Project no.: 619209

Project full title: Analysis of Massive Data STreams

Project Acronym: AMIDST

Deliverable no.: D4.2

Title of the deliverable: State of the art for learning in the context of the AMIDST framework

Contractual Date of Delivery to the CEC:	**29.02.2016**
Actual Date of Delivery to the CEC:	**29.02.2016**
Organisation name of lead contractor for this deliverable:	**NTNU**
Author(s):	**Helge Langseth, Anders L. Madsen, Thomas D. Nielsen, Antonio Salmerón**
Participants(s):	**P01, P02, P03, P04**
Work package contributing to the deliverable:	**WP4**
Nature:	**R**
Version:	**1.0**
Total number of pages:	**27**
Start date of project:	**1st January 2014 Duration: 36 month**

Project co-funded by the European Commission within the Seventh Framework Programme (2007-2013)		
Dissemination Level		
PU	Public	**X**
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Abstract:

In this document, we describe the state of the art for learning AMIDST models from data streams. The learning problem is put in a Bayesian setting, where scalable approximate learning can be done using variational Bayes. Furthermore, we look at extensions to accommodate drifting/non-stationary domains, and discuss scalable techniques for feature selection.

Keyword list: Bayesian learning, AMIDST model class, state of the art, variational Bayes, adaptation, concept drift, feature selection.

Contents

Document history

Version	Date	Author (Unit)	Description
v0.3	5/1-2016	Helge Langseth, Thomas D. Nielsen, Antonio Salmerón	Content of the document discussed and established
v0.6	24/2-2016	Helge Langseth, Anders L. Madsen, Thomas D. Nielsen, Antonio Salmerón	Initial version of document finished and reviewed by the PSRG
v1.0	29/2-2016	Helge Langseth, Anders L. Madsen, Thomas D. Nielsen, Antonio Salmerón	Final version of document

1 Executive summary

The aim of this document is to provide a thorough description of the state of the art for learning in Bayesian networks, but with a focus on the models relevant for the AMIDST project. The role of this deliverable is thus to document the scientific foundation for the developments carried out in WP4.

After an introductory discussion, the document has been structured according to the task division in WP4, with separate sections covering learning in AMIDST models with focus on scalability, extensions to accommodate non-stationarity, and efficient methods for feature selection. The included references have been analysed taking the AMIDST model class and its learning needs into account.

2 Introduction

2.1 Background

Probabilistic graphical models provide a well-founded and principled approach for performing inference in complex domains endowed with uncertainty. A probabilistic graphical model is a framework consisting of two parts: a qualitative component in the form of a graphical model encoding conditional independence assertions about the domain being modelled as well as a quantitative component consisting of a collection of local probability distributions adhering to the independence properties specified in the graphical model. Collectively, the two components provide a compact representation of the joint probability distribution over the domain being modelled.

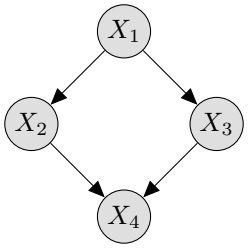

Figure 1: A Bayesian network with five variables.

Bayesian networks (BNs) [1–3] are a particular type of probabilistic graphical model that has enjoyed widespread attention in the last three decades. Figure 1 shows a BN representing the joint distribution of variables $X_1, \ldots, X_4$. Attached to each node, there is a conditional probability distribution given its parents in the network, so that the joint distribution factorises as

$$p(X_1, \ldots, X_4) = p(X_1)p(X_2|X_1)p(X_3|X_1)p(X_4|X_2, X_3).$$

In general, for a BN with N variables $\boldsymbol{X} = \{X_1, \ldots, X_N\}$, the joint distribution factorises as

$$p(\boldsymbol{X}) = \prod_{i=1}^{N} p(X_i|\mathrm{pa}\,(X_i)), \tag{1}$$

where $\mathrm{pa}\,(X_i)$ denotes the set of parents of X_i in the network.

Here, and later on in the document, we use capital letters like X_i to signify a random variable and let a lowercase letter, e.g., x_i, refer to a value or configuration of that variable. The domain of X_i is denoted Ω_{X_i}. Furthermore, we use boldface to represent *vectors*, e.g., $\boldsymbol{X} = \{X_1, \ldots, X_N\}$. When discussing models that develop over time, we will index a variable related to time t by a subscript t, e.g., $X_{i,t}$ or $\boldsymbol{X}_t$. Finally, we use $\boldsymbol{X}_{t_1:t_2}$ as a shorthand for $\boldsymbol{X}_{t_1:t_2} = \bigcup_{t=t_1}^{t_2} \boldsymbol{X}_t$.

Given a set of observed variables $\boldsymbol{X}_E \subset \boldsymbol{X}$ and a set of variables of interest $\boldsymbol{X}_I \subset \boldsymbol{X} \backslash \boldsymbol{X}_E$, *probabilistic inference* is the calculation of the posterior distribution $p(x_i|\boldsymbol{x}_E)$ for each $i \in I$. A thorough introduction to the state of the art for inference in BNs was given in Deliverable D3.1 [4]. In this document we assume that inference techniques for a given BN are available (inference has been treated in WP3 and WP5), and we will here only consider the state of the art related to *learning* a BN from data.

Consider again the factorization of the full joint distribution $p(\boldsymbol{x})$ in Equation (1). With this factorization we can efficiently represent the joint probability distribution $p(\boldsymbol{x})$, but it requires that $\mathrm{pa}\,(X_i)$, i.e., the *parent set* of X_i, is known for each $i = 1, \ldots, N$. The parents of X_i are exactly the nodes which have an outgoing edge pointing to X_i (e.g., $\mathrm{pa}\,(X_4) = \{X_2, X_3\}$ for the model in Figure 1). Algorithms for learning the parent sets, also known as *structure learning* algorithms, most often follow one of two approaches: *i*) search and score methods, see, e.g., [5] and *ii*) constraint-based techniques [6]. Structure learning was considered in Task 4.1 of the project, but as that task has already been concluded and since the focus there was on *static* models, the topic will not be discussed in this document.

As soon as the parent sets in Equation (1) are determined, the conditional distributions $p(X_i|\mathrm{pa}\,(X_i))$ can be learned from data. The approach taken in the AMIDST project is that the conditional distribution $p(X_i|\mathrm{pa}\,(X_i))$ is assumed to be a member of the conjugate exponential family (including, among others, the Gaussian and multinomial distributions). Therefore, parameter learning amounts to finding a close-to optimal *parameterization* of the given distributions. This is the task that will receive the most consideration in this document, and we will focus on its behavior in context of the *AMIDST model framework*.

2.2 The AMIDST model framework

We will limit our investigation of learning algorithms to those relevant for *the AMIDST model framework*. This subset of the *probabilistic graphical models* (PGMs) was defined in Deliverable D2.1 [7] and further discussed and refined in Deliverable D2.2 [8]. In its most general installment, the AMIDST model is a *dynamic* model (i.e., a model explicitly modelling time) that accommodates *hybrid* domains (that is, domains containing both discrete and continuous variables). The framework structure is a dynamic Bayesian network, often described as a two time-slice Bayesian network (2T-DBN). A 2T-DBN is a compact representation of a time-dependent probabilistic model, where the underlying assumption is that the first part of the model describes the generative model for data from time $t = 0$, and the second part describes the development over time, depicting both how the variables inside the time-slice interact locally, and their dependence on the previous time-slice. The 2T-DBN can be "unrolled" over time, so that the model defined at time-points $t = 0, 1, 2, \ldots, T$ generalizes the representation in Equation (1) as

follows:

$$\begin{aligned} p(\boldsymbol{X}_{0:T}) &= p(\boldsymbol{X}_0)\prod_{t=1}^{T} p(\boldsymbol{X}_t|\boldsymbol{X}_{1:t-1}) \\ &= \prod_{i=1}^{N} p(X_{i,0}|\text{pa}\,(X_{i,0}))\prod_{t=1}^{T}\prod_{i=1}^{N} p(X_{i,t}|\text{pa}\,(X_{i,t})). \end{aligned} \tag{2}$$

A 2T-DBN model explicitly enforces the *Markov-assumption* by requiring that $\text{pa}\,(X_{i,t}) \subseteq \{\boldsymbol{X}_{t-1}, \boldsymbol{X}_t\} \setminus \{X_{i,t}\}$, and the *stationarity-assumption* by not explicitly indexing the conditional distribution $p(X_{i,t}|\text{pa}\,(X_{i,t}))$ on t (i.e., it is a time-invariant function evaluated on values attained by the variables defined at time t).

