A BAYESIAN RECURRENT NEURAL NETWORK FOR UNSUPERVISED PATTERN RECOGNITION IN LARGE INCOMPLETE DATA SETS

Roland Orre
Mathematical Statistics, Stockholm University, SE-106 91 Stockholm, Sweden
NeuroLogic Sweden AB, Johan Enbergs v. 28, SE-171 61 Solna, Sweden
E-mail: orre@math.su.se, roland.orre@neurologic.se

Andrew Bate
WHO Collaborating Centre for International Drug Monitoring,
Uppsala Monitoring Centre (UMC), SE-753 20 Uppsala, Sweden
Div of Clinical Pharmacology, SE-901 85 Umeå University, Sweden
E-mail: andrew.bate@who-umc.org

G. Niklas Norén
WHO Collaborating Centre for International Drug Monitoring,
Uppsala Monitoring Centre (UMC), SE-753 20 Uppsala, Sweden
Mathematical Statistics, Stockholm University, SE-106 91 Stockholm, Sweden

Erik Swahn
WHO Collaborating Centre for International Drug Monitoring,
Uppsala Monitoring Centre (UMC), SE-753 20 Uppsala, Sweden

Stefan Arnborg
NADA, Royal Institute of Technology, SE-100 44 Stockholm, Sweden

I. Ralph Edwards
WHO Collaborating Centre for International Drug Monitoring,
Uppsala Monitoring Centre (UMC), SE-753 20 Uppsala, Sweden

Abstract

A recurrent neural network, modified to handle highly incomplete training data is described. Unsupervised pattern recognition is demonstrated in the WHO database of adverse drug reactions. Comparison is made to a well established method, AutoClass, and the performances of both methods is investigated on simulated data. The neural network method performs comparably to AutoClass in simulated data, and better than AutoClass in real world data. With its better scaling properties, the neural network is a promising tool for unsupervised pattern recognition in huge databases of incomplete observations.

1. Introduction

With the increasing volumes of data at one's disposal, and with ever increasing computational power, data mining methods have become a useful approach in finding previously unsuspected trends in data. Such methods have been successfully applied to credit card fraud detection[1], market basket analysis[2], classifying documents into categories[3] and many other applications[4]. A data mining approach has the potential to find new patterns in the data which may never have been suspected to exist, in particular when manual inspection is not practical.

This article proposes a new approach to unsupervised pattern recognition. The aim of this method is to identify groups of attribute values related through some unknown, underlying concept. In particular we are interested in situations where the full concept may never have been observed and thus recognition must be based on partial (and often noisy) observations. The method accomplishes this by identifying sets of attribute values with overall high levels of pairwise interdependency.

In analyzing adverse drug reaction (ADR) databases, it is valuable to find and characterize syndromes and other large concepts. A syndrome is a cluster of symptoms that reliably co-occur and identify subtypes of patients who are homogeneous[5]. Thus, the individual ADR terms that make up a syndrome are not necessarily strongly associated with the drug causing the syndrome, but should have some association to the other symptoms in the syndrome. Previous work in this area has focused on searching the database for complete observations of groups of co-occurring attribute values[6,7] (*i.e.* cases where all the symptoms of interest are reported together). However for large concepts it is very rare that all attribute values occur in a single observation. For example, in the world's largest ADR database, there are well characterized syndromes where the full set of symptoms has never been reported on a single case report[8]. The identification of underlying concepts from incomplete observations would also be of use in other applications throughout the area of market basket analysis.

Hopfield networks are efficient models for computation that have previously been successful at pattern recognition. However, for unsupervised pattern recognition on the large and very sparse data sets that are studied in this work, the standard Hopfield network does not perform well: only the two trivial patterns where either all nodes are active or all nodes are inactive are found. This is due to the high degree of co-inactivity between the nodes in such data, which leads to positive weights between all possible nodes in the standard Hopfield network. In contrast, a variant of the Hopfield network referred to as the recurrent Bayesian Confidence Propagation Neural Network (BCPNN)[9] uses weights which are less sensitive to co-inactivity of nodes and thus has the potential for efficient pattern recognition even for large and sparse data sets.

A feedforward BCPNN has been in routine use since 1998 for the detection of ADR signals in the WHO database[10], and can also be used as a Bayes classifier[9]. The recurrent BCPNN has previously been used to recall known prototypes[11], but here we present methods for unsupervised pattern recognition including changes to the recall phase of the algorithm that allows for better handling of incomplete training data.

We demonstrate the effectiveness of the recurrent BCPNN for unsupervised pattern recognition by applying it to real data — the WHO database of ADR reports. To give further insight into the performance of the method, we show its use in finding patterns in a set of simulated data sets, designed as models of the WHO database, but with known properties. Additionally we compare the performance of the recurrent BCPNN with that of AutoClass[12]. There is no standard comparison method for this particular application, but AutoClass is a standard method for clustering[13], and examination of its output class profiles gives similar information.

2. The recurrent BCPNN

The variant of the recurrent BCPNN on which the algorithm in this article is based was described by Lansner and Ekeberg[11]. Specifically, the activation levels in the recurrent BCPNN are graded between 0 and 1, and the transfer function is a truncated exponential. The nodes do not have recurrent self connections and the weight matrix is by definition symmetric. This type of network works as an iterative fixed point solver for pattern association[11].

The weights in the BCPNN are called Information Components (IC)[6]. IC_{ij} is a measure of the degree of association between attribute values i and j, and is defined as:

$$IC_{ij} = \log_2 \frac{p_{ij}}{p_i p_j} \qquad (1)$$

where p_i is the marginal probability of attribute value i, p_j is the marginal probability of attribute value j and p_{ij} is the joint probability of i and j combined.

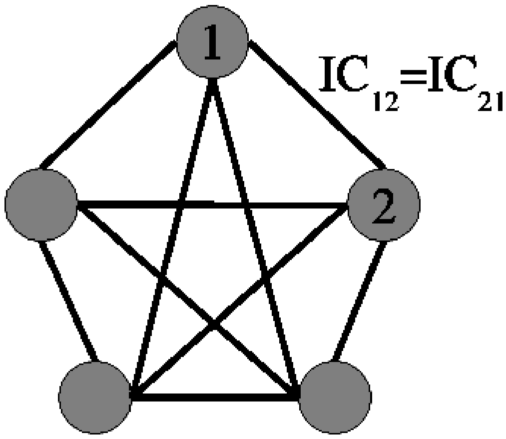

Figure 1: Schematic description of a simple recurrent BCPNN.

