[*]

Title	:	Unsupervised Anomaly Detection in Large Databases Using Bayesian Networks.
Authors	:	Antonio Cansado and Alvaro Soto
Affiliation	:	Pontificia Universidad Católica de Chile
Abbreviated Title	:	Anomaly Detection in Large Databases
Mailing Address	:	Departamento de Ciencia de la Computación Pontificia Universidad Católica de Chile Casilla 306 - Santiago 22 Chile

[*]This work is partially funded by FONDECYT grant 1030336.

Unsupervised Anomaly Detection in Large Databases Using Bayesian Networks

Antonio Cansado and Alvaro Soto

e-mail: [acansado,asoto]@ing.puc.cl

Department of Computer Science

Pontificia Universidad Catolica de Chile Casilla 306 - Santiago 22 - CHILE

December 26, 2007

Abstract

Today, there has been a massive proliferation of huge databases storing valuable information. The opportunities of an effective use of these new data sources are enormous, however, the huge size and dimensionality of current large databases call for new ideas to scale up current statistical and computational approaches. This paper presents an application of Artificial Intelligence technology to the problem of automatic detection of candidate anomalous records in a large database. We build our approach with three main goals in mind: 1)An effective detection of the records that are potentially anomalous, 2)A suitable selection of the subset of attributes that explains what makes a record anomalous, and 3)An efficient implementation that allows us to scale the approach to large databases. Our algorithm, called Bayesian Network Anomaly Detector (BNAD), uses the joint probability density function (pdf) provided by a Bayesian Network (BN) to achieve these goals. By using appropriate data structures, advanced caching techniques, the flexibility of Gaussian Mixture models, and the efficiency of BNs to model joint pdfs, BNAD manages to efficiently learn a suitable BN from a large dataset. We test BNAD using synthetic and real databases, the latter from the fields of manufacturing and astronomy, obtaining encouraging results.

1 Introduction

Today, technology is changing the way people produce and handle information. From business to science and engineering, there has been a massive proliferation of huge databases storing valuable information. The opportunities of an effective use of these new data sources, Gigabytes up to Terabytes, are enormous, however, traditional data analysis techniques have not kept in pace with the type of processing needed by the new volumes of data.

The huge size and dimensionality of current large databases require new ideas to scale up current statistical and computational approaches. This is especially critical when the system needs to interact with a human expert, as it is usually the case. If a data analysis tool takes weeks or months to return a result, the interaction between the analyst and the data is severely limited.

In this paper we present an application of AI technology to the problem of automatic detection of anomalous records in a large database. Although we refer to the problem as anomaly detection, this problem has been cast under different names, such as outlier detection, novelty detection, noise detection, deviation detection, or exception mining (Hodge and Austin 2004). The detection of anomalies in a database plays a critical role in many tasks. Depending on the application, these anomalies may correspond to fraudulent transactions in a financial database, strange celestial objects in an astrophysical data catalog, or records of faulty products in a production database, see (Hodge and Austin 2004) for a extensive list of applications.

We build our approach with three main goals in mind: Goal-1)An effective detection of the records that are potentially anomalous, Goal-2)A suitable selection of the subset of attributes that explains what makes a record anomalous, and Goal-3)An efficient implementation that allows us to scale the approach to large databases.

We follow a probabilistic approach by modeling the joint probability density function (pdf) of the attributes of the records in the database. This function provides a straight forward method to rank the records according to their oddness (Goal-1). While highly common records, well explained by the model, receive a high likelihood, strange records, poorly explained by the model, receive a low likelihood.

Although the probabilistic approach seems to be fruitful, the estimation of a joint pdf in a high dimensional space is an extremely challenging task. Any direct attempt to model the joint pdf would require to fit a prohibitively large number of parameters. Furthermore, searching for models in a large parameter space increases the chances of getting stuck in a local optimum (Mitchell 1997).

To handle the complexity associated to the estimation of the joint pdf, we use a Bayesian Network (BN) (Pearl 1988) (Lauritzen 1996) (Jensen 2001) (Neapolitan 2004). A BN provides an efficient graphical representation of a joint pdf by taking advantage of conditional independence relations inherent to most high dimensional datasets. These conditional independence relations among the attributes of the records in the database provide a suitable factorization of the joint pdf in terms of simpler local conditional probability functions, whose reduced dimensionality simplifies the estimation process.

Another key advantage of using BNs is the straightforward evaluation of the likelihood of each data point, which otherwise can be the bottleneck for the detection of strange objects. Furthermore, the local structure and conditional pdfs embedded in the network, provide relevant information about which groups of attributes are related to other groups, to what extent they are related, and under what circumstances. These features provide key information to find an explanation about the set of attributes and conditions that make an object anomalous (Goal-2).

In general, finding an appropriate BN to model the joint pdf involves two main steps: learning the structure of the network and learning the conditional pdfs that relate the nodes in the network (see Section 3.2 for more details). To learn the structure of the network, we extend the Sparse Candidate Algorithm (SCA) (Friedman et al. 1999) to the case of continuous variables. We choose SCA due to its ability to scale to large datasets. Instead of using traditional Greedy Hill Climbing (GHC) search over the space of possible structures (Chickering 1996b), where local changes, such as adding, deleting or reversing arcs, require $O(n^2)$ possible changes to each network of n variables, SCA efficiently shrinks the search space by selecting only the most relevant parents for each variable. The selection of relevant parents is guided by a scoring metric based on information theory (Friedman et al. 1999).

To learn the conditional pdfs that relate each node with its parents, we use a Gaussian Mixture model (GMM) trained with an accelerated version of the Expectation Maximization (EM) algorithm (Dempster et al. 1977). Our accelerated version of EM, described in (Soto et al. 2007), is based on the use of data summarization techniques, such as the one used in (Zhang et al. 1996) and (Moore 1999). By using these techniques, EM does not need to sweep over all the observations and dimensions during each iteration, but just over their summaries. This provides a great efficiency that allows us to scale the approach to large databases (Goal-3).

The use of GMMs to model the local conditional pdfs at each node provides a straightforward method to increase the extent of the search of the original SCA. This is provided by particular properties of GMMs. Following the regular search of SCA, after we run intensive computations to obtain a local joint pdf among a set of variables, it is possible to use properties from GMMs to obtain any marginal or conditional density among these variables by simply applying basic matrix operations, such as swapping or deleting columns and rows in the covariance matrix. Using such properties, we can increase the scope of the search for network structures without adding a significant computational load to our algorithm.

We call our algorithm Bayesian Network Anomaly Detector (BNAD). In this paper, we present the main components of BNAD, and we also evaluate its performance to detect anomalies in synthetic and real databases. The paper is organized as follows. Section 2 provides background information. Section 3 discusses relevant previous work. Section 4 presents the details of our algorithm. Section 5 shows the main results of applying BNAD to synthetic and real databases. Finally, Section 6 presents the main conclusions of this work.