Figure 2 shows an example of graphical structures defining a 2T-DBN model consisting of a transition network for time t (right) and its corresponding time slice $t = 0$ network (left). For time $t - 1$, only the (ingoing) *interface variables* are displayed, i.e., those variables whose values at time $t - 1$ have a direct effect on the variables at time t. Interface variables do not have parents or conditional probability distributions [9]. In our example, the interface variable $X_{1,t-1}$ has a direct effect on $X_{1,t}$. The transition distribution is expressed as

$$p(\boldsymbol{X}_t|\boldsymbol{X}_{t-1}) = p(X_{1,t}|X_{1,t-1})p(X_{2,t}|X_{1,t})p(X_{3,t}|X_{1,t})p(X_{4,t}|X_{2,t}, X_{3,t}).$$

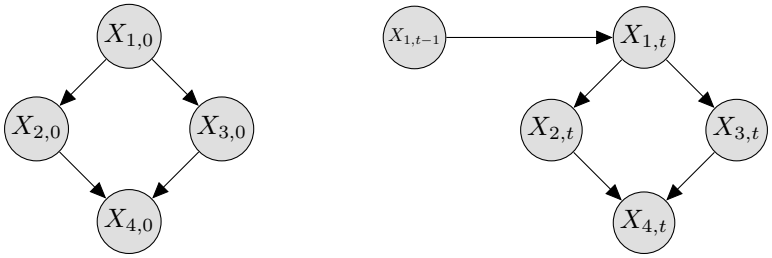

Figure 2: An example of the graphical structures corresponding to a 2T-DBN (right), and its corresponding network for time $t = 0$ (left).

Figure 3 shows the AMIDST model class, unrolled to show the time-points t and $t + 1$. The model has a fixed structure, with continuous variables following the Gaussian (or mixture of Gaussians) distribution, while discrete variables are categorical. Discrete variables can only have discrete parents.

The AMIDST requirement that learning should be seamlessly integrated with domain expertise during learning dictates a *Bayesian* approach to learning. Furthermore, our focus on dynamic models solidly anchors this report within the field of *Bayesian time series models* [10,11]. Of special interest are hidden Markov models (see, e.g., [7, Section 3.3.1]), Kalman filters and switching Kalman filter models [7, Section 3.3.2]. With this

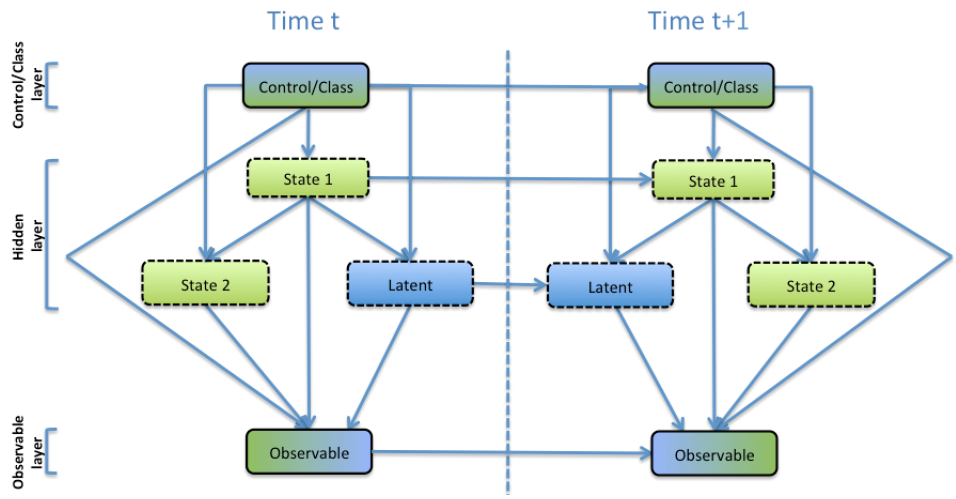

Figure 3: The most general dynamic model in the AMIDST model class.

focus, our work distinguishes itself from classical approaches to learning from streams, in which one assumes that for a model $\mathcal{M}$ it holds that $\boldsymbol{X}_s \perp\!\!\!\perp \boldsymbol{X}_t | \mathcal{M}$, for $s \neq t$. We note that the explicit modelling of dynamic aspects of the stream prevents us from using asynchronous parallelization techniques to analyze data from different time-slices in parallel. However, the extra modelling flexibility introduced is important for dealing with *non-stationary* streams, i.e., domains exhibiting *concept drift* [12, 13].

As the model structures in the AMIDST modelling frameworks are (semi-)fixed, our investigations will to a large extent by-pass the vast body of work on learning the *structure* of Bayesian networks. We will instead investigate the use of *feature selection* to learn models that are not prone to overfitting.

The rest of the document is structured as follows: In Section 3 we will discuss learning of parameterization of these models and their natural extensions assuming stationary (non-drifting) domains. Our focus will be on Bayesian learning methods, and how they scale up. We discuss extensions of the learning regime required to handle non-stationary domains in Section 4. This is followed by a description of *feature selection* from data streams in Section 5, and a conclusion in Section 6.

3 Parameter learning

3.1 Setting the scene

Let $\boldsymbol{\theta}$ denote the combination of all parameters of the model (that is, all parameterizations of the conditional distributions in Equation (1) for static models, Equation (2)

for dynamic models), and let $\mathcal{D} = \left\{ \boldsymbol{x}_{T_1}^{(1)}, \ldots, \boldsymbol{x}_{T_N}^{(N)} \right\}$ denote the dataset which we are learning from. $\mathcal{D}$ is a collection of N independent configurations over the variables of the domain, i.e., N realizations of the time series s.t. $\boldsymbol{x}_{T_j}^{(j)}$ is a realization over time-points $t = 0, \ldots, T_j$. Note that $\mathcal{D}$ may be *incomplete*, in case some of the observations are missing at random. Missingness can occur, for instance, if some of the variables are latent, or some data-collecting instrument fails.

Parameter learning in Bayesian networks amounts to estimating $\boldsymbol{\theta}$ using some optimization criterion that depends on $\mathcal{D}$. The learning generally comes in two different shapes. While the most commonly used approach (at least traditionally) is *maximum likelihood* learning, we will in the AMIDST project focus on *Bayesian* learning. From a simplistic point of view, the main difference between the two is that the Bayesian set-up regards $\boldsymbol{\theta}$ as a vector of *random variables*, and treats them on an equal footing as all other (unobservable) variables in the domain. From a modelling perspective this extension is straightforward: The 2T-DBN-model in Figure 2 extends the model in Figure 4 by explicitly representing $\boldsymbol{\theta} = \{\theta_{i,t}\}$, where $\theta_{i,t}$ is the parameterization of the conditional distribution function $p(X_{i,t}|\mathrm{pa}\,(X_{i,t}))$. Note that due to the *stationarity assumption* encoded in the 2T-DBN model, we have that $p(X_{i,t}|\mathrm{pa}\,(X_{i,t}))$ will not change with t as long as $t > 0$. The $\boldsymbol{\theta}$ parameters for $t > 0$ (right hand side of Figure 4 are therefore not indexed by t, we merely separate between the case $t = 0$ and $t > 0$ (giving $\boldsymbol{\theta}_0$ and $\boldsymbol{\theta}_>$, respectively). Typically, the case $t > 0$ is most important in streaming applications.

