© 2004 IEEE. Reprinted, with permission, from IEEE Transactions on Neural Networks 15(4), pp. 800–810, 2004.

Variational Learning and Bits-Back Coding: An Information-Theoretic View to Bayesian Learning

Antti Honkela and Harri Valpola

***Abstract*—The bits-back coding first introduced by Wallace in 1990 and later by Hinton and van Camp in 1993 provides an interesting link between Bayesian learning and information-theoretic minimum-description-length (MDL) learning approaches. The bits-back coding allows interpreting the cost function used in the variational Bayesian method called ensemble learning as a code length in addition to the Bayesian view of misfit of the posterior approximation and a lower bound of model evidence. Combining these two viewpoints provides interesting insights to the learning process and the functions of different parts of the model. In this paper, the problem of variational Bayesian learning of hierarchical latent variable models is used to demonstrate the benefits of the two views. The code-length interpretation provides new views to many parts of the problem such as model comparison and pruning and helps explain many phenomena occurring in learning.**

***Index Terms*—Bits-back coding, ensemble learning, hierarchical latent variable models, minimum description length, variational Bayesian learning.**

I. Introduction

THE problem of learning an optimal model for a given data set is usually divided into two subtasks: finding optimal values for parameters of a single model and finding the best model among a collection of different models. There is a large number of methods for solving the former problem ranging from *ad hoc* algorithms designed to mimic the assumed behavior of the human brain to others that minimize various cost functions. The second problem, model selection, is more difficult in general and it is consequently also more difficult to design good heuristics for this task.

Two wholesale solutions that can reliably tackle both parameter optimization and model selection exist: statistical Bayesian framework and information-theoretic minimum-description-length (MDL) principle. While these techniques are derived from very different starting points, both lead to very similar results and in some cases to exactly the same algorithms. It is thus reasonable to regard them as two viewpoints to the same underlying learning methodology.

Bayesian statistics can be derived from Cox's axioms stating that the subject should perform rationally with respect to the information he has of the world [1]. The beliefs of the subject are represented as probabilities. With given costs or utilities for different possible outcomes the Bayesian procedure provides the optimal decision based on the available data, thus being able to learn from the data optimally. MDL and the related minimum-message-length (MML) principle, on the other hand, are based on the premise of finding the simplest explanation to adequately model the available data. This is achieved by looking for the most compact encoding of the data. The corresponding model providing the most compact code is then the best model for the data.

The two frameworks, Bayesian statistics and MDL/MML, use two different languages, probabilities and code lengths, to speak of the same things. Both of these views offer unique benefits and using both of them leads to a more complete picture of the whole. While Bayesian statistics are better grounded, especially in relation to including utilities and decisions, the code length interpretation in MDL and MML is sometimes more concrete and can be of great help in understanding many of the phenomena occurring while applying the methods.

In this paper, we discuss the insights offered by the two viewpoints for practical problems we have encountered. The purpose of the paper is not to present new algorithms or theorems but to review and discuss the benefits offered by combining the two existing frameworks. The paper starts with a general introduction to Bayesian learning and MDL/MML in Section II. The specific link between the methods as formed by bits-back coding and variational Bayesian learning is presented in Section III. Section IV introduces the specific model and its learning procedure used in the examples in Section V. The paper concludes with discussion in Section VI.

II. Bayesian Learning and MDL

Let us consider the problem of finding a model for a given data set $\boldsymbol{X} = \{x_i(t)|i,t\}$. The model being fitted is parameterised with a set of parameters $\boldsymbol{\theta} = \{\theta_i|i\}$. The traditional solution to the problem is to use maximum-likelihood estimation to find a single set of values for the model parameters that maximizes the data likelihood $p(\boldsymbol{X}|\boldsymbol{\theta})$. This approach is susceptible to overfitting and does not offer any tools for model comparison and selection.

In Bayesian statistics, the values of the parameters can have a distribution too. Probability quantifies the subjective state of knowledge and all that is known about the parameters is contained in the posterior probability distribution $p(\boldsymbol{\theta}|\boldsymbol{X}, \mathcal{H})$ of the parameters given the data and all the assumptions related to

Manuscript received March 15, 2003; revised October 20, 2003. This work was supported by the European Commission project BLISS, and the Finnish Center of Excellence Programme (2000–2005) under the Project New Information Processing Principles.

The authors are with the Neural Networks Research Centre, Helsinki University of Technology, FI-02015 HUT, Finland (e-mail: antti.honkela@hut.fi; harri.valpola@hut.fi).

Digital Object Identifier 10.1109/TNN.2004.828762

1045-9227/04$20.00 © 2004 IEEE

using the given model $\mathcal{H}$. This distribution can be evaluated by means of the Bayes rule

$$p(\boldsymbol{\theta}|\boldsymbol{X},\mathcal{H}) = \frac{p(\boldsymbol{X}|\boldsymbol{\theta},\mathcal{H})p(\boldsymbol{\theta}|\mathcal{H})}{p(\boldsymbol{X}|\mathcal{H})}. \quad (1)$$

All predictions of the model are then evaluated by using all possible parameter values and weighting the results by the corresponding posterior probabilities, an operation known as marginalization. In this view, the contribution of any single parameter value to inference is zero as the probability of a single value under a continuous density is always zero.

Unfortunately the full posterior is too difficult to handle in most realistic problems and different ways to summarize it are required. These approximations range from point estimates such as maximum *a posteriori* (MAP) estimate to variational approximations and stochastic approximations provided by various Markov chain Monte Carlo (MCMC) methods. Because of the use of full distributions, exact Bayesian learning is insensitive to overfitting. It also provides a natural means of performing model comparison by using the same procedure to evaluate the posterior probabilities of different models. Different approximation techniques retain these properties to different extents. For a thorough introduction to Bayesian statistics, see, e.g., [2], [3].

Another interesting view to learning is provided by information theory and involves finding a model that can be used to encode the data in a compact manner. The idea of using compact coding for inductive inference was first proposed by Solomonoff in the early 1960s [4]. His approach was based on universal turing machines which limits its usefulness in practice. In 1968, Wallace and Boulton proposed the first learning algorithm based on minimum encoding and more classical statistical models [5]. Their approach, later known as MML inference, was interpreted as a tractable approximation to exact Bayesian inference. Since then there have been many specific algorithms based on the idea of finding the most compact representation for the data, some of which are more closely and some more distantly related to Bayesian learning [6]–[9]. An introduction to different approaches to minimum-encoding inference can be found in [10]–[12].

The fundamental idea behind minimum-encoding learning was distilled by Rissanen as the MDL principle [6]: choose the model that gives the shortest description of data. The description here means a code with which the receiver can reconstruct the original data. The MDL principle is intuitively appealing but it is not clear why it should provide a reasonable basis for learning. After all, the world is not a very simple thing to describe. The close relation of MDL with Bayesian statistics actually suggests that it is because of our limited knowledge that simplicity is a good thing.

The MDL principle leaves open many technical questions on what is a valid code but as it is not required to construct an actual code but rather to evaluate its length, Shannon's coding theorem [13] can be used to obtain a lower bound for the code length. Shannon's theorem states that data following a discrete distribution $p(x)$ cannot, on average, be coded using less than

$$L(x) = -\sum_i p(x_i)\log_2 p(x_i) = -\langle \log_2 p(x)\rangle \quad (2)$$

bits/sample. Here $\langle\cdot\rangle$ denotes expectation over the random variable. The discrete entropy defined in (2) can be generalized to continuous case by replacing the sum by a corresponding integral to evaluate the expectation over $p(x)$. The differential entropy thus derived is not an absolute quantity such as the discrete entropy but it nevertheless characterizes the length of the code required to encode the values of a variable. With this optimal code a single discrete data element x can be coded using $-\log_2 p(x)$ bits. In the continuous case coding arbitrary precision values with finite code is impossible and therefore the values are only coded up to a given fine tolerance ϵ. The corresponding code length of value x will then be approximately $-\log_2(p(x)\epsilon)$ bits.

The choice of 2 as the base of the logarithm above rises from the natural wish to evaluate the information content in bits. In many, especially continuous applications it is, however, more convenient to use natural logarithm instead. This amounts to changing the unit of measure from base-2 bits to base-e *nats*. In general nats can always be converted to bits by dividing the value by ln 2. From now on we will use notation log for the (natural) logarithm without explicitly stating the base e.

