theta^i_{j|\mathbf{h}}<\epsilon\right)}. \tag{5.13}$$

The results for USPS and MNIST data are shown in Tables 5.2(a) and 5.2(b), respectively. The number of extreme parameter values in BNCs with MCL parameters is larger than in BNCs with MM parameters, and the number of extreme parameter values in BNCs with MM parameters is larger than in BNCs with ML parameters. This suggests that classification using MCL parameters is more sensitive to parameter deviations than classification with MM parameters, and classification using MM parameters is more sensitive to deviations than classification with ML parameters.

Reduced-Precision Classification Performance

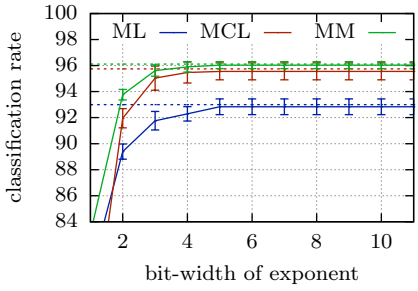
We evaluated the classification performance of BNCs with ML, MCL and MM parameters on the USPS, MNIST and TIMIT data. Results are shown in Figures 5.4, 5.5, and 5.6, respectively. Classification rates using full double-precision floating-point parameters are indicated by the dotted lines. The classification performance resulting from BNCs with reduced-precision ML, MCL, and MM parameters are shown by the solid lines. Reduced-precision parameters were determined by firstly learning parameters in double-precision, and secondly reducing the precision of these parameters. Even when using only 4 bits to represent the exponent and 1 bit to represent the mantissa, the classification rates are close to full-precision performance on USPS data. On MNIST and TIMIT data the results are similar when 4 and 2 bits are used to represent the mantissa, respectively.

Furthermore, we evaluated the classification rates of BNCs with reduced-precision parameters for training sets with varying sizes. The training sets were obtained by selecting the desired number of samples randomly from all available samples. The remaining

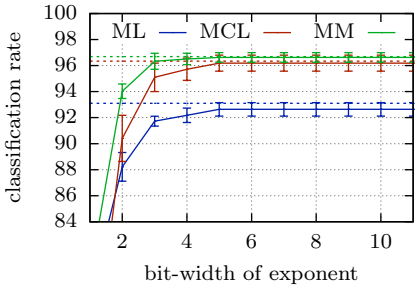


(a) NB

(b) TAN-CMI



(c) TAN-OMI-CR



(d) TAN-CR

Figure 5.4: Classification rates of BNCs with (a) NB, (b) TAN-CMI, (c) TAN-OMI-CR, and (d) TAN-CR structures using reduced-precision ML, MCL, and MM parameters on USPS data. The bit-width of the mantissa was fixed to 1 bit and the bit-width of the exponent was varied. The classification rates for full double-precision floating-point parameters are indicated by the horizontal dotted lines. Error bars indicate the 95 % confidence intervals of the mean classification rate over 5 different training/test splits.

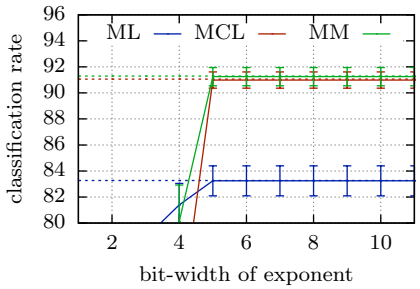


Figure 5.5: Classification rates of BNCs with NB structure using reduced-precision ML, MCL, and MM parameters on MNIST data. The bit-width of the mantissa was fixed to 4 bits and the bit-width of the exponent was varied. The classification rate for full double-precision floating-point parameters is indicated by the horizontal dotted lines. Error bars indicate the 95 % confidence intervals of the mean classification rate over 5 different training/test splits.

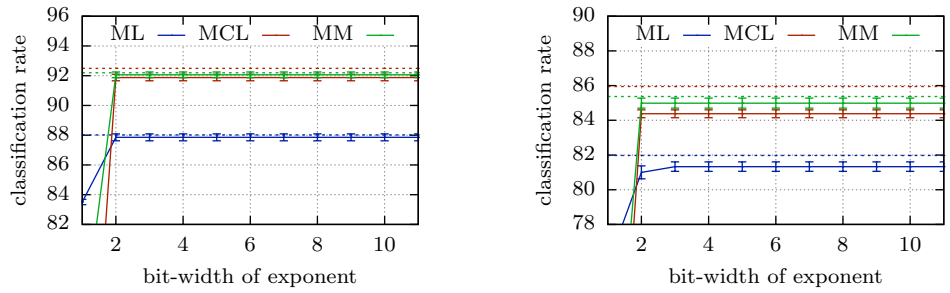
Table 5.2: Number of probability parameters $\theta^i_{j|\mathbf{h}}$ close to the extreme values 0 and 1. Additionally, the total number of parameters (# par.) and classification rates (CR) on the test set using parameters in full double-precision floating-point format on (a) USPS data and (b) MNIST data are shown.

(a) USPS

		$M_{0.05}$			$M_{0.01}$			CR		
structure	# par.	ML	MCL	MM	ML	MCL	MM	ML	MCL	MM
NB	8650	1478	4143	1837	364	2134	446	87.10	93.93	95.00
TAN-CMI	33040	12418	14712	13002	8271	9371	8428	91.90	95.70	95.37
TAN-OMI-CR	25380	6677	8167	7441	3486	3937	3624	92.40	95.73	95.40
TAN-CR	20840	5405	7344	6519	2666	3503	3009	92.57	95.97	95.87

(b) MNIST

		$M_{0.05}$			$M_{0.01}$			CR		
structure	# par.	ML	MCL	MM	ML	MCL	MM	ML	MCL	MM
NB	6720	3252	3289	3170	1784	1513	1520	83.73	92.00	91.97
TAN-CMI	38350	15772	25327	16790	8603	18647	9448	91.28	92.91	94.21
TAN-OMI-CR	44600	22488	29159	24048	13615	20419	15147	92.01	93.59	94.60
TAN-CR	39980	19557	25733	23308	11794	17702	16020	92.58	93.72	95.02



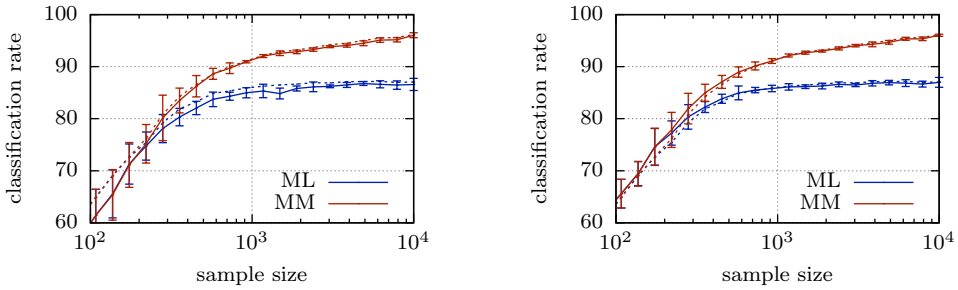
(a) 4 classes (b) 6 classes

Figure 5.6: Classification rates of BNCs with NB structure using ML, MCL, and MM parameters with reduced-precision on TIMIT data with (a) 4 classes and (b) 6 classes, respectively. The bit-width of the mantissa was fixed to 2 bits and the bit-width of the exponent was varied. The classification rates for full double-precision floating-point parameters are indicated by the horizontal dotted lines. Error bars indicate the 95 % confidence intervals of the mean classification rate over 5 different training/test splits.

samples were used as test set. For every sample size, 5 different training/test splits were evaluated. Results on USPS data are shown in Figure 5.7. Classification performance using reduced-precision parameters is close to optimal for all sample sizes.

5.2.5 Summary

In this section, we presented classification results of BNCs when reducing the precision of the probability parameters. Contrary to the authors' expectation, even discriminatively optimized BNCs are robust to distortions in the parameters resulting from the bit-width reduction. About 6 to 10 bits are necessary to represent each probability parameter while maintaining classification rates close to full-precision performance. This allows either to implement BNCs with reduced-precision floating point arithmetic or to cast the classification to the integer domain. In both cases, computational and run-time benefits arise when implementing BNCs on embedded systems or low-power computers.



(a) 3 bit exponent bit-width

(b) 5 bit exponent bit-width

Figure 5.7: Classification rates of BNCs with NB structures using reduced-precision ML and MM parameters on USPS data. The parameters were learned from training sets with varying sizes. The bit-width of the mantissa was fixed to 1 bit. The bit-width of the exponent is (a) 3 bits and (b) 5 bits, respectively. The classification rates for full double-precision floating-point parameters using the same training data are indicated by the dashed lines. Error bars indicate the 95 % confidence intervals of the mean classification rate over 5 different training/test splits.

5.3 Bounds for Bayesian Network Classifiers with Reduced-Precision Parameters[3]

In this section, we extend the promising results from the last section by presenting novel theoretical results and extended empirical results for BNCs with finite precision *fixed-point* parameters. All our results are based on the assumption that parameters are learned in full-precision and rounded to the desired precision for classification. We derive three types of bounds on the classification performance after parameter precision reduction and compare these in experiments. Additionally, we empirically compare the classification performance and robustness of BNCs with respect to precision reduction for different learning paradigms. In particular, we use generatively and discriminatively optimized parameters/structures [6, 23, 21, 7, 28, 25]. For generative parameter learning, we resort to Bayesian parameter estimation [3, 9]. This type of parameter learning results in a posterior distribution over the parameters, enabling us to consider the uncertainty in the parameter estimates in our bounds. Taking this uncertainty into account is crucial as the common assumptions of uniform and independent quantization error are incorrect and result in loose bounds. However, ML estimates do not provide this uncertainty information.

Our main results presented in this section are:

- Derivation of probabilistic and worst-case bounds on the classification performance of BNCs with quantized parameters.
- An empirical evaluation of these bounds on classical machine learning datasets.
- Empiric evidence that BNCs with discriminatively optimized parameters are not more robust to parameter quantization than BNCs with generatively optimized parameters.[4] However, classification performance using discriminatively optimized parameters is better when using bit-widths of 3 bits or more.
- Empiric evidence that BNCs with generatively optimized parameters and discriminatively optimized structures, i.e. structures optimized using a large margin score,

[3] This section is published as *On Bayesian Network Classifiers with Reduced Precision Parameters* in TPAMI [32].

[4] Robustness compares the number of changing classifications of generatively and discriminatively optimized BNCs with respect to parameter quantization.

(1) yield higher classification performance, and (2) are more robust to parameter quantization than BNCs with generatively optimized structures. The former statement holds for all considered bit-widths (1–10 bits), and the latter statement holds for small bit-widths (5/10 bits, depending on the dataset).

This part of this thesis is structured as follows: In Section 5.3.1 we introduce our notation, formally introduce BNCs, and describe methods for assessing the CPTs of BNCs. In Section 5.3.2 we derive bounds on the CR performance of BNCs with reduced-precision parameters and present experiments supporting our arguments in Section 5.3.3. We summarize our analysis in Section 5.3.4.

5.3.1 Background: Learning Bayesian Network Classifiers

We assume BNs with a fixed graph. As already mentioned, for learning the parameters $\mathcal{P}_\mathcal{G}$ of a BN two paradigms exist, namely generative parameter learning and discriminative parameter learning: An instance of the generative paradigm is ML parameter learning. Its objective is maximization of the likelihood of the data with respect to the parameters. ML parameter learning is not well suited for our needs, as it only results in a point estimate of the parameters and does not capture information on the distribution of the parameters. However, this distribution is important when investigating effects of precision reduction. Therefore, we resort to Bayesian parameter estimation using Dirichlet priors [9, 26]. We assume global parameter independence, i.e. the CPTs corresponding to the different nodes in the BNC are independent, and local parameter independence, i.e. the parameters for different parent states are independent [9]. In detail, assuming Dirichlet priors, the parameters[5] $\mathrm{P}(X_i|\mathbf{Pa}(X_i)=\mathbf{h})$ follow a Dirichlet distribution $\mathrm{Dir}\left(\widetilde{\boldsymbol{\alpha}}^i_\mathbf{h}\right)$ with concentration parameters

$$\widetilde{\boldsymbol{\alpha}}^i_\mathbf{h} = \left[\widetilde{\alpha}^i_{1,\mathbf{h}}, \ldots, \widetilde{\alpha}^i_{|\mathbf{val}(X_i)|,\mathbf{h}}\right], \tag{5.14}$$

i.e. $\mathrm{P}(X_i|\mathbf{Pa}(X_i)=\mathbf{h}) \sim \mathrm{Dir}\left(\widetilde{\boldsymbol{\alpha}}^i_\mathbf{h}\right)$. Given the training data $\mathcal{D}$, the posterior parameter distribution is

$$\mathrm{P}(X_i|\mathbf{Pa}(X_i)=\mathbf{h};\mathcal{D}) \sim \mathrm{Dir}\left(\widetilde{\boldsymbol{\alpha}}^i_\mathbf{h} + \mathbf{n}^i_\mathbf{h}\right), \tag{5.15}$$

where $\mathbf{n}^i_\mathbf{h}$ is a vector of frequency counts obtained from the training data. The j^{th} entry of $\mathbf{n}^i_\mathbf{h}$, denoted as $n^i_{j|\mathbf{h}}$, is the number of times $X_i = j$ together with $\mathbf{Pa}(X_i)=\mathbf{h}$ is observed in the training data. Each parameter *instantiation* is marginally beta distributed, i.e.

$$\Theta^i_{j|\mathbf{h}} = \mathrm{P}(X_i = j|\mathbf{Pa}(X_i)=\mathbf{h}) \sim \mathrm{Beta}\left(\alpha^i_{j|\mathbf{h}}, \beta^i_{j|\mathbf{h}}\right), \tag{5.16}$$

where

$$\alpha^i_{j|\mathbf{h}} = \widetilde{\alpha}^i_{j|\mathbf{h}} + n^i_{j|\mathbf{h}}, \text{ and} \tag{5.17}$$

$$\beta^i_{j|\mathbf{h}} = \sum_{\substack{j'=1 \\ j'\neq j}}^{|\mathbf{val}(X_i)|} \left(\widetilde{\alpha}^i_{j'|\mathbf{h}} + n^i_{j'|\mathbf{h}}\right). \tag{5.18}$$

From these beta distributions, maximum a-posteriori (MAP) parameters can be computed as

$$\theta^{i,\mathrm{MAP}}_{j|\mathbf{h}} = \frac{\alpha^i_{j|\mathbf{h}} - 1}{\alpha^i_{j|\mathbf{h}} + \beta^i_{j|\mathbf{h}} - 2}. \tag{5.19}$$

In contrast to generative parameter learning, *discriminative parameter learning* aims at identifying parameters leading to good CR performance. An instance of this discriminative

[5] For convenience, we denote the RVs $\mathrm{P}(X_i|\mathbf{Pa}(X_i)=\mathbf{h})$ as parameters.

paradigm is MM parameter learning, which is restated here for reference. We use the formulation proposed in Pernkopf et al. [25]: MM parameters $\mathcal{P}_{\mathcal{G}}^{\mathrm{MM}}$ are found as

$$\mathcal{P}_{\mathcal{G}}^{\mathrm{MM}} = \arg\max_{\mathcal{P}_{\mathcal{G}}} \prod_{n=1}^{N} \min\left(\gamma, d^{\mathcal{B}}(c^{(n)}, \mathbf{x}^{(n)})\right), \tag{5.20}$$

where $\min(\gamma, d^{\mathcal{B}}(c^{(n)}, \mathbf{x}^{(n)}))$ denotes the hinge loss and $d^{\mathcal{B}}(c^{(n)}, \mathbf{x}^{(n)})$ is the margin of the n^{th} sample given as

$$d^{\mathcal{B}}(c^{(n)}, \mathbf{x}^{(n)}) = \frac{\mathrm{P}^{\mathcal{B}}(c^{(n)}, \mathbf{x}^{(n)})}{\max_{c \neq c^{(n)}} \mathrm{P}^{\mathcal{B}}(c, \mathbf{x}^{(n)})}, \tag{5.21}$$

and $\gamma > 1$ is a parameter scaling the margin. In this way, the margin *measures* the likelihood of the n^{th} sample belonging to the correct class $c^{(n)}$ in relation to the strongest competing class. The n^{th} sample is correctly classified if $d^{\mathcal{B}}(c^{(n)}, \mathbf{x}^{(n)}) > 1$ and vice versa. This type of learning results in a point estimate for the parameters, i.e. no information on the distribution of the parameters is obtained.

5.3.2 Performance Bounds

In this section, we derive worst-case and probabilistic bounds on the classification rate of BNCs with CPTs represented by reduced-precision fixed-point numbers. Before deriving these bounds, we formalize the considered scenario.

Setting

When representing the parameters of BNCs in reduced-precision, one has to decide if to represent the probabilities or the logarithmic probabilities. Typically, log-probabilities are favored for numerical reasons, i.e. a large dynamic range is achieved and classification resorts to a simple addition. However, reduced-precision log-probabilities have the drawback that the resulting CPTs are in general not normalized. In contrast, when representing probabilities in reduced-precision, ensuring proper normalization is easy. Note that to overcome this normalization issue, undirected graphical models are an appealing alternative as well [27]. As we only consider classification tasks in this section, we focus on reduced-precision log-probabilities.

The logarithm of the joint probability induced by a BN $\mathcal{B}$ according to (2.5) is

$$\log \mathrm{P}^{\mathcal{B}}(C = c, \mathbf{X} = \mathbf{x}) = \sum_{i=0}^{L} \log \mathrm{P}(X_i = x_i | \mathbf{Pa}(X_i) = \mathbf{x}(\mathbf{Pa}(X_i))) \tag{5.22}$$

$$= \sum_{i=0}^{L} \sum_{\mathbf{h}} \sum_{j} \mathbf{1}_{([c,\mathbf{x}](X_i)=j \text{ and } [c,\mathbf{x}](\mathbf{Pa}(X_i))=\mathbf{h})} W_{j|\mathbf{h}}^{i} \tag{5.23}$$

$$= \boldsymbol{\phi}(c, \mathbf{x})^T \mathbf{W}, \tag{5.24}$$

where we used that we identify the class C also as X_0, and where $\boldsymbol{\phi}(c, \mathbf{x})$ collects the terms $\mathbf{1}_{([c,\mathbf{x}](X_i)=j \text{ and } [c,\mathbf{x}](\mathbf{Pa}(X_i))=\mathbf{h})}$ in a vector and $\mathbf{W}$ the terms $W_{j|\mathbf{h}}^{i} = \log \Theta_{j|\mathbf{h}}^{i}$, respectively. Consequently, the probability $\mathrm{P}^{\mathcal{B}}(C = c, \mathbf{X} = \mathbf{x})$ can be written as

$$\mathrm{P}^{\mathcal{B}}(C = c, \mathbf{X} = \mathbf{x}) = e^{\boldsymbol{\phi}(c,\mathbf{x})^T \mathbf{W}}. \tag{5.25}$$

In the remainder of this section, we consider the effect of quantizing the RVs $\mathbf{W}$ using fixed-point numbers with B bits. Quantization of these RVs is performed by rounding, i.e. let $w_{j|\mathbf{h}}^{i}$ denote an instantiation of $W_{j|\mathbf{h}}^{i}$ and $\mathcal{Q}_S^B(w_{j|\mathbf{h}}^{i})$ its quantized value using a scale factor $S > 0$. Then

$$\mathcal{Q}_S^B(w_{j|\mathbf{h}}^{i}) = 2^{-B} \left[\frac{w_{j|\mathbf{h}}^{i}/S}{2^{-B}} \right]_R, \tag{5.26}$$

where $[a]_R$ means that a is rounded to the closest integer number. Note that we can arbitrarily scale all parameters by some constant factor S without changing the class assignment of any sample. The usefulness and selection of the scale factor S is explained in the next section. In the remainder of this section we denote the width of the quantization interval as q, i.e. $q = 2^{-B}$. Additionally, for ease of notation we write $\mathcal{Q}(\cdot)$ instead of $\mathcal{Q}_S^B(\cdot)$ and assume some fixed S and B.

Selection of the Scale Factor

By appropriately selecting the scale factor S, we ensure that all parameters can be represented using a desired number of B bits — without the need for integer bits. This alleviates our analysis since only the fractional bits of the fixed-point representation of all parameters have to be considered. Note that in general the scale factor should be as small as possible to ensure that the scaled parameters exploit the full range of representable numbers. For example, if we use a 5 bit fixed-point representation and the parameters are in the range $[0, -20]$, then these parameters should rather be scaled to $[0, -(1-2^{-5})]$ than to $[0, -0.5]$.

For generative or discriminative parameters, S is selected as follows:

Generative parameters. We use MAP scaled log-parameters in the reduced-precision BNCs. Let $Z \sim \text{Beta}\left(\alpha^i_{j|\mathbf{h}}, \beta^i_{j|\mathbf{h}}\right)$ with shape parameters $\alpha^i_{j|\mathbf{h}}, \beta^i_{j|\mathbf{h}} > 1$. Further, let $Y = \frac{1}{S}\log(Z)$ be the corresponding scaled log-parameters, where $S > 0$. The distribution $f_Y^S(y)$ of Y can be easily determined as

$$f_Y^S(y) = S\frac{e^{\alpha^i_{j|\mathbf{h}} Sy}(1-e^{Sy})^{\beta^i_{j|\mathbf{h}}-1}}{\text{B}\left(\alpha^i_{j|\mathbf{h}}, \beta^i_{j|\mathbf{h}}\right)}, \tag{5.27}$$

where $\text{B}(c,d) = \int_0^1 u^{c-1}(1-u)^{d-1}\text{d}u$ is the beta function. Assuming that we quantize Y using B bits and that quantization extends over the negative real numbers, the possible values are $\mathcal{C}_B = \{-n \cdot 2^{-B} \colon n \in \mathbb{N}_0\}$. For all $n \in \mathbb{N}_0$, the probability of $\mathcal{Q}(Y) = -nq$ is [38]

$$\text{P}(\mathcal{Q}(Y) = -nq) = \int_{-nq-\frac{q}{2}}^{\min\{0,-nq+\frac{q}{2}\}} f_Y^S(y)\text{d}y \tag{5.28}$$

$$= \frac{\text{B}\left(e^{S\min\{0,-nq+\frac{q}{2}\}}; \alpha^i_{j|\mathbf{h}}, \beta^i_{j|\mathbf{h}}\right) - \text{B}\left(e^{-Snq-S\frac{q}{2}}; \alpha^i_{j|\mathbf{h}}, \beta^i_{j|\mathbf{h}}\right)}{\text{B}\left(\alpha^i_{j|\mathbf{h}}, \beta^i_{j|\mathbf{h}}\right)}, \tag{5.29}$$

where $\text{B}(k;c,d) = \int_0^k u^{c-1}(1-u)^{d-1}\text{d}u$ is the incomplete beta function. The distribution $f_Y^S(y)$ is unimodal, i.e. its only maximum m is attained[6] at

$$m = \frac{1}{S}\log\frac{\alpha^i_{j|\mathbf{h}}}{\alpha^i_{j|\mathbf{h}} + \beta^i_{j|\mathbf{h}} - 1}. \tag{5.30}$$

Hence, we can easily determine the quantized parameter value with highest a-posteriori probability as

$$n^* = \arg\max_{\widetilde{n}\in\mathbb{N}_0} \text{P}(\mathcal{Q}(Y) = -\widetilde{n}q) \tag{5.31}$$

$$= \arg\max_{n\in\{\lfloor m/q\rfloor, \lceil m/q\rceil\}} \text{P}(\mathcal{Q}(Y) = nq), \tag{5.32}$$

where $\lfloor\cdot\rfloor$ and $\lceil\cdot\rceil$ denote the floor and ceiling functions, respectively. To ensure that n^* can be represented using B bits, $S > 0$ is selected such that

$$|\arg\max_{n\in\{\lfloor m/q\rfloor, \lceil m/q\rceil\}} \text{P}(\mathcal{Q}_S^B(W^i_{j|\mathbf{h}}) = nq)| \leq 2^B - 1. \tag{5.33}$$

[6] The maximum can be determined by solving $\frac{\text{d}}{\text{d}y} f_Y^S(y) = 0$.