Parameter learning now amounts to calculating the probability distribution $p(\boldsymbol{\theta}|\mathcal{D})$, that is, it is reduced to inference in an (extended) Bayesian network model. Thereby, inference techniques – including approximate approaches like variational Bayes message passing, expectation propagation or importance sampling – can be used directly for learning.

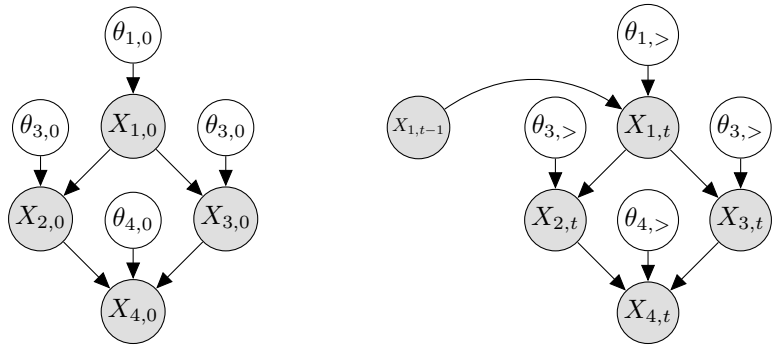

Figure 4: The 2T-DBN of Figure 2 is extended to a Bayesian setting by explicitly including the unknown parameters as latent variables.

In practice, the *a priori* marginal distributions for each θ_i must be declared before the learning can be performed. These prior distributions enable domain experts to express knowledge about the parameterization of the distributions that is combined with

information from the dataset to obtain the posterior information captured by $p(\boldsymbol{\theta}|\mathcal{D})$. The prior information consists of the definition of a distributional family for each θ_i together with a parameterization (the so-called *hyper-parameters*). In AMIDST we will ensure efficient calculation of posteriors by enforcing the distributional families to be the conjugate distribution of the likelihood-terms. The hyper-parameters are chosen freely, and this is the vehicle provided to encode prior (expert) knowledge.

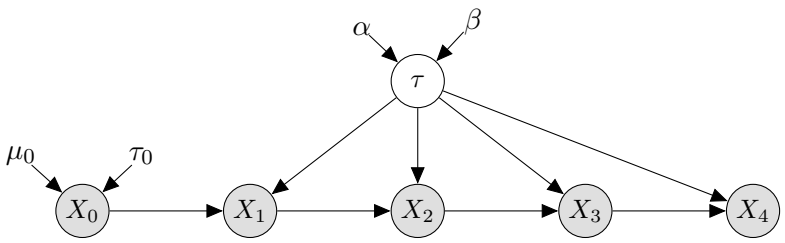

Figure 5: A detailed Bayesian network for the streaming variable X_t (observed at time $t = 0, \ldots, 4$). For $t > 0$, $X_t|\{X_{t-1} = x_{t-1}\}$ is assumed to follow a Gaussian distribution with mean x_{t-1} and precision τ.

Figure 5 shows a detailed description of the model for a streaming variable X_t for $t = 0, \ldots, 4$. For $t > 0$ we have the model $X_t|\{X_{t-1} = x_{t-1}\} \sim N\left(x_{t-1}, \tau^{-1}\right)$. Prior information about the parameter is encoded by assuming $\tau \sim \Gamma(\alpha, \beta)$ for given values of the hyper-parameters $\{\alpha, \beta\}$. Furthermore, $X_0 \sim N(\mu_0, \tau_0)$. The dataset $\mathcal{D} = \{x_0, x_1, x_2, x_3, x_4\}$ enables us to calculate

$$p(\tau|x_0, \ldots, x_4, \mu_0, \tau_0, \alpha, \beta) = \frac{p(x_0|\mu_0, \tau_0)p(\tau|\alpha, \beta)\prod_{t=1}^{4} p(x_t|x_{t-1}, \tau)}{p(x_0, \ldots, x_4|\mu_0, \tau_0, \alpha, \beta)}. \tag{3}$$

The calculation is efficient both in terms of space and time because the model is from the conjugate exponential family.

3.2 Scalable Bayesian learning from data streams

When considering inference and parameter learning in this document we will focus on *variational Bayes*(VB) inference. VB inference is a deterministic approximate inference technique, where we seek to iteratively optimise a variational approximation to the posterior distribution of interest [14, 15]. Let $\mathcal{Q}$ be the set of possible approximations; then the variational approximation to a posterior distribution $p(\boldsymbol{x}_I|\boldsymbol{X}_E = \boldsymbol{x}_E)$ is defined as

$$q^*_{\boldsymbol{x}_E}(\boldsymbol{x}_I) = \arg\min_{q \in \mathcal{Q}} D(q(\boldsymbol{x}_I)||p(\boldsymbol{x}_I|\boldsymbol{X}_E = \boldsymbol{x}_E)),$$

where $D(q||p)$ is the KL divergence between q and p. This is equivalent to maximizing a variational lower bound of the marginal log-likelihood of the model given the data, which in turn provides a model selection and comparison criteria.

The tractability of the above optimization problem is ensured by constraining the set of possible approximations $\mathcal{Q}$. A common approach is to employ a *variational mean-field* approximation of the posterior distribution, so that the approximation factorises over the individual variables involved, i.e.,

$$q^*_{\boldsymbol{x}_E}(\boldsymbol{x}_I) = \prod_{i \in I} q^*_{\boldsymbol{x}_E}(x_i). \tag{4}$$

During the optimisation of the variational mean-field one performs a coordinate ascent by iteratively updating the individual variational distributions while holding the others fixed [16]. Updating a variational distribution essentially involves calculating the variational expectation of the logarithm of the original conditional distributions of the model. This can be done efficiently and in closed form when the distributions involved are conjugate-exponential [17]. Variational Bayes techniques, while often being quick in practice, can suffer from slow convergence when parameters are strongly correlated in the exact posterior. Auxiliary variable techniques [18] extend the Bayesian models with auxiliary variables targeted to reduce the correlations, and have also been employed in Bayesian time-series models [19].

When performing the coordinate ascent updates, one often distinguishes between "local" latent variables and "global" latent variables. Local latent variables typically relates to a particular data point, whereas global latent variables represent model parameters that are shared across all components in the model. Using plate notation, the distinction between the two types of latent variables can be illustrated as in Figure 6, which also reveals the implied conditional independencies between the variables in the model.

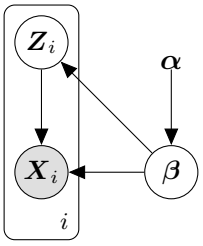

Figure 6: A plate representation of a simplified AMIDST model. Each observation variable $\boldsymbol{X}_i$ is associated with a local latent variables $\boldsymbol{Z}_i$. The local models are connected through the global latent variables $\boldsymbol{\beta}$, which are parameterized by $\boldsymbol{\alpha}$.

Straightforward variational coordinate ascent in models having both local and global latent variables, implies that in order to make a single update to the global latent variables one would have to make a complete pass through all the data. This is inefficient, in particular if we would expect that a smaller subset of the data is sufficient for making an informed update. This observation is the basis for *stochastic variational inference* (SVI) [20], which was proposed for scaling up VB to large datasets. The key idea being to utilize the setup of other stochastic gradient methods, and only use mini-batches (possibly singletons) of the data at each iteration of the algorithm. That is, one iteration

consists of sub-sampling a mini-batch of data and from that obtain a noisy estimate of the natural gradient of the variational lower bound. This noisy estimate is subsequently used to update the variational parameters in the model. Stochastic variational inference has also been particularized for Bayesian time-series models (Hidden Markov models, Hidden semi-Markov models, and infinite hidden Markov models) [21, 22]. Here, the full dataset is supposed to consist of several time-series, and a mini-batch corresponds to one specific time-series.