A simple way to use the MDL principle for learning is to postulate the following encoding: the data is modeled using a model $\mathcal{H}$ with parameters $\boldsymbol{\theta}$ which are encoded first and then the modeling error is encoded. This way the description length becomes

$$\begin{aligned} L(\boldsymbol{X}) &= L(\boldsymbol{\theta}) + L(\boldsymbol{X}|\boldsymbol{\theta}) \\ &= -\log\left(p(\boldsymbol{\theta}|\mathcal{H})\epsilon_\theta^{|\boldsymbol{\theta}|}\right) - \log\left(p(\boldsymbol{X}|\boldsymbol{\theta},\mathcal{H})\epsilon_x^{|\boldsymbol{X}|}\right) \end{aligned} \quad (3)$$

where $L(\boldsymbol{\theta}) = -\log(p(\boldsymbol{\theta}|\mathcal{H})\epsilon_\theta^{|\boldsymbol{\theta}|})$ is the description length of the parameters using a suitable (prior) distribution and precision ϵ_θ for them, $L(\boldsymbol{X}|\boldsymbol{\theta}) = -\log(p(\boldsymbol{X}|\boldsymbol{\theta},\mathcal{H})\epsilon_x^{|\boldsymbol{X}|})$ is the description length of the modeling error up to precision ϵ_x and $|\boldsymbol{X}|$ and $|\boldsymbol{\theta}|$ denote the numbers of elements in the sets $\boldsymbol{X}$ and $\boldsymbol{\theta}$, respectively. If the precisions ϵ_θ and ϵ_x are ignored, minimizing this description length is equivalent to finding the MAP estimate for the parameters.

III. Bits-Back Coding and Variational Methods

In 1990, Wallace presented an interesting new coding scheme to be used together with the MDL/MML learning principle [14]. The same idea has been later developed by numerous authors [15]–[19]. The name of bits-back coding is due to Hinton and van Camp [15], [16]. Wallace's scheme is based on the idea of using a code with redundant codewords and selecting the codeword according to some auxiliary information. The receiver can then later recover the auxiliary information, get his "bits back", by running the same learning algorithm and observing the choices that were made. The recovered bits compensate the additional cost caused by the redundancy in the codewords and it turns out that the resulting code length is never greater than when using nonredundant codes.

It also turns out that bits-back coding provides an interesting connection between information-theoretic MDL principle and Bayesian learning and especially variational approximation to it.

A. Bits-Back Coding

Let us consider the example above leading to the code length given in (3). Assuming for now that everything is discrete, the precisions ϵ_θ and ϵ_x can be ignored. Instead of choosing only one $\boldsymbol{\theta}$ or the corresponding code word, bits-back coding uses multiple alternatives that are chosen according to a distribution $q(\boldsymbol{\theta})$. The distribution $q(\boldsymbol{\theta})$ introduces redundancy in the coding and results in the expected code length of

$$\begin{aligned}\langle L(\boldsymbol{X})\rangle_{q(\boldsymbol{\theta})} &= \langle L(\boldsymbol{\theta})\rangle + \langle L(\boldsymbol{X}|\boldsymbol{\theta})\rangle \\ &= -\sum_{\boldsymbol{\theta}} q(\boldsymbol{\theta}) \log p(\boldsymbol{\theta}|\mathcal{H}) \\ &\quad -\sum_{\boldsymbol{\theta}} q(\boldsymbol{\theta}) \log p(\boldsymbol{X}|\boldsymbol{\theta},\mathcal{H}). \end{aligned} \tag{4}$$

This length is of course greater than the optimal $L(\boldsymbol{X})$ derived in (3) but now the code contains additional information in the choice of $\boldsymbol{\theta}$ that were used in the coding. The amount of this extra information that can be transmitted is given in (2) by the entropy of the distribution $q(\boldsymbol{\theta})$

$$H_q(\boldsymbol{\theta}) = -\sum_{\boldsymbol{\theta}} q(\boldsymbol{\theta}) \log q(\boldsymbol{\theta}). \tag{5}$$

This part can be used to carry other information than the original data $\boldsymbol{X}$ and should therefore not be included in the total description length of $\boldsymbol{X}$. This yields the following total expected bits-back code length of $\boldsymbol{X}$:

$$\begin{aligned} L_{q(\boldsymbol{\theta})}(\boldsymbol{X}) &= \langle L(\boldsymbol{X})\rangle_{q(\boldsymbol{\theta})} - H_q(\boldsymbol{\theta}) \\ &= \sum_{\boldsymbol{\theta}} q(\boldsymbol{\theta}) \log \frac{q(\boldsymbol{\theta})}{p(\boldsymbol{\theta}|\mathcal{H}) p(\boldsymbol{X}|\boldsymbol{\theta},\mathcal{H})}. \end{aligned} \tag{6}$$

The classical result of (3) is obtained as a special case when $q(\boldsymbol{\theta})$ is chosen to give probability of one to a single value $\boldsymbol{\theta}_0$ which maximizes $\langle L(\boldsymbol{X})\rangle_{q(\boldsymbol{\theta})}$ and nothing for the rest. In general it is possible to find a much shorter description by using a better $q(\boldsymbol{\theta})$.

It might not be immediately clear that the receiver can decode the secondary message hidden in the choice of $\boldsymbol{\theta}$. This can, however, be done with the following decoding scheme:

1) decode $\boldsymbol{\theta}$ from the first part of the message using $p(\boldsymbol{\theta}|\mathcal{H})$;
2) decode $\boldsymbol{X}$ from the second part of the message using $p(\boldsymbol{X}|\boldsymbol{\theta},\mathcal{H})$;
3) run the same algorithm as the sender used for computing the distribution $q(\boldsymbol{\theta})$ based on $\boldsymbol{X}$;
4) decode the secondary message from $\boldsymbol{\theta}$ using $q(\boldsymbol{\theta})$.

But what would be the optimal $q(\boldsymbol{\theta})$? The code length in (6) can be written as

$$\begin{aligned} L_{q(\boldsymbol{\theta})}(\boldsymbol{X}) &= \sum_{\boldsymbol{\theta}} q(\boldsymbol{\theta}) \log \frac{q(\boldsymbol{\theta})}{p(\boldsymbol{X},\boldsymbol{\theta}|\mathcal{H})} \\ &= \sum_{\boldsymbol{\theta}} q(\boldsymbol{\theta}) \log \frac{q(\boldsymbol{\theta})}{p(\boldsymbol{\theta}|\boldsymbol{X},\mathcal{H})} - \log p(\boldsymbol{X}|\mathcal{H}) \\ &= D\left(q(\boldsymbol{\theta})||p(\boldsymbol{\theta}|\boldsymbol{X},\mathcal{H})\right) - \log p(\boldsymbol{X}|\mathcal{H}) \end{aligned} \tag{7}$$

where $D(q(\boldsymbol{\theta})||p(\boldsymbol{\theta}|\boldsymbol{X},\mathcal{H}))$ is the Kullback–Leibler divergence between the coding distribution $q(\boldsymbol{\theta})$ and the posterior distribution of the parameters $p(\boldsymbol{\theta}|\boldsymbol{X},\mathcal{H})$ [20]. In the other term, $p(\boldsymbol{X}|\mathcal{H})$ is the model evidence which is independent of the values of $\boldsymbol{\theta}$ and also independent of $q(\boldsymbol{\theta})$. Thus, the code length can be minimized by minimizing the Kullback–Leibler divergence $D(q(\boldsymbol{\theta})||p(\boldsymbol{\theta}|\boldsymbol{X},\mathcal{H}))$. This can be done by setting $q(\boldsymbol{\theta}) = p(\boldsymbol{\theta}|\boldsymbol{X},\mathcal{H})$ in which case the divergence will be zero. Thus, the optimal $q(\boldsymbol{\theta})$ is equal to the Bayesian posterior distribution of the parameters. This results in code length $-\log p(\boldsymbol{X}|\mathcal{H})$ which is optimal according to Shannon's coding theorem. This shows that bits-back coding is actually an optimal coding scheme.