2 Background

In this section, we start by briefly describing the main elements of a BN along with the notation we use in the paper. Afterwards, we review the scoring functions used to explore the space of possible BN structures that model the data. Finally, we describe GMMs and their properties that are relevant to this work.

2.1 Bayesian Networks (BNs)

Consider a set $\mathbf{x} = \{x_1, \ldots, x_n\}$ of continuous random variables. We use lowercase boldface letters, such as $\mathbf{x}$, to denote sets of single random variables, such as x_i. A BN, also called belief network, is a directed acyclic graph G that represents independencies embodied in a given joint probability distribution over the set of variables x_i, $i \in \{1, \ldots, n\}$. The graph G associates a vertex or node V_i to each variable x_i. The joint pdf of the variables x_i can be factorized as $\prod_i^n p(x_i|Pa^G(x_i))$ where $Pa^G(x_i)$ is the set of direct parents of x_i in G. Given the values of $Pa^G(x_i)$, x_i is conditionally independent of all the variables that are not descendant of V_i in the graph G. Therefore, edges define dependency relations among variables. We denote by $B = \langle G, \theta \rangle$ the BN represented by graph G, where the parameters of each factor $p(x_i|Pa^G(x_i))$ are contained in θ. We also denote as $De^G(x_i)$ the set of descendants of x_i in G.

In general, finding an optimal structure for a BN is NP-complete (Chickering 1996a). Thus, it is not possible to search for an optimal structure using the whole space of feasible networks. Fortunately, studies have shown experimentally that a "good" network structure is often enough to provide a suitable approximation to the underlying model (Heckerman 1996).

2.2 Scoring Functions

A well known function to measure the distance from a "true" probability distribution $p(\mathbf{x})$ to an arbitrary probability distribution $q(\mathbf{x})$ is the *Kullback-Leibler* divergence (KLD) (Kullback and Leibler 1951), defined as

$$KLD\,(p(\mathbf{x})\,\|q(\mathbf{x})\,) = \int_{\mathbf{x}} p(\mathbf{x})\ \log \frac{p(\mathbf{x})}{q(\mathbf{x})}\ d\mathbf{x}. \tag{1}$$

Although not a symmetric function, therefore not a true metric distance, the KLD satisfies many important mathematical properties. In particular, it can be used to measure the degree of dependency between two random variables x_i and x_j by measuring the dissimilarity between the joint distribution $p(x_i, x_j)$ and the product of its marginals distributions $p(x_i)$ and $p(x_j)$,

$$KLD\,(p(x_i, x_j)\,\|p(x_i)p(x_j)\,) = \int_{x_i, x_j} p(x_i, x_j)\ \log \frac{p(x_i, x_j)}{p(x_i)p(x_j)}\ dx_j\ dx_i. \tag{2}$$

Eq. (2) is also known as the mutual information or relative entropy between the joint and the product distribution (Cover and Thomas 1991). As in the case of SCA, we use mutual information as the key metric to select the initial set of candidate parents to build the BN structure.

Using the KLD, (Friedman et al. 1999) defines Discrepancy between the empirical joint density of x_i and x_j, $\hat{p}(x_i, x_j)$, and the corresponding joint density implied by the current estimation B of the BN structure, $p_B(x_i, x_j)$, as

$$Disc(x_i, x_j) = KLD(\hat{p}(x_i, x_j)\,\|p_B(x_i, x_j)\,). \tag{3}$$

Discrepancy is the key score used by SCA to select a new candidate parent for a node. In this work we also use this metric to explore the space of BN structures.

2.3 Gaussian Mixture Model (GMM)

Given a finite set $\mathbf{x} = \{x_1, \ldots, x_n\}$ of continuous random variables, finding their joint probability distribution $p(\mathbf{x})$ in a large dimensional space is usually a challenging task. Fortunately, it has been proved (Silverman 1986) that such distribution can be approximated with arbitrary accuracy using a sum of Gaussian distributions weighted by membership probabilities w_h,

$$p(\mathbf{x}) = \sum_{h=1}^{k} w_h \, p_h(\mathbf{x}|\boldsymbol{\mu}_h, \boldsymbol{\Sigma}_h), \qquad \mathbf{x} \in \Re^n, \tag{4}$$

such that $\sum_h w_h = 1$, and where $p_h(\cdot|\boldsymbol{\mu}_h, \boldsymbol{\Sigma}_h)$ corresponds to the Gaussian distribution with mean $\boldsymbol{\mu}_h$ and covariance matrix $\boldsymbol{\Sigma}_h$,

$$p_h(\mathbf{x}|\boldsymbol{\mu}_h, \boldsymbol{\Sigma}_h) = \frac{1}{\sqrt{(2\pi)^n \, |\boldsymbol{\Sigma}_h|}} \, \exp\left\{-\frac{1}{2} \left(\mathbf{x} - \boldsymbol{\mu}_h\right)^t \boldsymbol{\Sigma}_h^{-1} \left(\mathbf{x} - \boldsymbol{\mu}_h\right)\right\}. \tag{5}$$

The number of Gaussian components, k, is determined by the complexity of $p(\cdot)$ and the desired accuracy of the approximation. The distribution in Eq. (4) is called a GMM. In general, the parameters of the model: means, covariance matrices, and membership probabilities, are unknown but can be found using the EM algorithm (Dempster et al. 1977).

Useful properties for GMMs appear, arising from those of Gaussian distributions. In particular, given a GMM with k components for $(\mathbf{x}, \mathbf{y})$, the conditional distribution of $\mathbf{x}$ given $\mathbf{y}$ corresponds to a GMM with the same number of components. In effect, we see that

$$\begin{aligned} p(\mathbf{x}|\mathbf{y}) \propto p(\mathbf{x}, \mathbf{y}) &= \sum_{h=1}^{k} w_h \, p_h(\mathbf{x}, \mathbf{y}) \\ &= \sum_{h=1}^{k} w_h \, p_h(\mathbf{y}) \, p_h(\mathbf{x}|\mathbf{y}). \end{aligned}$$

Given the properties of Gaussian distributions, the conditional distributions

$p_h(\mathbf{x}|\mathbf{y})$ are also Gaussian, and thus we obtain that

$$p(\mathbf{x}|\mathbf{y}) \quad \propto \quad \sum_{h=1}^{k} w_h^* \, p_h(\mathbf{x}|\mathbf{y}), \qquad (6)$$

which is the form of a GMM with k components. Expressions for the conditional means and covariance matrices involved in Eq. (6) are available in closed form (Anderson and Moore 1979). We will use the expression in (6) to efficiently increase the search space of BNAD.

3 Previous work

This section discusses relevant previous work. First, Section 3.1 discusses previous work on the problem of detecting anomalous records in databases. The discussion is mainly focused on works in the Artificial Intelligence (AI) domain, in particular, the Machine Learning community. Then, Section 3.2 describes relevant works related to learning the structure of a BN from data.