In its original formulation, stochastic variational inference assumes access to the full dataset from which data points are sampled. Nevertheless, a special case of the SVI algorithm applied in an online learning setting for latent Dirichlet allocation models [23] was proposed in [24]. Also, as noted in [25], SVI can also be adapted to a streaming context by having the algorithm visit each data point exactly once. A more robust, but also more computationally expensive, version of SVI was proposed in [26], where the natural gradient steps of SVI are replaced by so-called trust-region updates. This adaption of SVI has also been applied in a streaming context, where new data points are incrementally added to the database and empirical Bayes [26] is used to achieve a more robust setting of the hyper-parameters.

A more direct approach for applying variational inference in a streaming setting is pursued in [25] with the Streaming Distributed Asynchronous Bayes (SDA-Bayes) algorithm. The SDA-Bayes algorithm assumes a generative model where the variables at time s and t are conditionally independent given the parameterization $\boldsymbol{\Theta}$, i.e., $p(\boldsymbol{X}_{0:T}, \boldsymbol{\Theta}) = p(\boldsymbol{\Theta}) \prod_t p(\boldsymbol{X}_t|\boldsymbol{\Theta})$. The algorithm processes the data in mini-batches, iteratively updating the posterior distribution over $\boldsymbol{\Theta}$ after the t'th mini-batch $\mathcal{B}_t$:

$$p(\boldsymbol{\Theta}|\mathcal{B}_0, \ldots, \mathcal{B}_t) \propto p(\mathcal{B}_t|\boldsymbol{\Theta}) p(\boldsymbol{\Theta}|\mathcal{B}_0, \ldots, \mathcal{B}_{t-1}).$$

Rather than calculating the posterior distributions exactly, one will typically rely on approximate methods (like variational inference), in which case the procedure above will recursively calculate an approximation to the posterior. The sequential updating of $p(\boldsymbol{\Theta}|\cdot)$ can also be done in parallel by exploiting that

$$p(\boldsymbol{\Theta}|\mathcal{B}_0, \ldots, \mathcal{B}_t) \propto \left[\prod_{i=0}^{t} p(\mathcal{B}_i|\boldsymbol{\Theta}) \right] p(\boldsymbol{\Theta}) \propto \left[\prod_{i=0}^{t} p(\boldsymbol{\Theta}|\mathcal{B}_i) p(\boldsymbol{\Theta})^{-1} \right] p(\boldsymbol{\Theta}). \tag{5}$$

Hence, the local posteriors associated with the individual mini-batches are computed in parallel and afterwards combined to get the full posterior. Note again that when resorting to approximate inference when calculating the local posterior, the full posterior will also be approximate.

A subtle problem of doing distributed and decentralized learning according to Equation 5 is that a straightforward combination of the local posteriors may lead to inconsistencies in the full posterior. This problem is addressed in [27] in relation to hybrid models with focus on capturing parameter permutation symmetry. That is, permuting the

values of a subset of the parameters does not impact the posterior probability, in which case the model is non-identifiable. In order to resolve potential inconsistencies, [27] poses the task as a combinatorial optimization problem over the possible parameter permutation matrices. Although the optimization method is restricted to parametric models, [28] extends the scope of the general approach by adapting it to Bayesian non-parametric models. Neither of the two methods does, however, come with any guarantees on optimality of the resulting full posterior.

Alternative approaches for combining local posteriors have also been considered [29]. For example, [30] consider the combination of posterior distributions through their barycenter in Wasserstein space, [31] forces the local posteriors to agree on the overall shape of the full posterior by ensuring that they agree on the first two moments. In the same spirit, [32] imposes equivalency among the local posterior distributions by formulating the problem as a constrained optimization problem (constrained by parameter equivalence). This problem is subsequently solved using the alternating direction method of multipliers and by penalizing distribution differences through the Bregman-divergence induced by the log-partition function of the global latent variable.

When adopting the full AMIDST model class illustrated in Figure 3, the model itself imposes natural constraints on the parallelization opportunities. Specifically, in Figure 3 we see that the variables at two different time steps s and t are not conditional independent given the parameters in the model. In turn this means that the observations in time step $t+1$ can neither be processed before nor in parallel with the observations in time step t. Thus, scalable learning within this context is limited to the processing of the observations pertaining to a single time step as well the transition from one time step to the next. For the former task, the methods above are applicable whereas for the latter task the *factored frontier* algorithm [33] defines an approximate updating step when transitioning to a new time step, see also [34, 35].

4 Adaptation

The parameter learning discussed so far requires *stationary* domains. Assume we monitor a stream of data $\{\boldsymbol{x}_t = (\boldsymbol{z}_t, y_t)\}$, where Y_t is the variable we want to predict, and $\boldsymbol{Z}_t$ is the vector of features (or covariates) from which the prediction of Y_t is to be made. In our dynamic-model setting, the stationarity-assumption can be defined as

$$p(\boldsymbol{X}_t|\boldsymbol{X}_{t-k:t-1}, \boldsymbol{\Theta}) \equiv p(\boldsymbol{X}_s|\boldsymbol{X}_{s-k:s-1}, \boldsymbol{\Theta}) \text{ for all } s \text{ and all } t, \tag{6}$$

where k determines the Markov order (typically $k = 1$ or, if the model should not incorporate the dynamic aspect, $k = 0$). Notice that the model in Equation (6) is time invariant, because neither the probability function $p(\cdot)$ nor the parameterization $\boldsymbol{\Theta}$ are indexed by time.

However, learning algorithms using real-life data streams often operate in a dynamic setting, and in this section we will investigate how learning can be done in domains

exhibiting *concept drift* [12,13]. When discussing concept drift, it is beneficial to separate between *real* concept drift, for which $p(y_t|\boldsymbol{z}_t, \boldsymbol{x}_{t-k:t-1})$ changes over time, and *virtual* concept drift, where $p(y_t|\boldsymbol{z}_t, \boldsymbol{x}_{t-k:t-1})$ is stable, but $p(\boldsymbol{z}_t|\boldsymbol{x}_{t-k:t-1})$ drifts. Obviously, virtual drift is not as harmful for predictive ability as real concept drift, and can even be completely disregarded when working with discriminative models. From a modelling perspective, it is also helpful to separate between *sudden* (or abrupt) and *incremental* concept drift. The former is typically more pronounced and easier to detect, while the latter can be harder to pinpoint.

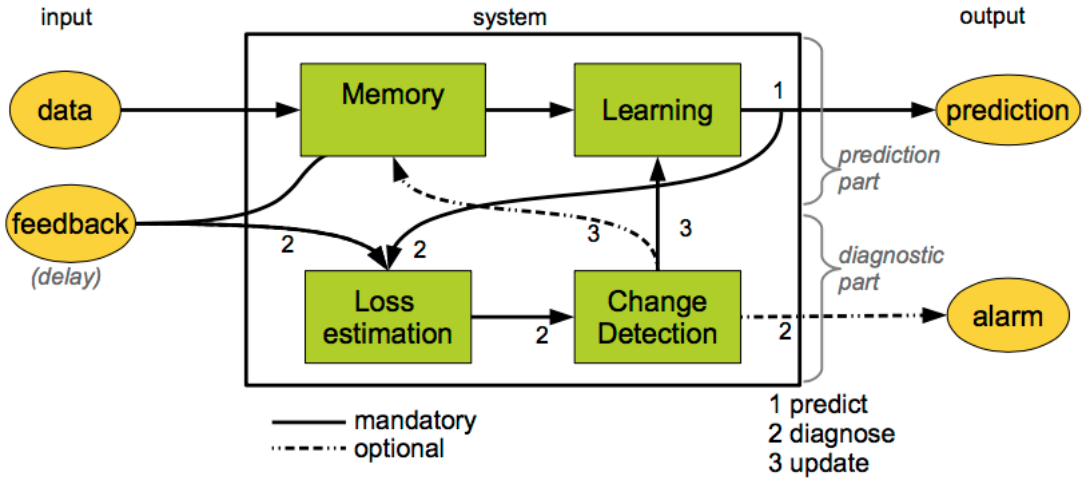

Figure 7: Gama et al.'s generic schema for an online adaptive learning algorithm [13, Figure 3].