2.1. *Training*

The recurrent BCPNN has two modes of operation: training and recall. In training mode the marginal counts c_i and c_j (the number of occurrences of each attribute value), the joint counts c_{ij} (the number of co-occurrences of each pair of attribute values) and the total number of cases N are computed.

The weights for the network are then calculated based on (1) using maximum likelihood estimates for the probabilities p_i, p_j and p_{ij} if $c_{ij} > 0$. If $c_{ij} = 0$, the IC is (for computational stability) set to a large negative number (here to $\log_2 1/N$, so that the IC values for combinations with $c_{ij} = 0$ are equal to the smallest possible IC value for a combination with $c_{ij} > 0$):

$$IC_{ij} = \begin{cases} 0 & c_i = 0 \text{ and/or } c_j = 0 \\ log_2\ 1/N & c_i, c_j > 0 \text{ and } c_{ij} = 0 \\ log_2 \frac{c_{ij}N}{c_i c_j} & \text{otherwise} \end{cases}$$

The maximum number of patterns that can be stored in a one layer recurrent neural network for a given number of nodes (K) has been derived analytically as[14]:

$$Z_{max} = 0.13 \cdot \left(\frac{K}{\log(K)} \right)^2 . \tag{2}$$

2.2. Recall

Let π_i denote the current activation level of node i and let $u_i = \log_2 \pi_i$. The update equations for the recurrent BCPNN are[11]:

$$\frac{du_i}{dt} = \beta_i + \sum_j IC_{ij} \pi_j - u_i \tag{3}$$

$$\pi_i = \begin{cases} 2^{u_i(t)} & u_i(t) < 0 \\ 1 & u_i(t) \geq 0 \end{cases}$$

where IC_{ij} is the weight between nodes i and j as defined in (1) and where β_i is an external input (below referred to as the stimulus).

The recall phase for the recurrent BCPNN consists of two consecutive relaxations of the network with different values of β_i. In each relaxation, the network is iterated to stability based on an Euler forward approximation to the differential equation above. In the first relaxation, $\beta_i = \log_2 x_i$ where $\mathbf{x} = \{x_1, \ldots, x_K\}$ is a given input pattern. In the second relaxation $\beta_i = \log_2 f_i$ where f_i is the average frequency for node i in training data($f_i = \frac{c_i}{N}$). The fixed point in the first relaxation is typically "between" the input pattern and the fixed point in the second relaxation. (To avoid $\log 0$, any x_i or f_i equal to zero is replaced by 1/N in the calculation of β_i.)

Each relaxation is deterministic, and the convergence to a fixed point has been proven[15]. The possible fixed points are local minima to the following Lyapunov energy function:

$$E = -\frac{1}{2} \sum_{i,j} IC_{ij} \pi_i \pi_j - \sum_i \beta_i \pi_i + \sum_i \int_0^{\pi_i} \log_2 x \, dx \tag{4}$$

3. Unsupervised pattern recognition with the recurrent BCPNN

Earlier, the recurrent BCPNN has been used for supervised pattern recognition[11]. Supervised pattern recognition with the recurrent BCPNN requires the true classes to be known for the training data, and the aim of the recall phase is to classify noisy new observations. The external input in each recall is thus the noisy observation to be classified.

In unsupervised pattern recognition, the recurrent BCPNN is trained on noisy and incomplete observations from the underlying classes, but the class labels are unknown. The aim of the recall phase is to extract information about the underlying classes based on the imperfect observations in training data. An important choice is how to set the external input for the recall phase. One possible approach is to use each of the noisy observations in the training data as external input in consecutive recall phases. This often requires a large number of recall iterations to be carried out. In addition, with a high degree of incompleteness in the training data, there is the risk that this approach will only yield the trivial pattern where all nodes have activities equal to their mean frequencies in training data (we refer to this as the empty pattern). An approach that seems to work better in practice is as follows:

1. As initial external input pattern use the nodes' average frequencies in the training data scaled to the interval 0-1

2. After each recall, update the external input pattern by setting any nodes with activity greater than 0.5 in a recalled pattern to 0 in the updated external input pattern; then rescale to the 0-1 interval

Pseudo code for the algorithm is presented in Tables 1 and 2. The idea is to start with an external input pattern that is likely to be close to some fixed point other than the empty pattern, and to modify

$\mathbf{P}$=find_patterns($\mathbf{y}$,$\mathbf{IC}$)

- $\bar{\mathbf{y}}$ is a K-by-1 matrix containing the average frequencies in the training data for each node
- N is the total number of cases in the training data
- $\mathbf{IC}$ is a K-by-K matrix of weights between nodes

```
% Define initial input pattern
x = ȳ
x = x − min(x)
x = x/ max(x)

Set γ to the smallest value between 0 and 2 so that
p=recall(IC,ȳ,N,x,γ) results in max(p) > 0.5

p=recall(IC,ȳ,N,x,γ)
while(max(p) > 0.5 & p not in P)
        Store p in P
        % Modify input pattern
        x(find(p > 0.5))=0
        x = x − min(x)
        x = x/ max(x)
        p=recall(IC,ȳ,N,x,γ)
end
return P
```

Table 1: Pseudo code for the find_patterns algorithm.

$\mathbf{p}$=recall($\mathbf{IC}$,$\bar{\mathbf{y}}$,N,$\mathbf{x}$,γ)

- $\bar{\mathbf{y}}$ is a K-by-1 matrix containing the average frequencies in training data for each node
- N is the total number of cases in the training data
- $\mathbf{IC}$ is a K-by-K matrix of weights between nodes
- $\mathbf{x}$ is a 1-by-K vector with the input pattern
- γ is the activation modifier

```
% Euler forward time step
s = 0.01
% For computational stability
ȳ = max(ȳ, 1/N)
x = max(x, 1/N)
% Initialization
π = 0
u = log2 1/N
% First relaxation
β = log2 x
t = β + γ + π · IC
while(max(| u − t |) > 0.001)
        t = β + γ + π · IC
        π_i = 2^(u_i + s·(t_i − u_i)), ∀i
        u = log2 π
end
% Second relaxation
β = log2 ȳ
t = β + γ + π · IC
while(max(| u − t |) > 0.001)
        t = β + γ + π · IC
        π_i = 2^(u_i + s·(t_i − u_i)), ∀i
        u = log2 π
end
return p
```

Table 2: Pseudo code for the recall algorithm used by find_patterns.

external input before each new recall so that the risk of ending up in the same fixed point twice is small.