3.1 Anomaly Detection

In the AI community, there have been several attempts to tackle the problem of detecting anomalies in databases (Anderson et al. 1995) (Lewis 1993). Most of these approaches are built upon either knowledge based systems, in particular expert systems (Jackson 1998), or case based reasoning techniques (Aamodt and Plaza 1994). The expert system approach encodes knowledge about possible anomalies as if-then rules. The case based reasoning approach bases the detection on the estimated distances to a set of known anomalies. Although the two approaches have shown success when applied to small problems, in the case of large scale problems, the huge amount of rules and domain knowledge required to implement suitable solutions, make these approaches too complex to be implemented.

The Machine Learning and the related Knowledge Discovery in Databases (KDD) communities have also tackled the problem, motivated mainly by applications on fraud detection, see (Kou et al. 2004) and (Hodge and Austin 2004) for reviews. Most of these applications are based on supervised learning techniques, such as supervised neural networks and decision trees. Their need for labeled data, however, limits the convenience of these approaches when dealing with complex large dimensional domains.

Unsupervised learning techniques, such as our approach, have also been used, clustering techniques being the favorite tool. Using clustering, anomalies are detected as small clusters or isolated points located in low density regions of the features space (Kou et al. 2004). Although there have been efforts to scale clustering approaches to large databases (Bradley et al. 1998) (Moore 1999) the main problem arises with dimensionality. In a large dimensional space, many unknown relations or patterns may be hidden in arbitrary subspaces, making it difficult to find suitable metrics to find the clusters. While it is possible to use dimensionality reduction techniques, such as Principal Components Analysis (Morrison 2004), the problem of finding the relevant subspaces in large dimensional datasets is still a dilemma for traditional clustering techniques, see (Parsons et al. 2004) for a recent review.

The last observation highlights one of the main strength of our approach, which is given by the factorization of the joint pdf provided by the BN. If we consider a BN from a clustering point of view, the factorization of the joint pdf provided by the BN can be understood as model fitting in selective dimensions or subspaces. In effect, each factor in the joint pdf is given by a local conditional pdf over a subset of variables. These subsets of variables correspond to relevant subspaces of the feature space. In each of these subspaces, we use GMMs to fit the main clusters and to find anomalies by identifying points with low likelihood. This selective model fitting is the main mechanism how, by focusing on key reduced subspaces, our approach avoids the Curse of Dimensionality problem (Mitchell 1997). Furthermore, by using this mechanism, our approach does not operate as a black box, but it provides a generative probabilistic model. This model helps to explain the sources of an anomaly by identifying the factors that account for a low likelihood.

3.2 Bayesian Network Structure Learning

The problem of learning the structure of a BN from data has been intensively investigated (Cooper and Herskovits 1992) (Chickering et al. 1995) (Friedman et al. 1999) (Spirtes et al. 2000) (Friedman and Koller 2003) (Moore and Wong 2003) (Teyssier and Koller 2005). In general, these works can be classified into two main approaches: the constraint based approach and the scoring based approach. The constraint based learning approach uses a statistical test to find dependency relations among the variables, which are then mapped to the network structure (Spirtes et al. 2000). The scoring based learning approach considers the search as an optimization problem, where the exploration of the space of network structures is guided by a statistically motivated metric that quantifies how each network models the data (Cooper and Herskovits 1992). In practice, due to the complexities associated to finding and applying a robust test to detect dependencies, the scoring based approach is currently the most used option and it is the approach we follow in this work.

Under the scoring based approach, the basic idea is to search for network structures using an adequate heuristic, and to assign a score that penalizes model complexity to each structure. These scores include penalized log-likelihood, such as Bayesian scores (Cooper and Herskovits 1992), and Information Theory based scores (Friedman et al. 1999).

A commonly used scoring function is the Bayesian Information Criterion (BIC) (Hoeting et al. 1999). BIC evaluates the fit of a BN to the data through the likelihood function, penalizing for additional model complexity. In this work we use the BIC scoring function. This score has a suitable connection to the probability that a given BN is the true model underlying the data. Furthermore, as we will discuss in section 4.1, it can be decomposed to facilitate its calculation.

As we noted before, we base our approach in the Sparse Candidate Algorithm (SCA) (Friedman et al. 1999). SCA is an iterative algorithm that uses statistically motivated scores and a current estimate of the BN, to restrict the possible parent set of each variable. By restricting the number of parents, the algorithm avoids examining candidates that are extremely unreasonable. At each iteration, the restricting phase is followed by a maximization step, where GHC is used to greedily keep the best arc addition, within the set of candidate parents.

In our work, we use a restricting phase, similar to SCA. We use, however, the properties of GMMs to work with continuous variables and to increase the search space of network structures without adding a significant computational load. Furthermore, we use special implementations of Discrepancy and BIC scoring that allow us to cache computations in order to scale the approach to large and high dimensional databases.

Another state-of-the-art algorithm to find network structures is the Optimal Reinsertion Algorithm (ORA) (Moore and Wong 2003). ORA succeeds on speeding up BN structure learning, while achieving higher scoring networks than GHC. Using a pre-computed cache, found by using AD-search (Anderson and Moore 1998), ORA avoids high computational costs required by adding, deleting, or inverting edges. Unfortunately, AD-search cannot be used with continuous attributes, as is our interest. Without the benefits of AD-search, ORA would require a high computational complexity to fit models to local joint pdfs, making it hardly useful to be applied to large databases.

Teyssier and Koller propose the Ordering-Based Search (OBS) algorithm (Teyssier and Koller 2005). This algorithm is based on searching over the space of orderings of the variables in the network, rather than over the standard space of network structures. A similar idea was used by Larrañaga et al. (Larrañaga et al. 1996), who applies genetic algorithms to conduct the search. Teyssier and Koller claim that the OBS algorithm reaches similar network scoring than the ORA, but with a better computational performance. Although, OBS seems to be a good alternative, the main disadvantage to scale the approach to large datasets is the calculation of the scores for the potential successors of the initial ordering, which requires the calculation of a large set of sufficient statistics.

4 Bayes Network Anomaly Detector (BNAD)

This section describes the main steps followed by BNAD to find a suitable BN to model the data. The algorithm works in an unsupervised way with unlabeled data and assuming that there is not missing data. The main goal is to find a BN able to detect potential anomalous records in large databases. We achieve this goal by integrating and developing several AI and data processing technologies. Specifically, by using appropriate data structures, advanced caching techniques, BNs efficiency to model joint pdfs, and the properties of GMMs, our algorithm manages to learn a suitable BN from a large dataset. Section 4.1 describes the basic search strategy used by BNAD, which is mainly based on SCA. Section 4.2 explains how we use properties from GMMs to increase the scope of the search for network structures.

4.1 Basic Search Strategy

Traditionally, scoring based learning algorithms use GHC to explore the space of networks structures. In this work, as in the case of SCA, we efficiently shrink the search space by statistically selecting the most probable parents for each variable. Furthermore, our algorithm searches the space using an accelerated implementation of EM to train GMMs (Soto et al. 2007).