A general schema for adaptive learning algorithms, taken from [13, Figure 3], is reproduced in Figure 7. In this setup, the system goes through three different phases:

1. **Predict:** Data in the form of queries, i.e., the stream $\{\boldsymbol{z}_t\}$, is fed to the system. A model is employed to generate a stream of predictions $\{\hat{y}_t\}$.

2. **Diagnose:** The "correct" value y_t for the query $\boldsymbol{z}_t$ is provided to the system, potentially with a delay. y_t is compared to the system-generated prediction $\hat{y}_t$, and the empirical loss $L_t = L(y_t, \hat{y}_t)$ is calculated. The stream of losses, $\{L_t\}$, is monitored over time, and undesirable changes (increased loss or reduced speed of loss improvement) can be flagged to the user.

3. **Update:** Finally, data in memory and the results of the change detector are combined with the current model to update it. The goal is to have a model that will work well given the data stream's *current* characteristics. An update may entail simply keeping the model as it is, refine it using all available data in memory, or do more elaborate considerations where, for instance, some data may be deemed as irrelevant and discarded before learning of a new model is performed.

This general schema gives a vehicle for modifying algorithms that were initially intended for stationary streams to also work under concept drift. As an example, Hoeffding Trees ("Very Fast Decision Trees", VFDTs) [36] define a framework for time-efficient learning from data streams, but the learning algorithm is only well-defined when the stream is stationary. The idea was therefore extended by [37], who kept the model consistent with a sliding window on the stream, re-checked consistency periodically, and grew alternative sub-trees as required. Similarly, Gama et al. [38] monitor the error rate of the learned model, and learn alternative subtrees when drift was detected. Hulten and Domingos [39] utilize the Hoeffding bound underlying the VFDT algorithm also for learning Bayesian network structures efficiently from data streams. That algorithm can now be extended to concept drift situations using the approaches of [37, 38]. The schematic approach to building adaptive prediction systems thus also has great potential in the literature devoted to probabilistic graphical models.

The generic schema of Figure 7 requires detection of change in the (internal) stream of losses, $\{L_t\}$. Several methods can be used here. For instance, both the Cumsum [40] and the Page-Hinkley [40] tests are based on the sequential probability test; [41] utilized P-charts from statistical quality control; [38, 42, 43] compare distributions for different time-windows following the idea of [44]. Finally, [45] utilized a Bayesian approach to average over the uncertainty about how much of the observed data $\boldsymbol{X}_{0:t}$ that is relevant for learning the model at time t.

It is worth noticing that the general principle of using a change detector to determine what data is relevant to learn from at a given point in time also *can* be employed when feedback in terms of the correct prediction is unavailable. In this case one can only look for changes in the input stream $\{\boldsymbol{z}_t\}$, however, and will thus be unable to separate between *real* and *virtual* drift. In these cases, one solution is to use a fixed-size sliding window, and only use data inside the window for learning (see, e.g., [12]); this is the simplistic alternative to methods using adaptive window lengths (see, e.g., [46]). While windowing techniques are conceptually easy and simple to implement, the methods require in their simplest form that a potentially large sub-sample of the data is kept in memory, and that one can drop the oldest elements out of memory when new elements come in. A simplification in that respect can be employed if the distribution is from the exponential family. In this case, one can monitor *sufficient statistics* instead of the full data-sample, and (to avoid remembering the sequencing of the observations), use exponential decay on the statistics to adapt to changing environments. This approach was employed by Olesen et al. [47] for learning parameterizations of discrete Bayesian networks. The sufficient statistics were defined as a set of (partial) counts $\{s_{ijk}\}$, and before a new data sample was incorporated into the set of sufficient statistics, each s_{ijk} was multiplied by a fixed scaler $q \in [0, 1]$.

Following a Bayesian approach for stationary streams, one will use the prediction model

$$p(y_t|\boldsymbol{z}_t, \boldsymbol{x}_{0:t-1}) = \int_{\boldsymbol{\Theta}} p(y_t|\boldsymbol{z}_t, \boldsymbol{x}_{t-k:t-1}, \boldsymbol{\Theta}) p(\boldsymbol{\Theta}|\boldsymbol{z}_t, \boldsymbol{x}_{0:t-1}) \, \mathrm{d}\boldsymbol{\Theta}.$$

This can easily be extended to drifting domains by allowing $\boldsymbol{\Theta}$ to change with t, and explicitly *model* the dynamics. This approach was, for instance, pursued by [48, 49]. Penny and Roberts [48] considered a logistic regression model, where the weights of the model at time t, $\boldsymbol{w}_t$, where modelled as Gaussian variables. The key idea was then to connect the variables over time, so that $\boldsymbol{w}_{t+1}|\boldsymbol{w}_t$ had expected value $\boldsymbol{w}_t$ and covariance determined by the a priori belief in the strength of the concept drift. While the posterior for $\boldsymbol{w}_t$ is not available in closed form, Penny and Roberts approximated it using a Gaussian with parameters determined using approximate inference techniques. A similar strategy was pursued by [50, 51], but with a support vector machine as the base classifier. Finally, Borchani et al. [52] used latent variables to capture concept drift. They started out with a dynamic Naive Bayes classifier, then extended the model using a sequence of latent variables H_t, which a priori was assumed to follow Brownian motion. The parameters of the classifier at time t where then scaled by H_t to incorporate the effect of a "global" external process. The authors argue that their setup can easily be extended to arbitrarily complicated drifting regimes by adding more latent variables exhibiting different behaviors. For instance, a discrete latent variable will work well with abrupt shifts (and in particular for reoccurring concepts), while a sequence of latent variables with strong autocorrelation is natural to model slowly drifting domains. Another advantage of this approach is that even if the model is designed to cope with drifting domains, it respects the stationary assumption of Equation (6) at the same time, therefore being able to leverage the efficient inference techniques described in Section 3.2.

In the PGM literature there are also approaches that adapt the underlying *network structure* of the prediction model. For instance, [53] learned augmented Naïve Bayes classifiers incrementally, thereby being able to handle concept drift situations. Similarly, [41] gradually increases the complexity of a Naïve Bayes classifier (up to k-dependence classifiers) as more *relevant* data is observed. An external change detector determines when concept drift is observed, and the classifier is updated accordingly (the structure can also be thinned when applicable). Finally, [54–56] all consider learning of general Bayesian network structure in drifting domains.

5 Variable selection

For datasets with a high number of variables, a key step is to determine the most informative variables for the model and exclude the less informative ones. Furthermore, some of the variables may be redundant in the context of others. Learning Bayesian networks with an excessive number of variables may yield models overfitting irrelevant aspects of the data, and therefore showing a poor predictive power. Also, the complexity of learning usually grows exponentially with the number of variables, and hence the computational cost of learning with all the available variables may become unaffordable.

AMIDST models are fundamentally oriented to classification, and therefore there is a distinguished variable or set of variables that constitute the class, where the rest of the

variables are referred to as *features* (see Figure 3). Feature selection methods can be grouped into three main blocks, *wrapper*, *filter* and *embedded* methods [57, 58]. There are also mixed filter-wrapper approaches, some of which are able to operate in high dimensional domains [59, 60].

Wrapper methods [61] explore the space of possible feature subsets, optimizing some model-dependent metric. It means that for each explored feature subset, a model is learnt and evaluated. Due to the size of the datasets and the time constraints, traditional wrapper approaches for variable selection are not viable within the context of the AMIDST project and therefore we will not review them in this document. Embedded methods [62] are also model-dependent. They use some specific property of the target model to guide the search procedure. On the other hand, filter methods [63] are independent of the underlying model. These methods use some score function to produce a ranking of the features. In general, a threshold is defined and variables below it are discarded. For instance, a threshold value of the score function or, alternatively, a maximum number of features may be determined.