Using the posterior directly is not feasible in most realistic problems and some simpler alternatives, leading to larger code lengths, must be used in practice. The typical simpler alternatives are suitably factorized distributions as discussed in Section III-C.

The MDL principle can be readily extended to continuous data. This introduces the term $-|\boldsymbol{X}| \log \epsilon_x$ related to the coding precision ϵ_x of the data in (3), but the term is separable and does not depend on the model that is used.

In contrast, if some model parameters are continuous, the MDL framework is complicated by the precision ϵ_θ of the parameters. Selecting too low a value for ϵ_θ will increase the code length $L(\boldsymbol{\theta})$ of the parameters and selecting too high a value will lead to suboptimal choices of parameters which increases the code length $L(\boldsymbol{X}|\boldsymbol{\theta})$. Bits-back coding with its redundant coding is particularly useful in the continuous case as it is possible to let ϵ_θ to be arbitrarily close to zero without increasing the code length corresponding to $\boldsymbol{X}$. This is because $-|\boldsymbol{\theta}| \log \epsilon_\theta$ appears both in $\langle L(\boldsymbol{X})\rangle_{q(\boldsymbol{\theta})}$ and in $H_q(\boldsymbol{\theta})$ but these terms have opposite signs in (6) and the terms related to the precision ϵ_θ thus cancel out.

The only difference to the discrete case is the additive constant $-|\boldsymbol{X}| \log \epsilon_x$. As it does not depend on the approximation $q(\boldsymbol{\theta})$ it can be ignored and the message length in both discrete and continuous cases can be denoted as

$$L_{q(\boldsymbol{\theta})}(\boldsymbol{X}) = \mathcal{C} = \left\langle \log \frac{q(\boldsymbol{\theta})}{p(\boldsymbol{X}|\boldsymbol{\theta},\mathcal{H}) p(\boldsymbol{\theta}|\mathcal{H})} \right\rangle \tag{8}$$

where $\langle\cdot\rangle$ denotes expectation over the distribution $q(\boldsymbol{\theta})$.

B. Relation to the EM Algorithm

Many existing learning methods can be seen as specific examples of using the above methodology. The expectation-maximization (EM) algorithm, for instance, can be viewed as a specific method to minimize a cost function of (8) [21]. In this case, the set of unknown variables consists of some unobserved data $\boldsymbol{S} = \{\mathbf{s}(t)|t\}$ and the standard model parameters $\boldsymbol{\theta}$. The approximation is chosen to be of the form $q(\boldsymbol{S},\boldsymbol{\theta}) = q(\boldsymbol{S})q(\boldsymbol{\theta})$ with $q(\boldsymbol{\theta})$ being restricted to a delta distribution with only one value of parameters having probability (mass) of one. The learning process then proceeds by alternating updates of $q(\boldsymbol{S})$ in the E-step and $q(\boldsymbol{\theta})$ in the M-step. In the basic algorithm, both of these updates set the

corresponding part to the optimum of (8) given the current value of the other part.

C. The Bayesian Interpretation and Variational Learning

From a Bayesian point of view, the bits-back coding suggests using some kind of approximation $q(\boldsymbol{\theta})$ to approximate the true posterior $p(\boldsymbol{\theta}|\boldsymbol{X}, \mathcal{H})$. These kinds of approaches have been used for some time under the name of variational methods or especially in statistical mechanics as mean field methods [22]–[25].

Variational methods are used to decrease the number of posterior dependencies in too complex models. This can be done for instance by decoupling variables from the model with so called variational transformations until the learning problem of the remaining structure can be solved in a tractable manner [24]. This sequential approach is very flexible but it is difficult to develop a general theory for it.

An alternative approach is to fix a single structure for the approximation and then find the optimal solution in that class of approximating probability distributions. This leads to another way of deriving an algorithm equivalent to bits-back learning by considering the minimization of the Kullback–Leibler divergence between the approximation and the true posterior. The resulting Bayesian-learning algorithm is often called *ensemble learning* [17], [26].

The posterior approximation $q(\boldsymbol{\theta})$ in ensemble learning is usually chosen to be a product of independent distributions for some easily separable sets of parameters. In an EM-like situation with some unobserved data $\boldsymbol{S}$, an approximation of the form $q(\boldsymbol{S}, \boldsymbol{\theta}) = q(\boldsymbol{S})q(\boldsymbol{\theta})$ is typically used. The factors $q(\boldsymbol{S})$ and $q(\boldsymbol{\theta})$ are often further factored to smaller pieces, sometimes until the fully factorized form $q(\boldsymbol{\theta}) = \prod_i q(\theta_i)$. As these methods can be seen as generalizations of the EM algorithm, they are sometimes called variational EM methods.

D. Bayesian and Bits-Back Model Comparison

From a Bayesian perspective there is no need for any special model selection tools. The whole learning problem can be viewed as finding the joint posterior probability $p(\boldsymbol{\theta}_i, \mathcal{H}_i|\boldsymbol{X})$ of the model parameters $\boldsymbol{\theta}_i$ and the models $\mathcal{H}_i$. This view allows setting aside strict boundaries between learning model parameters and structures.

The standard approach here is to marginalize over the parameters to compare different models. The required posterior probabilities of the models can then be obtained from the Bayes rule as $p(\mathcal{H}_i|\boldsymbol{X}) \propto p(\boldsymbol{X}|\mathcal{H}_i)p(\mathcal{H}_i)$. In practice, the posterior probabilities are often so different that only one candidate model contributes significantly to the predictions of the combined models. Thus, it is reasonable to approximate the marginalization by using only the single best model. Whichever of the alternatives is chosen, the key quantity to evaluate is the *model evidence* $p(\boldsymbol{X}|\mathcal{H}_i)$ [27], [28].

The ensemble learning procedure of minimizing the Kullback–Leibler divergence between an approximate posterior $q(\boldsymbol{\theta}_i, \mathcal{H}_i)$ and the exact $p(\boldsymbol{\theta}_i, \mathcal{H}_i|\boldsymbol{X})$ can also be applied to approximate the exact inference. Performing the minimization yields the approximation [26]

$$q(\mathcal{H}_i) \propto \exp\left(-L_{q(\boldsymbol{\theta}_i|\mathcal{H}_i)}(\boldsymbol{X})\right) p(\mathcal{H}_i) \tag{9}$$

suggesting the use of $\exp(-L_{q(\boldsymbol{\theta}_i|\mathcal{H}_i)}(\boldsymbol{X}))$ in place of model evidence. This approach is actually more practical as it takes into account the actual parameter values learnt for the model. The exact evidence ignores the distribution of the probability mass in the parameter space and may thus favor, for instance, models with internal symmetries that allow multiple parameterizations of the same model of which only one is used in practice.

The model evidence relates closely to the code length of the model in MDL framework. The relation for optimal code $L_{\text{optimal}}(\boldsymbol{X}) = -\log p(\boldsymbol{X}|\mathcal{H})$ shows that MDL and Bayesian approaches agree in principle on what is the best model. Because the construction of the corresponding code in the MDL case or even evaluation of its length require use of the exact posterior which is typically intractable, the result has only a theoretical interest.

The corresponding general result for arbitrary $q(\boldsymbol{\theta})$

$$L_{q(\boldsymbol{\theta})}(\boldsymbol{X}) = D\left(q(\boldsymbol{\theta})||p(\boldsymbol{\theta}|\boldsymbol{X}, \mathcal{H})\right) - \log p(\boldsymbol{X}|\mathcal{H}) \tag{10}$$

is much more interesting in practice, although its deeper message is also rather obvious. If an imperfect approximation to the posterior is used, the resulting code will be longer than the minimum by the amount of information discrepancy between the approximation and the true posterior as measured by the Kullback–Leibler divergence. Evaluating the divergence is usually not possible as it would again require using the exact posterior. In any case, the actual code length will provide a lower bound for model evidence as

$$L_{q(\boldsymbol{\theta})}(\boldsymbol{X}) \geq -\log p(\boldsymbol{X}|\mathcal{H}) \tag{11}$$

implying

$$p(\boldsymbol{X}|\mathcal{H}) \geq \exp\left(-L_{q(\boldsymbol{\theta})}(\boldsymbol{X})\right) \tag{12}$$

which parallels (9) in an interesting way.