Let $\hat{B} = \langle \hat{G}, \hat{\theta} \rangle$ be the initial estimation of the BN that models the data. In this initial estimation all the variables are considered independent, i.e., there are no edges in $\hat{G}$. Following (Friedman et al. 1999), BNAD searches for new network structures by iterating two basic steps: Restrict and Maximize.

- Restrict:

  For each $x_i \in \mathbf{x}$, compute the quantity $Disc(x_i, x_j)$, where x_j is a candidate parent for x_i in $\hat{G}$, i.e., $x_j \in \left\{ \mathbf{x} \setminus \{ Pa^{\hat{G}}(x_i) \cup x_i \cup De^{\hat{G}}(x_i) \} \right\}$. Afterwards, form the set $\mathbf{d}$ with the variables x_j with the highest D discrepancies.

To compute $Disc(x_i, x_j)$, we estimate $p(x_i, x_j)$ by the empirical distribution and $p_{\hat{B}}(x_i, x_j)$ by sampling a fixed number of observations from $\hat{B}$, adding the assumption that x_j is a parent of x_i. In other words, we estimate $p_B(x_i, x_j)$ by sampling x_i from $p(x_i|Pa^{\hat{G}}(x_i), x_j)$. In our experiments we use a set of 5000 samples from $p_{\hat{B}}$ to estimate $p_B(x_i, x_j)$. We limit the maximal number of parents for each node to 5.

- Maximize:

  For each new candidate parent x_j in the set $\mathbf{d}$, independently compute $BIC(B')$, where B' is given by $\langle \hat{G} \cup \{x_j \to x_i\},\ \theta' \rangle$ and the parameters in θ' are equal to those in $\hat{\theta}$, except for those related to variable x_i, which now has an additional parent x_j. If for the highest scoring BN, B'_{max}, $|BIC(B'_{max}) - BIC(\hat{B})| \leq \delta$, the algorithm stops, where δ is a user defined parameter of convergence. Otherwise, B'_{max} is used as $\hat{B}$ in the next iteration.

This algorithm requires $O(mn^2)$ operations to collect the required statistics, where m is the number of records in the database. It is important, however, to notice that a single pass over the m records is sufficient to compute every pairwise frequency in constant time. In order to efficiently compute Discrepancy, we discretize the observations in t bins. Equally sized bins (Scott 1992) are used in the discretization as an approximation to the real values of Discrepancy.

As we mentioned, BNAD uses BIC as the scoring function. Although other model selection scoring can be used, the key issue is to choose a scoring metric that can be decomposed according to the BN factorization of the joint pdf. In this way, we can reuse most partial computations. In the case of BIC, it is possible to show that:

$$BIC(B) = \sum_{i}^{n} BIC_c(x_i), \qquad (7)$$

where $BIC_c(x_i)$ corresponds to the BIC score of node V_i, conditional on its parents $Pa^G(x_i)$. Using this additive decomposition, we only need to compute BICs for the nodes whose parents have changed, which in BNAD occurs one at a time. The important observation is that all partial log-likelihoods used in the calculation of BIC, can be cached for latter use, avoiding $O(am)$ operations for each reused calculation, where a is the number of attributes needed to compute the partial log-likelihood.

BNAD detects candidate strange objects by evaluating the likelihood of each object and selecting the worst $\alpha\%$ of them. Deciding the correct value of α depends directly on the capacity of the BN to fit the data. If the training of the BN is successful, most objects will be accurately modeled, and only relevant anomalies will be displayed as low probability objects. It is also possible to find objects with very low probability but of no interest to the user, due to noise and other external factors.

4.2 Extended Search Strategy

When searching for network structures, the estimation of the local conditional pdfs is the bottleneck that limits a further search. As an example, when we test our approach running a regular implementation of EM to find a BN that models the database described in Section 5.2.2, it takes 5 days to find a suitable structure. In contrast, when we use our accelerated version of EM, it takes just 232 minutes. Although the accelerated version of EM helps to reduce the computational burden, our experiments indicate that the running time of this algorithm consumes approximately 90% of the total processing time. In this section we show how we are able to increase the scope of the search for network structures while avoiding extra expensive calls to EM.

In our search for network structures, we use our version of EM to estimate the joint pdf that relates each variable x_i with its candidate set of parents $p(x_i, Pa^{\hat{G}}(x_i))$. After estimating this joint density, we use the property in Eq. (6) to obtain the relevant conditional pdf for the network under consideration, i.e., $p(x_i|Pa^{\hat{G}}(x_i))$. As we indicated in Section 2.3, the interesting fact is that after estimating $p(x_i, Pa^{\hat{G}}(x_i))$, we can use properties from GMMs not only to estimate $p(x_i|Pa^{\hat{G}}(x_i))$ but also to find any marginal or conditional distributions among these variables by simply applying basic matrix operations, such as, swapping or deleting columns and rows.

As an example of the previous property, suppose that the Restrict phase of BNAD requires the estimation of $p(x_1|x_2, x_3, x_4)$. To achieve this, BNAD calls first the accelerated version of EM to estimate $p(x_1, x_2, x_3, x_4)$. After estimating this joint density, we can compute any conditional or marginal, such as $p(x_2|x_1, x_3, x_4)$, without the computational cost of a new EM call, but just using basic matrix operations. In this case, the calculation of $p(x_2|x_1, x_3, x_4)$ can be seen as an edge inversion process that allow us to explore a structure where x_2 is a parent of x_1. BNAD takes advantage of such properties to explore additional relations in the space of network structures without adding a significant computational load.

When we use properties of GMMs to expand the search, we need to take some considerations into account. First, we need to test only new edges that do not produce cycles in the network. Also, if a node has a set of pre-existing parents, these edges need to be removed in order to avoid an additional call to EM. Finally, besides the calculation of the new conditional pdf, there is an additional computation due to the calculation of the BIC scores for the new structure, which requires at least $O(am)$ operations, as it requires a new partial log-likelihood calculation.

To expand the search space of BNAD using the properties mentioned before, we need to modify the basic search scheme presented in Section 4.1. The Restrict phase is the same as before. At each iteration it provides the set $\mathbf{d}$ of best D candidates parents according to Discrepancy. In the case of the Maximize phase, it needs to be modified in order to evaluate the BIC scores for the new network structures that expand the search. These new network structures result from calculating marginal and conditional densities from the joint pdfs obtained for the D candidate parents found in the Restrict phase. In particular, if the Restrict phase specifies a candidate parent x_j for x_i, then, besides exploring the regular BN with the local relation $p(x_i|x_j, Pa^{\hat{G}}(x_i))$, we also calculate the BIC score for the related structures that satisfy the local relation $p(x_j|x_i, Pa^{\hat{G}}(x_i))$ or the local relations $p(x_k|x_i, x_j, Pa^{\hat{G}}(x_i)\backslash(x_k))$, where $x_k \in Pa^{\hat{G}}(x_i)\backslash(x_k)$. It is important to note that in this process we only consider new structures that do not contain cycles and do not require extra calls to EM. According to this, the new Maximize phase is given by:

- Maximize:

  For each new candidate parent x_j in the set $\mathbf{d}$ obtained in the Restrict phase, compute the BIC score for the following BNs:

- B' given by $\langle \hat{G} \cup \{x_j \rightarrow x_i\},\ \theta' \rangle$. To obtain θ', update in $\hat{\theta}$ the conditional local relation for x_i by computing $p(x_i|x_j, Pa^{\hat{G}}(x_i))$. Afterwards, compute $BIC(B')$.