In what follows, we present an analysis of the state of the art on filter and embedded methods for feature selection. The analysis is focused on methods able to deal with large amounts of variables and/or data streams.

5.1 Filter methods

Some of the most outstanding filter methods are based on the use of information-theoretic score functions, all of them related to the concept of *entropy* [64]. Throughout this exposition we shall assume a set of discrete or continuous variables $\boldsymbol{X} = \{X_1, \ldots, X_n\}$ and a class variable C which is always discrete. The entropy of a discrete variable $X \in \boldsymbol{X}$ is

$$H(X) = -\sum_{x \in \Omega_X} p(x) \log p(x),$$

where the summation is changed to integration if X is continuous. The entropy measures the uncertainty in the distribution of X. Given two variables $X_i, X_j \in \boldsymbol{X}$, the *conditional entropy* of X_i given X_j is

$$H(X_i|X_j) = -\sum_{x_j \in \Omega_{X_j}} p(x_j) \sum_{x_i \in \Omega_{X_i}} p(x_i|x_j) \log p(x_i|x_j),$$

and it quantifies the uncertainty that remains in the distribution of X_i after observing X_j.

The amount of information shared by two variables $X_i, X_j \in \boldsymbol{X}$ can be measured by their *mutual information* [64, 65]:

$$I(X_i, X_j) = H(X_i) - H(X_i|X_j). \tag{7}$$

It can be interpreted as the amount of uncertainty in X_i that is removed after observing X_j. Similarly, given $X_i, X_j, X_k \in \boldsymbol{X}$ the *conditional mutual information* between X_i and X_j given X_k can be defined as

$$I(X_i, X_j|X_k) = H(X_i|X_k) - H(X_i|X_j, X_k).$$

Another measure that is commonly used in filter methods is the *symmetric uncertainty*, defined as

$$\mathrm{SU}(X_i, X_j) = \frac{I(X_i, X_j)}{H(X_i) + H(X_j)}, \tag{8}$$

which is in fact a normalized version of the mutual information.

A thorough analysis of information-theoretic filter methods for variable selection is reported in [66]. The analysis is based on the *conditional likelihood* of the class variable given parameters $\boldsymbol{S}$ (features included in the model) and $\boldsymbol{\tau}$ (parameters of the distributions involved in the model) for a data set $\mathcal{D} = \{(\boldsymbol{x}^i, c^i), i = 1, \ldots, N\}$, defined as

$$\mathcal{L}(\boldsymbol{S}, \boldsymbol{\tau}|\mathcal{D}) = \prod_{i=1}^{N} q(c^i|\boldsymbol{x}^i, \boldsymbol{\tau})$$

where q is the learnt model. It is shown in [66] that the conditional likelihood is maximized by minimizing $I(\boldsymbol{X} \setminus \boldsymbol{S}, C|\boldsymbol{S})$. In other words, the conditional likelihood is maximized when the mutual information between the class and the variables not included in the model, given the variables actually included, is minimized.

From the perspective of data stream processing and scalability, it is worth pointing out that the information-theoretic score functions mentioned so far can be efficiently updated after the arrival of new data if the distributions involved belong to the exponential family. It is enough to store a set of sufficient statistics corresponding to the parameters of the model, which are just counts for discrete variables and first and second order sample moments for Gaussian variables.

Filter methods usually rank the variables independently from each other. This is the approach taken in [67], where vertical parallelization is used. Subsets of candidate variables are distributed through the available computing units, where the filter is actually applied. The features that are discarded receive one vote, and finally those with more votes are excluded from the model.

The use of filter methods when learning from data streams was analyzed in [68], where the variables are ranked according to their mutual information with the class, as defined in Equation (7). The sufficient statistics involved in the calculation of the mutual

information of each feature are stored and updated as new items come from the input stream. Naive Bayes and kNN classifiers are considered.

A similar approach is followed in [69]. Instead of the mutual information, the symmetric uncertainty in Equation (8) is used as filter measure. The symmetric uncertainty is computed from a sliding window on the input stream.

The problem of feature selection and concept drift are linked in [70], where it is assumed that a concept drift happens when the set of selected features changes. Features are ranked using the mutual information, but only those scarcely correlated with the already included features are appended to the model. Streams are handled by keeping records of the sufficient statistics involved in the calculation of the mutual information and the linear correlation coefficient between features.

A different idea is exploited in [71]. A Lasso regression model [72] is computed from a sliding window on the input stream. The features are ranked according to their corresponding regression coefficients in the regression model.

A general view of filter methods is given in [73]. Rather than using a fixed filter measure, the variables are ranked according to the reduction in expected loss when including them in the model.

5.2 Embedded methods

Embedded methods for feature selection are model-dependent in the sense that they base the selection procedure on some properties of the model. Embedded methods have been extensively used in regression problems. Typically, the selection procedure involves the use of the regression coefficient, using some kind of regularization in order to obtain values close to zero for the coefficient of irrelevant variables. That is the case of the work described in [74]. It is shown how Lasso regression models can be estimated from data streams. The joint distribution of the response and the covariates (features) is dynamically updated from a sliding window.

Lasso regression is also the underlying model in [75]. The regression parameters are learnt from streams using recursive least squares, where the covariance estimates at time $t + 1$ are learnt from the observations at time $t + 1$ and the existing estimates at time t. The importance of past observations is scaled down using exponential forgetting. A similar approach is taken in [76], where recursive partial least squares is solved using singular value decomposition with a Lasso-like penalty factor aimed at sparsification.

Stepwise variable selection is explored in [77] within the context of credit card fraud detection. Logistic regression models are used for classifying the credit card use. Sparse projection is inspired by principal component analysis is used to reduce the number of features in linear classifiers in [78].

Another important group of embedded methods rely on Bayesian techniques. A review of such methods, also focused on regression problems, can be found in [79]. Variable

selection in the Bayesian case is closely related to the so called Automatic Relevance Determination [18, 80, 81]. An example is the *adaptive shrinkage* technique, illustrated in Figure 8. The contribution of each variable X_i is modeled through a parameter β_i, over which a prior can be specified as $\beta_i|\tau_i^2 \sim \mathcal{N}(0, \tau_i^2)$. A prior must also be specified for τ_i^2 so that the distribution of β_i adopts the appropriate shape to represent the fact that β_i should concentrate around zero if X_i is irrelevant according to the data. The degree of sparseness is controlled by the prior on τ_i^2, usually a Γ distribution with parameters a and b is used. Finally, an indicator variable I_i can be included for each X_i, so that $I_i = 1$ if $|\beta_i| > c$ for some threshold $c > 0$, meaning that X_i is only included in the model in such case [82].

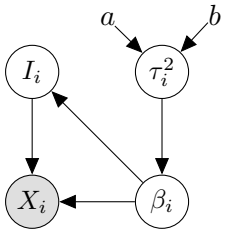

Figure 8: An illustration of Bayesian feature selection as a graphical model.

6 Conclusions

This document describes the state-of-the art of learning in the context of the AMIDST modelling framework. Due to the requirement that the learning should be able to leverage both expert knowledge and information hidden in the data, the focus for the document is on learning in a Bayesian setting. We have considered learning using *variational Bayes*, and described ideas used for speeding up learning in that context. Furthermore, we discussed the state of the art for learning from non-stationary data streams. Finally, since the AMIDST model class has a (semi-)fixed structure, the vast literature on learning Bayesian network structure was only mentioned in passing. Instead, we discussed scalable techniques for feature selection, which can be employed to avoid learning models that are prune to overfitting.

References

[1] Pearl, J.: Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann Publishers Inc., San Mateo, CA. (1988)

[2] Jensen, F.V., Nielsen, T.D.: Bayesian Networks and Decision Graphs. 2nd edn. Springer Publishing Company, Incorporated (2007)

[3] Kjærulff, U.B., Madsen, A.L.: Bayesian Networks and Influence Diagrams: A Guide to Construction and Analysis. 2nd edn. Springer Publishing Company, Incorporated (2013)

[4] Langseth, H., Madsen, A.L., Nielsen, T.D., Rumí, R., Salmerón, A.: State of the art of inference in hybrid and dynamic models (2014) Deliverable 3.1 of the AMIDST project, `amidst.eu`.