E. Combining the Two Views

The Bayesian and MDL views of ensemble learning and bits-back coding are two complementary ways of looking at the same thing that both offer unique benefits for the whole. The MDL approach provides a nice way to derive the cost function (8) and offers an interpretation even for the individual terms

$$\left\langle \log \frac{q(\theta_i)}{p(\theta_i|\mathcal{H})} \right\rangle \tag{13}$$

as code lengths of individual parameters.

From a purely Bayesian perspective, the cost function (10) calls for minimization of a somewhat ad hoc distance measure between the approximate and true posterior. As an additional bonus one also gets a convenient lower bound for model evidence as in (12). The resulting approximation does not appear to have a simple corresponding loss function [2] so from the Bayesian perspective it is not clear how the approximate marginalization using the approximate posterior relates to results of other methods. The Kullback–Leibler divergence is widely used for measuring the discrepancy between the approximate and the true posterior probability but in order to

minimize the expected loss of natural logarithmic score function, $D(p\|q)$ should be used instead of $D(q\|p)$. Unfortunately, these are not the same because Kullback–Leibler divergence is asymmetric and there is no apparent simple relation between the two forms [2], [19].

IV. Building Blocks for Hierarchical Models

The ensemble learning approach presented above has been used for a variety of different modeling problems ranging from learning multilayer neural networks [29] to learning hidden Markov models [30]. It has recently become very popular in the field of linear independent component analysis (ICA) [31]–[35]. The approach also provides suitable regularization for severely ill-posed nonlinear problems and has been successfully applied to nonlinear ICA [36]–[38] as well as nonlinear and switching state-space models [39], [40], to name a few examples. It also allows modeling of variance simultaneously with the mean in a way that would be impossible for methods based on conventional point estimates [41], [42].

In order to demonstrate the above principles in practice, we use a collection of "building blocks" called Bayes Blocks presented in [43]. These blocks allow easy definition and learning of many linear and nonlinear hierarchical latent variable model structures.

Most probabilistic models can be represented as graphical models. A graphical model is a directed acyclic graph (DAG) representation of the joint probability density of the model. The nodes of the graph correspond to different variables of the model and the edges between them represent dependence relations. The joint probability density of the model represented by a graph is a product of terms of the form p (node $|$ parents of node in the graph), where parents of node N are the nodes from which there are incoming edges to N. The intuitive interpretation of the graph is, thus, that the value of a variable represented by a node N is directly dependent only on the variables represented by the immediate parents of N, i.e., the distribution of the values of N is perfectly determined once the values of its parents are known [44], [45].

Let us assume that we are using a fully factorial posterior approximation

$$q(\boldsymbol{\theta}) = \prod_i q(\theta_i). \tag{14}$$

The case of a general (not necessarily fully) factorial approximation is similar with joint densities of groups of variables appearing as factors in (14) instead of densities $q(\theta_i)$ of individual variables.

The part of the cost function affected by variable θ_i of a graphical model can, up to an additive constant, be written as

$$\mathcal{C}(\theta_i) = \left\langle \log \frac{q(\theta_i)}{p\left(\theta_i|\mathrm{pa}(\theta_i), \mathcal{H}\right) p\left(\mathrm{ch}(\theta_i)|\theta_i, \mathrm{co\text{–}pa}(\theta_i), \mathcal{H}\right)} \right\rangle \tag{15}$$

where $\mathrm{pa}(\theta_i)$ denotes the parents, $\mathrm{ch}(\theta_i)$ the children and $\mathrm{co\text{–}pa}(\theta_i)$ the co-parents (other parents with at least one common child) of θ_i.

If a conjugate prior is used for θ_i, the optimal approximation $q(\theta_i)$ can typically be solved exactly assuming the distributions of all the other parameters $\boldsymbol{\theta} \setminus \{\theta_i\}$ are kept fixed. This leads to an alternating optimization algorithm where each parameter is updated in turn while all the others are kept fixed.

The restriction of using only conjugate priors severely limits the range of possible model structures as it is not possible to find reasonable conjugates for all distributions. This places restrictions on possible hierarchical models that can be handled.

A. Hierarchical Models of Variance

Let us consider for example the possible hierarchical priors for the parameters of a Gaussian distribution

$$\theta \sim N(\mu, \sigma^2). \tag{16}$$

The conjugate prior for the mean μ is Gaussian so it is easy to build a hierarchy for the means. The variances are, however, more difficult to handle. The conjugate prior of the variance σ^2 of a univariate Gaussian is the inverse gamma distribution

$$p(\sigma^2) = \mathrm{Inv\text{–}gamma}(\sigma^2; \alpha, \beta) = \frac{\beta^\alpha}{\Gamma(\alpha)} (\sigma^2)^{-(\alpha+1)} e^{-\frac{\beta}{\sigma^2}}. \tag{17}$$

Unfortunately, there is no simple conjugate prior for the parameter α of the inverse gamma distribution so it is not possible to handle a more complicated hierarchical model for the variance of a Gaussian using only conjugate priors.

The problem of building a hierarchical model for variance can be handled by using a nonconjugate log-normal prior for the variance. Thus, (16) can be written instead as

$$\theta \sim N\left(\mu, \exp(-v)\right) \tag{18}$$

where both μ and v have Gaussian priors and can thus be handled in a similar manner as θ. Thus, the parameterization allows construction of arbitrary hierarchies. As the prior of v is not conjugate, solving the approximate posterior $q(v)$ is more difficult. If $q(v)$ is restricted to be Gaussian it is nevertheless relatively easy to find the best approximation in this class of distributions as shown in [41]–[43].

Gaussian variables based on the model (18) form a good basis for building many different hierarchical-latent variable models. When combined with addition and multiplication which both transform Gaussian inputs to a Gaussian output, they allow for a wide class of linear and some nonlinear models. The nonlinearities here arise from the fact that a linear model for the variance of a variable results in a nonlinear model for the variable itself. More details on the exact implementation of the different blocks can be found in [41]–[43].

B. Variance Model for Independent Component Analysis

The hierarchical variance model described above can easily be used for performing ICA. The generative model for the observations $\mathbf{x}(t)$ in (noisy) ICA is

$$\mathbf{x}(t) = \mathbf{A}\mathbf{s}(t) + \mathbf{n}(t) \tag{19}$$

where $\mathbf{s}(t)$ are the independent sources, $\mathbf{A}$ is the mixing matrix and $\mathbf{n}(t)$ is Gaussian observation noise. In order to be able to separate the sources, $\mathbf{s}(t)$ must have a non-Gaussian prior. This

is achieved by using variance neurons to model the variance of each sample of $\mathbf{s}(t)$ separately as in

$$s_i(t) \sim N\left(0, \exp\left(-u_i(t)\right)\right). \tag{20}$$

The resulting Gaussian distribution with varying variance corresponds to a super-Gaussian prior distribution [41], [42]. Evaluation of the code length for this model is a lengthy but straightforward operation. For details on the evaluation of the cost function and the rest of the method, see [42].

C. Hierarchical Nonlinear Factor Analysis

Another application of the Bayes Blocks framework used in this paper is the hierarchical nonlinear factor analysis (HNFA) model presented in [38]. The generative model for nonlinear factor analysis is basically $\mathbf{x}(t) = \mathbf{C}\mathbf{s}(t) + \mathbf{f}(\mathbf{s}(t))$ with MLP-like structure $\mathbf{f}(\mathbf{s}(t)) = \mathbf{A}\boldsymbol{\phi}(\mathbf{B}\mathbf{s}(t) + \mathbf{b}) + \mathbf{a}$ and activation function $\phi(s) = \exp(-s^2)$ applied componentwise. In order to avoid problems caused by propagation of variance of the sources through multiple paths as explained in [36], the values of the hidden neurons are taken as latent variables $\mathbf{h}(t)$ so that

$$\mathbf{h}(t) \sim N\left(\mathbf{B}\mathbf{s}(t) + \mathbf{b}, \boldsymbol{\Sigma}_h\right) \tag{21}$$

and

$$\mathbf{x}(t) \sim N\left(\mathbf{C}\mathbf{s}(t) + \mathbf{A}\boldsymbol{\phi}\left(\mathbf{h}(t)\right) + \mathbf{a}, \boldsymbol{\Sigma}_x\right) \tag{22}$$

where $\boldsymbol{\Sigma}_h$ is the (diagonal) covariance matrix of the first-level sources and $\boldsymbol{\Sigma}_x$ is the noise covariance matrix.