When the associations in the data set are weak and the average node activities low, the recurrent BCPNN approach may fail to find any interesting fixed points, because the empty pattern is the only local minimum to the energy function. In this article we show that for such data sets, the interesting patterns may be found by introducing an activation modifier γ in the update equation:

$$\frac{du_i}{dt} = \gamma + \beta_i + \sum_j IC_{ij}\pi_j - u_i \tag{5}$$

Only positive γ values are used. The purpose of γ is to benefit node activity. Compared with the original energy function, the modified energy function:

$$\begin{aligned} E = &- \frac{1}{2}\sum_{i,j} IC_{ij}\pi_i\pi_j - \sum_i \beta_i\pi_i - \gamma\sum_i \pi_i + \\ &+ \sum_i \int_0^{\pi_i} \log_2 x\, dx \end{aligned} \tag{6}$$

has a term $\gamma\sum_i \pi_i$ that lowers the energy more for configurations with high node activity. Because γ does change the energy function, the use of a non-zero γ is limited to data sets where no patterns other than the empty pattern can be found with $\gamma = 0$. For each data set, γ is therefore set (through use of an interval bisection algorithm) to the smallest possible value for which a pattern other than the empty pattern is first recalled with the above defined search procedure. In order to preserve the similarity with the original energy function, only γ values smaller than 2 are allowed.

3.1. Computational requirements

Training of the recurrent BCPNN requires only a single pass over data. For each instance in the data set, it needs to update a subset of the $K^2/2$ counts c_{ij} (those that are non-zero). Therefore its time complexity is:

$$\mathcal{O}(NK^2) \tag{7}$$

(in fact often better, depending on the sparsity of data). All counts c_{ij} are stored in memory, so the memory requirements scale as:

$$\mathcal{O}(K^2) \tag{8}$$

(in fact often better, depending on the sparsity of data).

The time complexity of the the recall phase of the recurrent BCPNN is more difficult to determine. It has previously been estimated to be $\mathcal{O}(K \log K)$[11], but this was under the assumption that the number of active nodes in each prototype pattern was less than or equal to $\log K$. Without this requirement, the time complexity estimate must be adjusted to $\mathcal{O}(K^2)$. The setting of γ necessary for unsupervised pattern recognition requires a binary search with precision ϵ in the interval 0 to 2, where for each value of γ, a full recall must be performed. The time complexity of this binary search is $\log_2 \epsilon$ so the total time complexity for setting γ is equal to:

$$\mathcal{O}(K^2 \log_2 \epsilon) \tag{9}$$

This term dominates the time complexity for the full recall phase in unsupervised pattern recognition since our experiments do not indicate any dependence on K or N of the number of recalls required for each data set. As for the training phase, the memory requirements of the recurrent BCPNN recall phase scale as:

$$\mathcal{O}(K^2) \tag{10}$$

4. AutoClass

AutoClass is a Bayesian clustering algorithm, based on mixture models and expectation maximization (EM) iteration[12]. Clustering is the attempt to find a natural partitioning of cases in a data set into groups. AutoClass was originally developed to interpolate raw data from interplanetary probes and deep space explorations[16], and both AutoClass and its derivatives have been applied to other areas such as classification of bacteria[17].

For a specified number of classes, AutoClass starts by randomly assigning cases to classes and then estimates the relevant parameters of each class; thus, initial class profiles are determined. Cases are then reclassified and class profiles updated. This is repeated until convergence, and the whole process is repeated for differing numbers of classes. AutoClass then determines the optimal number of classes using Bayesian inference.

In the context of this paper, AutoClass's partitioning of data is not of primary focus. Instead we are interested in the estimated probabilities for different attribute values of cases in each class.

In our experiments we used AutoClass C. This is a public domain implementation of AutoClass, which can be downloaded from the Internet[18].

4.1. Computational requirements

For AutoClass, the training phase involves the scanning of data and the full expectation maximization iteration. The recall phase on the other hand is trivial: it consists of inspecting the final cluster profiles.

In contrast to the recurrent BCPNN, the training of AutoClass requires a pass over data for each expectation maximization iteration step. Since AutoClass iterates over different numbers of classes (N_c), tries several different starting points for each specified number of classes (N_{tries}) and in addition requires a large number of cycles (N_{cpt}) for each try in order to converge, its time complexity is [19]:

$$\mathcal{O}(NKN_cN_{tries}N_{cpt}) \tag{11}$$

In the training phase of AutoClass, the entire data set is stored in memory, so the memory requirements scale as [19]:

$$\mathcal{O}(NK) \tag{12}$$

The memory requirements for AutoClass could be improved to $\mathcal{O}(N_cK)$ by modifying the standard implementation, but this would adversely affect the already long run-times.

5. Syndrome detection in the WHO database

The WHO database is constantly monitored to find previously unsuspected ADRs. In June 2002 the database contained 2.8 million suspected ADR reports, collected from 67 countries. Most of the variables are discrete valued. The benefit of a data set of adverse reaction experiences with international coverage, for all marketed products, with data collected from 1967 is clear[20]. It is an attractive possibility to search this database for more complex and unexpected effects of combinations of drugs and ADRs.

The antipsychotic drug haloperiol has well characterized syndromes associated with it, and thus haloperidol reports were selected as a suitable subset of the WHO database for testing. The common literature source Martindale[21] describes five syndromes: the neuroleptic malignant syndrome (NMS), parkinsonism, acute dystonia, tardive dyskinesia and akathisia. To examine scalability, the larger data set of all antipsychotics reports was also examined in a separate experiment.

5.1. Method

The variables considered in these experiments are the 1887 ADR terms in the WHO adverse reaction terminology. Out of these, 721 had actually been reported together with haloperidol and 1391 had been reported together with any of the antipsychotics. There are 8902 case reports in the haloperidol data set, and 100 083 case reports in the antipsychotics data set. These data sets can be regarded as matrices, where each row represents a specific case and each column represents a specific ADR term.

Both the recurrent BCPNN and AutoClass were run on these two data sets. To improve efficiency, any non-occurring ADR terms were dropped from the input data. For AutoClass to converge on these data sets, we had to use a less strict convergence criterion (referred to as "converge" in the AutoClass C manual pages) recommended for exploratory data analysis. To be able to run AutoClass on the antipsychotics data set (1391 terms), we also changed the maximum number of attributes from the default 1000 to 10 000. The only restriction on the number of classes was that there should be more than one.

5.2. BCPNN results

For the haloperidol data set, the recurrent BCPNN produced thirteen output patterns that consist of between 4 and 17 ADR terms. The first output pattern recalled is made up of 17 ADR terms, *i.e.* 17 of the 1887 ADR terms were highlighted as members of this pattern. This first pattern is discussed in detail as an example of the clinical relevance of the work, and is listed in Table 3.