[5] Cooper, G.F., Herskovits, E.: A Bayesian method for the induction of probabilistic networks from data. Machine Learning Journal **9** (1992) 309–347

[6] Spirtes, P., Glymour, C., Scheines, R.: Causation, Prediction, and Search. Second edn. MIT Press (2000)

[7] Borchani, H., Fernández, A., Gundersen, O.E., Hovda, S., Langseth, H., Madsen, A.L., Martínez, A.M., Sáez-Martínez, R., Masegosa, A., Nielsen, T.D., Salmerón, A., Sørmo, F., Weidl, G.: The AMIDST modelling framework – Initial draft report (2014) Deliverable 2.1 of the AMIDST project, `amidst.eu`.

[8] Borchani, H., Fernández, A., Gundersen, O.E., Hovda, S., Langseth, H., Madsen, A.L., Martínez, A.M., Sáez, R., Masegosa, A.R., Nielsen, T.D., Salmerón, A., Sørmo, F., Weidl, G.: The AMIDST modelling framework – final report (2015) Deliverable 2.2 of the AMIDST project, `amidst.eu`.

[9] Koller, D., Friedman, N.: Probabilistic Graphical Models: Principles and Techniques. MIT Press (2009)

[10] West, M., Harrison, P.J.: Bayesian Forecasting & Dynamic Models. 2nd edn. Springer Verlag (1997)

[11] Barber, D., Cemgil, A.T., Chiappa, S.: Bayesian Time Series Models. Cambridge University Press, New York, NY, USA (2011)

[12] Widmer, G., Kubat, M.: Learning in the presence of concept drift and hidden contexts. Machine Learning **23**(1) (1996) 69–101

[13] Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., Bouchachia, A.: A survey on concept drift adaptation. ACM Computing Surveys **46**(4) (March 2014) 44:1–44:37

[14] Jordan, M.I., Ghahramani, Z., Jaakkola, T.S., Saul, L.K.: An introduction to variational methods for graphical models. Machine Learning **37** (1999) 183–233

[15] Attias, H.: A variational Bayesian framework for graphical models. Advances in neural information processing systems (2000) 209—215

[16] Jaakkola, T.S., Qi, Y.: Parameter expanded variational Bayesian methods. In: Advances in Neural Information Processing Systems. (2006) 1097–1104

[17] Beal, M.J.: Variational algorithms for approximate Bayesian inference. PhD thesis, Gatsby Computational Neuroscience Unit, University College London (2003)

[18] Qi, Y.A., Jaakkola, T.S.: Parameter expanded variational Bayesian methods. In: Advances in Neural Information Processing Systems 19, Proceedings of the Twentieth Annual Conference on Neural Information Processing Systems. (2006) 1097–1104

[19] Luttinen, J.: Fast variational Bayesian linear state-space model. In: Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2013, Berlin, Heidelberg, Springer (2013) 305–320

[20] Hoffman, M.D., Blei, D.M., Wang, C., Paisley, J.: Stochastic variational inference. Journal of Machine Learning Research **14** (2013) 1303–1347

[21] Johnson, M., Willsky, A.: Stochastic variational inference for Bayesian time series models. In Jebara, T., Xing, E.P., eds.: Proceedings of the 31st International Conference on Machine Learning (ICML-14), JMLR Workshop and Conference Proceedings (2014) 1854–1862

[22] Foti, N.J., Xu, J., Laird, D., Fox, E.B.: Stochastic Variational Inference for Hidden Markov Models. Advances in Neural Information Processing Systems 27 (2014) 1–9

[23] Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent Dirichlet Allocation. Journal of Machine Learning Research **3**(4-5) (may 2003) 993–1022

[24] Hoffman, M.D., Blei, D.M., Bach, F.: Online learning for latent dirichlet allocation. In: Advances in Neural Information Processing Systems. (2010) 1–9

[25] Broderick, T., Boyd, N., Wibisono, A., Wilson, A.C., Jordan, M.I.: Streaming variational Bayes. In: Advances in Neural Information Processing Systems 26. (2013) 1727–1735

[26] Theis, L., Hoffman, M.D.: A trust-region method for stochastic variational inference with applications to streaming data. In: Proceedings of the 32nd International Conference on Machine Learning. (2015)

[27] Campbell, T., How, J.P.: Approximate decentralized Bayesian inference. In: Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence, UAI 2014, Quebec City, Quebec, Canada, July 23-27, 2014. (2014) 102–111

[28] Campbell, T., Straub, J., Fisher III, J.W., How, J.P.: Streaming, distributed variational inference for Bayesian nonparametrics. In: Advances in Neural Information Processing Systems (NIPS). (2015)

[29] Minsker, S., Srivastava, S., Lin, L., Dunson, D.B.: Robust and scalable Bayes via a median of subset posterior measures (2014) `http://arxiv.org/abs/1403.2660`.

[30] Srivastava, S., Li, C., Dunson, D.B.: Scalable Bayes via Barycenter in Wasserstein Space (2015) http://arxiv.org/abs/1508.05880.

[31] Xu, M., Lakshminarayanan, B., Teh, Y.W., Zhu, J., Zhang, B.: Distributed Bayesian posterior sampling via moment sharing. In: Advances in Neural Information Processing Systems 27, Curran Associates, Inc. (2014) 3356–3364

[32] Babagholami-Mohamadabadi, B., Yoon, S., Pavlovic, V.: D-MFVI: distributed mean field variational inference using Bregman ADMM (2015) http://arxiv.org/abs/1507.00824.

[33] Murphy, K., Weiss, Y.: The factored frontier algorithm for approximate inference in DBNs. In: Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence (UAI). (2001) 378–385

[34] Boyen, X., Koller, D.: Tractable inference for complex stochastic processes. In: Proceedings of the Fourteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI-98). (1998) 33–42

[35] Boyen, X., Koller, D.: Exploiting the architecture of dynamic systems. In: Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI-99). (1999) 313–320

[36] Domingos, P., Hulten, G.: Mining high-speed data streams. In: Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. KDD '00, ACM (2000) 71–80

[37] Hulten, G., Spencer, L., Domingos, P.: Mining time-changing data streams. In: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. KDD '01, ACM (2001) 97–106

[38] Gama, J., Fernandes, R., Rocha, R.: Decision trees for mining data streams. Intelligent. Data Analysis **10**(1) (2006) 23–45

[39] Hulten, G., Domingos, P.: Mining complex models from arbitrarily large databases in constant time. In: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, NY, USA, ACM (2002) 525–531

[40] Page, E.S.: Continuous inspection schemes. Biometrika **41** (1954) 100–115

[41] Castillo, G., Gama, J.: Adaptive Bayesian network classifiers. Intelligent Data Analysis **13** (2009) 39–59

[42] Vorburger, P., Bernstein, A.: Entropy-based concept shift detection. In: Proceedings of the Thirteenth IEEE International Conference on Data Mining, IEEE Press (2006) 1113–1118

[43] Bach, S.H., Maloof, M.A.: Paired learners for concept drift. In: Proceedings of the Eighth IEEE International Conference on Data Mining, IEEE Press (2008) 23–32

[44] Kifer, D., Ben-David, S., Gehrke, J.: Detecting change in data streams. In: Proceedings of the Thirtieth International Conference on Very Large Data Bases - Volume 30. VLDB '04, VLDB Endowment (2004) 180–191

[45] Bach, S.H., Maloof, M.A.: A Bayesian approach to concept drift. In: Advances in Neural Information Processing Systems (NIPS). (2010) 127–135

[46] Bifet, A., Gavaldà, R.: Learning from time-changing data with adaptive windowing. In: Proceedings of the Seventh SIAM International Conference on Data Mining. (2007) 443–448