Hence, we need that $|\lfloor m/q \rfloor| \leq 2^B - 1$ (note that m is negative). This inequality certainly holds if $-m/q \leq 2^B - 1$. Thus it suffices to select S such that

$$S \geq \max_{i,j,\mathbf{h}} \left[-\frac{1}{1-q} \log \left(\frac{\alpha^i_{j|\mathbf{h}}}{\alpha^i_{j|\mathbf{h}} + \beta^i_{j|\mathbf{h}} - 1} \right) \right]. \tag{5.34}$$

Discriminative parameters. Only a point estimate of the MM parameters $\mathbf{w}^{\mathrm{MM}}$ is available. We scale these parameters such that they can be represented using B bits. Hence, we require $|w^{i,\mathrm{MM}}_{j|\mathbf{h}}|/S \leq 1 - 2^{-B}$ for all $|w^{i,\mathrm{MM}}_{j|\mathbf{h}}|$. This is satisfied by selecting

$$S \geq \max_{i,j,\mathbf{h}} \frac{|w^{i,\mathrm{MM}}_{j|\mathbf{h}}|}{1-q}. \tag{5.35}$$

On the Quantization Error

Given the training data and having performed parameter estimation, we want to obtain bounds on the CR of BNCs when using fixed-point numbers with B bits to represent their CPTs. As explained in Section 5.3.1, the components of the random vector $\boldsymbol{\Theta} = \exp(\mathbf{W})$ for Bayesian parameter estimates are marginally beta-distributed. A first observation is that in general the quantization error of the logarithmic parameters $\mathbf{W}$ is not uniform. This is especially true, when only few bits are used. For example, consider Figure 5.8; the distribution of the quantization error over quantization interval q of an RV $\log(Y)$, where Y is beta-distributed is shown. For convenience, the scale factor introduced above is set to $S = 1$. The RVs Y corresponding to the left and right subfigures are distributed as $Y \sim \mathrm{Beta}\left(1 \cdot 10^3, 9 \cdot 10^3\right)$ and $Y \sim \mathrm{Beta}\left(1 \cdot 10^5, 9 \cdot 10^5\right)$, respectively. When using sufficiently many bits, the quantization error is uniformly distributed in the quantization interval q. However, in the case of few bits, the distribution of the quantization error is not uniform. Consequently, assuming a uniformly distributed quantization error can be inappropriate. Furthermore, note that although both RVs have the same expected value, their quantization errors are different. As a consequence, also the average quantization error, indicated by the vertical red line in Figure 5.8, differs.

The setting considered above describes the situation when a fixed but unknown distribution of some RV is estimated from training sets of different sizes. The more training samples are observed, the more peaked the beta distribution becomes and the more concentrated is the quantization error (for small number of bits). This matches our intuition — the more training samples are available, the less uncertain are the parameter estimates. Consequently, the uncertainty about the expected quantization error is reduced with increasing sample size. This observation is reflected in the bounds derived below.

Worst-Case Bound

We are now able to derive a worst-case bound on the CR. For ease of notation, we state this bound for the single-sample case:

Theorem 6 (Worst-Case Bound)**.** *Let $\mathrm{P}^{\mathcal{B}}$ be the probability distribution defined by a BN, and let S, B, q be as introduced above. Further, let $(c, \mathbf{x})$ be a sample belonging to class c with feature instantiation $\mathbf{x}$. Assuming that the parameters of the BN are independent, the expected classification rate for this sample can be lower bounded as*

$$\mathbb{E}\left[\mathbf{1}_{\left(c = h_{\mathrm{P}^{\mathcal{B}}}(\mathbf{x})\right)}\right] \geq 1 - \min\left\{1, \sum_{c' \neq c} e^{S(L+1)q} \left[\prod_{(i,j,\mathbf{h}) \in A_1} M^i_{j|\mathbf{h}}\right] \left[\prod_{(i,j,\mathbf{h}) \in A_2} \widetilde{M}^i_{j|\mathbf{h}}\right]\right\}, \tag{5.36}$$

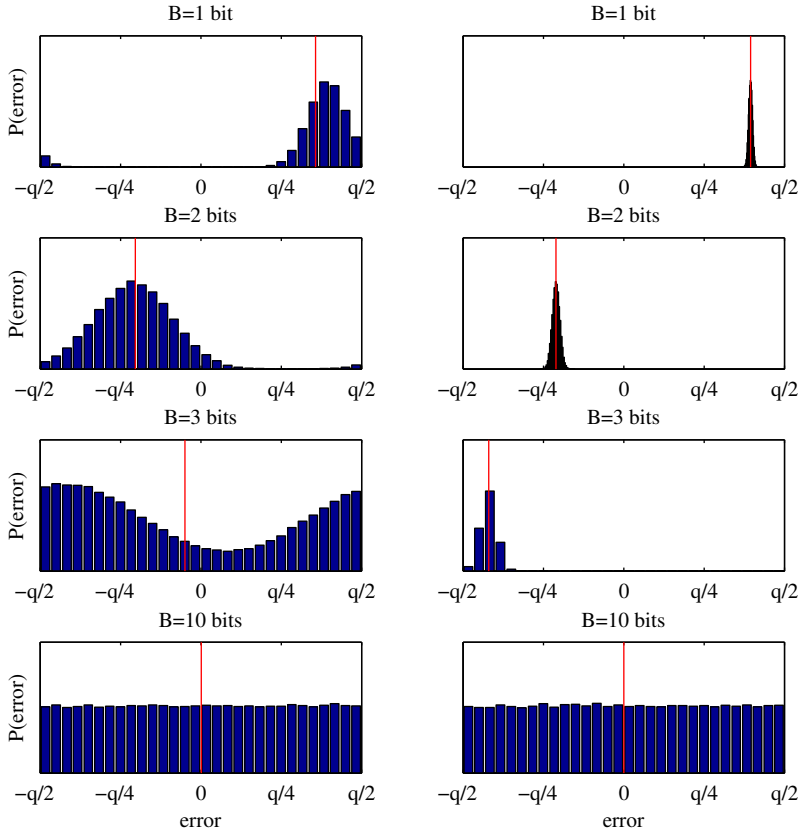


Figure 5.8: Quantization error of $\log(Y)$, where Y is beta-distributed. Error histograms are computed from 10^5 samples. Left: $Y \sim \text{Beta}\left(1 \cdot 10^3, 9 \cdot 10^3\right)$; Right: $Y \sim \text{Beta}\left(1 \cdot 10^5, 9 \cdot 10^5\right)$; in both cases $\mathbb{E}_{\text{P}(Y)}[\log Y] = -2.3026$; q is the quantization interval width; average quantization error (vertical red line).

where

$$A_1 = \{(i, j, \mathbf{h}) : \phi(c, \mathbf{x})^i_{j|\mathbf{h}} = 1\}, \tag{5.37}$$

$$A_2 = \{(i, j, \mathbf{h}) : \phi(c', \mathbf{x})^i_{j|\mathbf{h}} = 1\}, \tag{5.38}$$

$$M^i_{j|\mathbf{h}} = \frac{\alpha^i_{j|\mathbf{h}} + \beta^i_{j|\mathbf{h}} - 1}{\alpha^i_{j|\mathbf{h}} - 1}, \text{ and} \tag{5.39}$$

$$\widetilde{M}^i_{j|\mathbf{h}} = \frac{\alpha^i_{j|\mathbf{h}}}{\alpha^i_{j|\mathbf{h}} + \beta^i_{j|\mathbf{h}}}. \tag{5.40}$$

Before proceeding to the proof, we want to make some comments on the above bound: Typically this bound is not tight and rather conservative. Nevertheless, it allows for simple determination of a worst-case performance, by noting that the term

$$\left[\prod_{(i,j,\mathbf{h}) \in A_1} M^i_{j|\mathbf{h}}\right]\left[\prod_{(i,j,\mathbf{h}) \in A_2} \widetilde{M}^i_{j|\mathbf{h}}\right] \tag{5.41}$$

is constant for different choices of bits used for quantization. Hence, the lower bound in (5.36) can be evaluated by performing parameter learning once, computing the terms $M^i_{j|\mathbf{h}}$ and $\widetilde{M}^i_{j|\mathbf{h}}$ for all samples and classes and then performing a weighting by the exponential function $e^{S(L+1)q}$.

Proof of Theorem 6. The expected CR of a sample belonging to class c having feature instantiation $\mathbf{x}$ using BNC $\mathcal{B}$ with log-parameters $\mathbf{W}$ can be lower bounded by

$$\mathbb{E}\left[\mathbf{1}_{\left(c=h_{\mathrm{P}\mathcal{B}}(\mathbf{x})\right)}\right] = \mathrm{P}\left(\boldsymbol{\phi}(c,\mathbf{x})^T\mathcal{Q}(\mathbf{W}) > \max_{c'\neq c}\boldsymbol{\phi}(c',\mathbf{x})^T\mathcal{Q}(\mathbf{W})\right) \tag{5.42}$$

$$= 1-\mathrm{P}\left(\bigcup_{c'\neq c}\left[\left(\boldsymbol{\phi}(c,\mathbf{x})-\boldsymbol{\phi}(c',\mathbf{x})\right)^T\mathcal{Q}(\mathbf{W})\leq 0\right]\right) \tag{5.43}$$

$$\overset{(a)}{\geq} 1-\min\left\{1,\sum_{c'\neq c}\mathrm{P}\left(\left(\boldsymbol{\phi}(c,\mathbf{x})-\boldsymbol{\phi}(c',\mathbf{x})\right)^T\mathcal{Q}(\mathbf{W})\leq 0\right)\right\} \tag{5.44}$$

$$\overset{(b)}{\geq} 1-\min\left\{1,\sum_{c'\neq c}\mathrm{P}\left(\left(\boldsymbol{\phi}(c,\mathbf{x})-\boldsymbol{\phi}(c',\mathbf{x})\right)^T\frac{\mathbf{W}}{S}\leq q(L+1)\right)\right\}, \tag{5.45}$$

where the expectation is with respect to the distribution of the parameters $\mathbf{W}$. Inequality (a) is by the union bound and (b) is because

$$\left(\boldsymbol{\phi}(c,\mathbf{x})-\boldsymbol{\phi}(c',\mathbf{x})\right)^T\mathcal{Q}(\mathbf{W})\leq 0 \tag{5.46}$$

implies

$$\left(\boldsymbol{\phi}(c,\mathbf{x})-\boldsymbol{\phi}(c',\mathbf{x})\right)^T\frac{\mathbf{W}}{S}\leq 2(L+1)\frac{q}{2} \tag{5.47}$$

assuming the *worst-case quantization error of* $\frac{q}{2}$ for each $\frac{W^i_{j|\mathbf{h}}}{S}$. We can further bound the above expression by determining Chernoff-type bounds on the terms of the form

$$\mathrm{P}\left(\left(\boldsymbol{\phi}(c,\mathbf{x})-\boldsymbol{\phi}(c',\mathbf{x})\right)^T\mathbf{W}/S\leq(L+1)q\right). \tag{5.48}$$

In a first step, for $t>0$ we obtain

$$(5.48) = \mathrm{P}\left(e^{-t\left(\boldsymbol{\phi}(c,\mathbf{x})-\boldsymbol{\phi}(c',\mathbf{x})\right)^T\mathbf{W}}\geq e^{-t(L+1)qS}\right). \tag{5.49}$$

As $t>0$ can be arbitrarily selected,

$$(5.48)\overset{(a)}{\leq}\inf_{t>0}e^{t(L+1)qS}\prod_{(i,j,\mathbf{h})\in A_1}\mathbb{E}_{\mathrm{P}(W^i_{j|\mathbf{h}})}\left[e^{-tW^i_{j|\mathbf{h}}}\right]\prod_{(i,j,\mathbf{h})\in A_2}\mathbb{E}_{\mathrm{P}(W^i_{j|\mathbf{h}})}\left[e^{tW^i_{j|\mathbf{h}}}\right] \tag{5.50}$$

$$\overset{(b)}{\leq} e^{S(L+1)q}\prod_{(i,j,\mathbf{h})\in A_1}\mathbb{E}\left[e^{-W^i_{j|\mathbf{h}}}\right]\prod_{(i,j,\mathbf{h})\in A_2}\mathbb{E}\left[e^{W^i_{j|\mathbf{h}}}\right] \tag{5.51}$$

$$= e^{S(L+1)q}\left[\prod_{(i,j,\mathbf{h})\in A_1}M^i_{j|\mathbf{h}}\right]\left[\prod_{(i,j,\mathbf{h})\in A_2}\widetilde{M}^i_{j|\mathbf{h}}\right], \tag{5.52}$$

where (a) is by Markov's inequality and independence of the parameters[7], and (b) is by arbitrarily selecting $t=1$; $A_1=\{(i,j,\mathbf{h}):\phi(c,\mathbf{x})^i_{j|\mathbf{h}}=1\}$, $A_2=\{(i,j,\mathbf{h}):\phi(c',\mathbf{x})^i_{j|\mathbf{h}}=1\}$. Further, $\widetilde{M}^i_{j|\mathbf{h}}=\frac{\alpha^i_{j|\mathbf{h}}}{\alpha^i_{j|\mathbf{h}}+\beta^i_{j|\mathbf{h}}}$, i.e. the expectation of the beta-distributed RV $e^{W^i_{j|\mathbf{h}}}$. Furthermore, $M^i_{j|\mathbf{h}}$ is computed as follows: The RV $e^{-W^i_{j|\mathbf{h}}}$ is distributed as the RV $Z=1/Y$, where $Y\sim\mathrm{Beta}\left(\alpha^i_{j|\mathbf{h}},\beta^i_{j|\mathbf{h}}\right)$. The density $f_Z(z)$ can be computed [20] as

$$f_Z(z)=\begin{cases}0 & 0\leq z\leq 1,\\ \frac{(1/z)^{\alpha^i_{j|\mathbf{h}}+1}(1-1/z)^{\beta^i_{j|\mathbf{h}}-1}}{\mathrm{B}\left(\alpha^i_{j|\mathbf{h}},\beta^i_{j|\mathbf{h}}\right)} & 1<z.\end{cases} \tag{5.53}$$

[7] The assumed independence does not hold exactly for the RVs in $\mathbf{W}$ corresponding to the class node. Further, it does not hold for nodes independent of the class node. This later case however is uninteresting, as the introduced quantization error affects all classes in the same way.

Hence, the expected value evaluates to

$$M^i_{j|\mathbf{h}} = \mathbb{E}_{f_Z}[Z] \tag{5.54}$$

$$= \int_1^\infty z f_Z(z) \mathrm{d}z \tag{5.55}$$

$$= \frac{1}{\mathrm{B}\left(\alpha^i_{j|\mathbf{h}}, \beta^i_{j|\mathbf{h}}\right)} \frac{\Gamma(\alpha^i_{j|\mathbf{h}} - 1)\Gamma(\beta^i_{j|\mathbf{h}})}{\Gamma(\alpha^i_{j|\mathbf{h}} + \beta^i_{j|\mathbf{h}} - 1)} \tag{5.56}$$

$$= \frac{\alpha^i_{j|\mathbf{h}} + \beta^i_{j|\mathbf{h}} - 1}{\alpha^i_{j|\mathbf{h}} - 1}, \tag{5.57}$$

where $\Gamma(\cdot)$ is the gamma function and where we used $\mathrm{B}(c,d) = \frac{\Gamma(c)\Gamma(d)}{\Gamma(c+d)}$ and the recurrence equation $\Gamma(c) = \frac{\Gamma(c+1)}{c}$.

The final bound is derived by using (5.52) in (5.45), i.e.

$$\mathbb{E}\left[\mathbf{1}_{\left(c=h_{\mathrm{P}_\mathcal{B}}(\mathbf{x})\right)}\right] \geq 1 - \min\left\{1, \sum_{c' \neq c} e^{S(L+1)q} \left[\prod_{(i,j,\mathbf{h}) \in A_1} M^i_{j|\mathbf{h}}\right] \left[\prod_{(i,j,\mathbf{h}) \in A_2} \widetilde{M}^i_{j|\mathbf{h}}\right]\right\}. \tag{5.58}$$

□

Probabilistic Bound

The bound according to Theorem 6 is conservative and is, empirically, often very loose, cf. the experiments in Section 5.3.3. However, we can obtain a tighter bound by considering the stochastic nature of the quantization error. Due to quantization, each of the entries in $\mathbf{W}$ is distorted by a quantization error, i.e.

$$\mathcal{Q}(W^i_{j|\mathbf{h}}) = \frac{W^i_{j|\mathbf{h}}}{S} + E^i_{j|\mathbf{h}}, \tag{5.59}$$

where $E^i_{j|\mathbf{h}}$ is the error introduced by quantization. Note, that the error is a deterministic function of the scaled log-parameters $\frac{1}{S}\mathbf{W}$. Therefore, it depends on both the number of bits B and the scale factor S. The error $E^i_{j|\mathbf{h}}$ can assume values in $[-\frac{q}{2}, +\frac{q}{2}]$, cf. Figure 5.8. Similar to before, the expected classification rate of a sample belonging to class c with feature instantiation $\mathbf{x}$ can be bounded:

Theorem 7 (Probabilistic Bound)**.** *Let $\mathrm{P}^\mathcal{B}$ be the probability distribution defined by a BN, and let S, B, q be as introduced above. Further, let $(c, \mathbf{x})$ be a sample belonging to class c with feature instantiation $\mathbf{x}$. Assuming that the parameters of the BN are independent, the expected classification rate for this sample can be lower bounded as*

$$\mathbb{E}\left[\mathbf{1}_{\left(c=h_{\mathrm{P}_\mathcal{B}}(\mathbf{x})\right)}\right] \geq 1 - \min\left\{1, \sum_{c' \neq c} F^{2(L+1)} \left[\prod_{(i,j,\mathbf{h}) \in A_1} B^i_{j|\mathbf{h}}\right] \left[\prod_{(i,j,\mathbf{h}) \in A_2} \widetilde{B}^i_{j|\mathbf{h}}\right]\right\}, \tag{5.60}$$

where $F = \frac{e^{Sq/2} - e^{-Sq/2}}{Sq}$, where

$$B^i_{j|\mathbf{h}} = \frac{1}{\mathrm{B}\left(\alpha^i_{j|\mathbf{h}}, \beta^i_{j|\mathbf{h}}\right)} \sum_{k=0}^{\infty} e^{Skq} \cdot \left[\mathrm{B}\left(e^{S\min(0, -k\cdot q + q/2)}; \alpha^i_{j|\mathbf{h}}, \beta^i_{j|\mathbf{h}}\right) - \mathrm{B}\left(e^{S(-kq - q/2)}; \alpha^i_{j|\mathbf{h}}, \beta^i_{j|\mathbf{h}}\right)\right], \tag{5.61}$$

$$\widetilde{B}^i_{j|\mathbf{h}} = \frac{1}{\mathrm{B}\left(\alpha^i_{j|\mathbf{h}}, \beta^i_{j|\mathbf{h}}\right)} \sum_{k=0}^{\infty} e^{-Skq} \cdot \left[\mathrm{B}\left(e^{S\min(0, -k\cdot q + q/2)}; \alpha^i_{j|\mathbf{h}}, \beta^i_{j|\mathbf{h}}\right) - \mathrm{B}\left(e^{S(-kq - q/2)}; \alpha^i_{j|\mathbf{h}}, \beta^i_{j|\mathbf{h}}\right)\right], \tag{5.62}$$

and where A_1, A_2 are as in Theorem 6.

Before proceeding to the proof, we want to make a brief remark: For many combinations of α, β and q, the quantities $B^i_{j|\mathbf{h}}$ and $\widetilde{B}^i_{j|\mathbf{h}}$ can be approximated as

$$B^i_{j|\mathbf{h}} \approx \frac{1}{Sq} - \frac{e^{-Sq/2}}{Sq}\frac{\mathrm{B}(\alpha+1,\beta)}{\mathrm{B}(\alpha,\beta)} - \frac{1}{Sq}\left[\frac{\mathrm{B}\left(e^{-Sq/2};\alpha,\beta\right)}{\mathrm{B}(\alpha,\beta)} - e^{Sq/2}\frac{\mathrm{B}\left(e^{-Sq/2};\alpha+1,\beta\right)}{\mathrm{B}(\alpha,\beta)}\right]. \tag{5.63}$$

and

$$\widetilde{B}^i_{j|\mathbf{h}} \approx -\frac{1}{Sq} + \frac{e^{-Sq/2}}{Sq}\frac{\mathrm{B}(\alpha-1,\beta)}{\mathrm{B}(\alpha,\beta)} - \frac{1}{Sq}\left[e^{-Sq/2}\frac{\mathrm{B}\left(e^{-Sq/2};\alpha-1,\beta\right)}{\mathrm{B}(\alpha,\beta)} - \frac{\mathrm{B}\left(e^{-Sq/2};\alpha,\beta\right)}{\mathrm{B}(\alpha,\beta)}\right]. \tag{5.64}$$

Details on these approximations are provided after the proof. In cases in which these approximations are not accurate, Equations (5.61) and (5.62) can be approximated by partial sums with only few terms.