This first pattern includes the following symptoms: creatine phosphokinase increased, fever, death and hypertonia, which are typical of the NMS[22]. The NMS is a well-known characteristic set of symptoms constituting a reaction to haloperidol and other antipsychotic treatments, and the pathology of the disease is well characterized[21]. Some of the other reactions in Table 3 such as agitation and stupor can be considered precursors of the syndrome[23]. Therefore while the related clinical signs and symptoms are not listed in all literature search sources as being part of the NMS they are reasonably present as part of the pattern. Dysphagia was the only ADR term listed in the output pattern, which was neither listed in standard reference sources as a symptom of NMS, nor as a precursor. There are, however, literature reports of suspected haloperidol induced dysphagia[24,25,26], some of which refer to dysphagia as a symptom of

Adr no	Adr Name
A1202	Neuroleptic malignant syndrome
A0116	Hypertonia
A0725	Fever
A0154	Tremor
A0092	Confusion
A0791	Creatine phosphokinase increased
A0163	Agitation
A0091	Coma
A0093	Convulsions
A0224	Tachycardia
A0151	Stupor
A0210	Hypertension
A0043	Sweating increased
A0280	Dysphagia
A0576	Leukocytosis
A0156	Urinary incontinence
A0507	Apnoea

Table 3: The first pattern produced by the recurrent BCPNN for the haloperidol data.

Cij	i	A1202	A0116	A0725	A0154	A0092	A0791	A0163	A0091	A0093	A0224	A0151	A0210	A0043	A0280	A0576	A0156	A0507
j	Ci/Cj	723	585	517	357	348	270	217	174	174	145	143	125	108	108	92	66	60
A1202	723	-	109	171	23	29	126	17	20	11	39	24	27	22	8	43	6	3
A0116	585	109	-	121	67	43	88	26	20	13	16	40	24	26	19	17	8	6
A0725	517	171	121	-	33	38	109	18	25	14	47	30	43	35	9	38	3	4
A0154	357	23	67	33	-	24	9	8	6	11	11	12	8	20	8	3	5	1
A0092	348	29	43	38	24	-	25	39	7	9	14	10	8	11	5	10	16	2
A0791	270	126	88	109	9	25	-	7	13	5	25	14	19	11	5	47	5	5
A0163	217	17	26	18	8	39	7	-	6	6	5	10	5	5	4	2	3	2
A0091	174	20	20	25	6	7	13	6	-	19	9	2	5	6	2	1	5	6
A0093	174	11	13	14	11	9	5	6	19	-	3	8	8	1	3	1	1	7
A0224	145	39	16	47	11	14	25	5	9	3	-	6	29	18	2	7	1	1
A0151	143	24	40	30	12	10	14	10	2	8	6	-	1	5	4	3	2	5
A0210	125	27	24	43	8	8	19	5	5	8	29	1	-	4	2	6	3	1
A0043	108	22	26	35	20	11	11	5	6	1	18	5	4	-	5	3	2	1
A0280	108	8	19	9	8	5	5	4	2	3	2	4	2	5	-	1	1	2
A0576	92	43	17	38	3	10	47	2	1	1	7	3	6	3	1	-	2	1
A0156	66	6	8	3	5	16	5	3	5	1	1	2	3	2	1	2	-	2
A0507	60	3	6	4	1	2	5	2	6	7	1	5	1	1	2	1	2	-

ICij	i	A1202	A0116	A0725	A0154	A0092	A0791	A0163	A0091	A0093	A0224	A0151	A0210	A0043	A0280	A0576	A0156	A0507
j	Ci/Cj	723	585	517	357	348	270	217	174	174	145	143	125	108	108	92	66	60
A1202	723	-	1.16	1.98	-0.38	-0.00	2.48	-0.09	0.46	-0.40	1.69	1.01	1.37	1.28	-0.17	2.48	0.12	-0.74
A0116	585	1.16	-	1.79	1.47	0.87	2.27	0.82	0.76	0.14	0.71	2.05	1.51	1.83	1.38	1.45	0.84	0.56
A0725	517	1.98	1.79	-	0.63	0.87	2.76	0.47	1.27	0.43	2.44	1.81	2.52	2.44	0.48	2.79	-0.40	0.16
A0154	357	-0.38	1.47	0.63	-	0.74	-0.31	-0.16	-0.26	0.61	0.88	1.02	0.63	2.17	0.84	-0.34	0.88	-1.31
A0092	348	-0.00	0.87	0.87	0.74	-	1.20	2.16	-0.00	0.36	1.26	0.80	0.67	1.34	0.20	1.43	2.59	-0.27
A0791	270	2.48	2.27	2.76	-0.31	1.20	-	0.05	1.26	-0.12	2.47	1.65	2.28	1.71	0.57	4.03	1.28	1.42
A0163	217	-0.09	0.82	0.47	-0.16	2.16	0.05	-	0.46	0.46	0.46	1.48	0.67	0.88	0.56	-0.21	0.86	0.41
A0091	174	0.46	0.76	1.27	-0.26	-0.00	1.26	0.46	-	2.44	1.63	-0.52	0.99	1.47	-0.12	-0.89	1.91	2.31
A0093	174	-0.40	0.14	0.43	0.61	0.36	-0.12	0.46	2.44	-	0.04	1.48	1.67	-1.12	0.47	-0.89	-0.41	2.54
A0224	145	1.69	0.71	2.44	0.88	1.26	2.47	0.46	1.63	0.04	-	1.32	3.79	3.31	0.14	2.18	-0.15	-0.01
A0151	143	1.01	2.05	1.81	1.02	0.80	1.65	1.48	-0.52	1.48	1.32	-	-1.05	1.49	1.16	0.98	0.87	2.33
A0210	125	1.37	1.51	2.52	0.63	0.67	2.28	0.67	0.99	1.67	3.79	-1.05	-	1.36	0.36	2.17	1.65	0.21
A0043	108	1.28	1.83	2.44	2.17	1.34	1.71	0.88	1.47	-1.12	3.31	1.49	1.36	-	1.89	1.38	1.28	0.42
A0280	108	-0.17	1.38	0.48	0.84	0.20	0.57	0.56	-0.12	0.47	0.14	1.16	0.36	1.89	-	-0.20	0.28	1.42
A0576	92	2.48	1.45	2.79	-0.34	1.43	4.03	-0.21	-0.89	-0.89	2.18	0.98	2.17	1.38	-0.20	-	1.51	0.65
A0156	66	0.12	0.84	-0.40	0.88	2.59	1.28	0.86	1.91	-0.41	-0.15	0.87	1.65	1.28	0.28	1.51	-	2.13
A0507	60	-0.74	0.56	0.16	-1.31	-0.27	1.42	0.41	2.31	2.54	-0.01	2.33	0.21	0.42	1.42	0.65	2.13	-

Table 4: The upper table displays joint counts c_{ij} for all pairs of nodes in the first pattern produced by the recurrent BCPNN on the haloperidol data. The lower table shows the corresponding weights (IC_{ij}). In both tables, a shaded cell indicates a negative IC_{ij} for the pair.