- B' given by $\langle G' \cup \{x_i \rightarrow x_j\} \cup \{Pa^{\hat{G}}(x_i) \rightarrow x_j\},\ \theta' \rangle$. To obtain G', remove existing parents of x_j in $\hat{G}$. Then, add $\{x_i \cup Pa^{\hat{G}}(x_i)\}$ as new parents of x_j. If the resulting network B' is cyclic rollback. Otherwise obtain θ' by updating in $\hat{\theta}$ the conditional local relation for x_j. To do this, use properties of GMM to calculate $p(x_j|x_i, Pa^{\hat{G}}(x_i))$ from $p(x_i, x_j, Pa^{\hat{G}}(x_i))$. Afterwards, compute $BIC(B')$.

- B' given by $\langle G' \cup \{x_i \rightarrow x_k\} \cup \{Pa^{\hat{G}}(x_i) \backslash (x_k) \rightarrow x_k\} \cup \{x_j \rightarrow x_k\},\ \theta' \rangle$. To obtain G', remove existing parents of x_k in $\hat{G}$. Then, add $\{x_i \cup x_j \cup Pa^{\hat{G}}(x_i) \backslash (x_k)\}$ as new parents of x_k. If the resulting network B' is cyclic rollback. Otherwise obtain θ' by updating in $\hat{\theta}$ the conditional local relation for x_k. To do this, use properties of GMM to calculate $p(x_k|x_i, x_j, Pa^{\hat{G}}(x_i) \backslash (x_k))$ from $p(x_i, x_j, Pa^{\hat{G}}(x_i))$. Afterwards, compute $BIC(B')$.

  As before, keep just the highest scoring BN, B'_{max}. If $|BIC(B'_{max}) - BIC(\hat{B})| \leq \delta$, the algorithm stops, otherwise, B'_{max} is used as $\hat{B}$ in the next iteration.

This algorithm can be seen as a particular edge reversing strategy using GMM properties. Due to the new calculations, mainly the new BIC scores, the computational cost will be slightly higher than the regular search strategy of SCA, however, the search space becomes wider without the need of new expensive EM calls. This may help to skip from local maximums of the scoring function.

5 Experimental Results

In this section we use synthetic and real databases to illustrate the main features of our approach. First, we use synthetic databases to perform 3 experiments oriented to demonstrate the abilities of BNAD to achieve each of the 3 main goals pursued by this work: 1)Effective detection of potential anomalous records, 2)Selection of attributes that explains the source of an anomaly, and 3)Efficient implementation to scale to large databases. Afterwards, we test the abilities of BNAD to detect anomalous records in 2 real databases.

In all the experiments, we use the accelerated version of EM. We keep all cache of EM calculations and BIC functions in disk for computational efficiency. To compute the Discrepancy metric score, we notice that using less than 5 bins tends to increase the error, while using more than 15 bins does not provide additional benefits, thus, we choose to work with $t = 10$ bins. Also, we use 5000 samples in the calculation of Discrepancy since we notice that a greater number of samples does not provide major changes in the final network structure. Furthermore, the memory needed to create frequency tables seems to be quite irrelevant, being approximately 16MB for a configuration with 200 variables and 10 bins each, and 256MB for 400 variables and 20 bins. All the experiments were conducted using a Pentium D 3.2GHz, except Section 5.1.2 and Section 5.2.2 that use an Athlon 3200+ CPU.

5.1 Synthetic databases

We create synthetic databases using the following strategy. Given the number of components k of a GMM and a set of n variables x_i, $i \in \{1, \ldots, n\}$:

- For $h = 1, \ldots, k$, create a random vector $\boldsymbol{\mu}_h$ with the mean value of each GMM component.

- For $h = 1, \ldots, k$, create an orthogonal base of n vectors which represents the covariance matrix $\boldsymbol{\Sigma}_h$ of each GMM component. This matrix is positive definite by construction.

- Create a random membership vector $\boldsymbol{w}$, such that, $\sum_{h=1}^{k} w_h = 1$.

- Let G be an empty graph. For each node $x_i \in \mathbf{x}$, select a random number l of parent nodes x_j, such that, $0 \leq l < i - 1$ and $1 \leq j < i - 1$. Add all the resulting relations $x_j \rightarrow x_i$ to G.

- For each node $x_i \in \mathbf{x}$ create a conditional GMM, $p(x_i|Pa^G(x_i))$, that satisfies the restrictions imposed by G and the joint pdf given by $\boldsymbol{\Sigma}_h$, $\boldsymbol{\mu}_h$, and w_h, $1 \leq h \leq k$.

This algorithm creates an acyclic BN with consistent conditional probabilities. No cycles are possible due to the restriction on edge order. Consistency is guaranteed by taking all conditional pdfs from partitions of the same joint pdf modeled through a GMM with k components.

5.1.1 Experiment 1: Detection of anomalous records

We first evaluate the ability of BNAD to detect anomalies and we also compare its performance respect to GHC. For this purpose, we generate $m = 10,000$ observations of a BN with $n = 20$ nodes, and $k = 10$ Gaussian components in the GMMs. We artificially add to this database anomalous records consisting of a valid instance of the database, for which c attributes have been modified with random values taken within the range of the variable. We just accept as anomalies, records that are located in areas of low density under the generating model. For this experiment, we test two cases. In both we artificially introduce 10 anomalies in the database. In one case we modify $c = 4$ variables and in the other we modify $c = 8$ variables.

The results are shown in Table 1. As it can be seen, the models obtained by BNAD and GHC present similar BIC scores. These scores are also similar to the score obtained for the true underlying model. In order to evaluate the quality of the classifications obtained by both algorithms, we use the concepts of sensitivity and specificity, which correspond to the percentage of well classified anomalies and well classified regular observations, respectively. We set the sensitivity to 90%, and we define the cut-off point as the minimum number of observations we need to explore, in increasing order of likelihood to attain that sensitivity. For both algorithms, this strategy determines a specificity of 90%.

The cut-off points in Table 1 show that both algorithms are able to rank the 10 anomalies among the group of lowest likelihood records. In both cases, we need to explore similar number of observations to detect 90% of the anomalies. As expected, the number of observations we need to explore to detect the anomalies decreases considerably when anomalies become more evident, as is the case where we modify 8 of the attributes instead of 4. In this case, 90% of the anomalies are properly detected with less than 1% of false positives.

The big advantage of BNAD over GHC is in terms of efficiency. In effect, BNAD was able to reduce by 33% the number of EM calls made by GHC. It is important to notice that classification based on the true model attained similar results. This implies that errors in classification using GHC and BNAD are mainly due to the natural variability of the data.