[47] Olesen, K.G., Lauritzen, S.L., Jensen, F.V.: aHUGIN: A system creating adaptive causal probabilistic networks. In: UAI '92: Proceedings of the Eighth Annual Conference on Uncertainty in Artificial Intelligence, Stanford University, Stanford, CA, USA, July 17-19, 1992. (1992) 223–229

[48] Penny, W.D., Roberts, S.J.: Dynamic logistic regression. In: Proceedings of the International Joint Conference on Neural Networks (IJCNN'99). Volume 3. (1999) 1562–1567

[49] Garnett, R.: Learning from data streams with concept drift. PhD thesis, Oxford University (2010)

[50] Turkov, P., Krasotkina, O., Mottl, V.: Bayesian approach to the concept drift in the pattern recognition problems. In Perner, P., ed.: Proceedings of the 8th International Conference on Machine Learning and Data Mining in Pattern Recognition, Springer Berlin Heidelberg (2012) 1–10

[51] Turkov, P., Krasotkina, O., Mottl, V.: The Bayesian logistic regression in pattern recognition problems under concept drift. In: Proceedings of the 21st International Conference on Pattern Recognition. (2012) 2976–2979

[52] Borchani, H., Martínez, A.M., Masegosa, A., Langseth, H., Nielsen, T.D., Salmerón, A., Fernández, A., Madsen, A.L., Sáez, R.: Modeling concept drift: A probabilistic graphical model based approach. In: In proceedings of The Fourteenth International Symposium on Intelligent Data Analysis. (2015) 72–83

[53] Alcobé, J.R.: Incremental augmented naive Bayes classifiers. In: Proceedings of the 16th Eureopean Conference on Artificial Intelligence, ECAI'2004. (2004) 539–543

[54] Lam, W., Bacchus, F.: Using new data to refine a Bayesian network. In: Proceedings of the Tenth International Conference on Uncertainty in Artificial Intelligence. UAI'94, San Francisco, CA, USA, Morgan Kaufmann Publishers Inc. (1994) 383–390

[55] Friedman, N., Goldszmidt, M.: Sequential update of Bayesian network structure. In: Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence. UAI'97, San Francisco, CA, USA, Morgan Kaufmann Publishers Inc. (1997) 165–174

[56] Nielsen, S.H., Nielsen, T.D.: Adapting Bayes network structures to non-stationary domains. International Journal of Approximate Reasoning **49**(2) (2008) 379 – 397

[57] Kumar, V., Minz, S.: Feature selection: A literature review. Smart Computing Review **4**(3) (2014) 211–229

[58] Li, J., Cheng, K., Wang, S., Morstatter, F., Trevino, R., Tang, J., Liu, H.: Feature selection: A data perspective. arXiv preprint arXiv:1601.07996 (2016)

[59] Bermejo, P., Gámez, J., Puerta, J.: A GRASP algorithm for fast hybrid (filter-wrapper) feature subset selection in high-dimensional datasets. Pattern Recognition Letters **32** (2011) 701–711

[60] Bermejo, P., de la Ossa, L., Gámez, J., Puerta, J.: Fast wrapper feature subset selection in high-dimensional datasets by means of filter re-ranking. Knowledge-Based Systems **25** (2012) 35–44

[61] John, G., Kohavi, R., Pfleger, P.: Irrelevant features and the subset selection problem. In: Machine Learning: Proceedings of the Eleventh International Conference, Morgan Kaufmann (1994) 121–129

[62] Lan, T., Chapelle, O., Weston, J., Elisseeff, A.: Embedded methods. In Guyon, I., Gunn, S., Nikravesh, M., Zadeh, L., eds.: Feature extraction. Fundations and applications. Volume 207 of Studies in Fuzziness and Soft Computing. Springer (2006) 137–165

[63] Duch, W.: Filter methods. In Guyon, I., Gunn, S., Nikravesh, M., Zadeh, L., eds.: Feature extraction. Fundations and applications. Volume 207 of Studies in Fuzziness and Soft Computing. Springer (2006) 89–117

[64] Cover, T.M., Thomas, J.A.: Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing). 2nd edn. Wiley-Interscience (2006)

[65] Shannon, C.: A mathematical theory of communication. Bell Systems Technical Journal **27**(3) (1948) 379–423

[66] Brown, G., Pocock, A., Zhao, M.J., Luján, M.: Conditional likelihood maximisation: A unifying framework for information theoretic feature selection. Journal of Machine Learning Research **13** (January 2012) 27–66

[67] Morán-Fernández, L., Bolón-Canedo, V., Alonso-Betanzos, A.: A time efficient approach for distributed feature selection partitioning by features. CAEPIA 2015. Lecture Notes in Artificial Intelligence **9422** (2015) 245–254

[68] Katakis, I., Tsoumakas, K., Vlahavas, I.: Dynamic feature space and incremental feature selection for the classification of textual data streams. In: in ECML/PKDD-2006 International Workshop on Knowledge Discovery from Data Streams. (2006) 107–116

[69] Lutu, P.: Fast feature selection for naive Bayes classification in data stream mining. In: Proceedings of the World Congress on Engineering (WCE 2013). Volume III. (2013) 1549–1554

[70] Zhou, X., Li, S., Chang, C., Wu, J., Liu, K.: Information-value-based feature selection algorithm for anomaly detection over data streams. Technical Gazette **21** (2014) 223–232

[71] Huang, H., Yoo, S., Kasiviswanathan, S.: Unsupervised feature selection on data streams. In: Proceedings of the 24th ACM International on Conference on Information and Knowledge Management. (2015) 1031–1040

[72] Tibshirani, R.: Regression shrinkage and selection via the Lasso. Journal of the Royal Statistical Society, Series B **58** (1994) 267–288

[73] Ahmed, S., Narasimhan, H., Agarwal, S.: Bayes optimal feature selection for supervised learning with general performance measures. In: Proceedings of the 31st Conference on Uncertainty in Artificial Intelligence. (2015)

[74] Anagnostopoulos, C., Adams, N.M., Hand, D.J.: Deciding what to observe next: Adaptive variable selection for regression in multivariate data streams. In: Proceedings of the 2008 ACM Symposium on Applied Computing. SAC '08, New York, NY, USA, ACM (2008) 961–965

[75] Anagnostopoulos, C., Tasoulis, D., Hand, D.J., Adams, N.M.: Online optimization for variable selection in data streams. In: Proceedings of the 2008 Conference on ECAI 2008: 18th European Conference on Artificial Intelligence, Amsterdam, The Netherlands, The Netherlands, IOS Press (2008) 132–136

[76] McWilliams, B., Montana, G.: Sparse partial least squares regression for on-line variable selection with multivariate data streams. Statistical Analysis and Data Mining **3**(3) (2010) 170–193

[77] Ise, M., Niimi, A., Konishi, O.: Feature selection in large scale data stream for credit card fraud detection. In: Proceedings of the Fifth International Workshop on Computational Intelligence and Applications. (2009) 202–207

[78] Wang, J., Zhao, P., Hoi, S., Jin, R.: Online feature selection and its applications. IEEE Transactions on Knowledge and Data Engineering **26**(3) (March 2014) 698–710

[79] O'Hara, R., Sillanpää, M.: A review of Bayesian variable selection methods: what, how and which. Bayesian Analysis **4**(1) (2009) 85–118

[80] Bishop, C.M.: Variational principal components. In: Proceedings Ninth International Conference on Artificial Neural Networks, ICANN'99, IEE (January 1999) 509–514

[81] Bishop, C.M., Tipping, M.E.: Variational relevance vector machines. In: Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence. UAI'00, San Francisco, CA, USA, Morgan Kaufmann Publishers Inc. (2000) 46–53

[82] Hoti, F., Sillanpää, M.: Bayesian mapping of genotype x expression interactions in quantitative and qualitative traits. Heredity **97** (2006) 4–18