Proof of Theorem 7. The expected classification rate of a sample belonging to class c with feature instantiation $\mathbf{x}$ is given as

$$\mathbb{E}\left[\mathbf{1}_{\left(c=h_{\mathrm{P}\mathcal{B}}(\mathbf{x})\right)}\right] = \mathrm{P}\left(\boldsymbol{\phi}(c,\mathbf{x})^T\mathcal{Q}(\mathbf{W}) > \max_{c'\neq c}\boldsymbol{\phi}(c,\mathbf{x})^T\mathcal{Q}(\mathbf{W})\right) \tag{5.65}$$

$$\geq 1 - \min\left\{1, \sum_{c'\neq c}\mathrm{P}\left(\left(\boldsymbol{\phi}(c,\mathbf{x})-\boldsymbol{\phi}(c',\mathbf{x})\right)^T\mathcal{Q}(\mathbf{W}) \leq 0\right)\right\}, \tag{5.66}$$

where the quantization error is included in $\mathcal{Q}(\mathbf{W})$. Chernoff-type bounds for the terms

$$\mathrm{P}\left(\left(\boldsymbol{\phi}(c,\mathbf{x})-\boldsymbol{\phi}(c',\mathbf{x})\right)^T\mathcal{Q}(\mathbf{W}) \leq 0\right) \tag{5.67}$$

read as

$$(5.67) \leq \left[\prod_{(i,j,\mathbf{h})\in A_1}\mathbb{E}\left[e^{S\mathcal{Q}(W^i_{j|\mathbf{h}})}\right]\right]\left[\prod_{(i,j,\mathbf{h})\in A_2}\mathbb{E}\left[e^{-S\mathcal{Q}(W^i_{j|\mathbf{h}})}\right]\right] \tag{5.68}$$

For ease of notation, let $Y = \log(Z)$ and $Z \sim \mathrm{B}(\alpha,\beta)$. Then, the quantities in (5.68) read as

$$\mathbb{E}\left[e^{\pm S\mathcal{Q}(Y)}\right] = \int_{-\infty}^{0} e^{\pm Sq[y/q]_R} f_Y^S(y)\mathrm{d}y, \tag{5.69}$$

where Y corresponds to $W^i_{j|\mathbf{h}}$. As $e^{\pm Sq[y/q]_R}$ is constant for all y in any fixed quantization interval, we obtain

$$(5.69) = \sum_{k=-\infty}^{0}\int_{kq-q/2}^{\min(0,kq+q/2)} e^{\pm Skq} f_Y^S(y)\mathrm{d}y \tag{5.70}$$

$$\overset{(a)}{=} \sum_{k=0}^{\infty}\frac{e^{\mp Skq}}{\mathrm{B}(\alpha,\beta)}\int_{e^{S(-kq-q/2))}}^{e^{S\min(0,-kq+q/2)}} w^{\alpha-1}(1-w))^{\beta-1}\mathrm{d}w \tag{5.71}$$

$$= \frac{1}{\mathrm{B}(\alpha,\beta)}\sum_{k=0}^{\infty} e^{\mp Skq}\cdot \mathrm{B}(w;\alpha,\beta)\Big|_{e^{S(-kq-q/2)}}^{e^{S\min(0,-kq+q/2)}}, \tag{5.72}$$

where (a) is by substituting $w = e^{Sy}$. Hence, using (5.72) and (5.68) in (5.66) results in the desired bound. □

In the following, we present the derivation of the approximations for the terms $B^i_{j|\mathbf{h}}$ and $\widetilde{B}^i_{j|\mathbf{h}}$ mentioned above.

Approximations. Consider Equation (5.72). For the lower limit and the negative-sign case of the term $e^{\mp Skq}$,

$$\sum_{k=0}^{\infty} e^{-Skq}\mathrm{B}\Big(e^{S(-kq-q/2)};\alpha,\beta\Big) \overset{(a)}{\approx} \int_0^{\infty} e^{-Sxq}\mathrm{B}\Big(e^{-Sqx}e^{-Sq/2};\alpha,\beta\Big)\mathrm{d}x \tag{5.73}$$

$$\overset{(b)}{=} \frac{1}{Sq}\int_0^1 \mathrm{B}\Big(ye^{-Sq/2};\alpha,\beta\Big)\mathrm{d}y \tag{5.74}$$

$$\overset{(c)}{=} \frac{1}{Sq}\left[\mathrm{B}\Big(e^{-Sq/2};\alpha,\beta\Big) - \int_0^1 (ye^{-Sq/2})^{\alpha}(1-ye^{-Sq/2})^{\beta-1}\mathrm{d}y\right] \tag{5.75}$$

$$\overset{(d)}{=} \frac{1}{Sq}\left[\mathrm{B}\Big(e^{-Sq/2};\alpha,\beta\Big) - e^{Sq/2}\int_0^{e^{-Sq/2}} z^{\alpha}(1-z)^{\beta-1}\mathrm{d}z\right] \tag{5.76}$$

$$= \frac{1}{Sq}\Big[\mathrm{B}\Big(e^{-Sq/2};\alpha,\beta\Big) - e^{Sq/2}\mathrm{B}\Big(e^{-Sq/2};\alpha+1,\beta\Big)\Big], \tag{5.77}$$

where (a) is by approximating the sum by an integral, (b) by substituting $y = e^{-Sqx}$, (c) by integration by parts, and (d) by substituting $z = ye^{-Sq/2}$. Thus,

$$\frac{1}{\mathrm{B}(\alpha,\beta)}\sum_{k=0}^{\infty} e^{-Skq}\mathrm{B}\Big(e^{S(-2k-1)q/2};\alpha,\beta\Big) \approx \frac{1}{Sq}\left[\frac{\mathrm{B}\Big(e^{-Sq/2};\alpha,\beta\Big)}{\mathrm{B}(\alpha,\beta)} - e^{Sq/2}\frac{\mathrm{B}\Big(e^{-Sq/2};\alpha+1,\beta\Big)}{\mathrm{B}(\alpha,\beta)}\right]. \tag{5.78}$$

Similarly, for the upper limit in (5.72),

$$\frac{1}{\mathrm{B}(\alpha,\beta)}\sum_{k=0}^{\infty} e^{-Skq}\mathrm{B}\Big(\min(1,e^{S(-2k+1)q/2});\alpha,\beta\Big) \approx \frac{1}{Sq} - \frac{e^{-Sq/2}}{Sq}\frac{\mathrm{B}(\alpha+1,\beta)}{\mathrm{B}(\alpha,\beta)}. \tag{5.79}$$

Hence, using (5.77) and (5.79) in (5.72) we obtain

$$B^i_{j|\mathbf{h}} \approx \frac{1}{Sq} - \frac{e^{-Sq/2}}{Sq}\frac{\mathrm{B}(\alpha+1,\beta)}{\mathrm{B}(\alpha,\beta)} - \frac{1}{Sq}\left[\frac{\mathrm{B}\Big(e^{-Sq/2};\alpha,\beta\Big)}{\mathrm{B}(\alpha,\beta)} - e^{Sq/2}\frac{\mathrm{B}\Big(e^{-Sq/2};\alpha+1,\beta\Big)}{\mathrm{B}(\alpha,\beta)}\right]. \tag{5.80}$$

Analogously, we obtain

$$\widetilde{B}^i_{j|\mathbf{h}} \approx -\frac{1}{Sq} + \frac{e^{-Sq/2}}{Sq}\frac{\mathrm{B}(\alpha-1,\beta)}{\mathrm{B}(\alpha,\beta)} - \frac{1}{Sq}\left[e^{-Sq/2}\frac{\mathrm{B}\Big(e^{-Sq/2};\alpha-1,\beta\Big)}{\mathrm{B}(\alpha,\beta)} - \frac{\mathrm{B}\Big(e^{-Sq/2};\alpha,\beta\Big)}{\mathrm{B}(\alpha,\beta)}\right]. \tag{5.81}$$

Probabilistic Bound Assuming Uniform and Independent Quantization Errors

In the following, to emphasize the need for considering an accurate model of the quantization error in (5.59), we determine CR bounds by assuming that the quantization error is uniform and independent of $\mathbf{W}$. This assumption is common although often inappropriate when analyzing quantization effects. In experiments, cf. Section 5.3.3, the resulting bounds are looser than the ones in Theorem 7. The bound can be stated as follows:

Theorem 8 (Uniform and Independent Error Bound)**.** *Let $\mathrm{P}^{\mathcal{B}}$ be the probability distribution defined by a BN, and let S, B, q be as introduced. Further, let $(c,\mathbf{x})$ be a sample belonging to class c with feature instantiation $\mathbf{x}$. Assuming that the parameters of the BN are independent and that the quantization errors of the parameters are independent and uniformly distributed within a quantization interval, the expected classification rate for this sample can be lower bounded as*

$$\mathbb{E}\left[\mathbf{1}_{\left(c=h_{\mathrm{P}^{\mathcal{B}}}(\mathbf{x})\right)}\right] \geq 1 - \min\left\{1, \sum_{c'\neq c} F^{2(L+1)}\left[\prod_{(i,j,\mathbf{h})\in A_1} M^i_{j|\mathbf{h}}\right]\left[\prod_{(i,j,\mathbf{h})\in A_2} \widetilde{M}^i_{j|\mathbf{h}}\right]\right\}, \tag{5.82}$$

where $F = \frac{e^{Sq/2}-e^{-Sq/2}}{Sq}$, and A_1, A_2, $M^i_{j|\mathbf{h}}$, $\widetilde{M}^i_{j|\mathbf{h}}$ are as in Theorem 6.

Proof. Using B bits for quantization, $E^i_{j|\mathbf{h}} \sim U(\pm\frac{q}{2})$, where $U(\pm a)$ denotes a uniform distribution on the interval $[-a,+a]$. Hence,

$$\mathbb{E}_{E^i_{j|\mathbf{h}}\sim U(\pm\frac{q}{2})}\left[e^{SE^i_{j|\mathbf{h}}}\right] = \mathbb{E}_{E^i_{j|\mathbf{h}}\sim U(\pm\frac{q}{2})}\left[e^{-SE^i_{j|\mathbf{h}}}\right] \tag{5.83}$$

$$= \int_{-q/2}^{q/2} e^{Sx}\frac{1}{q}\mathrm{d}x = \frac{e^{Sq/2}-e^{-Sq/2}}{Sq}. \tag{5.84}$$

Thus, similar as in the proof of Theorem 7, we obtain

$$\mathrm{P}\left(\left(\boldsymbol{\phi}(c,\mathbf{x})-\boldsymbol{\phi}(c',\mathbf{x})\right)^T\left(\mathbf{W}/S+\mathbf{E}\right)\leq 0\right) \leq \prod_{(i,j,\mathbf{h})\in A_1}\mathbb{E}\left[e^{-W^i_{j|\mathbf{h}}}\right]\mathbb{E}\left[e^{-SE^i_{j|\mathbf{h}}}\right]\prod_{(i,j,\mathbf{h})\in A_2}\mathbb{E}\left[e^{W^i_{j|\mathbf{h}}}\right]\mathbb{E}\left[e^{SE^i_{j|\mathbf{h}}}\right]. \tag{5.85}$$

Consequently,

$$\sum_{c'\neq c}\mathrm{P}\left(\left(\boldsymbol{\phi}(c,\mathbf{x})-\boldsymbol{\phi}(c',\mathbf{x})\right)^T\left(\mathbf{W}/S+\mathbf{E}\right)\leq 0\right) \leq \sum_{c'\neq c^{(n)}} F^{2(L+1)}\left[\prod_{(i,j,\mathbf{h})\in A_1} M^i_{j|\mathbf{h}}\right]\left[\prod_{(i,j,\mathbf{h})\in A_2}\widetilde{M}^i_{j|\mathbf{h}}\right], \tag{5.86}$$

where $F = \frac{e^{Sq/2}-e^{-Sq/2}}{Sq}$. □

5.3.3 Experiments

The derived bounds are evaluated on real-world datasets. Additionally, we compare generatively and discriminatively optimized BNCs with respect to *classification performance* and *robustness* against parameter quantization. The investigation of classification performance compares the absolute CRs of generatively and discriminatively optimized BNCs with respect to parameter quantization, while the investigation of robustness compares the number of changing classifications due to parameter quantization of these two types of BNCs.

In all experiments, we select the scale factor S as the minimum value such that all parameters can be represented using B bits, cf. Section 5.3.2. We perform experiments for USPS, MNIST and DC-Mall data, cf. Section 2.3.

Classification Performance Bounds

We evaluate the bounds derived in Section 5.3.2. For all datasets, we perform Bayesian parameter estimation, where we assume a uniform parameter prior, i.e. $\widetilde{\alpha}^i_{j|\mathbf{h}} = 1$. For $B \in \{1,\ldots,10\}$ bits we determine the scale factor as stated above and quantize the MAP parameters. BNCs using the resulting reduced-precision parameters are evaluated by computing the CR performance on the test set. Additionally, the bounds on the CR derived in the previous section are computed. As reference, we also determine the CR performance of BNCs using double-precision MAP parameters. Furthermore, we compare BNCs with NB and generatively optimized TAN structures. These TAN structures are learned using the conditional mutual information (TAN-CMI) criterion [6]. We did not consider more general structures, e.g. 2-trees for augmenting NB, because (1) there is no significant increase in classification performance on several standard benchmarks for these structures [23], (2) inference complexity scales with the tree-width of the corresponding moralized graph, i.e. computational complexity increases drastically which conflicts with our interest for computationally highly efficient models, and (3) more general models have more parameters to be estimated (from the same number of samples) and therefore parameter estimates have a higher variance.

Experimental results are shown in Figure 5.9. CR performance of the reduced-precision BNCs increases with the number of bits and is close to optimal when using 3 to 4 bits for

Table 5.3: Number of parameters of BNCs for different datasets and structures.

Dataset	Structure	#Parameters	Samples/Parameter
USPS	NB	8650	0.92
	TAN-CMI	33040	0.24
	TAN-MM	31320	0.26
MNIST	NB	6720	8.93
	TAN-CMI	38350	1.56
	TAN-MM	39370	1.52
DC-Mall	NB	21490	12.19
	TAN-CMI	574406	0.46
	TAN-MM	484813	0.54

all datasets[8]. The worst-case bounds are conservative and rather loose. In contrast, the probabilistic bounds are much tighter. The bounds derived assuming uniform and independent quantization errors, cf. Theorem 8, are less tight than those obtained by assuming beta distributed parameters, cf. Theorem 7.

While BNCs with TAN structures typically achieve better CRs, the determined bounds for TAN structures are looser than the bounds for NB structures. This is because in the case of TAN structures, less samples are available for estimating the CPTs, cf. Table 5.3. Therefore, parameter estimates are more uncertain and quantization effects can have larger impact. This effect is strongest for the USPS dataset which consists only of a small number of training samples, i.e. the number of samples per parameter, as shown in Table 5.3, is low.

To emphasize the dependence of the bounds on the sample size, we performed the following experiment: For MNIST data, we trained BNCs with TAN-CMI structure on 10%, 20%, 50%, and 100% of the samples and computed the performance bounds, respectively. The results are shown in Figure 5.10. With increasing sample size, the bounds become tighter because the variance of the parameter estimates reduces.

Classification Performance of BNCs Optimized for Large Margin

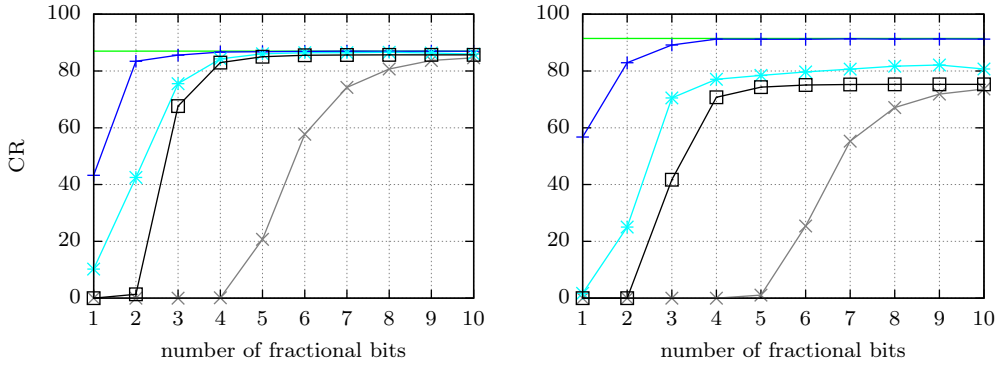
We compare the classification performance of BNCs with generatively and discriminatively optimized parameters/structure with respect to parameter quantization. One motivation for this is that determining whether parameter quantization changes the decision of a BNC $\mathcal{B}$ for a sample belonging to class c with feature instantiation $\mathbf{x}$ or not, is equivalent to determining whether the error introduced by quantization is larger than its log margin, i.e.

$$\log d^{\mathcal{B}}(c,\mathbf{x}) = \log \mathrm{P}^{\mathcal{B}}(c,\mathbf{x}) - \max_{c' \neq c} \log \mathrm{P}^{\mathcal{B}}(c',\mathbf{x}). \tag{5.87}$$

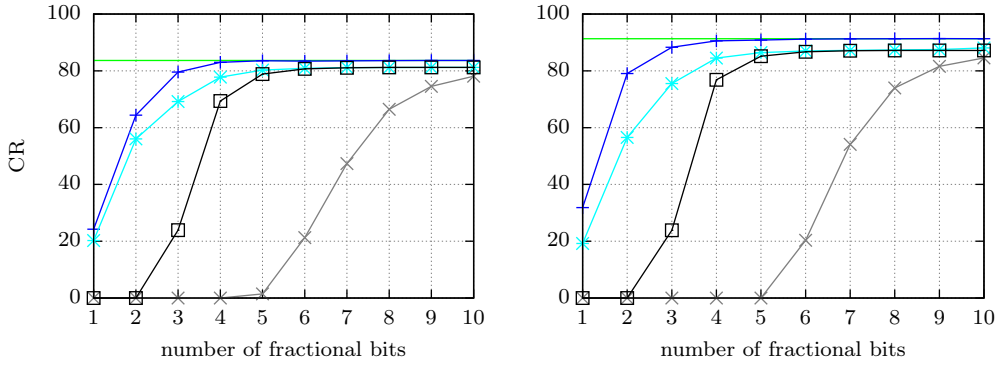
Maximizing this margin is essentially the objective of large margin training of BNCs [25, 8, 22, 24]. Hence, the assumption that BNCs optimized for a large margin are more *robust* to parameter quantization than for other BNC learning approaches is obtruding (this assumption is also supported by experimental results presented in [33, 34]). In the following, we compare two different approaches for obtaining large margin BNCs.

Discriminatively versus generatively optimized parameters. We compare the classification performance of BNCs with MAP parameters and of BNCs with MM parameters over varying numbers of bits used for quantization. MM parameters are determined using the algorithm described in [25]. The structures considered are NB and TAN-CMI. The results for USPS, MNIST and DC-Mall data are shown in Figure 5.11. Our hypothesis is that the

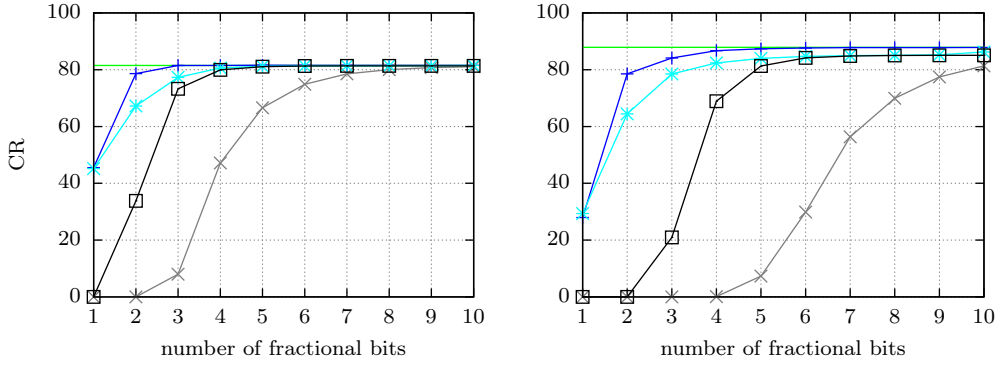
[8] Classification requires adding up $L+1$ reduced-precision parameters and performing a maximum operation. Hence, additional $\log_2(L+1)$ bits are required to avoid an overflow during summation. For easier comparison across different datasets, these additional bits are not considered in the presented figures.



(a) MAP parameters, USPS, left: NB, right: TAN-CMI



(b) MAP parameters, MNIST, left: NB, right: TAN-CMI



(c) MAP parameters, DC-Mall, left: NB, right: TAN-CMI

Figure 5.9: CRs and bounds on the CRs of BNCs with reduced-precision parameters for varying number of bits; worst-case bounds (grey ×), probabilistic bound (cyan ∗), probabilistic bound assuming uniform quantization error (black □), reduced-precision MAP parameters (blue +), full-precision MAP parameters (green, no marker).

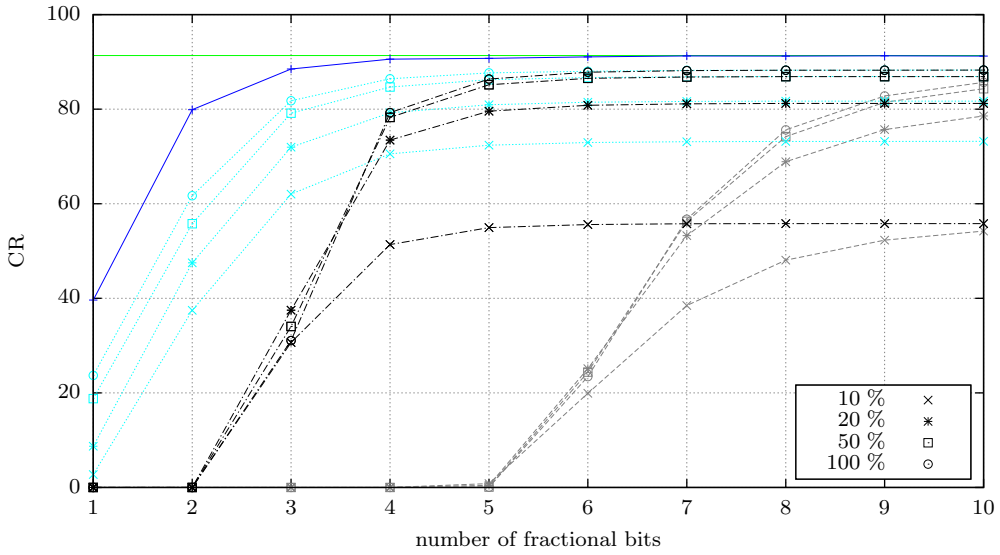


Figure 5.10: Sample size dependence of the proposed performance bounds for MNIST data and BNCs with TAN-CMI structure; worst-case bounds (dashed grey), probabilistic bounds (dotted cyan), probabilistic bounds assuming uniform quantization error (dash-dotted black), reduced-precision MAP parameters (solid blue), full-precision MAP parameters (green, horizontal); bounds are learned on 10%, 20%, 50% and 100% of the training data (from bottom to top).

Table 5.4: Comparison of the CRs of BNCs with MAP and MM parameters; a plus (minus) sign indicates that for the corresponding dataset/structure/number of bits BNCs with MM (MAP) parameters have a significantly higher CR.

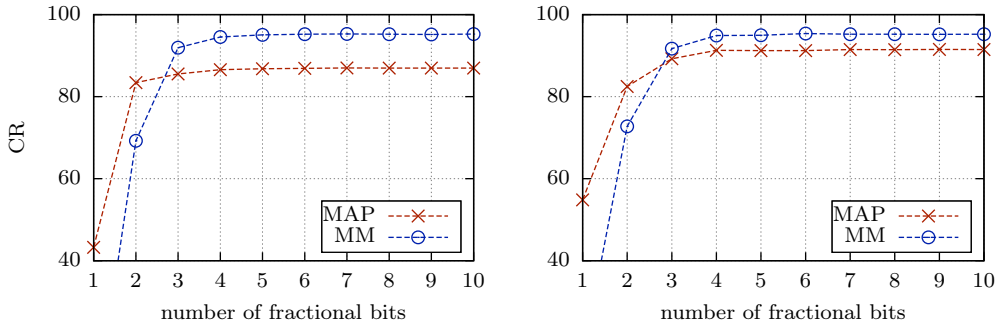
		number of bits									
Dataset	Structure	1	2	3	4	5	6	7	8	9	10
USPS	NB	−	−	+	+	+	+	+	+	+	+
	TAN-CMI	−	−	+	+	+	+	+	+	+	+
MNIST	NB	−	−	+	+	+	+	+	+	+	+
	TAN-CMI	−	−	+	+	+	+	+	+	+	+
DC-Mall	NB	−	+	−	+	+	+	+	+	+	+
	TAN-CMI	−	−	+	+	+	+	+	+	+	+

classification rate of BNCs with MM parameters is higher than that of BNCs with MAP parameters, i.e.