D. Parameter Estimation

The basic learning procedure for models of the above form is straightforward; process the variables one at a time and update the corresponding factors of the approximation, while the others are kept fixed. The factors of the approximation are always updated to minimize the corresponding cost as presented in (15). This is usually not the most efficient way to perform the minimization, however, and the convergence of the learning process can be accelerated with a simple procedure as presented in [46]. This speedup was used in all the experiments.

E. Model Selection and Pruning

The above procedure works well for a single fixed model structure. However, as the flexibility of the building-block framework makes it easy to use many different kinds of models, it is often desirable to use and compare different model structures.

According to the MDL principle, the best model for the data is the one that yields the shortest code length for the data. Even though the absolute values of the cost function are, especially in the continuous case, more difficult to interpret, the differences tell directly how much more likely one model is than another. With typical differences of more than 100 nats leading to exp(100)-fold difference in likelihoods, it is easy to justify using only the most likely model as suggested in Section III-D.

Even though the code length allows easy comparison of different models it is not very efficient to run the learning algorithm separately for each possible candidate model structure. Instead, it would be desirable to adjust the model structure during learning. These adjustments can be roughly broken into two different kinds of operations; adding new structures and pruning old ones away.

With the cost function as a guide, pruning could be implemented by trying to remove a part, iterating the algorithm for a while, and see if the change shortened the code and return the removed part back if not. In order to make this more simple, the additional iterations are skipped and instead of actually removing a part its value is set equal to constant zero and the change of the code length is observed. This is slightly biased toward keeping existing structure because other parts of the model could often make up for the removed part if given the chance via iterating the learning algorithm for a while. This bias can be countered by making the removal even if it would seem to result in only a small enough increase in the code length.

With working method for pruning, addition is easy to handle by adding the candidate structure, iterating the algorithm for a while to get the values of the new variables updated, and then apply the above check whether they should be pruned away.

The HNFA method presented in [38] is a good example of learning the structure of the model. The model starts with a linear mapping from sources $\mathbf{s}(t)$ to the data $\mathbf{x}(t)$ and gradually builds the nonlinearity by adding the hidden nodes $h_i(t)$. The incoming weights of the added nodes are initialized randomly. In the beginning, all the generated candidates are added but later only the best looking ones are selected because the probability that a random candidate will survive is low. After adding new hidden nodes the learning algorithm is run for 30 iterations before applying pruning to remove unsuccessful additions and other redundant parts. In addition to complete hidden nodes, the individual weights of the weight matrices are also pruned and added in a similar manner. For more details on the procedure, see [38].

V. Experiments

To illustrate the presented method, we performed a set of experiments using different hierarchical latent variable model structures.[1] The purpose of these experiments is not to present the best performance these methods can achieve in modeling the data used here, but rather to demonstrate how the information-theoretic viewpoint helps in understanding the results and sometimes provides insights that suggest improvements in the models and learning algorithms.

A. Hierarchical Nonlinear Factor Analysis

As the first example, we shall review the development of the learning algorithm of the HNFA method which is based on the previously developed nonlinear factor analysis (NFA) method [36]. The main difference is that in HNFA, the hidden nodes $h_i(t)$ are latent variables with a noise model, while in NFA they are deterministic.

[1]The source code for the Bayes Blocks library used in most of the simulations is available at http://www.cis.hut.fi/projects/bayes/software/

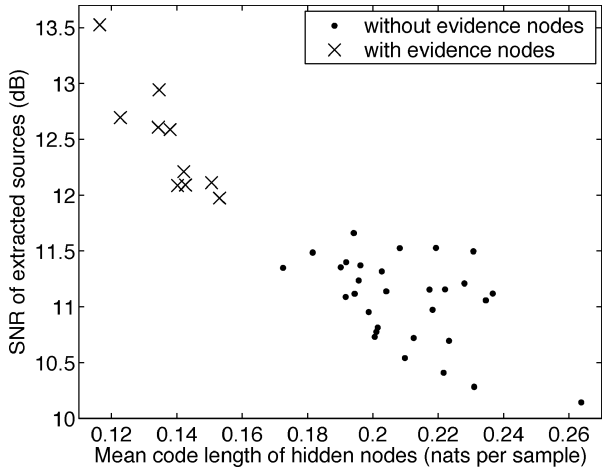

Fig. 1. The attained SNR of the separated sources as a function of mean code length of the hidden node values in the HNFA experiments with and without evidence nodes.

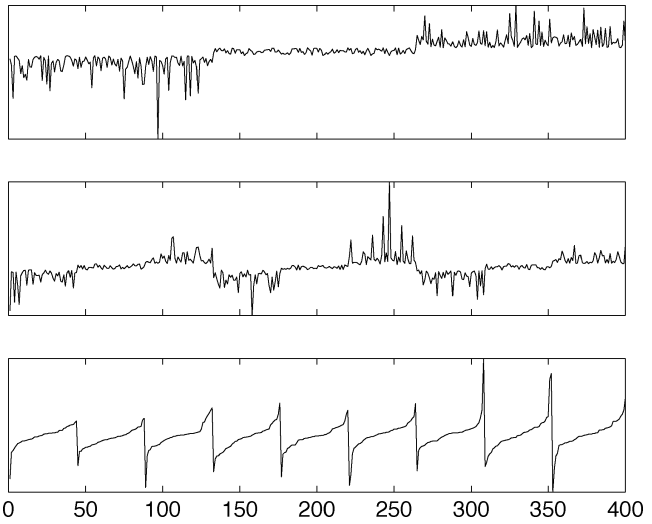

Fig. 2. The three original sources in the spike experiment. The source samples are actually i.i.d. but they have been sorted to show some structure in them.

Originally, HNFA performed unexpectedly poorly compared to NFA. The terms of the cost function shown in (13) have an information-theoretic interpretation as coding lengths of different parameters and this view proved invaluable in analysing the performance. It turned out that the coding length of several hidden nodes $h_i(t)$ was comparable to that of the sources $s_j(t)$. In other words, the hidden nodes which were supposed to be auxiliary variables for computing the nonlinear mapping from sources $s_j(t)$ to the observations had taken over and acted as sources. The mapping from hidden nodes to observations is linear and the model was thus effectively more linear than it should have been.

The analysis lead to a significant improvement in the learning procedure. As explained in [38], the method now relies on so called evidence nodes to restrict the hidden nodes $h_i(t)$ from acting as additional sources. During learning, the influence of the evidence nodes decays to zero after which they are removed. However, their influence on the hidden nodes during the early phases of learning is enough to give the sources $s_j(t)$ a competitive advantage in representing the data. This lead to significantly better solutions both in terms of signal-to-noise ratio (SNR) of the extracted source components and value of the cost function.

The plot in Fig. 1 shows the attained SNRs as a function of the mean code length of the hidden node variables $h_i(t)$ in experiments with and without evidence nodes. The 20-dimensional data set and the method for measuring the SNR are the same as in [38]. The figure shows that evidence nodes decrease the average code lengths of the hidden nodes and increase the SNRs. Consistent with the view that the main reason for the impoved performance is the reduced tendency for hidden neurons to represent the data, there is a significant correlation between the average code length and the SNR within the experiments of each individual type.

Note that the experiments without evidence nodes shown here were not the ones whose analysis lead to the introduction of the evidence nodes. The original experiments had even worse performance but since then there have been several refinements in the learning procedure. In order to isolate the effect of the evidence nodes, the learning schemes used in the experiments shown in the figure differ only in their use of the evidence nodes.

B. Spikes in ICA

Using standard ICA with unsuitable data set can cause two related types of artifacts: spikes and bumps [47]–[49]. We shall next discuss how the conditions in which they occur differ and how this behavior can be easily understood from an information-theoretic point of view.