Section 5.1.3 shows further results that compare the performance of BNAD with respect to GHC for a database with $m = 20,000$ observations of a BN with $n = 50$ nodes. In particular, Figure 3 shows the complete ROC curve for this case.

5.1.2 Experiment 2: Detection of the sources of an anomaly

The previous analysis bases the detection of a candidate anomaly on the evaluation of the complete log-likelihood under the estimated BN. As mentioned earlier, one of the major benefits of using a BN for anomaly detection corresponds to the factorization of the joint pdf of the attributes in the intended database. One of our hypothesis is that for each candidate anomaly, the information of the relative values of the log-likelihood associated to each factor provides important information about the sources of the anomaly.

To test the previous hypothesis, we conduct an experiment dedicated to explore which factors of the BN representation influence the most a low log-likelihood value for each anomaly. To perform this test, we construct a synthetic database consisting of $m = 10,000$ observations of a BN with $n = 20$ nodes, and $k = 10$ Gaussian components in the joint GMM. We randomly modify, afterwards, the values of each of the 20 attributes in the database, one at a time. Once attribute i has been modified, we measure the impact of the modification in three ways. We measure its *total* influence, which is the effect the modification has over the complete log-likelihood of the observations. We measure its *direct* influence, which is the effect the modification has over the partial log-likelihood of that node in particular. Finally, we measure its *indirect* influence, which is the effect the modification has over those partial log-likelihoods that include x_i

as a parent.

The results of the previous experiment are shown in Figure 1. The X axis shows the attribute that was modified each time. The Y axis shows the difference in log-likelihood between original and modified values divided by the original value. The figure shows that, a small effect on the overall log-likelihood becomes larger when we consider only those partial log-likelihoods that include the source of the anomaly. We can appreciate that *direct* influence is notable and could be easily detected by statistics. Less important but still notable is *indirect* influence. As for *total* log-likelihoods, although we can appreciate differences, there are neither direct nor indirect strong influences. As a consequence, if we only use total values, we would hardly differentiate between anomalous and regular records. In contrast, by using the information in the relevant factors, it is possible to increase the sensitivity of the detector and also to identify the causes of an anomaly. Therefore, BN factorization can be very effective in anomaly detection and its importance has been ignored in previous works.

5.1.3 Experiment 3: Scaling to a large database

In order to test the ability of BNAD to scale to a large database, we generate a synthetic database consisting of 500,000 records and 50 dimensions. In this case we obtain the samples from a BN with local conditional probabilities modeled as a GMM with 10 components. As in the previous cases, we insert in this database a set of 500 artificial anomalies for which $c = 1$ attribute has been modified with random values taken within the range of the variable.

We use subsets of the previous database to test how BNAD scales with respect to the number of records. In particular, we run BNAD using subsets consisting on 20,000, 80,000, 160,000 and 500,000 records, all of them with 50 dimensions. The results are shown in Figure 2. This Figure shows that due to the EM implementation used, BNAD scales almost linearly with respect to the number of records in the database.

To compare the processing time of BNAD with respect to GHC, we run GHC for the subset of 20,000 records and 50 dimensions (we avoid running GHC for the other datasets given that it takes more than a week to converge). Figure 3 shows the resulting ROC curve for BNAD and GHC for this case. In terms of accuracy in the detection of the anomalies both algorithms show a very similar performance. Nonetheless, the main difference relies in the computational load: while GHC takes 64 hours to find a suitable BN, BNAD takes just 12 hours to find a BN with a similar performance. These processing times are strongly related to the number of EM calls performed by each algorithm, precisely 5203 for GHC and 929 for BNAD.

5.2 Real databases

5.2.1 Flaw detection in Metallic Pieces

We test our algorithm in a flaw detection application using a database containing information extracted from X-ray images of regular and faulty metallic pieces (Mery et al. 2003). The database consists of 28 variables and 22,936 observations, where each record contains information about visual features of a specific region of each metallic piece. The database was previously labeled by a human expert, who added to each record a binary attribute indicating if the record corresponds to a regular or a faulty piece. The total number of faulty records in the database was 60. Given that our approach corresponds to an unsupervised method, we use the classification labels just as ground truth to evaluate the ability of our algorithm to detect the true anomalies.

Again we compare the performance of BNAD with respect to GHC. Figure 4 shows the ROC curves for both algorithms. In this case, BNAD outperformed GHC: it ranked all 60 flaws within the 1079 elements with lowest likelihood whereas GHC needed 1684 to collect all flaws. This shows the advantage of using a wider search space that to some extent avoids local optimums. Another important advantage of BNAD is again provided by the computational load. Concretely, BNAD performed 250 calls to EM whereas GHC performed 991, which correspond to processing times of 15 and 119 minutes respectively.

It is interesting to mention that these results are comparable to the performance of the supervised classifier documented in (Mery et al. 2003). In that work, a neural network classifier trained with the same database, was able to achieve a classification accuracy of 95%. It needed, however, to be provided with true labeled data during training, being this a relevant disadvantage with respect to our method that operates in a total unsupervised mode.

One important note is that we did not design BNAD to work as a flaw detection algorithm, only as an anomaly or outlier detector, however, it was still useful as a filter for such applications. The main reason is that only few objects were indeed flaws, therefore it was possible to consider them as anomalies. We think that using BNAD as a filter of candidates flaws may simplify the labeling process to train a supervised classifier.

5.2.2 Strange objects detection in Astronomy

We also test our algorithm using an astronomical database from the Canadian-France-Hawaii Telescope (CFHT). The database consists of 79 variables and 104386 objects, corresponding to information related to color and shape of galaxies. Following the advice of the astronomers that provided the database, we only consider 15 attributes. The idea is to find unusual shapes and colors which could describe interesting new galaxies.

Given that in this case we do not have ground truth data about real strange objects in the database, we artificially insert anomalous records using a similar strategy as described in Section 5.1.1. In this case, we modify 4 variables of each of the 104 anomalies inserted in the database. After modeling, we find that 80.7% of the anomalies artificially inserted appear among the 1% of the objects with lowest likelihood, corresponding to a rate of 0.97% false positives. It is important to note that these false positives might be real strange objects within the database.

In the case of the real anomalies, we lack of the domain knowledge to quantify the relevance of the candidate anomalies detected by BNAD, however, we sent a list of the 1000 records with lowest likelihood to experts of the Canada-France Legacy Project, who provided the database. As a feedback, the experts confirmed us that there were indeed a great number of interesting objects to them among the 100 records with lowest likelihood.

6 Conclusions

In this paper, we presented a new algorithm for the detection of candidate anomalies in large databases, composed of thousands of records and high dimensional datasets with floating-point domains.