NMS[27,28]. The highlighting of dysphagia can thus be considered a true positive finding. This emphasizes the ability of this method to highlight less well established associations, which will be a goal of the method in the detection of previously unsuspected syndromes.

The recurrent BCPNN highlights ADR terms which overall have high pairwise dependencies between members of the pattern. To illustrate this, Table 4 shows the strength of pairwise dependencies between the seventeen ADRs in the first output pattern. The ADRs are listed both as columns and rows, and their marginal counts, c_i and c_j, are shown in the second column and the second row. The upper table shows the joint counts c_{ij} and the lower table shows the pairwise strengths of association (IC_{ij}) between the ADRs which were included in the first output pattern for the haloperidol data. The highlighted ADRs were not among the most commonly reported for haloperidol. Additionally, the highlighted ADRs are not amongst those that have the highest IC values between the drug and individual ADRs. This emphasizes that more complex dependencies between ADRs are detected with this technique and that a clinically relevant pattern can be recognized, which might have been missed by pure individual pairwise analysis alone. Only hypertonia (A0116) has positive ICs with all other ADRs in the pattern. Every ADR is reported on at least one report with each of the other ADRs in the highlighted pattern, although this is not a necessary criterion for inclusion of an ADR in an output pattern.

The next four output patterns produced by the recurrent BCPNN are listed in Tables 5-8. Parkinsonism, which is made up of symptoms like abnormal gait, tremor, involuntary muscle contractions and increased salivation, is clearly detected (see Table 5). Also acute dystonia shown by clenching of muscles, oculogyric crisis and laryngeal dystonia is detected (see Table 8). However, akathisia, a condition of restlessness and anxiety, is not detected, and neither is tardive dyskinesia, a syndrome of orofacial movements, which is further characterized by sucking, chewing, involuntary movement and repetitive piano playing hand movements.

To detect three of the five most well known syndromes for haloperidol is a very convincing result, as syndromes are sometimes poorly characterized, and all symptoms are rarely seen together, and still less often reported. The use of a variety of different clinical terms for the same entity within a syndrome is a challenge to the interpretation of patterns, and is a limitation to the generation of patterns because of dilution. Akathisia is a probable example of this because restlessness and anxiety are relatively nonspecific and can be reported using a variety of different words. Tardive dyskinesia on the other hand is often reported as a single ADR term in the database and not as a collection of signs and symptoms.

Thus, this method has effectively associated the pattern of ADRs that make up the NMS, and other syndromes associated with haloperidol. This shows that clinically important patterns can be found using the recurrent BCPNN in the WHO database of side-effect reports. The importance of the result is

Adr no	Adr Name
A0087	Aphasia
A0088	Ataxia
A0092	Confusion
A0093	Convulsions
A0108	Gait abnormal
A0116	Hypertonia
A0150	Speech disorder
A0154	Tremor
A0155	Muscle contractions involuntary
A0183	Insomnia
A0197	Somnolence
A0199	Thinking abnormal
A0222	Saliva increased

Table 5: The second pattern produced by the recurrent BCPNN for the haloperidol data.

Adr no	Adr Name
A0106	Extrapyramidal disorder
A0114	Hyperkinesia
A0116	Hypertonia
A0118	Hypokinesia
A0150	Speech disorder
A0154	Tremor
A0172	Depression
A0188	Nervousness
A0222	Saliva increased
A0280	Dysphagia
A0724	Fatigue
A0731	Rigors

Table 6: The third pattern produced by the recurrent BCPNN for the haloperidol data.

Adr no	Adr Name
A0091	Coma
A0093	Convulsions
A0144	Respiratory depression
A0198	Suicide attempt
A0208	Bradycardia
A0212	Hypotension
A0437	Cardiac arrest
A0501	Cyanosis
A0502	ECG abnormal
A0507	Apnoea
A0722	Death

Table 7: The fourth pattern produced by the recurrent BCPNN for the haloperidol data.

Adr no	Adr Name
A0043	Sweating increased
A0068	Dystonia
A0077	Torticollis
A0102	Dyskinesia
A0108	Gait abnormal
A0116	Hypertonia
A0132	Oculogyric crisis
A0134	Opisthotonos
A0150	Speech disorder
A0154	Tremor
A0155	Muscle contractions involuntary
A0514	Dyspnoea

Table 8: The fifth pattern produced by the recurrent BCPNN for the haloperidol data.

Adr no	Adr Name	P(Adr)
A0106	Extrapyramidal disorder	0.16
A0068	Dystonia	0.11
A0116	Hypertonia	0.05
A0102	Dyskinesia	0.05
A0154	Tremor	0.05
A0197	Somnolence	0.04
A0114	Hyperkinesia	0.04
A0092	Confusion	0.04
A0132	Oculogyric crisis	0.04
A0163	Agitation	0.03
A0150	Speech disorder	0.03
A1065	Dyskinesia tardive	0.03
A0222	Saliva increased	0.02
A0027	Rash	0.02
A0179	Hallucination	0.02
A0108	Gait abnormal	0.02
A1202	Neuroleptic malignant syndrome	0.02
A0908	Leucopenia	0.02

Table 9: The larger (74% of all cases) of the two patterns produced by AutoClass for the haloperidol data. NMS is Neuroleptic malignant syndrome.

Adr no	Adr Name	P(Adr)
A1202	Neuroleptic malignant syndrome	0.23
A0722	Death	0.20
A0725	Fever	0.18
A0791	Creatine phosphokinase increased	0.12
A0116	Hypertonia	0.12
A0058	Injection site reaction	0.07
A0212	Hypotension	0.07
A0437	Cardiac arrest	0.06
A0091	Coma	0.06
A0224	Tachycardia	0.04
A0057	Injection site pain	0.04
A0055	Injection site mass	0.04
A0093	Convulsions	0.04
A0151	Stupor	0.04
A0092	Confusion	0.03
A0210	Hypertension	0.03
A0576	Leukocytosis	0.03
A0106	Extrapyramidal disorder	0.03
A0360	Sgpt increased	0.03
A0359	Sgot increased	0.03
A0043	Sweating increased	0.03
A0197	Somnolence	0.03
A0893	Hyperpyrexia malignant	0.02
A0163	Agitation	0.02

Table 10: The smaller (26% of all cases) of the two patterns produced by AutoClass for the haloperidol data. NMS is Neuroleptic malignant syndrome. CPI is Creatine phosphokinase increased.

emphasized by the complete absence of any other method, apart from manual inspection, for detecting such clusters of ADRs.