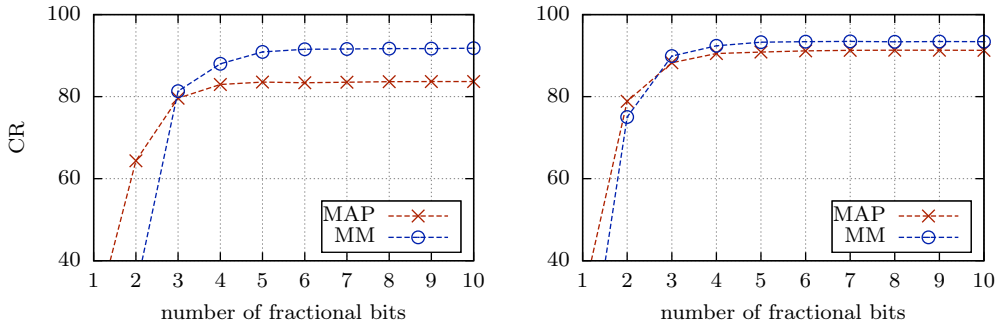
$$\mathrm{CR}\Big(h_{\mathcal{Q}(\mathbf{w}^{\mathrm{MM}})}\Big) \geq \mathrm{CR}\Big(h_{\mathcal{Q}(\mathbf{w}^{\mathrm{MAP}})}\Big), \tag{5.88}$$

where, in abuse of notation, $h_{\mathcal{Q}(\mathbf{w})}$ is the BNC induced by the quantized parameters $\mathcal{Q}(\mathbf{w})$. We use a one-tailed dependent t-test for paired samples at significance level 0.01 for testing significance. Results are summarized in Table 5.4. For all but small bit-widths and all datasets and structures, discriminatively optimized parameters yield significantly higher CRs.

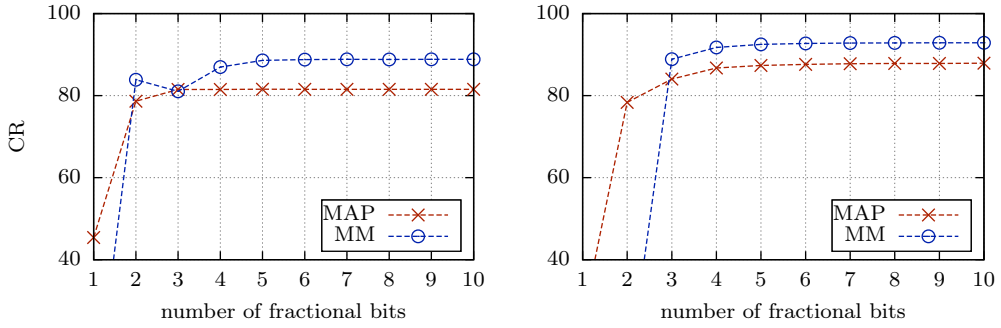
Discriminatively optimized BN structures. We compare the CR performance of BNCs with MAP parameters using TAN-CMI and margin-optimized TAN structures (TAN-MM). TAN-MM structures are determined using the margin objective (5.21) embedded in a hinge loss function as objective for scoring BN structures. Optimization is performed using a greedy hill climbing heuristic. Details are provided in [24]. CR results for MNIST and DC-Mall data are shown in Figure 5.12. Formally, our hypothesis is that the classification rate



(a) USPS, NB, left: NB, right: TAN-CMI



(b) MNIST, left: NB, right: TAN-CMI



(c) DC-Mall, left: NB, right: TAN-CMI

Figure 5.11: CRs of BNCs with MAP and MM parameters over varying number of bits; MAP parameters (red); MM parameters (blue).

of BNCs with TAN-MM structure is higher than that of BNCs with TAN-CMI structures, i.e.

$$\mathrm{CR}\Big(h_{\mathcal{Q}(\mathbf{w}^{\text{TAN-MM}})}\Big) \geq \mathrm{CR}\Big(h_{\mathcal{Q}(\mathbf{w}^{\text{TAN-CMI}})}\Big), \tag{5.89}$$

where $\mathbf{w}^{\text{TAN-MM}}$ denotes the MAP parameters for the TAN-MM structure and $\mathbf{w}^{\text{TAN-CMI}}$ denotes the MAP parameters for the TAN-CMI structure, respectively. We performed the same statistical tests as above. In all cases, i.e. for all datasets and considered bit-widths (1 to 10 bits), the CR of BNCs with TAN-MM structure is significantly higher.

Robustness of BNCs Optimized for Large Margin

We compare the robustness of BNCs with generatively and discriminatively optimized parameters and structure with respect to parameter quantization. We denote a classifier as robust if only a small number of classifications change due to parameter quantization —

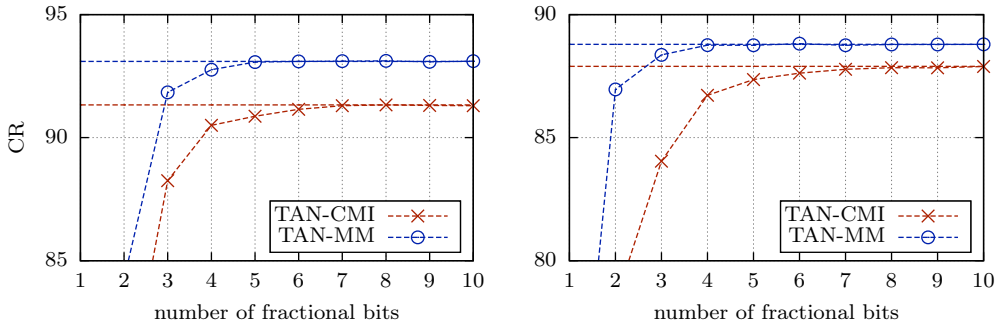


Figure 5.12: CRs of BNCs with MAP parameters over varying number of bits, TAN-CMI and TAN-MM structures; Left: MNIST, Right: DC-Mall.

Table 5.5: Comparison of the robustness of BNCs with MAP and MM parameters; a plus (minus) sign indicates that for the corresponding dataset/structure/number of bits BNCs with MM (MAP) parameters are significantly more robust.

		number of bits									
Dataset	Structure	1	2	3	4	5	6	7	8	9	10
USPS	NB	–	–								
	TAN-CMI	–	–				+				
MNIST	NB	–	–	–	–	–	–				
	TAN-CMI	–	–								
DC-Mall	NB	–	–	–	–	–	–	–	–	–	–
	TAN-CMI	–	–	+	+	+	+	+	+	+	

this is formalized below. The assumption, that BNCs with parameters/structures optimized for a large margin are more robust than other BNCs seems likely (this assumption is also supported by experimental results presented in [33, 34]). However, this higher robustness cannot be observed empirically in all cases. Again, we compare two different approaches for obtaining large margin BNCs.

Discriminatively versus generatively optimized parameters. Our hypothesis for testing robustness is

$$\mathbb{E}\left[\mathbf{1}_{\left(h_{\mathbf{w}^{\text{MM}}}(\mathbf{X})=h_{\mathcal{Q}(\mathbf{w}^{\text{MM}})}(\mathbf{X})\right)}\right] \geq \mathbb{E}\left[\mathbf{1}_{\left(h_{\mathbf{w}^{\text{MAP}}}(\mathbf{X})=h_{\mathcal{Q}(\mathbf{w}^{\text{MAP}})}(\mathbf{X})\right)}\right]. \tag{5.90}$$

Significance of results is assessed using a dependent t-test for paired samples at significance level 0.01. Then, BNCs with discriminatively optimized parameters are almost never significantly more robust to parameter quantization, cf. Table 5.5 (the only exception is BNCs with TAN-CMI structure for DC-Mall data). This can be explained with results from sensitivity analysis — discriminatively optimized BNCs have more extreme parameters, i.e. parameters close to zero or one, than generatively optimized BNCs [33]. Nevertheless, using only 3 to 4 bits, the discriminatively optimized parameters yield higher absolute CRs, cf. Section 5.3.3 and Table 5.4.

Discriminatively optimized BN structures. Our hypothesis is that BNCs with large margin structures are more robust to quantization, i.e.

$$\mathbb{E}\left[\mathbf{1}_{\left(h_{\mathbf{w}^{\text{TAN-MM}}}(\mathbf{X})=h_{\mathcal{Q}(\mathbf{w}^{\text{TAN-MM}})}(\mathbf{X})\right)}\right] \geq \mathbb{E}\left[\mathbf{1}_{\left(h_{\mathbf{w}^{\text{TAN-CMI}}}(\mathbf{X})=h_{\mathcal{Q}(\mathbf{w}^{\text{TAN-CMI}})}(\mathbf{X})\right)}\right]. \tag{5.91}$$

For assessing results, we used the same statistical test as above. BNCs with structures optimized for a large margin show a higher robustness to parameter quantization than TAN-CMI structures, cf. Table 5.6. Additionally, the CR performance using margin-optimized

Table 5.6: Comparison of the robustness of BNCs with TAN-CMI and TAN-MM structures and MAP parameters; a plus sign indicates that for the corresponding dataset/number of bits BNCs with TAN-MM structure are significantly more robust.

	number of bits									
Dataset	1	2	3	4	5	6	7	8	9	10
USPS	+	+	+	+	+					
MNIST	+	+	+	+	+					
DC-Mall	+	+	+	+	+	+	+	+	+	+

structures is always better, cf. Section 5.3.3 and Figure 5.12. Hence, in this case the large margin structures seem favorable.

5.3.4 Summary

We considered BNCs with reduced-precision parameters and derived an easy to evaluate worst-case bound on the CR performance. Furthermore, a probabilistic bound on the CR and approximations for the expected value of the quantized parameters were derived. In experiments, we evaluated the performance of reduced-precision BNCs and the derived bounds. Only 3 to 4 bits for representing each parameter are necessary to achieve CR performance close to double-precision floating-point performance. We investigated classification performance and robustness of BNCs with generatively and discriminatively optimized parameters and structures with respect to parameter quantization. While discriminatively optimized parameters do not show higher robustness than generatively optimized parameters, the CR performance of BNCs in the former case is already better when using only 2 to 3 bits for representing each parameter. When using discriminatively optimized TAN-MM structures, robustness to parameter quantization as well as CR performance is increased.

5.4 Learning Reduced-Precision Parameters With Full-Precision Arithmetic[9]

In this section, we consider learning reduced-precision parameters for BNCs using full-precision computations. In contrast to before, parameters are not initially learned in full-precision and then rounded, but parameters are optimized over the space of reduced-precision parameters. We claim that learning reduced-precision parameters by explicitly considering the reduced-precision constraints of the destination platform is advantageous. Furthermore, we argue that discriminatively optimized BNCs achieve a good trade-off between accuracy and required computational resources. This is motivated in Figure 5.13 for the *satimage* dataset from the UCI repository [2]. The model 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 (RD) and that of parameters optimized for resource constraint environments using branch and bound (BB) techniques is presented — details on parameter learning using the hybrid generative-discriminative objective from Chapter 4 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 better. 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 BNs using BB are the best choice.

[9] Published as *Integer Bayesian Network Classifiers* [35] in the proceedings of ECML 2014.

Note, that 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.[10] Furthermore, when memory consumption is an issue, non-linear SVMs are disadvantageous because all support-vectors must be stored for classification.

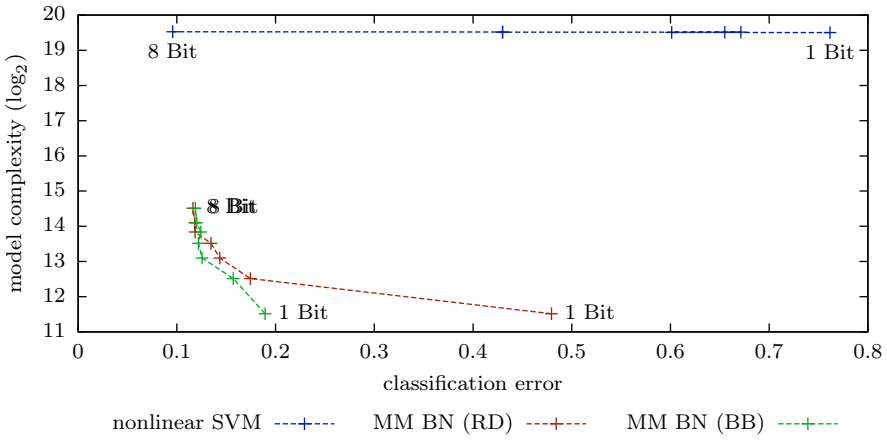


Figure 5.13: Model complexities versus classification errors. *Nonlinear SVM* refers to SVMs with radial-basis-function kernels, *MM BN (RD)* refers to MM BNs with parameters obtained by rounding, and *MM BN (BB)* refers to MM BNs with parameters obtained by the method proposed in this section.

In this section, we devise algorithms for efficiently learning high performance low complexity models. While we already showed in Section 5.2 that parameters of BNCs can be mapped to the integer domain without considerable loss in CR performance, we take the analysis further: A principled approach for BNC parameter learning of margin-maximizing parameters over a discrete search space, i.e. MM (BB) 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 MM parameters.

Our main contributions can be summarized as follows:

- An efficient algorithm for computing margin maximizing reduced-precision parameters. The algorithm is based on the BB algorithm and a set of greedy heuristics. This offers a gain in computation time and makes learning tractable.
- Experiments demonstrating that reduced-precision BNs with small bit-widths can be widely applied. We especially show that a very low number of reduced-precision bits is often sufficient to obtain classification performance close to full-precision MM BNs. 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 part of the thesis is structured as follows: In Section 5.4.1, an efficient algorithm for learning margin maximizing reduced-precision parameters of BNCs is provided. Section 5.4.2 considers reduced-precision parameter learning by rounding from a theoretical perspective. Experimental results are provided in Section 5.4.3. We conclude our investigations in Section 5.4.4.

[10] 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 NB or 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.

5.4.1 Bayesian Network Classifiers with Reduced-Precision Parameters

In this section, we review how to represent reduced-precision parameters of BNCs by integer parameters and present an algorithm for determining margin maximizing reduced-precision parameters for BNCs.

Integer Representation of BNCs With Reduced-Precision Parameters

According to (2.5), the BN $\mathcal{B}$ assigns the probability

$$\mathrm{P}^{\mathcal{B}}(\mathbf{x}) = \prod_{i=0}^{L} \mathrm{P}(X_i = \mathbf{x}(X_i) | \mathbf{Pa}(X_i) = \mathbf{x}(\mathbf{Pa}(X_i))), \tag{5.92}$$

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.

$$\log \mathrm{P}^{\mathcal{B}}(\mathbf{x}) = \sum_{i=0}^{L} \log \mathrm{P}(X_i = \mathbf{x}(X_i) | \mathbf{Pa}(X_i) = \mathbf{x}(\mathbf{Pa}(X_i))). \tag{5.93}$$

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

$$w^i_{j|\mathbf{h}} = \log \mathrm{P}(X_i = j | \mathbf{Pa}(X_i) = \mathbf{h}) = -\sum_{k=-b_f}^{b_i-1} b^{i,k}_{j|\mathbf{h}} \cdot 2^k, \tag{5.94}$$

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, ignoring the possibility of underflows, all $w^i_{j|\mathbf{h}}$ are in the set of negative fixed-point numbers $-\mathbb{B}^{b_i}_{b_f}$ with b_i integer bits and b_f fractional bits, i.e.

$$w^i_{j|\mathbf{h}} \in -\mathbb{B}^{b_i}_{b_f} = -\left\{ \sum_{k=-b_f}^{b_i-1} d_k \cdot 2^k : d_k \in \{0,1\} \right\}. \tag{5.95}$$

Introducing this in (5.93) and scaling by 2^{b_f} results in

$$2^{b_f} \log \mathrm{P}^{\mathcal{B}}(\mathbf{x}) = -\sum_{i=0}^{L} \sum_{k=-b_f}^{b_i-1} b^{i,k}_{\mathbf{x}(X_i)|\mathbf{x}(\mathbf{Pa}(X_i))} \cdot 2^{k+b_f}, \tag{5.96}$$

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

$$\log_2(L+1) + b_i + b_f \tag{5.97}$$

bits are required to calculate the joint probability. This transformation to the integer domain, as already shown in Section 5.2.3, 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.

Learning Reduced-Precision BNCs Using Full-Precision Arithmetic

In principle, parameters for BNCs with reduced-precision parameters can be determined by first learning BNC 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 (5.98) and we aim at a more principled approach.

We advocate the hybrid generative-discriminative parameter learning according to [22], cf. Chapter 4. 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

$$\mathcal{P}_{\mathcal{G}}^{\mathrm{MM}} = \arg\max_{\mathcal{P}_{\mathcal{G}}} \left[\sum_{n=1}^{N} \log \mathrm{P}^{\mathcal{B}}(c^{(n)}, \mathbf{x}^{(n)}) + \lambda \sum_{n=1}^{N} \min\left(\gamma, \log \mathrm{P}^{\mathcal{B}}(c^{(n)}, \mathbf{x}^{(n)}) - \max_{c \neq c^{(n)}} \log \mathrm{P}^{\mathcal{B}}(c, \mathbf{x}^{(n)}) \right) \right], \tag{5.98}$$

where $\mathrm{P}^{\mathcal{B}}(C, \mathbf{X})$ is the joint distribution in (2.5) induced by the BN $(\mathcal{G}, \mathcal{P}_{\mathcal{G}})$, λ is a trade-off parameter between likelihood and margin, i.e. generative and discriminative optimization, and γ is 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 alternative class, i.e. $\log \mathrm{P}^{\mathcal{B}}(c^{(n)}, \mathbf{x}^{(n)}) - \max_{c \neq c^{(n)}} \log \mathrm{P}^{\mathcal{B}}(c, \mathbf{x}^{(n)})$. Consequently, a sample is classified correctly iff it has positive margin and incorrectly otherwise. Both parameters λ and γ are typically set using cross-validation.

Our approach is based on the BB procedure [14], exploiting convexity of (5.98) under suitable parametrization. The implications will become clear immediately. Optimization of the MM criterion can be represented as

$$\begin{aligned} \underset{\mathbf{w}}{\text{maximize}} \quad & \sum_{n=1}^{N} \boldsymbol{\phi}(c^{(n)}, \mathbf{x}^{(n)})^T \mathbf{w} + \lambda \sum_{n=1}^{N} \min\left(\gamma, \boldsymbol{\phi}(c^{(n)}, \mathbf{x}^{(n)})^T \mathbf{w} - \max_{c \neq c^{(n)}} \boldsymbol{\phi}(c, \mathbf{x}^{(n)})^T \mathbf{w} \right) \\ \text{s.t.} \quad & \sum_{j=1}^{|\mathbf{val}(X_i)|} \exp(w^i_{j|\mathbf{h}}) = 1 \qquad \forall i, \mathbf{h}, \end{aligned} \tag{5.99}$$

where we exploit that any log-probability $\log \mathrm{P}^{\mathcal{B}}(c, \mathbf{x})$ can be written as

$$\log \mathrm{P}^{\mathcal{B}}(c, \mathbf{x}) = \boldsymbol{\phi}(c, \mathbf{x})^T \mathbf{w}, \tag{5.100}$$

cf. Section 2.2.1. The above problem in (5.99) is nonconvex and hard to solve. However, when relaxing the normalization constraints to

$$\sum_{j=1}^{|\mathbf{val}(X_i)|} \exp(w^i_{j|\mathbf{h}}) \leq 1, \tag{5.101}$$

the problem becomes convex and can hence be solved efficiently. If all components of $\sum_{n=1}^{N} \boldsymbol{\phi}(c^{(n)}, \mathbf{x}^{(n)})$ are positive, e.g. when applying Laplace smoothing, then (5.101) is automatically satisfied with equality by any optimal solution of the relaxed problem, i.e. the original constraints are recovered [22].

For learning reduced-precision parameters, we restrict the parameters $\mathbf{w}$ to $-\mathbb{B}_{b_f}^{b_i}$ and further relax the normalization constraints to

$$\sum_{j=1}^{|\mathbf{val}(X_i)|} \exp(w^i_{j|\mathbf{h}}) \leq 1 + \xi(|\mathbf{val}(X_i)|, b_i, b_f), \qquad \forall i, \mathbf{h} \tag{5.102}$$

where $\xi(|\mathbf{val}(X_i)|, b_i, b_f)$ is an additive constant depending on $|\mathbf{val}(X_i)|$, b_i and b_f. 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 required, 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 additional constant, our optimization problem would be infeasible. In our experiments, we set

$$\xi(|\mathbf{val}(X_i)|, b_i, b_f) = \max\left(0, \frac{|\mathbf{val}(X_i)|}{2}[\exp(M) + \exp(M + 2^{-b_f})] - 1\right), \tag{5.103}$$

allowing at least half of the parameters of every CPD $\mathrm{P}(X_i|\mathbf{Pa}(X_i))$ to take on values larger than M. Note that $\xi(|\mathbf{val}(X_i)|, b_i, b_f)$ quickly goes down to zero with increasing b_i. Thus, our final optimization problem for learning reduced-precision parameters using full-precision arithmetic is

$$\begin{aligned}
\underset{\mathbf{w}}{\text{maximize}} \quad & \sum_{n=1}^{N} \boldsymbol{\phi}(c^{(n)}, \mathbf{x}^{(n)})^T \mathbf{w} + \lambda \sum_{n=1}^{N} \min\left(\gamma, \boldsymbol{\phi}(c^{(n)}, \mathbf{x}^{(n)})^T \mathbf{w} - \max_{c \neq c^{(n)}} \boldsymbol{\phi}(c, \mathbf{x}^{(n)})^T \mathbf{w}\right) \\
\text{s.t.} \quad & \sum_{j=1}^{|\mathbf{val}(X_i)|} \exp(w^i_{j|\mathbf{h}}) \leq 1 + \xi(|\mathbf{val}(X_i)|, b_i, b_f) \qquad \forall i, \mathbf{h}, \\
& w^i_{j|\mathbf{h}} \in -\mathbb{B}^{b_i}_{b_f} \qquad \forall i, j, \mathbf{h},
\end{aligned} \tag{5.104}$$

where we additionally include the quantization constraint $w^i_{j|\mathbf{h}} \in -\mathbb{B}^{b_i}_{b_f}$. For efficiently finding (global) minimizers of (5.104), we propose to use a BB algorithm [11] and greedy heuristics for creating candidate solutions and branching orders:

Branch and Bound Algorithm. The optimal BNC 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 (5.104) using upper and lower bounds for $w^i_{j|\mathbf{h}}$ depending on the considered leaf of the search tree, i.e. the subproblem corresponding to the k^{th} leaf is given as

$$\begin{aligned}
\underset{\mathbf{w}}{\text{maximize}} \quad & \sum_{n=1}^{N} \boldsymbol{\phi}(c^{(n)}, \mathbf{x}^{(n)})^T \mathbf{w} + \lambda \sum_{n=1}^{N} \min\left(\gamma, \boldsymbol{\phi}(c^{(n)}, \mathbf{x}^{(n)})^T \mathbf{w} - \max_{c \neq c^{(n)}} \boldsymbol{\phi}(c, \mathbf{x}^{(n)})^T \mathbf{w}\right) \\
\text{s.t.} \quad & \sum_{j=1}^{|\mathbf{val}(X_i)|} \exp(w^i_{j|\mathbf{h}}) \leq 1 + \xi(|\mathbf{val}(X_i)|, b_i, b_f) \qquad \forall i, \mathbf{h}, \\
& l^{i,(k)}_{j|\mathbf{h}} \leq w^i_{j]\mathbf{h}} \leq u^{i,(k)}_{j|\mathbf{h}} \qquad \forall i, j, \mathbf{h},
\end{aligned}$$

where $l^{i,(k)}_{j|\mathbf{h}}, u^{i,(k)}_{j|\mathbf{h}} \in [-\infty, 0]$ are the lower and upper bounds on the parameters, respectively. These subproblems are convex and can be exactly and efficiently solved.[11] If the determined solution does not fit the required precision for all parameters, the algorithm performs one of the following options:

(a) *Bound*. If no global maximizer is to be found within the bounds, the algorithm prunes the whole subtree (this happens if the best feasible solution found so far is better (has larger objective) than the optimal solution of the subproblem of the current leaf).