The representational power of Gaussian Mixture Models, together with an optimized version of the Expectation Maximization algorithm, and caching strategies throughout the whole implementation, provided an efficient algorithm for Bayesian Network structure learning. The results of our proposed algorithm – Bayesian Network Anomaly Detector (BNAD) – on synthetic and real databases indicate that it is possible to use a simple, yet powerful, model for estimating the joint probability density function (pdf) of the domain variables. We showed that this joint pdf can be used to effectively detect anomalies as low likelihood elements. In fact, in the real case of flaw detection in metallic pieces, all flaws were found within the least probable (lowest likelihood) 4.7% elements. Moreover, as likelihood evaluation is fast, once the BN is trained, it can be used as a real time filter of candidate anomalies which extends its use in real-life applications.

Following the results from (Friedman et al. 1999), we use a similar scoring metric that favors relevant areas of the search space by pruning BNs with unlikely related variables. In this paper, we showed that these inherited results not only apply to discrete data, but also to data with continuous domains. Furthermore, BNAD was able to achieve similar results than those of Greedy Hill Climbing using less computational cost.

We also showed that the joint probability factorization provided by the BN can help in the detection of rare objects. It profits from the conditional probability of each variable given its parents to signal which are most distinguishing attributes for a given record. This constitutes a major advantage of BN with respect to other “black-box” alternatives. We believe that further research in this field could point-out even more impressive results.

In terms of computational complexity, by including an accelerated version of EM in combination with caching techniques, BNAD shows attracting results in scalability, being quasi-linear in the number of elements. In this way, BNAD is able to reduce processing time from several days to hours, making feasible the processing of larger datasets. Furthermore, BNAD also increases the search space for BN structures without the burden of additional EM calls by using GMM properties that provide marginal and conditional distributions by simply applying basic matrix operations.

As future work, we are currently exploring the use of active learning techniques to add semantic feedback from an expert to efficiently search the set of candidate anomalies provided by BNAD. We are also exploring the use of sub-space clustering techniques to directly find relevant sub-spaces to search for anomalous records.

Acknowledgments

We would like to thank astronomers Felipe Barrientos and David Schade for their contribution to the results presented in section 5.2.2. This work was partially funded by FONDECYT grant 1030336.

References

Aamodt, A. and E. Plaza (1994). Case-based reasoning: Foundational issues, methodological variations, and system approaches. *Artificial Intelligence Communications 7*(1), 39–59.

Anderson, B. and A. Moore (1998). AD-trees for fast counting and for fast learning of association rules. In *Proceedings of 4th International Conference on Knowledge Discovery and Data Mining*, pp. 134–138.

Anderson, B. D. and J. B. Moore (1979). *Optimal Filtering.* Prentice Hall.

Anderson, D., T. Frivold, and A. Valdes (1995). Next-generation intrusion detection expert system (NIDES): A summary. Technical Report SRI–CSL–95–07, Computer Science Laboratory, SRI International.

Bradley, P. S., U. Fayyad, and C. Reina (1998). Scaling EM (Expectation Maximization) clustering to large databases. Technical Report MSR–TR–98–35, Microsoft Research.

Chickering, D. M. (1996a). Learning Bayesian networks is NP-complete. In D. H. Fisher and H.-J. Lenz (Eds.), *Learning from Data: Artificial Intelligence and Statistics V*, pp. 121–130. Springer-Verlag.

Chickering, D. M. (1996b). Learning equivalence classes of Bayesian network structures. In *Proceedings of 12th Conference on Uncertainty in Artificial Intelligence*, pp. 150–157.

Chickering, D. M., D. Geiger, and D. Heckerman (1995). Learning Bayesian networks: Search methods and experimental results. In *Proceedings of 5th Conference on Artificial Intelligence and Statistics*, pp. 112–128.

Cooper, G. F. and E. Herskovits (1992). A Bayesian method for the induction of probabilistic networks from data. *Machine Learning 9*(4), 309–347.

Cover, T. M. and J. A. Thomas (1991). *Elements of Information Theory.* John

Wiley and Sons, Inc.

Dempster, A., N. Laird, and D. Rubin (1977). Maximum likelihood from incomplete data via the EM algorithm (with discussion). *Journal of the Royal Statistical Society, Series B 39*(1), 1–38.

Friedman, N. and D. Koller (2003). Being Bayesian about network structure: A Bayesian approach to structure discovery in Bayesian networks. *Machine Learning 50*(1–2), 95–125.

Friedman, N., I. Nachman, and D. Peér (1999). Learning Bayesian network structure from massive datasets: The Sparse Candidate algorithm. In *Proceedings of 15th Conference on Uncertainty in Artificial Intelligence*, pp. 206–215.

Heckerman, D. (1996). Bayesian Networks for knowledge discovery. In U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy (Eds.), *Advances in Knowledge Discovery and Data Mining*, pp. 273–305. MIT Press.

Hodge, V. J. and J. Austin (2004). A survey of outlier detection methodologies. *Artificial Intelligence Review 22*(2), 85–126.

Hoeting, J. A., D. Madigan, A. E. Raftery, and C. T. Volinsky (1999). Bayesian model averaging: A tutorial. *Statistical Science 14*(4), 382–417.

Jackson, P. (1998). *Introduction to Expert Systems.* Addison Wesley.

Jensen, F. V. (2001). *Bayesian Networks and Decision Graphs.* Springer-Verlag.

Kou, Y., C.-T. Lu, S. Sirwongwattana, and Y.-P. Huang (2004). Survey of fraud detection techniques. In *Proceedings of the 2004 IEEE International Conference on Networking, Sensing and Control*, pp. 749–754.

Kullback, S. and R. A. Leibler (1951). On information and sufficiency. *Annals of Mathematical Statistics 22*(1), 79–86.

Larrañaga, P., C. M. Kuijpers, R. H. Murga, and Y. Yurramendi (1996). Learning Bayesian network structures by searching for the best ordering with genetic algorithms. *IEEE Transactions on Systems, Man and Cybernetics 26*(4), 487–

493.

Lauritzen, S. L. (1996). *Graphical Models*. Oxford: Clarendon Press.

Lewis, L. M. (1993). A case based reasoning approach to the management of faults in communication networks. In *Proceedings of 12th Annual Joint Conference of the IEEE Computer and Communications Societies. INFOCOM 1993*, pp. 1422–1429.

Mery, D., R. R. da Silva, L. P. Caloba, and J. M. Rebello (2003). Pattern recognition in the automatic inspection of aluminium castings. *Insight 45*(7), 431–439.

Mitchell, T. (1997). *Machine Learning*. McGraw Hill.

Moore, A. (1999). Very fast EM-based mixture model clustering using multiresolution KD-trees. In *Proceedings of the 11th Conference on Advances in Neural Information Processing Systems, NIPS*, pp. 543–549.

Moore, A. and W.-K. Wong (2003). Optimal reinsertion: A new search operator for accelerated and more accurate Bayesian network structure learning. In *Proceedings of 20th International Conference on Machine Learning, ICML*, pp. 552–559.

Morrison, D. F. (2004). *Multivariate Statistical Methods*. Duxbury Advanced Series.

Neapolitan, R. E. (2004). *Learning Bayesian Networks*. Prentice Hall.