For the antipsychotics subset of the WHO database, the recurrent BCPNN generated 12 output patterns. The patterns were overall rather large, with an average of 26 ADR terms per pattern. These patterns were also clinically valid and coherent.

5.3. AutoClass results

The AutoClass solution to the clustering of haloperidol data had only two classes. These are presented in Table 9 and Table 10 together with the probabilities for different ADR terms given the class. The pattern corresponding to the larger cluster of cases (Table 9) contains characteristic elements of parkinsonism, acute dystonia, tardive dyskinesia and NMS. The other pattern (Table 10) does contain some of the NMS symptoms, but many unrelated terms as well. AutoClass consequently did not effectively identify any of the syndromes associated with haloperidol. For the antipsychotics data set, the AutoClass solution again had only two classes, which were a mixture of several groups of symptoms.

5.4. Run times

To run the recurrent BCPNN on the haloperidol data set required 3.6 seconds for the training phase and 6.5 seconds for the recall phase. The time required to train the recurrent BCPNN on the antipsychotics data set was 43 seconds, and the time required to recall the 12 patterns was 4.3 seconds.

Run times for AutoClass depend on how many EM iterations are carried out and in this choice there is a trade-off between performance and speed. For the haloperidol data, each EM iteration took at least five minutes (average run times increase with the number of components in the assumed mixture). For the antipsychotics data set, each EM iteration took at least 30 minutes (the increase is due to the larger number of cases and variables in this data set). In most situations there will be a need to run one or more EM iterations for a range of different numbers of mixture components, so the total run times will typically be in the order of hours. In our experiments AutoClass was run for over 20 hours each on both the haloperidol and the antipsychotics data sets.

All timings were performed on a computer equipped with a Pentium III 1.4 GHz processor and 3 GB of RAM.

6. Experiments on simulated data

To further explore the performance of the recurrent BCPNN, we also applied it to a number of simulated data sets with known properties. For comparison, AutoClass was run on the same data sets using the default search parameters.

6.1. Data generation

The simulated data sets are based on noisy and incomplete samples of two artificial prototype patterns (see Figure 2), as well as added pure noise samples. In each of these data sets there are 81 binary attributes.

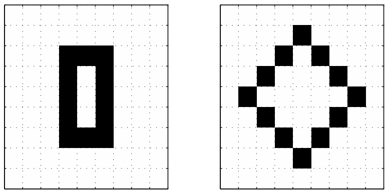

Figure 2: The two patterns used as prototype patterns. It is only for visual purposes they are organized as 9 x 9 matrices, rather than as 1 x 81 vectors; to the method, spatial configuration is irrelevant.

Each sample can be seen as a model of an ADR report entered into the WHO database — either an incomplete and noisy observation of a syndrome (sample of a prototype pattern) or an unrelated incident (noise). For the prototype samples, incompleteness was simulated by only using a random subset of the nodes in each prototype. Noise is then introduced by activating a random subset of the then inactive nodes. Pure noise samples were generated through random activation of a given proportion of the 81 nodes. This was carried out so that the same number of nodes were active in the pure noise samples as in the prototype samples.

Figures 3 and 4 display examples of samples from two of the simulated data sets.

For each simulated data set, 400 prototype samples were used with each sample having a 0.5 probability of coming from either prototype. Each data set had a specified level of completeness and noise.

The completeness levels were: 100%, 75%, 50% and 25%, and the noise levels were: 0%, 25%, 50% and 75% (relative to the average number of active

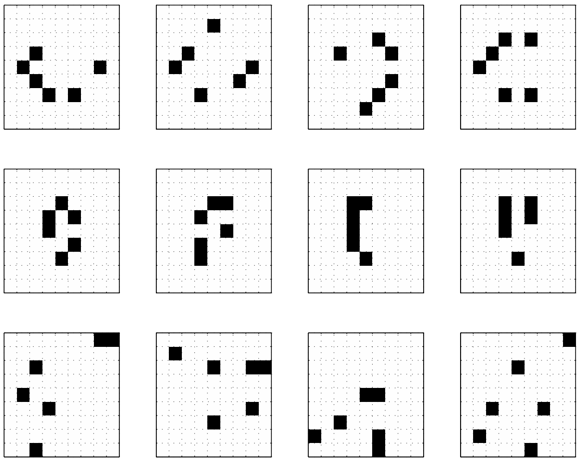

Figure 3: 12 training samples at completeness level 50% and noise level 0%. The top four samples are taken from the diamond prototype, the middle four from the rectangle prototype and the bottom four are pure noise. The underlying prototypes are barely distinguishable.

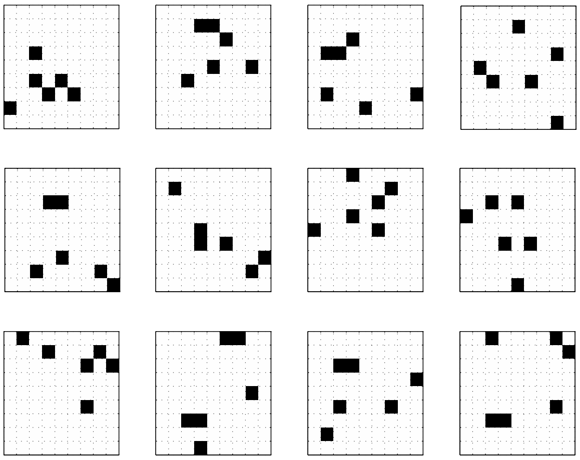

Figure 4: 12 training samples at completeness level 25% and noise level 25%. The top four samples are taken from the diamond prototype, the middle four from the rectangle prototype and the bottom four are pure noise. The underlying prototypes are no longer distinguishable.

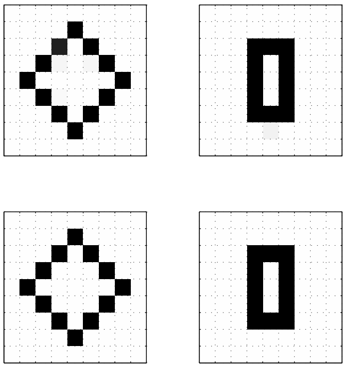

Figure 5: Sample output from the recurrent BCPNN at completeness level 50% and noise level 50%, with 4000 pure noise samples. The top row displays the raw output, and the bottom row displays the thresholded patterns.