Spikes are components whose energy is concentrated to a single observation with values at all other time instants being very close to zero. Spike-like signals maximize the kurtosis [50] of the signal as well as many contrast functions. If the dimensionality of the data is high compared to the number of samples, spikes can be found relatively easily and thus, algorithms attempting to maximize the kurtosis will yield such results unless care is taken to restrict the model in a suitable way.

ICA approaches based on MDL framework are not susceptible to such overfitting as using a source to model only one sample of the data is not very economical. Anything that can be gained in shorter description for the data at that sample will be lost in the description length needed for the corresponding parameters.

The emergence of spikes was tested with an example using artificial data. The data set consisted of 400 samples of 300 dimensional observations. The data was generated by mixing the three super-Gaussian sources shown in Fig. 2 along with two Gaussian white-noise sources using an orthogonal mixing matrix to five dimensions of the observations. The remaining 295 dimensions of the data contained only Gaussian white noise. This way the data set was already white and no reduction of dimensionality was possible.

The data set was tested using FastICA [50] and our bits-back ICA. Due to the extreme difficulty of the problem, both methods were initialized to the correct solution. This showed whether the optimization criterion was reasonable in that there was an optimum corresponding to the correct solution. If the model can

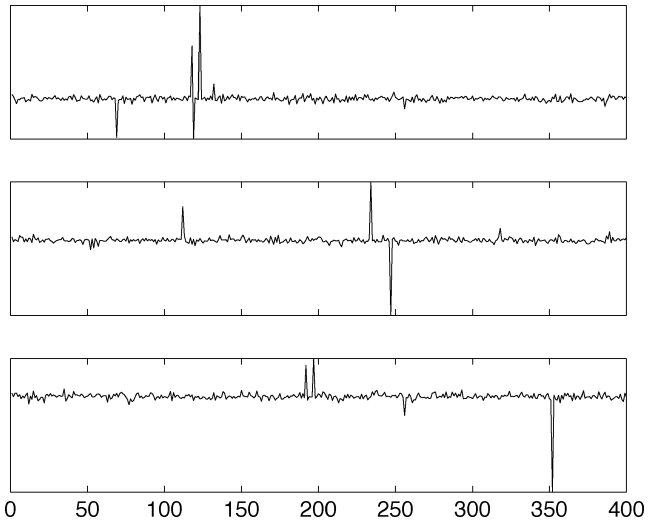

Fig. 3. Three sources recovered by FastICA in the spike experiment.

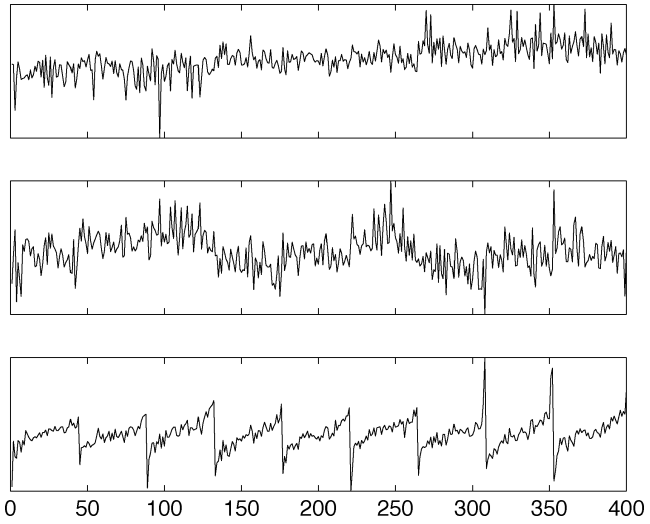

Fig. 4. Three sources recovered by bits-back ICA in the spike experiment.

stick to the correct solution, it should be able to find it with a good optimization algorithm but if it does not, there is no hope of it ever finding the solution.

The results of the different methods are illustrated in Fig. 3 for FastICA and Fig. 4 for bits-back ICA. Even though FastICA was initialized to the correct solution, it drifted away from it to a more spiky solution. The linear correlations of the sources recovered by FastICA with the original sources were -0.19, -0.27, and -0.24. Corresponding correlations between the sources recovered by bits-back ICA and the original sources were 0.86, 0.66, and 0.94. If the number of samples was decreased to be equal to the dimensionality of the data, FastICA found perfect spikes with only one essentially nonzero component. When the number of the added noise dimensions was decreased to a small enough value, FastICA recovered the original sources easily. The results attained with bits-back ICA varied only little depending on the dimensionality of the observations.

When the number of samples was increased from 400 to 4000, FastICA could retain the correct solution when initialized to it

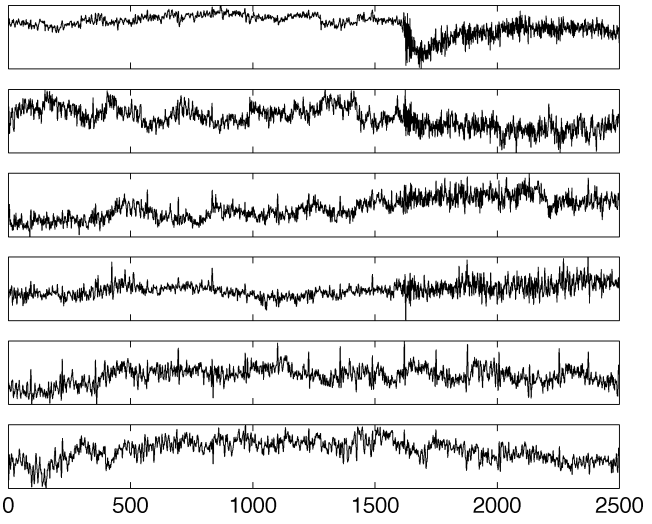

Fig. 5. Six-example time series of the 122 channels in the MEG data set.

and could find two of the components even without the correct initialization. This shows that the results obtained with FastICA do not stay the same when the amount of data is increased and the results obtained with a small number of samples are just artifacts.

C. Bumps in MEG Data

Bumps are the counterpart of spikes for temporally correlated data. Unlike spikes discussed in the previous section, bumps cannot be dismissed as overlearning results since they arise from the fact that the model cannot properly represent the temporal dependencies and persist even when large amounts of data is used.

The energy of a bump signal is concentrated over a few consecutive samples in the temporal direction. Like spikes, they too have a very high kurtosis. Using a learning algorithm based on the MDL framework will not help against bumps, however, because they present real savings in the description length. A bumpy solution for a static ICA model with temporally correlated data is actually a temporal model where the time series has been segmented and each component is used to model a single segment of the data. As each source can be used to model several observations, the model is able to attain a shorter description for the original data. Bumps can be best avoided by using a temporal model suitable for the data but there are also other methods that can be used to suppress them [48], [49].

To illustrate the formation of bumps with real data, we performed some experiments using biomedical magnetoencephalogram (MEG) measurements used in [51]. The MEG data consists of signals originating from brain activity measured with an array of magnetic sensors. The data has 122 channels corresponding to magnetic fields measured in two directions in 61 locations around the head of the subject. The signals are contaminated by external artifacts such as a digital watch and heart beat as well as eye movements and blinks. A set of 2500 samples of the original data set was used. A part of the data set is illustrated in Fig. 5.

The MEG measurements were studied using linear ICA based on the bits-back methodology as described in Section IV-B.

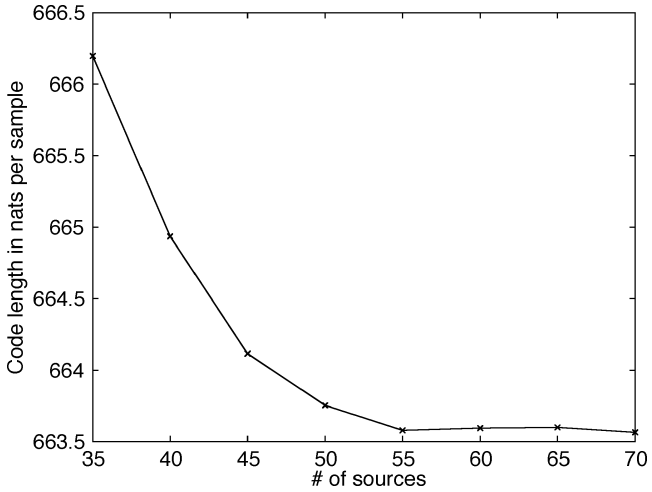

Fig. 6. The total code length in the MEG experiment with ICA model as a function of the number of sources.