(b) *Branch*. Alternatively, the algorithm creates two new problems by adding new lower and upper bounds to one of the parameters (branching variable) which does not satisfy the desired precision. If multiple parameters do not satisfy the desired precision,

[11] When solving these problems using gradient-ascent methods, a projection onto the feasible set must be performed in every iteration. In the original publication, we used a general purpose convex solver for computing these projections. However, this is very time-consuming. Thus, we derived a specialized projection algorithm that shows much better performance. This algorithm is presented in Appendix 5.A.

i.e. different branching variables are possible, we use the *branching heuristic* described below to select the branching variable. Furthermore, to efficiently prune subtrees, it is important to generate good lower bounds for the objective at this stage, cf. the paragraph on *rounding heuristics* below.

Subproblems in the search tree are processed in order of their highest achievable objective value (the achievable objective value of subproblem k is upper bounded by the objective value of the relaxed problem of the parent of k according to the search tree). In this way, the subproblem of the search tree with highest upper-bound is processed next. The BB algorithm terminates either after a specified amount of time, returning the best solution found so far (anytime solution), or after there are no more open subproblems. In the latter case, the found solution is the global optimizer of (5.104).

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 two simple rounding heuristics. If any of these heuristics yields a better feasible solution for (5.104) than the best solution found so far, the best solution is updated.

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

- *Rounding*: The idea of this heuristic is that by rounding the intermediate solution $\hat{\mathbf{w}}$, we may be able to create a solution that is almost as good as $\hat{\mathbf{w}}$ in terms of the objective, i.e. close to the best solution that can be found within the considered bounds. Therefore, set

$$\hat{a}^i_{j|\mathbf{h}} = \max\left(M, \left[\frac{\hat{w}^i_{j|\mathbf{h}}}{q}\right]_R q\right), \tag{5.105}$$

  where $[\cdot]_R$ denotes rounding to the closest integer, $q = 2^{-b_f}$ 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 $\mathbf{a}$ is feasible for (5.104).[12]

- *Gradient Guided Rounding*: Let $\mathbf{g}$ be the gradient of the objective at $\hat{\mathbf{w}}$. The intuition of this heuristic is that the gradient conveys information about directions in which the objective would increase if there were no constraints — we try to exploit this information by rounding according to the sign information of the gradient. Therefore,

$$\hat{b}^i_{j|\mathbf{h}} = \begin{cases} \left\lceil \frac{\hat{w}^i_{j|\mathbf{h}}}{q} \right\rceil q & \text{if } g^i_{j|\mathbf{h}} > 0, \text{and} \\ \max\left(M, \left\lfloor \frac{\hat{w}^i_{j|\mathbf{h}}}{q} \right\rfloor q\right) & \text{if } g^i_{j|\mathbf{h}} \leq 0, \end{cases} \tag{5.106}$$

  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.

$$(i', j', \mathbf{h}') = \arg\max_{i,j,\mathbf{h}} \left| \hat{w}^i_{j|\mathbf{h}} - \left[\frac{\hat{w}^i_{j|\mathbf{h}}}{q}\right]_R q \right|. \tag{5.107}$$

[12] A description of the algorithm used for projection is provided in Appendix 5.B

5.4.2 Theoretical Analysis: Approximate Bayesian Network Classifiers by Rounding

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. Performing a similar analysis for BNCs with MM BB parameters is much more difficult for the following two reasons: (1) The objective for learning margin maximizing parameters does not decompose as a product of conditional probabilities. (2) There is no closed-form solution for the parameters.

We start by analyzing the KL-divergence introduced by rounding, i.e. we consider 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 5 (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*

$$\widetilde{w}^i_{j|\mathbf{h}} = \left[\frac{w^i_{j|\mathbf{h}}}{q}\right]_R q, \tag{5.108}$$

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.

$$\mathcal{D}(\mathbf{w}^i_{\cdot|\mathbf{h}} \| \log\alpha + \widetilde{\mathbf{w}}^i_{\cdot|\mathbf{h}}) \leq q, \tag{5.109}$$

where $\alpha = (\sum_j \exp(\widetilde{w}^i_{j|\mathbf{h}}))^{-1}$ ensures renormalization such that $\sum_j \exp(\log\alpha + \widetilde{w}^i_{j|\mathbf{h}}) = 1$.

Proof. We calculate

$$\begin{aligned}
\mathcal{D}(\mathbf{w}^i_{\cdot|\mathbf{h}} \| \log\alpha + \widetilde{\mathbf{w}}^i_{\cdot|\mathbf{h}}) &= \sum_j \exp(w^i_{j|\mathbf{h}}) \log \frac{\exp(w^i_{j|\mathbf{h}})}{\alpha \exp(\widetilde{w}^i_{j|\mathbf{h}})} && (5.110)\\
&= \sum_j \exp(w^i_{j|\mathbf{h}}) \left[(w^i_{j|\mathbf{h}} - \widetilde{w}^i_{j|\mathbf{h}}) - \log\alpha\right] && (5.111)\\
&\overset{(a)}{\leq} \sum_j \exp(w^i_{j|\mathbf{h}}) \left[\frac{q}{2} - \log\alpha\right] && (5.112)\\
&= \frac{q}{2} - \log\alpha, && (5.113)
\end{aligned}$$

where (a) is because $\widetilde{\mathbf{w}}^i_{\cdot|\mathbf{h}}$ is derived from $\mathbf{w}^i_{\cdot|\mathbf{h}}$ by rounding the parameters, i.e. $(w^i_{j|\mathbf{h}} - \widetilde{w}^i_{j|\mathbf{h}}) \leq \frac{q}{2}$. It remains to upper bound $-\log\alpha$. Straightforward calculation yields

$$-\log\alpha = \log \sum_j \exp(\widetilde{w}^i_{j|\mathbf{h}}) \leq \log \sum_j \exp(w^i_{j|\mathbf{h}} + \frac{q}{2}) = \frac{q}{2}. \tag{5.114}$$

Hence, the statement follows. □

This bound is tight. Assuming that sufficient (integer) bits are used, no log-probabilities have to be truncated (truncation must be performed if some $w^i_{j|\mathbf{h}} \leq -2^{b_i} + 2^{-b_f}$, i.e. $w^i_{j|\mathbf{h}}$ is smaller than the smallest value that can be represented using the chosen number format). In this case, 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 truncated. Still, a bound on the KL-divergence can be derived:

Lemma 6 (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*

$$\widetilde{w}^i_{j|\mathbf{h}} = \max\left(M, \left[\frac{w^i_{j|\mathbf{h}}}{q}\right]_R q\right), \tag{5.115}$$

where $q = 2^{-b_f}$ is the quantization interval and where M is the minimal representable log-probability. Then the KL-divergence between the optimal and the renormalized approximate distribution is bounded as

$$\mathcal{D}(\mathbf{w}^i_{\cdot|\mathbf{h}} \| \log\alpha + \widetilde{\mathbf{w}}^i_{\cdot|\mathbf{h}}) \leq \frac{3q}{2} + |\mathbf{val}(X_i)|\exp(M), \tag{5.116}$$

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

Typically, $M = -2^{b_i} + 2^{-b_f}$. Hence, also in this case the bound decays with an increasing number of bits. One can further observe a dependency on the size of individual CPTs.

Both, Lemma 5 and 6, 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 rounding heuristics for obtaining good candidate solutions in the BB algorithm.

5.4.3 Experimental Results

In the following, we present classification experiments using the MM objective in (5.104). In particular, we use BNCs with parameters determined as follows:

- *branch and bound* (BB): These reduced-precision parameters are obtained using the BB algorithm presented in Section 5.4.1.
- *rounded* (RD): Parameters are obtained by rounding double-precision log-parameters to the desired number format. If necessary, parameters are truncated.
- *double-precision* (DP): Double-precision parameters are obtained by solving (5.99) using methods proposed in [22].

Classification Experiments

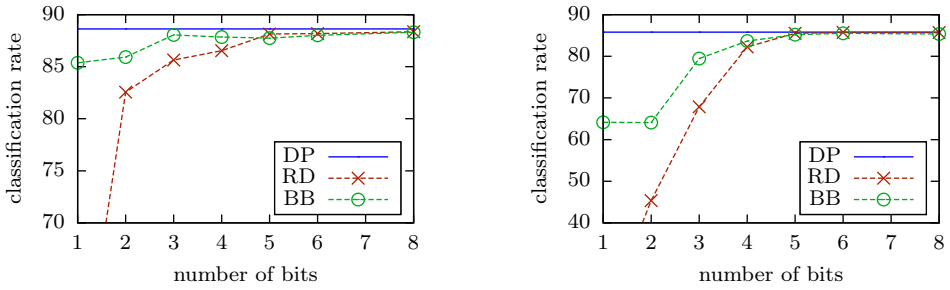
We consider classification experiments for four real world datasets, i.e. USPS data, MNIST data and satimage/letter data from the UCI repository. Details on these datasets are available in Section 2.3. On these datasets, we compare the CR performance of BNCs with BB, RD, and DP parameters.[13]

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 held out validation data is maximized. For learning BB parameters with B bits, we determine the splitting into integer bits and fractional bits, as well as the hyper-parameters λ and γ in (5.104), using 5-fold cross-validation. For every setting of the parameters B, λ and γ, 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 5.4.1.

The observed CRs are shown in Figures 5.14, 5.15 and 5.16, 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-CMI structures is shown. Only 5 to 6 bits for RD parameters are necessary to achieve CRs close to DP CRs. BNCs with BB parameters achieve

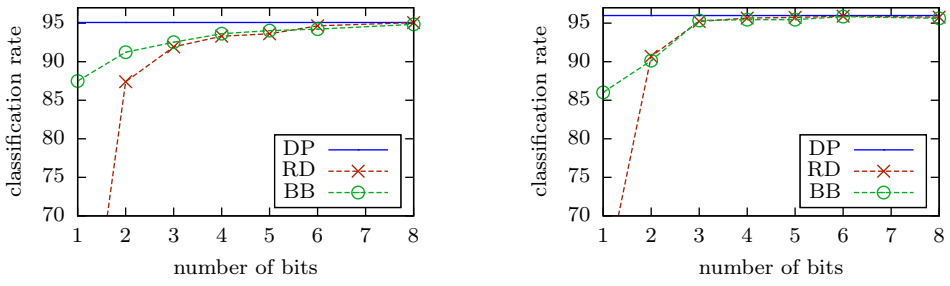
[13] As mentioned in Section 5.4.1, up to $\log_2(L+1)+b_i+b_f$ bits are necessary for classification using BNCs with reduced-precision parameters. In the presented experiments, we assume that these additional bits are available, i.e. summation does not cause overflows.

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

Figure 5.14: CRs for *satimage* and *letter* data of BNCs with BB, RD and DP parameters for NB structures.



(a) NB (b) TAN-CMI

Figure 5.15: CRs for USPS data of BNCs with BB, RD and DP parameters for NB and TAN-CMI structures.

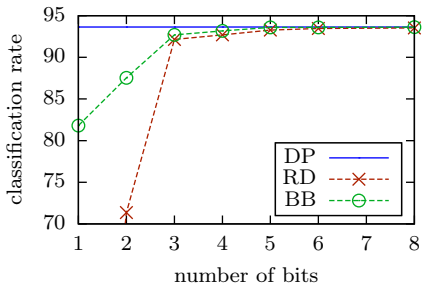


Figure 5.16: CRs for MNIST of BNCs using NB structure with DP, RD and BB parameters.

One important aspect of reduced-precision parameters, is their lower memory usage. This is exemplarily shown for USPS and MNIST data and NB and TAN-CMI structures in Table 5.7. The reduction in storage requirements by a factor of ~ 10 can positively influence the memory access when implementing BNCs on embedded hardware.

Table 5.7: Memory usage for parameter storage of DP, BB and RD parameters in reduced-precision.

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

5.4.4 Summary

We presented an efficient algorithm for computing margin maximizing reduced-precision parameters using full-precision arithmetic. If desired, these parameters can be scaled to integer values. The subproblems of the proposed algorithm are convex and can be solved efficiently.

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 support to understand the implications of implementing BNCs on embedded hardware and can greatly reduce the storage requirements and thus the time required for memory access.

5.5 Learning Reduced-Precision Parameters With Reduced-Precision Arithmetic

In this section, we very briefly consider learning reduced-precision parameters for BNCs using reduced-precision computations only. We present some algorithms we experimented with and show some preliminary experimental results. The algorithms are ad hoc and we did not perform any theoretical analysis. We start by considering learning ML parameters in Section 5.5.1 and then move on to learning MM parameters in Section 5.5.2. Please note that experiments do not cover all important details and effects. A more detailed investigation is subject to future work.

We claim that learning using reduced-precision arithmetic is most useful in online settings, i.e. parameters are updated on a per-sample basis. This online learning scenario captures the important case in which initially pre-computed parameters are used and these parameters are updated online as new samples become available, e.g. adaptation of a hearing-aid to a new acoustic environment. In this setting, learning using reduced-precision computations requires specialized algorithms, i.e. gradient-descent (or gradient-ascent) procedures using reduced-precision arithmetic do not perform well. This is mainly because we cannot perform exact projections that are necessary when we want to learn normalized parameters satisfying the sum-to-one constraints. Another issue may arise because the learning rate cannot be annealed arbitrarily. However, we do not find this issue as import as the inexact projections.

5.5.1 Maximum Likelihood Parameters

We consider an online algorithm for learning ML parameters. Recall the ML objective for the offline scenario, i.e.

$$\mathcal{P}_{\mathcal{G}}^{\mathrm{ML}} = \arg\max_{\mathcal{P}_{\mathcal{G}}} \prod_{n=1}^{N} \mathrm{P}^{\mathcal{B}}(c^{(n)}, \mathbf{x}^{(n)}), \tag{5.117}$$

or equivalently,

$$\mathbf{w}^{\mathrm{ML}} = \arg\max_{\mathbf{w}} \sum_{n=1}^{N} \boldsymbol{\phi}(c^{(n)}, \mathbf{x}^{(n)})^T \mathbf{w} \quad \text{s.t.} \sum_j \exp(w^i_{j|\mathbf{h}}) = 1, \forall i, j, \mathbf{h}. \tag{5.118}$$

In an online scenario, not all samples are available for learning at once but are available one at a time; the parameters $\mathbf{w}^{\mathrm{ML},t}$ at time-step t are updated according to the gradient of a single sample $(c, \mathbf{x})$ (or, alternatively, a batch of samples) and projected such that they satisfy the sum-to-one constraints, i.e.

$$\mathbf{w}^{\mathrm{ML},t+1} = \Pi\left[\mathbf{w}^{\mathrm{ML},t} + \eta\left(\nabla_{\mathbf{w}} \boldsymbol{\phi}(c, \mathbf{x})^T \mathbf{w}\right)(\mathbf{w}^{\mathrm{ML},t})\right] \tag{5.119}$$

$$= \Pi\left[\mathbf{w}^{\mathrm{ML},t} + \eta \boldsymbol{\phi}(c, \mathbf{x})\right], \tag{5.120}$$

where η is the learning rate, where $\nabla_{\mathbf{w}}(f)(\mathbf{a})$ denotes the gradient of f with respect to $\mathbf{w}$ at $\mathbf{a}$, and where $\Pi[\mathbf{w}]$ denotes the ℓ_2-norm projection of the parameter vector $\mathbf{w}$ onto the set of normalized parameter vectors. Note that the gradient has a simple form: it consists only of zeros and ones, where the ones are *indicators* of *active* entries in the CPTs of sample $(c, \mathbf{x})$. Furthermore, assuming normalized parameters at time-step t, the direction of the gradient is always such that the parameters $\mathbf{w}^{\mathrm{ML},t+1}$ are super-normalized. Consequently, after projection the parameters satisfy the sum-to-one constraints.

We continue by analyzing the effect of using reduced-precision arithmetic on the online learning algorithm. Therefore, we performed the following experiment: Assume that the projection can only be approximately performed. We *simulate* the approximate projection by performing an exact projection and subsequently adding quantization noise (this is similar to reduced-precision analysis in signal processing [18]). We sample the noise from a Gaussian distribution with zero mean and with variance $\sigma^2 = q^2/12$, where $q = 2^{-b_f}$. For the satimage dataset we construct BNCs with TAN-CMI structure. As initial parameters we use rounded ML parameters computed from one tenth of the training data. Then, we present the classifier further samples in an online manner and update the parameters according to (5.120). During learning, we set the learning rate η to $\eta = \eta_0/\sqrt{1+t}$, where η_0 is some constant (η_0 is tuned by hand such that the test set performance is maximized). The resulting classification performance is shown in Figures 5.17a and 5.17b for the exact and the approximate projection, respectively. One can observe, that the algorithm does not properly learn using the approximate projection. Thus, it seems crucial to perform the projections rather accurately. To circumvent the need for accurate projections, we propose a method that avoids computing a projection at all in the following.

Consider again the offline parameter learning case. ML parameters can be computed in closed-form by computing relative frequencies, i.e.

$$\theta^i_{j|\mathbf{h}} = \frac{m^i_{j|\mathbf{h}}}{m^i_{\mathbf{h}}}, \tag{5.121}$$

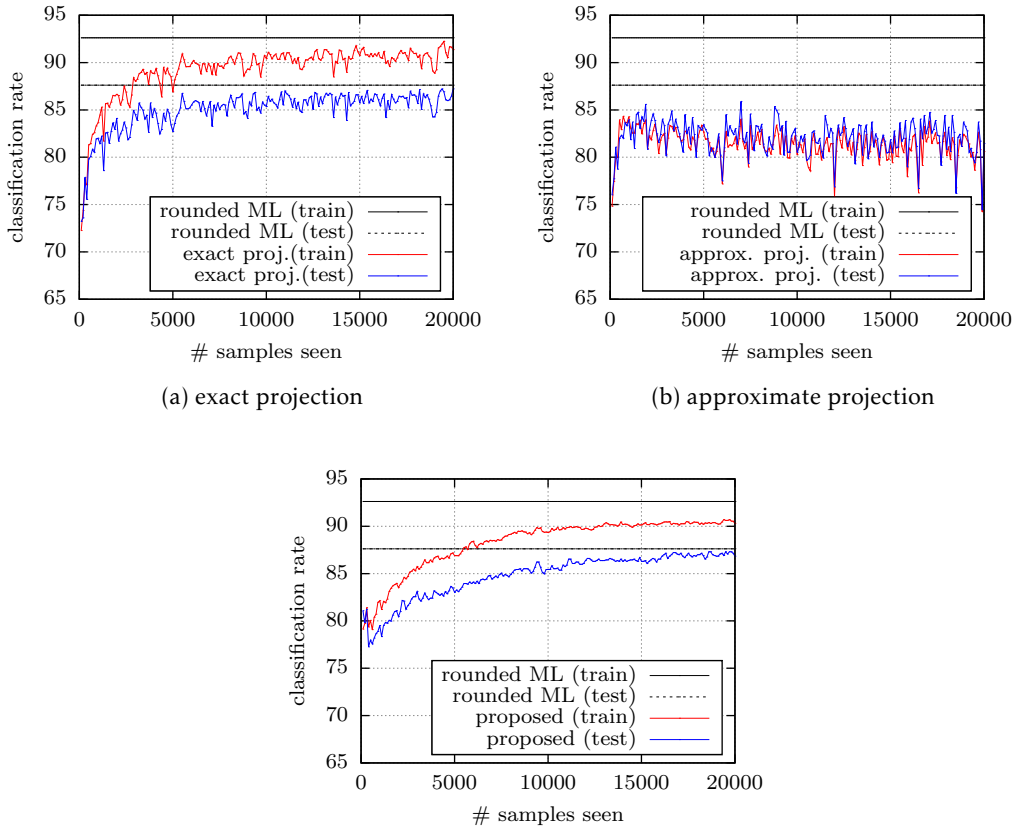
where

$$m^i_{j|\mathbf{h}} = \sum_{n=1}^{N} \boldsymbol{\phi}(c^{(n)}, \mathbf{x}^{(n)})^i_{j|\mathbf{h}}, \text{ and} \tag{5.122}$$

$$m^i_{\mathbf{h}} = \sum_j m^i_{j|\mathbf{h}}. \tag{5.123}$$

This can be easily extended to online learning. Assume that the counts $m^{i,t}_{j|\mathbf{h}}$ at time t are given and that a sample $(c^t, \mathbf{x}^t)$ is presented to the learning algorithm. Then, the counts are updated according to

$$m^{i,t+1}_{j|\mathbf{h}} = m^{i,t}_{j|\mathbf{h}} + \boldsymbol{\phi}(c^t, \mathbf{x}^t)^i_{j|\mathbf{h}}. \tag{5.124}$$



(a) exact projection

(b) approximate projection

(c) proposed algorithm

Figure 5.17: Classification performance of BNCs with TAN-CMI structure for satimage data in an online learning scenario; (a) Online ML parameter learning with exact projection after each parameter update, (b) online ML parameter learning with approximate projection after each parameter update (see text for details), (c) proposed algorithm for online ML parameter learning.

Exploiting these counts, the logarithm of the ML parameters $\theta_{j|\mathbf{h}}^{i,t}$ at time t can be computed as

$$w_{j|\mathbf{h}}^{i,t} = \log\left(\frac{m_{j|\mathbf{h}}^{i,t}}{m_{\mathbf{h}}^{i,t}}\right), \tag{5.125}$$

where similarly to before

$$m_{\mathbf{h}}^{i,t} = \sum_j m_{j|\mathbf{h}}^{i,t}. \tag{5.126}$$

A straightforward approximation of (5.125) is to (approximately) compute the counts $m_{j|\mathbf{h}}^{i,t}$ and $m_{\mathbf{h}}^{i,t}$, respectively, and to use a lookup table to determine $w_{j|\mathbf{h}}^{i,t}$. The lookup table can be indexed in terms of $m_{j|\mathbf{h}}^{i,t}$ and $m_{\mathbf{h}}^{i,t}$ and stores values for $w_{j|\mathbf{h}}^{i,t}$ in the desired reduced-precision format. To limit the maximum size of the lookup table and the bit-width required for the counters for $m_{j|\mathbf{h}}^{i,t}$ and $m_{\mathbf{h}}^{i,t}$, we assume some maximum integer number M. We pre-compute the lookup table L such that

$$L(i,j) = \left\lceil \frac{\log_2(i/j)}{q} \right\rfloor_R \cdot q, \tag{5.127}$$

where q is the quantization interval of the desired fixed-point representation, where $\log_2(\cdot)$ denotes the base-2 logarithm, and where i and j are in the range $0,\ldots,M-1$. Given sample $(c^t, \mathbf{x}^t)$, the counts $m_{j|\mathbf{h}}^{i,t+1}$ and $m_{\mathbf{h}}^{i,t+1}$ are computed according to Algorithm 4 from the counts $m_{j|\mathbf{h}}^{i,t}$ and $m_{\mathbf{h}}^{i,t}$. To guarantee that the counts stay in range, the algorithm identifies counters that reach their maximum value, and halves these counters as well as all other counters corresponding to the same CPTs. This division by 2 can be implemented as a bitwise shift operation.