Parsons, L., E. Haque, and H. Liu (2004). Subspace clustering for high dimensional data: A review. *ACM SIGKDD Explorations Newsletter 6*(1), 90–105.

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann.

Scott, D. W. (1992). *Multivariate Density Estimation: Theory, Practice, and Visualization*. John Wiley and Sons, Inc.

Silverman, B. W. (1986). *Density Estimation for Statistics and Data Analysis*. Chapman and Hall.

Soto, A., F. Zavala, and A. Araneda (2007). An accelerated algorithm for density estimation in large databases using Gaussian mixtures. *Cybernetics and Systems 38*(2), 123–139.

Spirtes, P., C. Glymour, and R. Scheines (2000). *Causation, Prediction, and Search.* The MIT Press.

Teyssier, M. and D. Koller (2005). Ordering-based search: A simple and effective algorithm for learning Bayesian networks. In *Proceedings of 21st Conference on Uncertainty in Artificial Intelligence*, pp. 584–590.

Zhang, T., R. Ramakrishnan, and M. Livny (1996). Birch: An efficient data clustering method for very large databases. In *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data*, pp. 103–114.

Algorithm	BIC	$c = 4$	$c = 8$	EM calls
		Cut-off points		
GHC	-425.075	933	38	850
BNAD	-422.317	942	25	572
True Model	-458.023	1008	98	-

Table 1: Performance of BNAD compared to GHC based on synthetic databases. Cut-off points are determined such that sensitivity equals 90%. Both methods attain specificity of 90%.

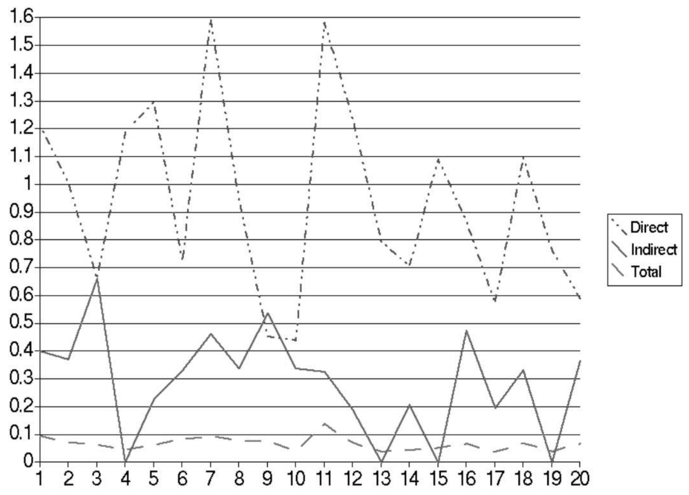

Figure 1: Effect of the presence of anomalies over Total and Partial Log-likelihoods on a set of 20 variable. The X axis shows the attribute that was modified. The Y axis shows the difference in log-likelihood between original and modified values divided by the original value.

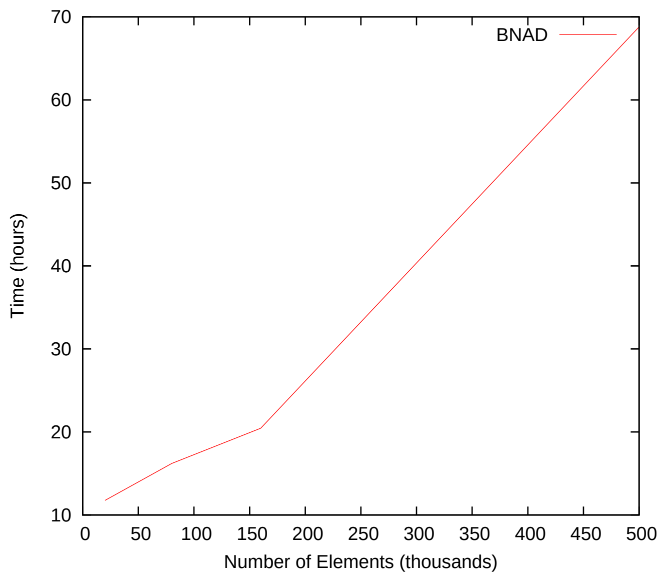

Figure 2: Scalability of BNAD with respect to the number of records for a databases with 50 dimensions.

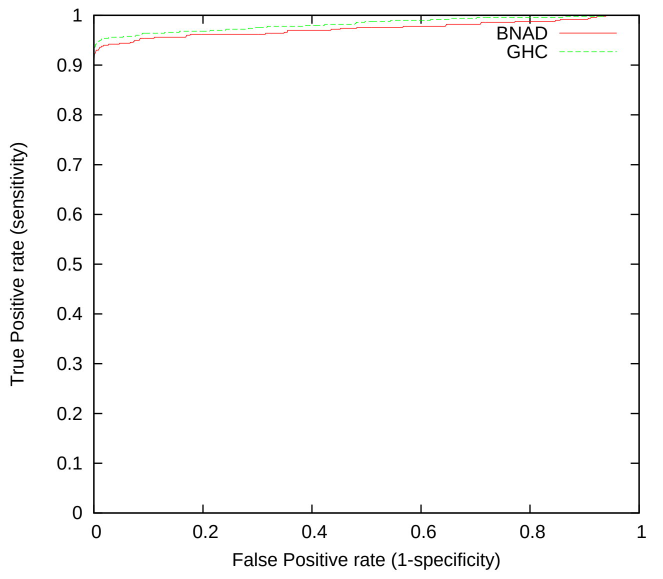

Figure 3: Performance of BNAD compared to GHC measured by a ROC curve in a database with 20,000 records and 50 variables.

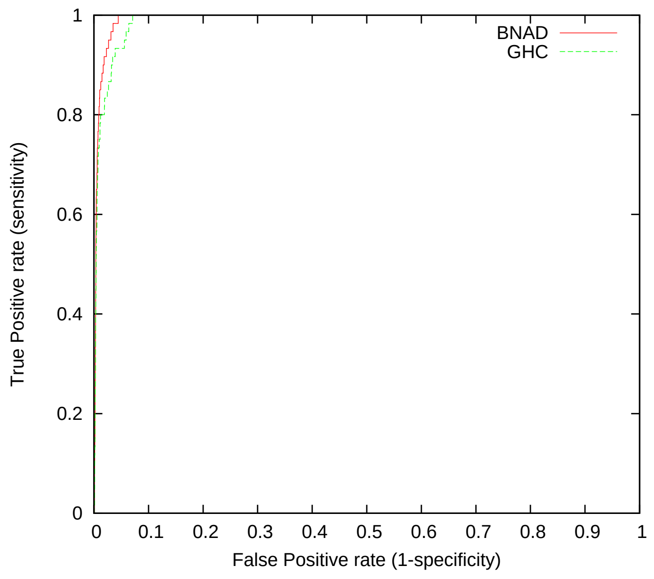

Figure 4: Performance of BNAD compared to GHC measured by a ROC curve in metallic pieces database.