The model is only able to represent super-Gaussian sources but that is sufficient for demonstrating the formation of bumps. In the variational Bayesian framework it is straightforward to use mixtures of Gaussians model for the source distribution if sub-Gaussian sources need to be extracted [31]–[35].

The attained code lengths using models with different numbers of sources are illustrated in Fig. 6. The figure shows a minimum around 55 sources, although the absolutely shortest code length was attained with a model initialized with 70 sources. However, 17 out of the 70 sources had been pruned in the best model leaving 53 active sources in the end. In general, having too few sources seems to be more harmful than having too many of them.

As the data has strong temporal correlations but the sources lack any dynamics in the model, some of the extracted sources correspond to bumps as illustrated in Fig. 7. Practically all ICA methods are susceptible to the same problem as illustrated by the findings of FastICA in Fig. 8. The difference in appearance of the bumps can easily be explained in that FastICA is essentially looking for spikes that happen to have temporal correlations, whereas bits-back ICA is trying to find a compact description of the data and therefore prefers wider bumps that describe a longer segment of the data.

The emergence of bumps can, of course, be avoided with additional care. One solution is high-pass filtering, which eliminates the strongest temporal dependencies. This was used in [51], but it risks losing important information as well. A safer solution is to use a temporal model for the sources [49].

VI. Discussion

In this paper, we have presented a review of the benefits offered by combining the views offered by Bayesian statistics and information theory. The bits-back coding scheme and the resulting expression for code length in (7) and its consequences suggest interesting connections between Bayesian and minimum-encoding learning approaches. First, it provides an alternative justification for using exact Bayesian inference with full posterior instead of different point estimates such as MAP.

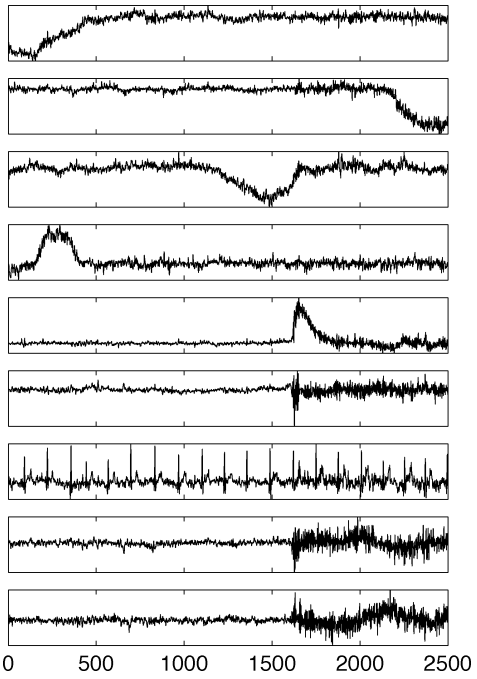

Fig. 7. Nine sources extracted from the MEG data set by bits-back ICA. The four topmost sources are bumps. The fifth looks like a bump but is really part of the signal associated with the beginning of a biting artifact at that point. The four lowermost sources are examples of desired artifact signals.

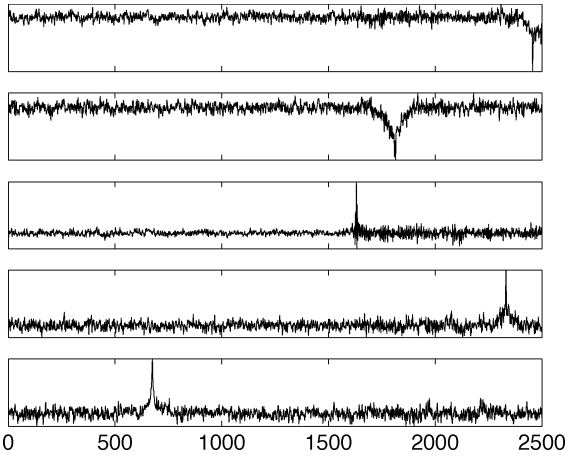

Fig. 8. Five bumpy sources extracted from the MEG data set by FastICA.

If the full posterior happens to be intractable, the best way to tackle the problem is to use an approximation that makes the problem tractable while being as close to the true posterior as possible, as measured by the Kullback–Leibler divergence.

Second, while similar methods have been used before as mean field approximations motivated by statistical mechanics, the information-theoretic viewpoint allows interpreting parts of the cost function as the code lengths of different parameters. This can help distinguish the most important parameters from the less important ones. Finally, the information-theoretic view also provides intuitive explanations of many phenomena through simple consideration of what can and cannot help in producing a shorter code for the data. This makes it easy to understand why it is sometimes not reasonable to build a higher level model for some parameters if their description length

is already so low that it is not possible to justify adding new parameters to model them.

One of the specific problems explained by the code length interpretation are the overfitting phenomena studied in Section V-B. The approach also saves bits-back ICA from spikes that are found by most cumulant and contrast function based methods. The emergence of bumps with temporally correlated data is intuitively understandable as well.

Methods based on careful application of the MDL principle are generally very resistant to overfitting. If they are used in a proper way, it is very difficult to make them go wrong. There are some exceptions related to the method being able to describe two real numbers to an arbitrary precision using only one number to describe them but these are usually easy to identify and avoid.

Bayesian statistics and MDL principle provide two alternative views to the problem of learning from data. As both methods can be used to derive the same practical algorithms, it is easy to combine the two complementary approaches. Using both principles together allows new explanations and solutions to be found for many practical problems encountered in constructing and training models.

ACKNOWLEDGMENT

The authors would like to thank M. Harva and T. Östman for their help with the experiments. Additionally, they would like to thank J. Särelä, R. Vigário, and P. Pajunen for useful comments and discussions.

REFERENCES

[1] R. T. Cox, "Probability, frequency and reasonable expectation," *Amer. J. Phys.*, vol. 14, no. 1, pp. 1–13, 1946.

[2] J. M. Bernardo and A. F. M. Smith, *Bayesian Theory*. New York: Wiley, 2000.

[3] A. Gelman, J. Carlin, H. Stern, and D. Rubin, *Bayesian Data Analysis*. Boca Raton, Florida: Chapman Hall/CRC Press, 1995.

[4] R. J. Solomonoff, "A formal theory of inductive inference," *Inform. Control*, vol. 7, pp. 1–22, 1964.

[5] C. S. Wallace and D. M. Boulton, "An information measure for classification," *Comput. J.*, vol. 11, no. 2, pp. 185–194, 1968.

[6] J. Rissanen, "Modeling by shortest data description," *Automatica*, vol. 14, no. 5, pp. 465–471, 1978.

[7] ——, "Stochastic complexity," *J. Roy. Statistical Soc. (Ser. B)*, vol. 49, no. 3, pp. 223–239, 1987.

[8] C. S. Wallace and P. R. Freeman, "Estimation and inference by compact coding," *J. Roy. Statistical Soc. (Ser. B)*, vol. 49, no. 3, pp. 240–265, 1987.

[9] J. Rissanen, *Stochastic Complexity in Statistical Inquiry*, Singapore: World Scientific, 1989.

[10] J. J. Oliver and D. Hand, "Introduction to Minimum Encoding Inference," Dept. Comput. Sci. Monash Univ., Clayton, Victoria, Australia, Tech. Rep. 205, 1994.

[11] J. J. Oliver and R. A. Baxter, "MML and Bayesianism: Similarities and Differences," Dept. Comput. Sci. Monash Univ., Clayton, Victoria, Australia, Tech. Rep. 206, 1994.

[12] R. A. Baxter and J. J. Oliver, "MDL and MML: Similarities and Differences," Dept. Comput. Sci. Monash Univ., Clayton, Victoria, Australia, Tech. Rep. 207, 1994.

[13] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, pp. 379–423, 1948.

[14] C. S. Wallace, "Classification by minimum-message-length inference," in *Proc. Advances in Computing and Information—ICCI '90*, vol. 468, S. G. Aki, F. Fiala, and W. W. Koczkodaj, Eds. Berlin, Germany, 1990, pp. 72–81.