Algorithm 4 Reduced-precision ML online learning

Require: Old counts $m_{j|\mathbf{h}}^{i,t}$; sample $(c^t, \mathbf{x}^t)$

```
m_{j|h}^{i,t+1} ← m_{j|h}^{i,t} + φ(c^t, x^t)^i_{j|h}    ∀i, j, h      ▷ update counts
for i, j, h do
    if m_{j|h}^{i,t+1} = M then                                        ▷ maximum value of counter reached?
        m_{j|h}^{i,t+1} ← ⌊m_{j|h}^{i,t+1}/2⌋    ∀j                    ▷ halve counters of considered CPT
    end if
end for
return m_{j|h}^{i,t+1}
```

We performed experiments using $M = 127$, i.e. we used counters with 7 bits. The number of integer bits and fractional bits was set to 4 bits each. Initially, we set all counts to zero, i.e. $m_{j|\mathbf{h}}^{i,0} = 0$, respectively. For the cumulative counts, i.e. $m_{\mathbf{h}}^{i,t}$ in (5.125), we did not limit the number of bits (for real implementations the necessary number of bits for this counter can be computed from the bit-width of the individual counters that are summed up and the graph structure of the considered BNC). Logarithmic parameters $w_{j|\mathbf{h}}^{i,t}$ are computed using the lookup table described above and using Algorithm 5. The classification performance during online learning is shown in Figure 5.17c. We can observe, that the algorithm behaves pleasant and the limited range of the used counters does not seem to affect classification performance (compared to the classification performance using rounded ML parameters computed using full-precision computations and all training samples).

Further experimental results are shown in Table 5.8 for more datasets and different classifier structures. All samples from the training set were presented to the proposed

Algorithm 5 Computation of logarithmic probabilities from lookup table

Require: Counts $m_{j|\mathbf{h}}^{i,t}$ and $m_{\mathbf{h}}^{i,t}$; lookup table L of size $(M+1)\times(M+1)$

```
div ← 0
while m_h^{i,t} > M do                          ▷ ensure that index into lookup table is in range
    m_h^{i,t} ← ⌊m_h^{i,t}/2⌋
    div ← div + 1
end while
w_{j|h}^{i,t} ← L(m_{j,h}^{i,t}, m_h^{i,t})  ∀j          ▷ get log-probability from lookup table
while div > 0 and ∀j: w_{j|h}^{i,t} > (−2^{b_i} + 2^{b_f}) + 1 do      ▷ revise index correction
    w_{j|h}^{i,t} ← w_{j|h}^{i,t} − 1  ∀j
    div ← div − 1
end while
return w_{j|h}^{i,t}
```

algorithm for five times in random order. The absolute reduction in classification rate is at most 2.3% for the considered datasets and below 2.8% relative. Thus the proposed reduced-precision computation scheme seems to be sufficiently accurate to yield good classification performance while employing only range-limited counters and a lookup table of size $(M+1)\times(M+1)$. Clearly, the performance of the proposed method can be improved by using larger and more accurate lookup tables and counters with larger bit-width.

Table 5.8: Learning ML parameters using reduced-precision computations; CRs using ML parameters according to (5.121) are denoted as *ML exact*, CRs using reduced-precision ML parameters computed according to Algorithm 4 using only reduced-precision arithmetic are denoted as *ML proposed*; *red. abs.* and *red. rel.* denote the absolute and the relative reduction in CR of the BNC using exact ML parameters to the BNC using the proposed reduced-precision ML parameters, respectively.

		CR [%]			
Dataset	Structure	ML exact	ML proposed	red. abs.	red. rel.
satimage	NB	83.26	82.79	0.47	0.56
satimage	TAN-CMI	87.62	87.46	0.16	0.18
USPS	NB	86.89	86.34	0.55	0.63
USPS	TAN-CMI	91.39	90.05	1.34	1.46
MNIST	NB	82.88	80.61	2.26	2.73
MNIST	TAN-CMI	93.11	93.00	0.11	0.12

5.5.2 Maximum Margin Parameters

The MM objective is

$$\mathbf{w}^{\mathrm{MM}} = \arg\max_{\mathbf{w}} \left[\sum_{n=1}^{N} \boldsymbol{\phi}(c^{(n)}, \mathbf{x}^{(n)})^T \mathbf{w} + \lambda \sum_{n=1}^{N} \min\left(\gamma, \min_{c \neq c^{(n)}} \left[(\boldsymbol{\phi}(c^{(n)}, \mathbf{x}^{(n)}) - \boldsymbol{\phi}(c, \mathbf{x}^{(n)}))^T \mathbf{w} \right] \right) \right], \tag{5.128}$$

where we used the notation and formulation from Chapter 4. Note that we rearranged terms and replaced the soft-hinge with a hard version and the soft-max by a regular max because these operations would be difficult to implement in reduced-precision. In the online learning case, given sample $(c, \mathbf{x})$, the parameters $\mathbf{w}^{\mathrm{MM},t+1}$ at time $t+1$ are computed

from the parameters $\mathbf{w}^{\mathrm{MM},t}$ at time t as

$$\mathbf{w}^{\mathrm{MM},t+1} = \Pi\Big[\mathbf{w}^{\mathrm{MM},t} + \eta\boldsymbol{\phi}(c,\mathbf{x}) + \eta\lambda\mathbf{g}(c,\mathbf{x})\Big], \tag{5.129}$$

where

$$\mathbf{g}(c,\mathbf{x}) = \begin{cases} 0 & \min\limits_{c'\neq c}\Big[(\boldsymbol{\phi}(c,\mathbf{x}) - \boldsymbol{\phi}(c',\mathbf{x}))^T\mathbf{w}\Big] \geq \gamma, \\ \boldsymbol{\phi}(c,\mathbf{x}) - \boldsymbol{\phi}(c',\mathbf{x}) & \text{o.w., where } c' = \arg\min_{c'}\Big[(\boldsymbol{\phi}(c,\mathbf{x}) - \boldsymbol{\phi}(c',\mathbf{x}))^T\mathbf{w}\Big] \end{cases} \tag{5.130}$$

and where, as before, $\Pi[\mathbf{w}]$ denotes the ℓ_2-norm projection of the parameter vector $\mathbf{w}$ onto the set of normalized parameter vectors.

For learning MM parameters, a similar observation with respect to the accuracy of the projection can be made as for ML parameters. But we cannot proceed exactly as in the case of learning ML parameters because we cannot compute MM parameters in closed-form. As in the ML parameter learning case, the gradient for the parameter update has a rather simple form, but the projection to satisfy the sum-to-one constraints is difficult to compute. Therefore, for online MM parameter learning, we propose Algorithm 6 that is similar to Algorithm 4 in Section 5.5.1, i.e. we avoid to compute the projection explicitly. From the counts computed by the algorithm, log-probabilities can be computed using Algorithm 5. Note that the proposed algorithm does not exactly optimize (5.128) but a, not explicitly defined, surrogate. The idea behind the algorithm is to (1) to optimize the likelihood term in (5.128) as in the algorithm for ML parameter learning, and (2) to optimize the margin term by increasing the likelihood for the correct class and simultaneously decreasing the likelihood for the strongest competitor class. Note that the idea of optimizing the margin term as explained above is similar in spirit to that of discriminative frequency estimates [29]. However, discriminative frequency estimates do not optimize a margin term but a term more closely related to the class-conditional likelihood.

We performed experiments using the same setup as in Section 5.5.1. We set the hyper-parameters $\lambda = 4$ and $\gamma = 0.25$ without performing cross-validation. For this setup, we observed the classification performance summarized in Table 5.9. While the results are not as good as those of the exact MM solution, we can clearly observe an improvement in classification performance using the proposed MM parameter learning method over the proposed ML parameter learning method, cf. Table 5.8. A proper investigation using cross-validation for setting hyper-parameters is subject to future work. Furthermore, a detailed investigation of effects decreasing the CR, e.g. bit-width of counters, should be conducted in the future.

Table 5.9: Learning MM parameters using reduced-precision computations; CRs using MM parameters according to (5.128) are denoted as *MM exact*, CRs using reduced-precision MM parameters computed according to Algorithm 6 are denoted as *MM proposed*; *red. abs.* and *red. rel.* denote the absolute and the relative reduction in CR of the BNC using exact MM parameters to the BNC using the proposed MM learning method, respectively.

		CR [%]			
Dataset	Structure	MM exact	MM proposed	red. abs.	red. rel.
satimage	NB	87.07	86.68	0.39	0.45
satimage	TAN-CMI	86.84	86.60	0.23	0.27
USPS	NB	93.91	93.17	0.74	0.79
USPS	TAN-CMI	93.01	93.50	−0.49	−0.53
MNIST	NB	90.49	87.92	2.57	2.84
MNIST	TAN-CMI	93.49	93.83	−0.34	−0.36

Algorithm 6 Reduced-precision MM online learning

Require: Old counts $m^{i,t}_{j|\mathbf{h}}$; sample $(c^t, \mathbf{x}^t)$; hyper-parameters $\gamma, \lambda \in \mathbb{N}_+$ for MM formulation

$m^{i,t+1}_{j|\mathbf{h}} \leftarrow m^{i,t}_{j|\mathbf{h}} + \phi(c^t, \mathbf{x}^t)^i_{j|\mathbf{h}} \quad \forall i, j, \mathbf{h}$ ▷ update counts (likelihood term)
for $i, j, \mathbf{h}$ **do** ▷ ensure that parameters stay in range
  if $m^{i,t+1}_{j|\mathbf{h}} = M$ **then**
    $m^{i,t+1}_{j|\mathbf{h}} \leftarrow \lfloor m^{i,t+1}_{j|\mathbf{h}}/2 \rfloor \quad \forall j$
  end if
end for
$c' \leftarrow$ strongest competitor of class c for features $\mathbf{x}$
if $\left[(\phi(c^t, \mathbf{x}^{(n)}) - \phi(c', \mathbf{x}^{(n)}))^T \mathbf{w} < \gamma\right]$ **then**
  for $k = 1, \ldots, \lambda$ **do**
    $m^{i,t+1}_{j|\mathbf{h}} \leftarrow m^{i,t}_{j|\mathbf{h}} + \phi(c^t, \mathbf{x}^t)^i_{j|\mathbf{h}} \quad \forall i, j, \mathbf{h}$ ▷ update counts (margin term)
    $m^{i,t+1}_{j|\mathbf{h}} \leftarrow m^{i,t}_{j|\mathbf{h}} - \phi(c', \mathbf{x}^t)^i_{j|\mathbf{h}} \quad \forall i, j, \mathbf{h}$ ▷ update counts (margin term)
    for $i, j, \mathbf{h}$ **do** ▷ ensure that parameters stay in range
      if $m^{i,t+1}_{j|\mathbf{h}} = 0$ **then**
        $m^{i,t+1}_{j|\mathbf{h}} \leftarrow m^{i,t+1}_{j|\mathbf{h}} + 1 \quad \forall j$
      end if
      if $m^{i,t+1}_{j|\mathbf{h}} = M$ **then**
        $m^{i,t+1}_{j|\mathbf{h}} \leftarrow \lfloor m^{i,t+1}_{j|\mathbf{h}}/2 \rfloor \quad \forall j$
      end if
    end for
  end for
end if
return $m^{i,t+1}_{j|\mathbf{h}}$

5.6 Open Questions and Directions for Future Research

The results presented in this chapter only form the basis of an interesting research field, letting a large number of questions unanswered. We discuss some of them in the following:

- **Underlying graphical model.** Recently, Piatkowski et al. [27] proposed an integer approximation for undirected graphical models. Clearly, undirected graphical models solve the normalization problem because of their definition, i.e. normalization is achieved by dividing the specified unnormalized probability densities by the partition function. But, computing the partition function can be expensive. While this is unproblematic for classification of fully observed data, it can make a significant difference when marginalization is necessary because of hidden variables or for other inference scenarios. Thus it would be interesting to compare approaches for reduced-precision directed graphical models and undirected graphical models.

- **More complex BNCs and structure learning.** Our theoretical and empirical investigations consider only BNs with simple structures, i.e. NB and TAN. It is interesting to investigate how results carry over to more general and complex BNs, e.g. HMMs. For the class of HMMs it may also be possible to derive performance bounds using results from propagation of error theory (by applying the theory to the forward-backward recursion that is used to compute marginals, etc.). For arbitrary BNs, derivation of useful bounds might be difficult.

- **General message passing.** It seems important to analyze message passing in general, i.e. the behavior of the sum-product and the max-product algorithm under quantization noise. Such an analysis could yield a high level view on the subject of reduced-precision BNs and could help to guide the development of specialized algorithms. This analysis could extend studies by Ihler et al. [12] that investigate effects of message approximations, for example caused by rounding, on loopy belief propagation. However, their focus is mainly on convergence issues and does not cover effects on the classification performance.

- **Structure learning.** 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 the parameters degrades classification performance as little as possible.

- **Mixed-precision designs.** Mixed-precision designs could be of interest. These designs would allow one to spend more computational resources, i.e. more bits for parameter storage and computation, on *crucial* parts of a BN, while saving resources in other parts. Crucial parts of a BN could be those parts that if quantized coarsely severely degrade classification performance or severely degrade marginalization accuracy for some variables of interest

- **Generative parameters.** Learning generatively optimized reduced-precision parameters considering reduced-precision constraints of the destination platform should be considered, cf. Appendix 5.C. In a first step, we want to boost performance of this type of learning to obtain better results.

- **Real-world implementation.** We work towards a real-world reduced-precision implementation of BNCs that could be used in hearing aids for acoustic scene classification. Once the implementation is completed, we want to compare it to full-precision implementations with respect to speedup, power consumption and classification performance.

Appendix

5.A A Projection Algorithm[14]

Consider the problem of projecting a given point $\xi^* \in \mathbb{R}^L$ onto the set $\mathcal{S} = \{\mathbf{x} \in \mathbb{R}^L : a_l \leq x_l \leq b_l, \sum_l \exp(x_l) \leq z\}$, i.e. the optimization problem

$$\begin{aligned} \underset{\xi}{\text{minimize}} \quad & \frac{1}{2}\|\xi^* - \xi\|_2^2 \\ \text{s.t.} \quad & \sum_l \exp(\xi_l) \leq z, \text{ and} \\ & a_l \leq \xi_l \leq b_l, \quad \forall l, \end{aligned} \tag{5.131}$$

where $z > 0$ is a real constant, and where a_l, b_l are lower and upper bounds for the parameters in each dimension, respectively. If $z = 1$, the set $\mathcal{S}$ describes all sub-normalized probability mass functions over L logarithmic probabilities ξ_l. Problem (5.131) is similar to the problem considered in Appendix 4.G but additionally includes box-constraints.

5.A.1 Algorithm

The projection (5.131) can be computed using the specialized projection algorithm presented in Algorithm 7. The definitions of $\xi(\gamma)$ and $\xi_l(\gamma)$, as well as a proof of correctness of the algorithm, are provided in the next section.

The algorithm works as follows: In lines 1–3, it is verified if ξ^* is feasible for (5.131) and, in the case, ξ^* is returned as the desired projection. Otherwise, in lines 4–7, the projection ξ^B of ξ^* onto the box-constraints given by a_l, b_l ($l = 1, \ldots, L$) is computed; if ξ^B is feasible for (5.131), ξ^B is returned as the desired projection. If ξ^B is not the sought after projection, then the constraint $\exp(\xi_l) \leq z$ must be active for an optimal solution of (5.131). The closest point to ξ^* that satisfies $\exp(\xi_l) \leq z$ with equality is uniquely parameterized by some γ^+, cf. proofs in next section. This γ^+ is found in lines 8–13. Exploiting γ^+, the optimal solution to (5.131) is computed and returned in lines 14–15.

Note that for root finding in line 13 of the algorithm, a multitude of algorithms can be used [10]. We used the bisection method in our experiments.

5.A.2 Correctness of the Algorithm

To prove the correctness of Algorithm 7, we need a collection of helpful lemmata which are given below. Most proofs are based on the observation that (5.131) is convex and consider the Lagrangian of (5.131), i.e.

$$\mathcal{L}(\xi, \alpha, \beta, \gamma) = \frac{1}{2}\|\xi^* - \xi\|_2^2 + \sum_l \alpha_l(a_l - \xi_l) + \sum_l \beta_l(\xi_l - b_l) + \gamma\left(\sum_l \exp(\xi_l) - z\right), \tag{5.132}$$

where $\alpha_l \geq 0$ is the Lagrange multiplier of $a_l \leq \xi_l$, where $\beta_l \geq 0$ is the Lagrange multiplier of $\xi_l \leq b_l$, and where $\gamma \geq 0$ is the Lagrange multiplier of $\sum_l \exp(\xi_l) \leq z$. The stationarity condition of the Lagrangian is

$$\xi_l - \xi_l^* - \alpha_l + \beta_l + \gamma \exp(\xi_l) = 0, \quad \forall l. \tag{5.133}$$

Lemma 7. *For fixed $\gamma \geq 0$ and $a_l < b_l$, there are three mutually excluding possibilities to simultaneously satisfy the stationarity condition* (5.133)*, the complementary slackness conditions of the box-constraints, and the box-constraints:*

[14] The algorithm presented in this section was derived after the original paper [35] was published. For computing the projections for the simulations in that paper, we used a general purpose convex optimizer. This was however disadvantageous in terms of runtime.

Algorithm 7 Projection algorithm for solving (5.131)

Require: Point $\boldsymbol{\xi}^*$ to project, lower bounds $\mathbf{a}$, upper bounds $\mathbf{b}$, normalization z

if $\boldsymbol{\xi}^*$ feasible for (5.131) **then** ▷ Nothing to do?
  return $\boldsymbol{\xi}^*$
end if
$\xi_l^B \leftarrow \min(b_l, \max(a_l, \xi_l^*)) \quad \forall l$ ▷ Project $\boldsymbol{\xi}^*$ using box-constraints only
if $\sum_l \exp(\xi_l^B) \leq z$ **then** ▷ Feasible for (5.131)?
  return $\boldsymbol{\xi}_B$
end if
$\gamma_{\min} \leftarrow 0$ ▷ Lower bound on γ
$\gamma_{\max} \leftarrow 1$
while $\sum_l \exp(\xi_l(\gamma_{\max})) > z$ **do** ▷ Find upper bound $\gamma_{\max}$ on γ
  $\gamma_{\max} \leftarrow 2\gamma_{\max}$
end while
$\gamma^+ \leftarrow$ Root of $g(\gamma) = [\sum_l \exp(\xi_l(\gamma)) - z]$ in the interval $[\gamma_{\min}, \gamma_{\max}]$ ▷ Optimal γ
$\xi_l^+ \leftarrow \xi_l(\gamma^+) \quad \forall l$ ▷ Construct solution
return $\boldsymbol{\xi}^+$

(1) $\xi_l = \xi_l^* - W(\gamma \exp(\xi_l^*))$ *solves* $\xi_l - \xi_l^* + \gamma \exp(\xi_l) = 0$ *such that* $a_l < \xi_l < b_l$.

(2) $\xi_l = a_l$ *and* $\alpha_l = a_l - \xi_l^* + \gamma \exp(a_l)$, $\alpha_l \geq 0$.

(3) $\xi_l = b_l$ *and* $\beta_l = -b_l + \xi_l^* - \gamma \exp(b_l)$, $\beta_l \geq 0$.

In the above equations, $W(\cdot)$ *is the real branch of the Lambert W function [37].*

Proof. (1) implies ¬(2). Assume that (1) is true. Then $a_l - \xi_l^* + \gamma \exp(a_l) < 0$. Hence, (2) cannot be satisfied by any $\alpha_l \geq 0$.

(1) implies ¬(3). Similarly as above, $b_l - \xi_l^* + \gamma \exp(b_l) > 0$. Hence, (3) cannot be satisfied by any $\beta_l \geq 0$.

(2) implies ¬(1). Contrapositive of ((1) implies ¬(2)).

(3) implies ¬(1). Contrapositive of ((1) implies ¬(3)).

(2) implies ¬(3). Assume that (2) is true, i.e. $\xi_l = a_l$ and $\alpha_l = a_l - \xi_l^* + \gamma \exp(a_l)$, $\alpha_l \geq 0$. Thus, $a_l - \xi_l^* + \gamma \exp(a_l) \geq 0$ and, therefore, $b_l - \xi_l^* + \gamma \exp(b_l) > 0$. Thus, (3) must not be true for any $\beta_l \geq 0$.

(3) implies ¬(2). Analog.

□

Lemma 7 ensures that for every γ, there is a unique solution $\boldsymbol{\xi}$ satisfying the stationarity condition, the complementary slackness conditions of the box-constraints, and the box-constraints. We denote this solution as $\boldsymbol{\xi}(\gamma)$. Note that for some fixed γ, $\boldsymbol{\xi}(\gamma)$ does not necessarily satisfy the complementary slackness condition $\gamma(\sum_l \exp(\xi_l(\gamma)) - z) = 0$. Furthermore, note that if some parameter has identical upper and lower bounds, i.e. $a_l = b_l$, then this parameter can be fixed to a_l.

Lemma 8. *The auxiliary function* $f(\gamma) = \sum_l \exp(\xi_l(\gamma))$ *is continuous and monotonically decreasing in* γ.

Proof. Fix some l. We show that $\exp(\xi_l(\gamma))$ is continuous and monotonically decreasing in γ. Consequently, also the auxiliary function $f(\gamma)$ is continuous and monotonically decreasing in γ because it is a sum of functions with these properties.

Regarding continuity: Consider the cases in Lemma 7. In case (1), $\xi_l(\gamma)$ is continuous and monotone in γ because these properties are *inherited* from the Lambert W function. In cases (2) and (3), $\xi_l(\gamma) = a_l$ if γ is larger than some constant γ' and $\xi_l(\gamma) = b_l$ if γ is

smaller than some constant γ'', respectively. Hence, it suffices to verify that $\exp(\xi_l(\gamma))$ is continuous at γ' and γ'', which is clearly the case.

Regarding monotonicity: In case (1) in Lemma 7, $\xi_l(\gamma) = \xi_l^* - W(\gamma \exp(\xi_l^*))$. The Lambert W function is monotonically increasing in $[0, \infty)$. Consequently, $\xi_l(\gamma)$ is monotonically decreasing in γ, and thus also $\exp(\xi_l(\gamma))$. □

Theorem 9 (Correctness of Algorithm 7)**.** *Algorithm 7 returns an optimal solution for the projection* (5.131).

Proof. If $\boldsymbol{\xi}^*$ is feasible for the projection, this solution is returned. Thus, assume that $\boldsymbol{\xi}^*$ is not feasible.

If $\boldsymbol{\xi}^*$ projected onto only the box-constraints is feasible, then this solution is returned. Otherwise, the constrained $\sum_l \exp(\xi_l) \leq z$ must be active at an optimal solution. We assume this in the following.