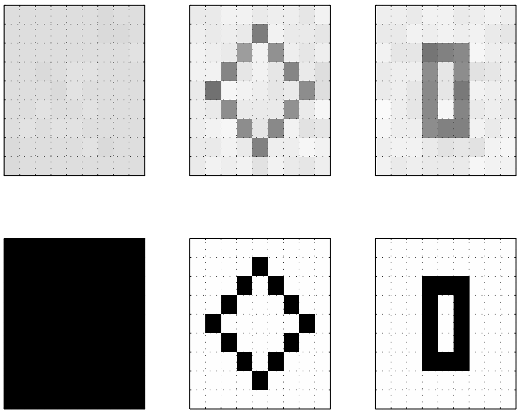

Figure 6: Sample output from AutoClass at completeness level 50% and noise level 50%, with 4000 pure noise samples. The top row displays the raw output, and the bottom row displays the thresholded patterns.

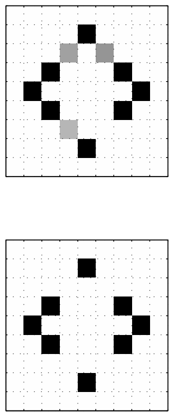

Figure 7: Sample output from the recurrent BCPNN at completeness level 25% and noise level 0%, with 4000 pure noise samples. The top row displays the raw output, and the bottom row displays the thresholded patterns.

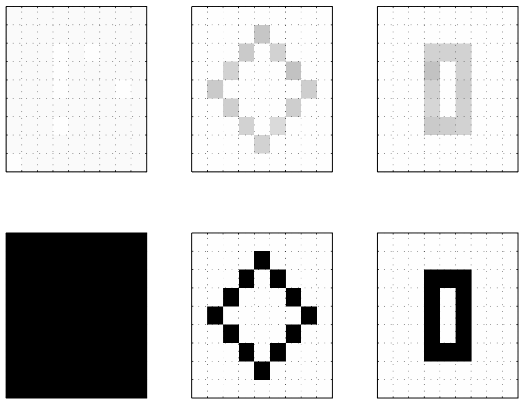

Figure 8: Sample output from AutoClass at completeness level 25% and noise level 0%, with 4000 pure noise samples. The top row displays the raw output, and the bottom row displays the thresholded patterns.

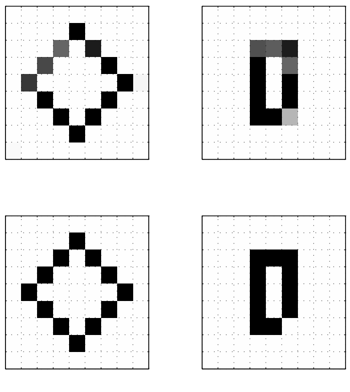

Figure 9: Sample output from the recurrent BCPNN at completeness level 25% and noise level 25%, with 2000 pure noise samples. The top row displays the raw output, and the bottom row displays the thresholded patterns.

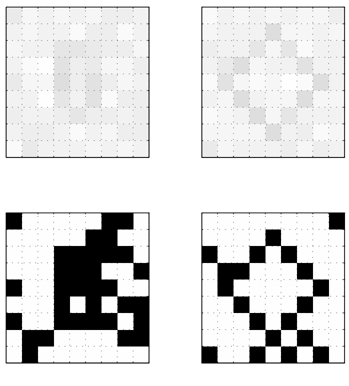

Figure 10: Sample output from AutoClass at completeness level 25% and noise level 25%, with 2000 pure noise samples. The top row displays the raw output, and the bottom row displays the thresholded patterns.

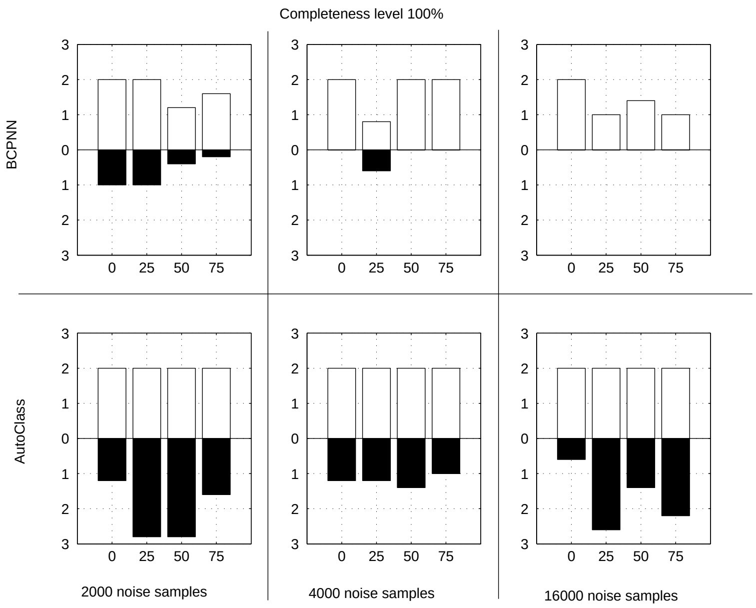

Figure 11: Average number of recalled patterns that were correct (white) and incorrect (black) for varying levels of added noise.

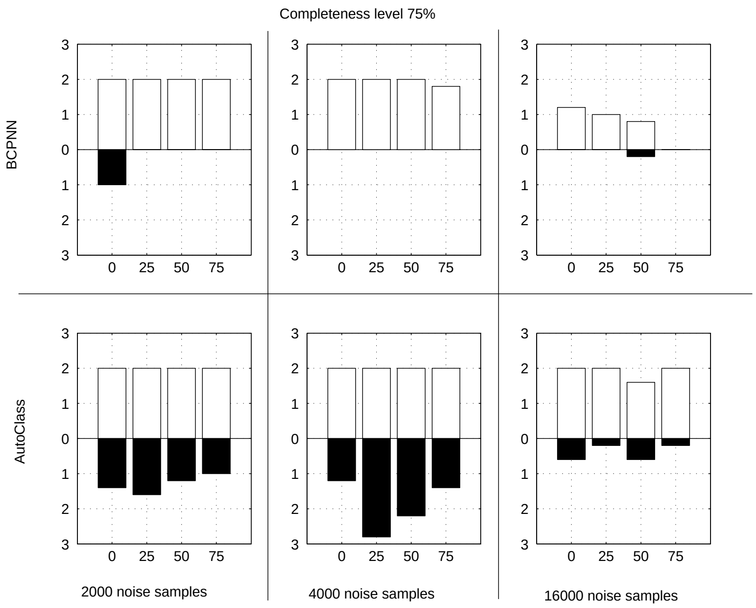

Figure 12: Average number of recalled patterns that were correct (white) and incorrect (black) for varying levels of added noise.