[15] G. E. Hinton and D. van Camp, "Keeping neural networks simple by minimizing the description length of the weights," in *Proc. 6th Annu. ACM Conf. Computational Learning Theory*, Santa Cruz, CA, 1993, pp. 5–13.

[16] G. E. Hinton and R. S. Zemel, "Autoencoders, minimum description length and Helmholtz free energy," in *Advances in Neural Information Processing Systems 6*, J. D. Cowan, G. Tesauro, and J. Alspector, Eds. Cambridge, MA: MIT Press, 1998, pp. 3–10.

[17] D. J. C. MacKay, "Developments in probabilistic modeling with neural networks—ensemble learning," in *Neural Networks: Artificial Intelligence and Industrial Applications. Proc. 3rd Annu. Symp. Neural Networks*, 1995, pp. 191–198.

[18] B. J. Frey and G. E. Hinton, "Efficient stochastic source coding and an application to a Bayesian network source model," *Comput. J.*, vol. 40, no. 2/3, pp. 157–165, 1997.

[19] B. J. Frey, *Graphical Models for Machine Learning and Digital Communication*. Cambridge, MA: MIT Press, 1998.

[20] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. New York: Wiley, 1991.

[21] R. M. Neal and G. E. Hinton, "A view of the EM algorithm that justifies incremental, sparse, and other variants," in *Learning in Graphical Models*, M. I. Jordan, Ed. Cambridge, MA: MIT Press, 1999, pp. 355–368.

[22] J. S. Rustagi, *Variational Methods in Statistics*. New York: Academic, 1976.

[23] G. Parisi, *Statistical Field Theory*. Reading, MA: Addison-Wesley, 1988.

[24] M. Jordan, Z. Ghahramani, T. Jaakkola, and L. Saul, "An introduction to variational methods for graphical models," in *Learning in Graphical Models*, M. Jordan, Ed. Cambridge, MA: MIT Press, 1999, pp. 105–161.

[25] M. Opper and D. Saad, Eds., *Advanced Mean Field Methods: Theory and Practice*. Cambridge, MA: MIT Press, 2001.

[26] H. Lappalainen and J. Miskin, "Ensemble learning," in *Advances in Independent Component Analysis*, M. Girolami, Ed. Berlin, Germany: Springer-Verlag, 2000, pp. 75–92.

[27] D. J. C. MacKay, "Bayesian interpolation," *Neural Comput.*, vol. 4, no. 3, pp. 415–447, 1992.

[28] ——, *Information Theory, Inference and Learning Algorithms*. Cambridge, U.K.: Cambridge Univ. Press, 2003.

[29] D. Barber and C. Bishop, "Ensemble learning for multi-layer networks," in *Advances in Neural Information Processing Systems 10*, M. Jordan, M. Kearns, and S. Solla, Eds. Cambridge, MA: MIT Press, 1998, pp. 395–401.

[30] Ensemble Learning for Hidden Markov Models, D. J. C. MacKay. (1997). [Online]. Available: http://wol.ra.phy.cam.ac.uk/mackay/

[31] H. Attias, "Independent factor analysis," *Neural Computat.*, vol. 11, no. 4, pp. 803–851, 1999.

[32] H. Lappalainen, "Ensemble learning for independent component analysis," in *Proc. Int. Workshop Independent Component Analysis Signal Separation (ICA'99)*, Aussois, France, 1999, pp. 7–12.

[33] H. Attias, "ICA, graphical models and variational methods," in *Independent Component Analysis: Principles and Practice*, S. Roberts and R. Everson, Eds. Cambridge, U.K.: Cambridge Univ. Press, 2001, pp. 95–112.

[34] J. Miskin and D. J. C. MacKay, "Ensemble learning for blind source separation," in *Independent Component Analysis: Principles and Practice*, S. Roberts and R. Everson, Eds: Cambridge Univ. Press, 2001, pp. 209–233.

[35] W. Penny, R. Everson, and S. Roberts, "ICA: model order selection and dynamic source models," in *Independent Component Analysis: Principles and Practice*, S. Roberts and R. Everson, Eds. Cambridge, U.K.: Cambridge Univ. Press, 2001, pp. 299–314.

[36] H. Lappalainen and A. Honkela, "Bayesian nonlinear independent component analysis by multi-layer perceptrons," in *Advances in Independent Component Analysis*, M. Girolami, Ed. Berlin, Germany: Springer-Verlag, 2000, pp. 93–121.

[37] H. Valpola, E. Oja, A. Ilin, A. Honkela, and J. Karhunen, "Nonlinear blind source separation by variational Bayesian learning," *IEICE Trans. Fundamentals Electron. Commun. Comput. Sci.*, vol. E86-A, no. 3, pp. 532–541, 2003.

[38] H. Valpola, T. Östman, and J. Karhunen, "Nonlinear independent factor analysis by hierarchical models," in *Proc. 4th Int. Symp. on Independent Component Analysis and Blind Signal Separation (ICA2003)*, Nara, Japan, 2003, pp. 257–262.

[39] H. Valpola and J. Karhunen, "An unsupervised ensemble learning method for nonlinear dynamic state-space models," *Neural Computat.*, vol. 14, no. 11, pp. 2647–2692, 2002.
[40] Z. Ghahramani and G. E. Hinton, "Variational learning for switching state-space models," *Neural Computat.*, vol. 12, no. 4, pp. 963–996, 2000.
[41] H. Valpola, M. Harva, and J. Karhunen, "Hierarchical models of variance sources," in *Proc. 4th Int. Symp. Independent Component Analysis Blind Signal Separation (ICA2003)*, Nara, Japan, 2003, pp. 83–88.
[42] ——, "Hierarchical models of variance sources," *Signal Processing*, vol. 84, no. 2, pp. 267–282, 2004.
[43] H. Valpola, T. Raiko, and J. Karhunen, "Building blocks for hierarchical latent variable models," in *Proc. 3rd Int. Conf. Independent Component Analysis Signal Separation (ICA2001)*, San Diego, CA, 2001, pp. 710–715.
[44] M. Jordan, Ed., *Learning in Graphical Models*. Cambridge, MA: MIT Press, 1999.
[45] R. Cowell, "Introduction to inference for Bayesian networks," in *Learning in Graphical Models*, M. I. Jordan, Ed. Cambridge, MA: MIT Press, 1999, pp. 9–26.
[46] A. Honkela, H. Valpola, and J. Karhunen, "Accelerating cyclic update algorithms for parameter estimation by pattern searches," *Neural Processing Lett.*, vol. 17, no. 2, pp. 191–203, 2003.
[47] A. Hyvärinen, J. Särelä, and R. Vigário, "Spikes and bumps: artifacts generated by independent component analysis with insufficient sample size," in *Proc. Int. Workshop Independent Component Analysis Signal Separation (ICA'99)*, Aussois, France, 1999, pp. 425–429.
[48] J. Särelä and R. Vigário, "Overlearning problem in high-order ICA: analysis and solutions," *J. Mach. Learning Res.*, vol. 4, pp. 1447–1469, Dec. 2003.
[49] ——, "A Bayesian Approach to Overlearning in ICA: A Comparison Study," Lab. Comput. Inform. Sci., Helsinki Univ. Technol., Finland, Tech. Rep. A70, 2003.
[50] A. Hyvärinen, J. Karhunen, and E. Oja, *Independent Component Analysis*. New York: Wiley, 2001.
[51] R. Vigário, J. Särelä, V. Jousmäki, M. Hämäläinen, and E. Oja, "Independent component approach to the analysis of EEG and MEG recordings," *IEEE Trans. Bio-Med. Eng.*, vol. 47, pp. 589–593, May 2000.

Antti Honkela received the M.Sc. degree in mathematics from the Helsinki University of Technology, Espoo, Finland, in 2001. He is currently working toward the D.Sc. degree in approximate Bayesian inference methods at the same university.

He has been working at the Neural Networks Research Centre, Helsinki University of Technology, since 1999.

Harri Valpola received the M.Sc. degree in technical physics and the D.Sc. degree in computer science from the Helsinki University of Technology, Espoo, Finland, in 1996 and 2000, respectively.

He has been working with the Neural Networks Research Centre, Helsinki University of Technology, since 1993. Currently, he is working on a humanoid–robot project in the Artificial Intelligence Laboratory University of Zurich, Switzerland.