Because of Lemma 8, the auxiliary function $g(\gamma) = \sum_l \exp(\xi_l(\gamma)) - z$ is continuous and monotone. By construction $\boldsymbol{\xi}(0) = \boldsymbol{\xi}(\gamma_{\min})$ corresponds to the projection onto the box-constraints only. Therefore, by assumption, $g(\gamma_{\min}) > 0$. The algorithm finds an upper bound on γ, namely $\gamma_{\max}$, such that $g(\gamma_{\max}) \leq 0$. Consequently, using any root finding algorithm of choice, we can find γ^+ such that $g(\gamma^+) = 0$. Exploiting γ^+, the vector $\boldsymbol{\xi}(\gamma^+)$ satisfies the stationary conditions, the complementary slackness conditions, and all constraints. Thus, using γ^+, we can compute the optimal projection. □

5.B Projection Onto Fixed-Point Parameters

The algorithm described in this section is used as part of the rounding heuristics in the BB algorithm described in Section 5.4.1.

Rounding of the intermediate solutions according to the proposed heuristics ensures that the components of the resulting vector $\hat{\mathbf{a}}$ (or $\hat{\mathbf{b}}$, respectively) satisfies the desired precision, i.e. $\hat{a}^i_{j|\mathbf{h}}, \hat{b}^i_{j|\mathbf{h}} \in -\mathbb{B}^{b_i}_{b_f}$. However, due to this rounding, the sub-normalization constraints may become violated. Thus, the vector $\hat{\mathbf{a}}$ (or $\hat{\mathbf{b}}$) must be projected onto the feasible set, i.e. we must solve

$$\begin{aligned} \underset{\mathbf{a}}{\text{minimize}} \quad & \frac{1}{2}\|\hat{\mathbf{a}} - \mathbf{a}\|_2^2 \\ \text{s.t.} \quad & \sum_{j=1}^{|\mathbf{val}(X_i)|} \exp(a^i_{j|\mathbf{h}}) \leq 1 + \xi(|\mathbf{val}(X_i)|, b_i, b_f) \qquad \forall i, \mathbf{h}, \\ & a^i_{j|\mathbf{h}} \in -\mathbb{B}^{b_i}_{b_f} \qquad \forall i, j, \mathbf{h}. \end{aligned} \tag{5.134}$$

Solving this projection exactly is difficult, thus we use Algorithm 8 to perform an approximate projection. The intuition of this algorithm is as follows: considering the parameters $\mathbf{a}^i_{\mathbf{h}}$ corresponding to the CPT indexed by $i, \mathbf{h}$, reduce the entries of the coordinates that make the largest possible *advance* towards the feasible set while ensuring that $a^i_{j|\mathbf{h}} \in -\mathbb{B}^{b_i}_{b_f}$.

Clearly, the output of the algorithm is feasible for (5.104).

5.C Bayesian Networks with Generatively Optimized Reduced-Precision Parameters

In the main text of this chapter, we considered only BNs with discriminatively optimized parameters. However, also BNs with generatively optimized ML parameters can be of interest. The theoretical analysis of the effect of rounding given in Section 5.4.2 clearly also applies in this case.

In this section, we derive algorithms for computing likelihood-maximizing reduced-precision parameters. Similarly to the discriminative case, they are based on the BB algorithm.

Algorithm 8 Algorithm for approximate projection

Require: Vector $\hat{\mathbf{a}}$ to project with $\hat{a}^i_{j|\mathbf{h}} \in -\mathbb{B}^{b_i}_{b_f}$; number of integer bits b_i and fractional bits b_f

Ensure: $a^i_{j|\mathbf{h}} \in -\mathbb{B}^{b_i}_{b_f}$; feasibility of (5.134), i.e. $\sum_j \exp(M) \leq 1 + \xi(|\mathbf{val}(X_i)|, b_i, b_f)$, where $M = -2^{b_i} + 2^{-b_f}$

```
q ← 2^{-b_f}                                   ▷ quantization interval
a ← â
for all i, h do
    while Σ_j exp(a^i_{j|h}) ≤ 1 + ξ(|val(X_i)|, b_i, b_f) do   ▷ repeat until feasible
        j' ← arg max_j a^i_{j|h}               ▷ find largest component
        a^i_{j'|h} ← a^i_{j'|h} − q            ▷ reduce component by q
    end while
end for
return a
```

5.C.1 Generative Objective

We use the same notation as for the discriminative case, cf. Section 5.4.

In principle, generative parameters can be computed by first learning BN parameters using full-precision floating-point computations and subsequent rounding to the desired number format. However, these parameters will in general not be (sub-)normalized and we aim at a more principled approach.

Our approach is inspired, by learning the *most generative reduced-precision parameters*. The exact meaning will become clear immediately. Classical ML parameter learning for BNs can be stated as

$$\begin{aligned} \underset{\mathbf{w}}{\text{maximize}} \quad & \sum_{n=1}^{N} \boldsymbol{\phi}(c^{(n)}, \mathbf{x}^{(n)})^T \mathbf{w} \\ \text{s.t.} \quad & \sum_{j=1}^{|\mathbf{val}(X_i)|} \exp(w^i_{j|\mathbf{h}}) = 1 \qquad \forall i, \mathbf{h}. \end{aligned} \tag{5.135}$$

Note that we can immediately relax the constraints to

$$\sum_{j=1}^{|\mathbf{val}(X_i)|} \exp(w^i_{j|\mathbf{h}}) \leq 1 \qquad \forall i, \mathbf{h}, \tag{5.136}$$

without changing the optimal solution as long as $\sum_{n=1}^{N} \boldsymbol{\phi}(c^{(n)}, \mathbf{x}^{(n)})^i_{\mathbf{h}} \neq \mathbf{0}$, where $\boldsymbol{\phi}(c^{(n)}, \mathbf{x}^{(n)})^i_{\mathbf{h}}$ is the subset of entries of $\boldsymbol{\phi}(c^{(n)}, \mathbf{x}^{(n)})$ corresponding to node i given parent state $\mathbf{h}$.

We now adopt (5.135) for learning reduced-precision fixed-point parameters. We apply the above relaxation and restrict the parameters in $\mathbf{w}$ to the set $-\mathbb{B}^{b_i}_{b_f}$. Then, maximum likelihood learning *translates* to

$$\begin{aligned} \underset{\mathbf{w}}{\text{maximize}} \quad & \sum_{n=1}^{N} \boldsymbol{\phi}(c^{(n)}, \mathbf{x}^{(n)})^T \mathbf{w} \\ \text{s.t.} \quad & \sum_{j=1}^{|\mathbf{val}(X_i)|} \exp(w^i_{j|\mathbf{h}}) \leq 1 \qquad \forall i, \mathbf{h} \\ & \mathbf{w}^i_{j|\mathbf{h}} \in -\mathbb{B}^{b_i}_{b_f} \qquad \forall i, j, \mathbf{h}. \end{aligned} \tag{5.137}$$

The above problem is nonlinear and hard to solve.

To solve problem (5.137), we resort to a BB algorithm. Note that (5.137) decomposes according to structure $\mathcal{G}$ of the BN, i.e. instead of solving the problem directly, one can solve the (smaller) subproblems

$$\begin{aligned} \underset{\mathbf{w}_{\mathbf{h}}^i}{\text{maximize}} \quad & \sum_{n=1}^{N} \boldsymbol{\phi}(c^{(n)}, \mathbf{x}^{(n)})_{\mathbf{h}}^{i,T} \mathbf{w}_{\mathbf{h}}^i \\ \text{s.t.} \quad & \sum_{j=1}^{|\mathbf{val}(X_i)|} \exp(w_{j|\mathbf{h}}^i) \leq 1 \\ & w_{j|\mathbf{h}}^i \in -\mathbb{B}_{b_f}^{b_i} \qquad \forall j \end{aligned} \tag{5.138}$$

for all possible i and $\mathbf{h}$, where $\mathbf{w}_{\mathbf{h}}^i$ is the subset of entries of $\mathbf{w}$ corresponding to node i given parent state $\mathbf{h}$. Consequently, the reduced-precision parameters can be learned for every CPT separately. Note, that in general the solution to the above problem will not be properly normalized, i.e. $\sum_j \exp(w_{j|\mathbf{h}}^i) < 1$.

To this end, note that for low bit-widths the solutions of (5.138) can at a first look be counter-intuitive. For example, consider a RV C with no parents and two states. Assume that $\sum_{n=1}^{N} \boldsymbol{\phi}(c^{(n)}) = [100, 10]$. Then, the corresponding ML parameters are $\mathbf{w}^{\text{ML}} = [-0.0953, -2.3979]$. We want to determine parameters that can be represented using two bits (and no fractional bits). By rounding down the ML parameters to the closest integer values, we obtain $\mathbf{w}^{\text{RD}} = [-1, -3]$, with $\exp(-1) + \exp(-3) = 0.4177$. In contrast, when solving (5.138), we obtain $\mathbf{w} = [-1, -1]$, with $\exp(-1) + \exp(-1) = 0.7358$. While $\mathbf{w}$ is *better normalized*, both states of the variable are equally likely.

5.C.2 Likelihood Maximizing Reduced-Precision Parameters

For efficiently finding global minimizers of (5.138), we propose to use a BB algorithm [11] and a greedy heuristic for solving intermediate optimization problems.

Branch and Bound Algorithm.

The optimal reduced-precision BN parameters in the sense of (5.138) have to be searched in a discrete solution space, i.e. $\mathbf{w}_{j,\mathbf{h}}^i \in -\mathbb{B}_{b_f}^{b_i}$. For optimization, a BB algorithm is used. Refer to Section 5.4 for a detailed description of the BB algorithm. For our considerations, it is only important to note that the algorithm has to solve a large number of optimization problems with similar form as (5.138), where the constraints $w_{j|\mathbf{h}}^i \in -\mathbb{B}_{b_f}^{b_i}$ are replaced by lower and upper bounds on $w_{j|\mathbf{h}}^i$ according to the search tree — details are provided below.

Greedy Algorithm.

To enhance the efficiency of the BB algorithm, we provide a greedy method to solve problems of the form

$$\begin{aligned} \underset{\mathbf{w}_{\mathbf{h}}^i}{\text{maximize}} \quad & \mathbf{a}_{\mathbf{h}}^{i,T} \mathbf{w}_{\mathbf{h}}^i \\ \text{s.t.} \quad & \sum_{j=1}^{|\mathbf{val}(X_i)|} \exp(w_{j|\mathbf{h}}^i) \leq 1 \\ & l_j \leq w_{j|\mathbf{h}}^i \leq u_j \qquad \forall j. \end{aligned} \tag{5.139}$$

These problems are solved by the BB algorithm for every leaf node in the search tree. In detail: Problem (5.139) is solved for some variable X_i assuming parent state $\mathbf{h}$, i.e. the corresponding frequency counts are $\mathbf{a}_{\mathbf{h}}^i = \sum_{n=1}^{N} \boldsymbol{\phi}(c^{(n)}, \mathbf{x}^{(n)})_{\mathbf{h}}^i$. According to the search tree,

the lower bounds and upper bounds are $\mathbf{l}_{\mathbf{h}}^{i}$ and $\mathbf{u}_{\mathbf{h}}^{i}$, respectively. To solve the problem, we propose to use Algorithm 9. Thus, the BB algorithm invokes Algorithm 9 with parameters $(\mathbf{a}_{\mathbf{h}}^{i}, \mathbf{l}_{\mathbf{h}}^{i}, \mathbf{u}_{\mathbf{h}}^{i}, 1)$, where the last element of the tuple corresponds to the normalization constant A. This constant is initially set to 1 as we are interested in parameters that are as normalized as possible.

Algorithm 9 determines the active constraints of an optimal solution of (5.139) in a greedy manner and thereby solves the optimization problem. In more detail: First, the algorithm checks whether setting all entries to the corresponding upper bounds is feasible — if this is the case, then this maximizes the objective in (5.139). Otherwise, the ML solution and its clipped variant $\mathbf{w}^{B}$ are computed. Then we distinguish three cases: (1) If the clipped version is super-normalized and there exist some entries that are smaller than their lower bounds, we set these entries to their lower bounds (the idea is that the bounds must be satisfied in any case; hence we set variables that are *unimportant* in terms of the objective to their lower bounds; in this way we have the largest possible freedom in selecting the values of the remaining variables without violating the normalization constraints). (2) Otherwise, if there exist entries in the ML solution that are larger than the corresponding upper bounds, we perform the same steps as above (with lower and upper bounds exchanged). (3) Otherwise, the ML solution is feasible and corresponds to the optimal solution. Finally, the algorithm performs the same operation for all variables whose values have not been determined already but with reduced normalization A.

Despite its greedy nature, the algorithm optimally solves (5.139), as ensured by the following theorem.

Theorem 10 (Optimality of Greedy ML)**.** *If the input* $(\mathbf{a}, \mathbf{l}, \mathbf{u}, A)$ *to Algorithm 9 corresponds to a feasible optimization problem according to* (5.139) *and if* $a_j > 0$ *for all* j*, then the algorithm solves* (5.139) *optimally.*

The proof of optimality is rather lengthy and technical; it is provided in Appendix 5.D.

5.C.3 Remarks on the Greedy Algorithm

We do not present any experimental results; they are not competitive (refer to the end of Section 5.C.1 for an explanation). Nevertheless, the greedy algorithm for solving box-constrained ML learning clearly has a wider range of applications than learning reduced-precision parameters. For example, this type of learning is of interest in cases in which a domain expert can specify probability ranges for certain events, e.g. the probability of having fever given that a person suffers from cough. To the best of our knowledge, the proposed algorithm is advantageous over other algorithms for the same purpose present in the literature:

- Tong and Ji [31] considered parameter learning for BNs with *qualitative constraints*, e.g. *range constraints* and *relationship constraints*. The range constraints case corresponds to ML parameter learning under box-constraints. For learning, they propose an algorithm for finding the active range constraints. However, this algorithm essentially tests all possible pairs of active constraints (although they provide a sufficient condition to prune impossible constraints). Nevertheless, for CPTs with large number of states and many feasible combinations of the constraint, this algorithm becomes infeasible (without pruning constraints, and given constraints for every possible state, there are 2^{L} constraints combinations for a CPT with L states).

- Niculescu et al. [17] considered learning of BNs with various types of constraints, e.g. parameter sharing constraints, constrained Dirichlet priors, and *upper bounds on sums of parameters*. This last type of constraints include ML parameter learning under box-constraints, where only upper bounds are given, as a special case. Thus, their considerations are more general in the sense that they consider sums of parameters, but less general in the sense that they cannot consider lower bounds. We are confident, that our algorithm can easily be generalized to also incorporate constraints on sums of parameters.

Algorithm 9 MLconstr($\mathbf{a}$, $\mathbf{l}$, $\mathbf{u}$, A): Solve problem (5.139)

Require: $(\mathbf{a}, \mathbf{l}, \mathbf{u}, A)$, where $\mathbf{a}$ are frequency counts, $\mathbf{l}$ lower bounds, $\mathbf{u}$ upper bounds, and A is a normalization constant

Ensure: Input problem is feasible; $\mathbf{w}$ solves (5.139)

$\mathcal{V} \leftarrow \{1, 2, \ldots, \text{length}(\mathbf{a})\}$ ▷ $\mathcal{V}$ is a set of vector indices of $\mathbf{a}$

if $\sum_{j \in \mathcal{V}} \exp(u_j) \leq A$ **then** ▷ Is setting $\mathbf{w}$ to $\mathbf{u}$ is feasible?

  $\mathbf{w} \leftarrow \mathbf{u}$

  return $\mathbf{w}$

end if

$\mathbf{w}^{\text{ML}} \leftarrow \log\left(A \frac{\mathbf{a}}{\sum_{j'} a_{j'}}\right)$ ▷ Compute ML solution (element-wise)

$\mathbf{w} \leftarrow \mathbf{w}^{\text{ML}}$ ▷ Initialize $\mathbf{w}$ to ML solution

$\mathbf{w}^B \leftarrow \min(\max(\mathbf{w}^{\text{ML}}, \mathbf{l}), \mathbf{u})$ ▷ $\mathbf{w}^B$ is clipped version of $\mathbf{w}$

if $\sum_{j \in \mathcal{V}} \exp(w_j^B) > A$ and $\exists j \colon w_j^{\text{ML}} < l_j$ **then** ▷ $\mathbf{w}^B$ is super-normalized

  $\mathcal{C} \leftarrow \{j \colon w_j^{\text{ML}} < l_j\}$

  $\mathbf{w}_{\mathcal{C}} \leftarrow \mathbf{l}_{\mathcal{C}}$

else if $\exists j \colon w_j^{\text{ML}} > u_j$ **then** ▷ $\mathbf{w}^B$ is sub-normalized

  $\mathcal{C} \leftarrow \{j \colon w_j^{\text{ML}} > u_j\}$

  $\mathbf{w}_{\mathcal{C}} \leftarrow \mathbf{u}_{\mathcal{C}}$

else

  return $\mathbf{w}$

end if

$\mathcal{V}^c \leftarrow \mathcal{V} \setminus \{\mathcal{C}\}$ ▷ Entries in $\mathbf{w}$ that are not set yet

$A \leftarrow A - \sum_{j \in \mathcal{C}} \exp(w_j)$

if $\mathcal{V}^c \neq \emptyset$ **then** ▷ Are we done?

  $\mathbf{w}_{\mathcal{V}^c} \leftarrow \text{MLconstr}(\mathbf{a}_{\mathcal{V}^c}, \mathbf{l}_{\mathcal{V}^c}, \mathbf{u}_{\mathcal{V}^c}, A)$ ▷ Recursive call for unset variables

end if

return $\mathbf{w}$

5.D Proof of Correctness of Greedy Maximum Likelihood Learning

For convenience, we restate the optimization problem of interest for a single CPT, i.e.

$$\begin{aligned} \underset{\mathbf{w}}{\text{maximize}} \quad & \mathbf{a}^T \mathbf{w} \\ \text{s.t.} \quad & \sum_j \exp(w_j) \leq 1 \\ & l_j \leq w_j \leq u_j \qquad \forall j. \end{aligned} \tag{5.140}$$

The Lagrangian of this problem is given as

$$\mathcal{L}(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \gamma) = \mathbf{a}^T \mathbf{w} + \sum_j \alpha_j \left(w_j - l_j\right) + \sum_j \beta_j \left(u_j - w_j\right) + \gamma \left(1 - \sum_j \exp(w_j)\right), \tag{5.141}$$

where $\alpha_j \geq 0$ is the Lagrangian multiplier of $l_j \leq w_j$, where $\beta_j \geq 0$ is the Lagrangian multiplier of $w_j \leq u_j$, and where $\gamma \geq 0$ is the Lagrangian multiplier of $\sum_j \exp(w_j) \leq 1$.

For reference, we also state the ML parameter learning problem, i.e.

$$\begin{aligned} \underset{\mathbf{w}}{\text{maximize}} \quad & \mathbf{a}^T \mathbf{w} \\ \text{s.t.} \quad & \sum_j \exp(w_j) \leq 1. \end{aligned} \tag{5.142}$$

The corresponding Lagrangian is

$$\mathcal{L}^{\text{ML}}(\mathbf{w}, \gamma^{\text{ML}}) = \mathbf{a}^T \mathbf{w} + \gamma^{\text{ML}} \left(1 - \sum_j \exp(w_j)\right), \tag{5.143}$$

where $\gamma^{\text{ML}} \geq 0$ is the Lagrangian multiplier of $\sum_j \exp(w_j) \leq 1$. Note that whenever $\mathbf{a} \neq \mathbf{0}$, the stationarity condition of (5.143) implies that

$$\gamma^{\text{ML}} = \frac{1}{\sum_j \exp(w_j)} \sum_j a_j. \tag{5.144}$$

Furthermore, for $\mathbf{a} \neq \mathbf{0}$, the optimal solution $\mathbf{w}^{\text{ML}}$ of (5.142) satisfies $\sum_j \exp(w_j^{\text{ML}}) = 1$.

We can now state the proof of correctness of the greedy algorithm:

Proof of Theorem 10. The proof is by induction over the number of elements n of $\mathbf{a}$. The base case, where $\mathbf{a}$ consists of only a single element is clearly solved optimally by the algorithm. In the inductive step $(n-1) \rightarrow n$, we assume that $\mathbf{a}$ has $n \geq 2$ elements and that we can solve problems where $\mathbf{a}$ has size less than or equal to $(n-1)$ optimally. Denote by $\mathbf{w}^{\text{ML}}$ and $\mathbf{w}^{\text{B}}$ the quantities according to lines 6 and 8 of the algorithm, respectively. We distinguish the three cases corresponding to lines 9, 12 and 15 of the algorithm, respectively:

- **Line 9.** Assume $\sum_j \exp(w_j^B) > A$ and that there exists some $w_j^{\text{ML}} < l_j$, i.e. the condition in line 9 is satisfied. Set $\mathcal{C} = \{j \colon w_j^{\text{ML}} < l_j\}$. To show that the algorithm is optimal, we have to prove that any optimal solution $\mathbf{w}^*$ of (5.140) satisfies $w_j^* = l_j$ for all $j \in \mathcal{C}$. We show this by contradiction. Therefore, let $\mathbf{w}^*$ be an optimal solution of (5.140) and assume that there exists a $k \in \mathcal{C}$ that satisfies $w_k^* > l_k$. The contradiction is obtained by deriving the statements $\gamma < \gamma^{\text{ML}}$ and $\gamma > \gamma^{\text{ML}}$.

  $\gamma < \gamma^{\text{ML}}$. Because of the complementary slackness conditions $\alpha_k(w_k^* - l_k) = 0$, the Lagrangian multiplier α_k must be zero. Hence, the stationarity condition from (5.141) for variable k reads as

  $$-a_k + \beta_k + \gamma \exp(w_k^*) = 0, \tag{5.145}$$

with $\beta_k \geq 0$ and $\gamma \geq 0$. Similarly, the stationarity condition of (5.143) of variable k of the ML problem reads as

$$-a_k + \gamma^{\mathrm{ML}} \exp(w_k^{\mathrm{ML}}) = 0, \tag{5.146}$$

where $\gamma^{\mathrm{ML}} \geq 0$. Setting (5.145) equal to (5.146), we obtain

$$\beta_k + \gamma \exp(w_k^*) = \gamma^{\mathrm{ML}} \exp(w_k^{\mathrm{ML}}). \tag{5.147}$$

As $\beta_k \geq 0$, $w_k^* > w_k^{\mathrm{ML}}$ and since $\gamma^{\mathrm{ML}} > 0$ (cf. Equation (5.144)),

$$\gamma < \gamma^{\mathrm{ML}}. \tag{5.148}$$

$\gamma > \gamma^{\mathrm{ML}}$. We start with a claim:

Claim: As $w_k^* > l_k > w_k^{\mathrm{ML}}$, there exists some w_m^* that satisfies $w_m^* < w_m^{\mathrm{ML}}$ and $w_m^* < u_m$.

Proof. Assume that $w_k^* > l_k > w_k^{\mathrm{ML}}$ and that all w_i^* satisfy $w_i^* \geq w_i^{\mathrm{ML}}$ or $w_i^* \geq u_i$. Set $\mathcal{A} = \{l \colon w_l^* \geq u_l \text{ or } w_l^{\mathrm{ML}} \geq u_l\}$ and $\mathcal{B} = \{l \colon l \notin \mathcal{A}\}$. Then, as $\mathbf{w}^*$ satisfies the constraints of (5.140),

$$\begin{aligned}
\sum_i \exp(w_i^*) &= \sum_{i \in \mathcal{A}} \exp(u_i) + \sum_{i \in \mathcal{B}} \exp(w_i^*) && (5.149)\\
&= \sum_{i \in (\mathcal{A} \setminus \{k\})} \exp(u_i) + \sum_{i \in (\mathcal{B} \setminus \{k\})} \exp(w_i^*) + \exp(w_k^*) && (5.150)\\
&> \sum_{i \in (\mathcal{A} \setminus \{k\})} \exp(u_i) + \sum_{i \in (\mathcal{B} \setminus \{k\})} \exp(w_i^*) + \exp(l_k) && (5.151)\\
&\geq \sum_i \exp(w_i^B) && (5.152)\\
&> A, && (5.153)
\end{aligned}$$

where the strict inequality is because of $w_k^* > l_k$. This violates the normalization constraint. □

According to the above claim, there is some m such that w_m^* satisfies $w_m^* < u_m$. Therefore, the Lagrangian multiplier β_m of an optimal solution must satisfy $\beta_m = 0$. Thus, we obtain — similarly as above — from the stationarity conditions of (5.141) and (5.143), the equation

$$-a_m - \alpha_m + \gamma \exp(w_m^*) = -a_m + \gamma^{\mathrm{ML}} \exp(w_m^{\mathrm{ML}}), \tag{5.154}$$

where $\alpha_m \geq 0$ is the Lagrangian multiplier of the constraint $l_m \leq w_m$. Consequently,

$$\gamma \exp(w_m^*) \geq \gamma^{\mathrm{ML}} \exp(w_m^{\mathrm{ML}}). \tag{5.155}$$

As $w_m^* < w_m^{\mathrm{ML}}$ (cf. claim above) and because of $\gamma^{\mathrm{ML}} > 0$, we obtain

$$\gamma > \gamma^{\mathrm{ML}}. \tag{5.156}$$

Thus (5.148) contradicts (5.156). Therefore, any optimal solution of the constrained problem must satisfy that $w_j^* = l_j$ for all $j \in \mathcal{C}$. The algorithm fixes these variables to their lower bounds in line 11 and solves the *remaining* subproblem in line 21. This subproblem has strictly less than n variables and can therefore be solved optimally according to the induction hypothesis. Note that the subproblem is clearly feasible. Consequently, this branch of the algorithm is optimal.

- **Line 12.** Assume there exists some $w_j^{\mathrm{ML}} > u_j$ and that $\sum_j \exp(w_j^B) \leq A$ or that all $w_j^{\mathrm{ML}} \geq l_j$, i.e. we consider the condition in line 12 is satisfied. Set $\mathcal{C} = \{j \colon w_j^{\mathrm{ML}} > u_j\}$. We have to prove that in this case any optimal solution $\mathbf{w}^*$ of (5.140) satisfies $\forall j \in \mathcal{C} \colon w_j^* = u_j$. We prove this by contraction, i.e. assume that some $k \in \mathcal{C}$ satisfies $w_k^* < u_k$.

  Note that the conditions $\forall j \colon w_j^{\mathrm{ML}} \geq l_j$ and $\exists k \colon w_k^{\mathrm{ML}} > u_k$ imply

  $$\sum_j \exp(w_j^B) < \sum_j \exp(w_j^{\mathrm{ML}}) \tag{5.157}$$

  $$= A. \tag{5.158}$$

  Consequently, it suffices to prove the statement for the conditions $\exists j \colon w_j^{\mathrm{ML}} > u_j$ and $\sum_j \exp(w_j^B) \leq A$. Again we derive a contradiction by deducing the statements $\gamma < \gamma^{\mathrm{ML}}$ and $\gamma > \gamma^{\mathrm{ML}}$.

  $\gamma > \gamma^{\mathrm{ML}}$. Because of the assumption $w_k^* < u_k$, the complementary slackness condition $\beta_k(u_k - w_k^*) = 0$ implies $\beta_k = 0$. Equating the stationarity conditions for variable k of (5.141) and (5.143) yields

  $$-a_k - \alpha_k + \gamma \exp(w_k^*) = -a_k + \gamma^{\mathrm{ML}} \exp(w_k^{\mathrm{ML}}). \tag{5.159}$$

  Consequently, because $w_k^* < w_k^{\mathrm{ML}}$ and $\alpha_k \geq 0$, $\gamma > \gamma^{\mathrm{ML}}$.

  $\gamma < \gamma^{\mathrm{ML}}$. Again, we start with a claim:

  Claim: Because of $w_k^{\mathrm{ML}} > u_k > w_k^*$, there is some m such that $w_m^* > w_m^{\mathrm{ML}}$ and $w_m^* > l_m$.

  Proof. We prove this by contradiction. Thus, assume that $\forall i : w_i^* \leq w_i^{\mathrm{ML}}$ or $w_i^* \leq l_i$. Define $\mathcal{A} = \{l \colon w_l^* \leq l_l \text{ or } w_l^{\mathrm{ML}} \leq l_l\}$ and $\mathcal{B} = \{l \colon l \notin \mathcal{A}\}$. Then,

  $$\sum_i \exp(w_i^*) = \sum_{i \in \mathcal{A}} \exp(l_i) + \sum_{i \in \mathcal{B}} \exp(w_i^*) \tag{5.160}$$

  $$= \sum_{i \in \mathcal{A} \setminus \{k\}} \exp(l_i) + \sum_{i \in \mathcal{B} \setminus \{k\}} \exp(w_i^*) + \exp(w_k^*) \tag{5.161}$$

  $$\leq \sum_{i \in \mathcal{A} \setminus \{k\}} \exp(w_i^B) + \sum_{i \in \mathcal{B} \setminus \{k\}} \exp(w_i^B) + \exp(w_k^*) \tag{5.162}$$

  $$< \sum_i \exp(w_i^B) \tag{5.163}$$

  $$\leq A. \tag{5.164}$$

  But then it is trivial to improve upon $\mathbf{w}^*$ by increasing w_k^* (which is feasible). Thus, this contradicts optimality of $\mathbf{w}^*$. □

  From the claim and the stationarity conditions for variable m, we immediately get

  $$-a_m + \beta_m + \gamma \exp(w_m^*) = -a_m + \gamma^{\mathrm{ML}} \exp(w_m^{\mathrm{ML}}). \tag{5.165}$$

  Consequently, $\gamma < \gamma^{\mathrm{ML}}$.

We derived a contradiction above, thus our assumption that there is some $k \in \mathcal{C}$ that satisfies $w_k^* < u_k$, must be incorrect. Therefore, any optimal solution of the constraint problem must satisfy that $w_j^* = u_j$ for all $j \in \mathcal{C}$. The algorithm fixes these variables to their upper bounds in line 14 and solves the *remaining* subproblem in line 21. This subproblem has strictly less than n variables and can therefore be solved optimally according to the induction hypothesis. Note that the subproblem is clearly feasible. Consequently, this branch of the algorithm is optimal.

- **Line 15.** The only remaining case deals with the conditions

$$\left(\sum_k \exp(w_k^B) \leq A \text{ or } \forall k\colon w_k^{\mathrm{ML}} \geq l_k\right) \text{ and } \forall k\colon w_k^{\mathrm{ML}} \leq u_k. \tag{5.166}$$

  We consider the following two cases:

  – Assume $\sum_k \exp(w_k^B) \leq A$ and $\forall k\colon w_k^{\mathrm{ML}} \leq u_k$. Furthermore, assume $\exists j : w_j^{\mathrm{ML}} < l_j$ and set $\mathcal{A} = \{l\colon w_l^{\mathrm{ML}} < l_l\}$, $\mathcal{B} = \{l\colon l \notin \mathcal{A}\}$. Then,

$$\begin{aligned} A &= \sum_j \exp(w_j^{\mathrm{ML}}) && (5.167)\\ &= \sum_{j\in\mathcal{A}} \exp(w_j^{\mathrm{ML}}) + \sum_{j\in\mathcal{B}} \exp(w_j^{\mathrm{ML}}) && (5.168)\\ &< \sum_{j\in\mathcal{A}} \exp(l_j) + \sum_{j\in\mathcal{B}} \exp(w_j^{\mathrm{ML}}) && (5.169)\\ &\leq \sum_j \exp(w_j^B) && (5.170)\\ &\leq A. && (5.171) \end{aligned}$$

    This is a contradiction, thus $\forall k\colon w_k^{\mathrm{ML}} \geq l_k$ must hold. Consequently, it suffices to consider the case described below.

  – If $\forall k\colon w_k^{\mathrm{ML}} \geq l_k$ and $\forall k\colon w_k^{\mathrm{ML}} \leq u_k$ are satisfied, then the ML solution is feasible for the constrained problem (5.140) and thus solves it optimally.

As all branches of the algorithm are optimal, the whole algorithm is optimal. □

Bibliography

[1] Davide Anguita, Alessandro Ghio, Stefano Pischiutta, and Sandro Ridella. A Support Vector Machine with Integer Parameters. *Neurocomputing*, 72(1-3):480–489, 2008.

[2] Kevin Bache and Moshe Lichman. UCI Machine Learning Repository, 2013. URL `http://archive.ics.uci.edu/ml`.

[3] José M. Bernardo and Adrian F. M. Smith. *Bayesian Theory*. John Wiley & Sons, New York, 1994.

[4] Hei Chan and Adnan Darwiche. When do Numbers Really Matter? *Journal of Artificial Intelligence Research*, 17(1):265–287, 2002.

[5] Hei Chan and Adnan Darwiche. Sensitivity Analysis in Bayesian Networks: From Single to Multiple Parameters. In *Uncertainty in Artificial Intelligence (UAI)*, pages 67–75, 2004.

[6] Nir Friedman, Dan Geiger, and Moises Goldszmidt. Bayesian Network Classifiers. *Machine Learning*, 29:131–163, 1997.

[7] Russell Greiner, Xiaoyuan Su, Bin Shen, and Wei Zhou. Structural Extension to Logistic Regression: Discriminative Parameter Learning of Belief Net Classifiers. *Machine Learning*, 59(3):297–322, 2005.

[8] Yuhong Guo, Dana Wilkinson, and Dale Schuurmans. Maximum Margin Bayesian Networks. In *Uncertainty in Artificial Intelligence (UAI)*, pages 233–242, 2005.

[9] David Heckerman, Dan Geiger, and David M. Chickering. Learning Bayesian Networks: The Combination of Knowledge and Statistical Data. *Machine Learning*, 20: 197–243, 1995.

[10] Joe D. Hoffman and Steven Frankel. *Numerical Methods for Engineers and Scientists*. CRC Press, 2nd edition, 2001.

[11] Reiner Horst and Hoang Tuy. *Global Optimization: Deterministic Approaches*. Springer-Verlag, 1996.

[12] Alexander T. Ihler, John W. Fischer III, and Alan S. Willsky. Loopy Belief Propagation: Convergence and Effects of Message Errors. *Journal of Machine Learning Research*, 6: 905–936, 2005.

[13] Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

[14] Ailsa H. Land and Alison G. Doig. An Automatic Method of Solving Discrete Programming Problems. *Econometrica*, 28(3):pp. 497–520, 1960.

[15] Dong-U. Lee, Altaf Abdul Gaffar, Ray C. C. Cheung, Oskar Mencer, Wayne Luk, and George A. Constantinides. Accuracy-Guaranteed Bit-Width Optimization. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 25(10):1990–2000, 2006.

[16] Jean-Michel Muller, Nicolas Brisebarre, Florent de Dinechin, Claude-Pierre Jeannerod, Vincent Lefèvre, Guillaume Melquiond, Nathalie Revol, Damien Stehlé, and Serge Torres. *Handbook of Floating-Point Arithmetic*. Birkhäuser Boston, 2010.

[17] Radu Stefan Niculescu, Tom M. Mitchell, and R. Bharat Rao. Bayesian Network Learning with Parameter Constraints. *Journal of Machine Learning Research*, 7:1357–1383, 2006.

[18] Alan V. Oppenheim, Ronald W. Schafer, and John R. Buck. *Discrete-time Signal Processing*. Prentice-Hall, Inc., 2nd edition, 1999.

[19] Michael L. Overton. *Numerical Computing with IEEE Floating Point Arithmetic - Including One Theorem, One Rule of Thumb, and One Hundred and One Exercices*. Society for Industrial and Applied Mathematics (SIAM), 2001.

[20] Athanasios Papoulis. *Probability, Random Variables, and Stochastic Processes*. Mc-Graw Hill, 1984.

[21] Robert Peharz and Franz Pernkopf. Exact Maximum Margin Structure Learning of Bayesian Networks. In *International Conference on Machine Learning (ICML)*, 2012.

[22] Robert Peharz, Sebastian Tschiatschek, and Franz Pernkopf. The Most Generative Maximum Margin Bayesian Networks. In *International Conference on Machine Learning (ICML)*, volume 28, pages 235–243, 2013.

[23] Franz Pernkopf and Jeff A. Bilmes. Efficient Heuristics for Discriminative Structure Learning of Bayesian Network Classifiers. *Journal of Machine Learning Research*, 11: 2323–2360, 2010.

[24] Franz Pernkopf, Michael Wohlmayr, and Manfred Mücke. Maximum Margin Structure Learning of Bayesian Network Classifiers. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2076–2079, 2011.

[25] Franz Pernkopf, Michael Wohlmayr, and Sebastian Tschiatschek. Maximum Margin Bayesian Network Classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 34(3):521–531, 2012.

[26] Franz Pernkopf, Robert Peharz, and Sebastian Tschiatschek. *Introduction to Probabilistic Graphical Models*, volume 1, chapter 18, pages 989–1064. Elsevier, 2014.

[27] Nico Piatkowski, Lee Sangkyun, and Katharina Morik. The Integer Approximation of Undirected Graphical Models. In *International Conference on Pattern Recognition Applications and Methods (ICPRAM)*, 2014.

[28] Teemu Roos, Hannes Wettig, Peter Grünwald, Petri Myllymäki, and Henry Tirri. On Discriminative Bayesian Network Classifiers and Logistic Regression. *Journal of Machine Learning Research*, 59(3):267–296, 2005.

[29] Jiang Su, Harry Zhang, Charles X. Ling, and Stan Matwin. Discriminative Parameter Learning for Bayesian Networks. In *International Conference on Machine Learning (ICML)*, pages 1016–1023. ACM, 2008.

[30] Jonathan Ying Fai Tong, David Nagle, and Rob. A. Rutenbar. Reducing Power by Optimizing the Necessary Precision/Range of Floating-point Arithmetic. *IEEE Transactions on Very Large Scale Integration Systems*, 8(3):273–285, 2000.

[31] Yan Tong and Qiang Ji. Learning Bayesian Networks with Qualitative Constraints. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, 2008.

[32] Sebastian Tschiatschek and Franz Pernkopf. On Reduced Precision Bayesian Network Classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, to be published.

[33] Sebastian Tschiatschek, Peter Reinprecht, Manfred Mücke, and Franz Pernkopf. Bayesian Network Classifiers with Reduced Precision Parameters. In *European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD)*, pages 74–89, 2012.

[34] Sebastian Tschiatschek, Carlos Eduardo Cancino Chacón, and Franz Pernkopf. Bounds for Bayesian Network Classifiers with Reduced Precision Parameters. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3357–3361, 2013.

[35] Sebastian Tschiatschek, Karin Paul, and Franz Pernkopf. Integer Bayesian Network Classifiers. In *European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD)*, to be published.

[36] Haiqin Wang. Using Sensitivity Analysis for Selective Parameter Update in Bayesian Network Learning. In *Association for the Advancement of Artificial Intelligence (AAAI)*, 2002.

[37] Eric W. Weisstein. *CRC Concise Encyclopedia of Mathematics*. Chapman and Hall/CRC, 1999.

[38] Bernard Widrow, István Kollár, and Ming-Chang Liu. Statistical Theory of Quantization. *IEEE Transactions on Instrumentation and Measurement*, 45(2):353–361, 1996.

[39] Marco Zaffalon. Credal Networks Classification. Technical report, 1999.

6 Conclusion and Future Work

In this thesis, we studied properties of maximum margin Bayesian networks (MM BNs), i.e. Bayesian network classifiers (BNCs) with parameters optimized for a large probabilistic margin. These MM BNs are inspired by support vector machines (SVMs) that aim to separate samples from different classes by a large margin in some feature space. Our investigations focused on three main topics, namely asymptotic consistency, hybrid learning, and reduced-precision analysis.

With respect to asymptotic consistency, we analyzed two definitions of MM BNs available in the literature. We found that both definitions are in general not asymptotically consistent. This result fits into results for general multiclass loss functions. We provided some specialized results that also take the true data distribution into account and that can ensure consistency, even if the used loss is not uniformly consistent. The identified deficiency of MM BNs is not eminent in the binary-class case. In experiments we demonstrated that MM BNs are nevertheless able to compensate for model-mismatch, i.e. when the true data distribution and the distributions representable by the learned models do not match. Furthermore, we proposed several approaches that potentially resolve the identified deficiencies.

In terms of hybrid learning, we extended MM BNs by incorporating a generative component into the maximum margin (MM) objective. This component consists of normalized frequency counts and is combined with the MM objective by a trade-off factor. In this novel formulation, MM BNs can be either interpreted as a linear SVM with a special regularizer or as a Bayesian network (BN) that optimizes both a generative and discriminative objective. The hybrid model interpretation allows one to naturally deal with missing features scenarios, simply by marginalization of the missing features. Furthermore, semi-supervised learning can be easily performed by considering unlabeled data in the generative component. Additionally, any probabilistic query is plausible in this model due to the generative component. State-of-the-art performance of the novel MM BN formulation is established in experiments and efficient algorithms for parameter learning in large scale scenarios are presented.

Furthermore, we considered reduced-precision implementations of BNCs. We focused on BNCs optimized for a large margin, either in terms of their structure or their parameters. In preliminary experiments, we investigated the classification performance of BNCs with parameters rounded to some specified precision. These experiments reveal that BNCs are well suited for reduced-precision implementations. We continued by deriving several types of classification performance bounds. These bounds can be used to analyze worst-case performance due to parameter rounding. In experiments, these bounds were evaluated and BNCs optimized for a large margin (both in terms of their parameters and their structure) were compared to generatively optimized BNCs in terms of robustness to parameter quantization and in terms of absolute classification performance. We found that MM BNs achieve higher classification rates even for low bit-widths, while BNCs with generatively optimized parameters are more robust to parameter quantization for low bit-widths meaning that less classifications change due to quantization. We ended our reduced-precision considerations by proposing an alternative to determining reduced-precision parameters for MM BNs by rounding. We slightly modified our formulation of MM BNs and proposed algorithms for maximizing this modified criterion over the search space of reduced-precision parameters. In several experiments, we demonstrated that parameters learned in this way yield better classification performance than parameters obtained by rounding. The proposed algorithm can therefore be used to determine reduced-precision parameters for BNCs with good classification performance before using these classifiers on a reduced-precision computing platform. Furthermore, we considered learning of reduced-precision parameters using reduced-precision arithmetic only. For this purpose, we proposed algorithms for learning generatively and discriminatively optimized parameters and demonstrated their effectiveness.

Some of the algorithms presented in this thesis have a wider range of possible appli-

cations than those they were used for. For example, the projection algorithms developed in Chapters 4 and 5 can be used whenever a projection of parameters onto the set of sub-normalized logarithmic probabilities over discrete variables is required. Such a projection is for example needed when performing projected gradient-ascent/descent, where parameters need to be projected to ensure (sub)-normalization. The presented projection algorithms are more efficient than general purpose projection algorithms.

Future Work

There are several intriguing research questions that could not be answered in the course of this thesis. Some of them, but by far not all, are summarized in the following:

- **Asymptotic consistency.** The inconsistent formulation of MM BNs should be analyzed in more detail.
  - We derived necessary conditions for the consistency in the case of fully connected graphs; however, the question whether these conditions are sufficient is unanswered.
  - A class of underlying data distributions could be identified for which the used loss function is consistent, i.e. instead of striving for universal consistency, consistency for some classes of distributions could be considered.
  - The *origin* of inconsistency in the case of model match should be analyzed in more detail, e.g. for the case of BNCs with naive Bayes (NB) structure.
  - The inconsistency could be resolved by using consistent loss functions.
- **Generative MM BNs.** The proposed formulation for generative MM BNs should be studied in further experiments. Especially, the partial generative nature of these models should be exploited:
  - How well do these models perform on more general probabilistic queries, e.g. computation of feature posteriors in case of missing features?
  - How well can we use the proposed objective for models including latent variables? Can the derived theoretical results be extended to such models? (For example: By introducing latent variables we will loose the convexity of the objective and cannot hope to have unique solutions, but we could still obtain normalized solutions automatically — does this hold true?)
- **Reduced-Precision Analysis.**
  - The obtained results could be extended to more general graph structures. Furthermore, analysis of related models, e.g. sum-product networks (SPNs), could be performed. SPNs represent the inference machine of probabilistic models and directly represent inference as a sequence of summations and multiplications. Thus, they are amenable for being studied using similar tools to those used in this thesis.
  - Reduced-precision analysis could be conducted for approximate inference algorithms like loopy belief propagation, with focus on classification performance. This would extent investigations by Ihler et al. that consider message propagation errors due to approximate messages (for example caused by rounding errors). However, their focus is on convergence issues and does not cover effects on the classification performance.

7 List of Publications

[1] Robert Peharz, Sebastian Tschiatschek, and Franz Pernkopf. The Most Generative Maximum Margin Bayesian Networks. In *International Conference on Machine Learning (ICML)*, volume 28, pages 235–243, 2013.

[2] Franz Pernkopf, Michael Wohlmayr, and Sebastian Tschiatschek. Maximum Margin Bayesian Network Classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 34(3):521–531, 2012.

[3] Franz Pernkopf, Robert Peharz, and Sebastian Tschiatschek. *Introduction to Probabilistic Graphical Models*, volume 1, chapter 18, pages 989–1064. Elsevier, 2014.

[4] Martin Ratajczak, Sebastian Tschiatschek, and Franz Pernkopf. Sum-Product Networks for Structured Prediction: Context- Specific Deep Conditional Random Fields. Technical report, 2014.

[5] Sebastian Tschiatschek and Franz Pernkopf. Convex Combinations of Maximum Margin Bayesian Network Classifiers. In *International Conference on Pattern Recognition Applications and Methods (ICPRAM)*, 2012.

[6] Sebastian Tschiatschek and Franz Pernkopf. Asymptotic Optimality of Maximum Margin Bayesian Networks. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 590–598, 2013.

[7] Sebastian Tschiatschek and Franz Pernkopf. On Reduced Precision Bayesian Network Classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, to be published.

[8] Sebastian Tschiatschek, Nikolaus Mutsam, and Franz Pernkopf. Handling Missing Features in Maximum Margin Bayesian Network Classifiers. In *International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6, 2012.

[9] Sebastian Tschiatschek, Peter Reinprecht, Manfred Mücke, and Franz Pernkopf. Bayesian Network Classifiers with Reduced Precision Parameters. In *European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD)*, pages 74–89, 2012.

[10] Sebastian Tschiatschek, Carlos Eduardo Cancino Chacón, and Franz Pernkopf. Bounds for Bayesian Network Classifiers with Reduced Precision Parameters. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3357–3361, 2013.

[11] Sebastian Tschiatschek, Rishabh Iyer, Haochen Wei, and Jeff Bilmes. Learning Mixtures of Submodular Functions for Image Collection Summarization. In *Neural Information Processing Systems (NIPS)*, submitted.

[12] Sebastian Tschiatschek, Karin Paul, and Franz Pernkopf. Integer Bayesian Network Classifiers. In *European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD)*, to be published.