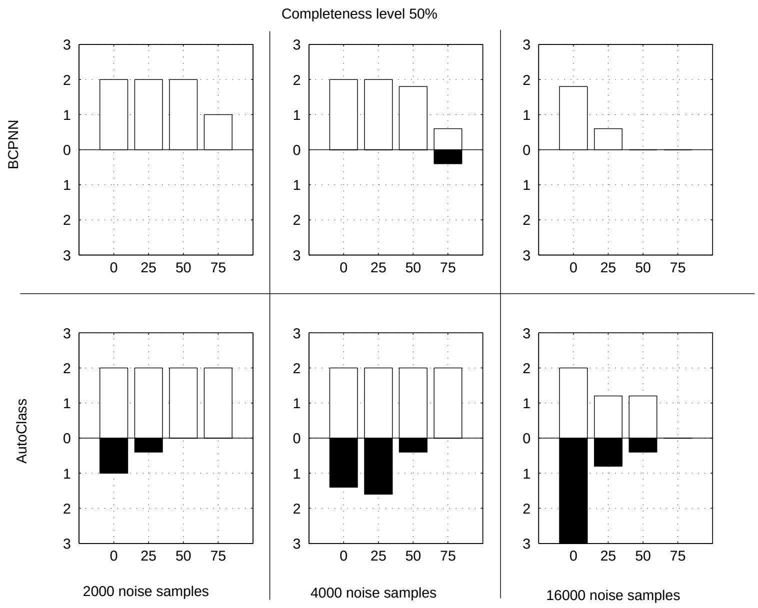

Figure 13: Average number of recalled patterns that were correct (white) and incorrect (black) for varying levels of added noise.

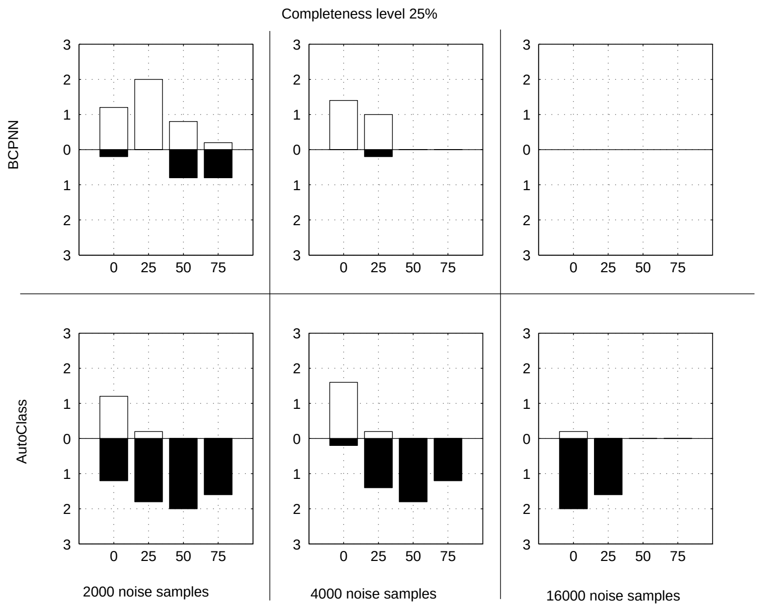

Figure 14: Average number of recalled patterns that were correct (white) and incorrect (black) for varying levels of added noise.

nodes in the prototypes). In addition a varying number of pure noise samples were added: 2000, 4000 and 16000. Thus in each experiment 48 ($4 \cdot 4 \cdot 3$) data sets were simulated and examined.

6.2. Methods for data analysis

Both AutoClass and the recurrent BCPNN were run on each of the 48 simulated data sets. This whole process (including the simulation of data) was repeated five times, in order to get more reliable estimates for the performance of the methods.

For data sets with low levels of completeness, AutoClass gives weak output patterns (the probability for each node to be on is overall lower). For comparability, both AutoClass outputs and recurrent BCPNN outputs were therefore thresholded at $0.5 \cdot \max(\pi_i)$. The thresholded patterns were considered to be true positives if the Hamming distance to either of the prototypes was less than 5 (with a limit of one true positive per prototype) — otherwise as false positives.

Since AutoClass is a clustering algorithm it will, under perfect performance, output a cluster containing all the pure noise samples. Although such a cluster does not correspond to one of our original prototype patterns, we do not consider it to be a false finding. Therefore for the simulated data sets any AutoClass output pattern where all nodes are highlighted is excluded from the analysis.

6.3. Results

Figures 5-10 display sample outputs of the two methods for different simulated data sets. Figures 11- 14 display the average number of true and false patterns found for each combination of completeness and noise.

Overall, both the recurrent BCPNN and AutoClass perform well on a large portion of the simulated data sets. In general, both methods effectively highlight the true prototype patterns. On average AutoClass detects 1.5 of the 2 underlying prototypes and the recurrent BCPNN 1.2. Low levels of completeness combined with high levels of noise, however, result in reduced performance for both methods.

While AutoClass detects true prototype patterns more effectively than the recurrent BCPNN (especially when a large number of pure noise samples are added), AutoClass also generates more false positive patterns (on average 1.2 compared to 0.2 for the recurrent BCPNN).

The focus of this article is inference based on incomplete observations. For 25% completeness, the recurrent BCPNN generates on average 0.6 correct and 0.2 incorrect patterns, whereas AutoClass generates 0.3 correct and 1.2 incorrect patterns. While both methods perform understandably worse in this domain, the recurrent BCPNN is the better of the two.

Imperfectly recalled prototypes are observed for the recurrent BCPNN but rarely so for AutoClass. More thorough inspection of the results indicate that whereas the performance of the recurrent BCPNN gradually deteriorates as incompleteness and noise levels increase, AutoClass tends to find the underlying prototypes either well or not at all.

7. Discussion

Our results on the simulated data indicate that for a wide range of noise and completeness levels, both methods are effective (97% of the correctly recalled patterns are in fact within a Hamming distance of 1). However, it is noteworthy that for the most interesting type of data sets (those with low completeness) the recurrent BCPNN performs better.

The results on the real world WHO ADR data sets are significantly better for the recurrent BCPNN than for AutoClass. The poor performance of AutoClass on these data sets may in part be due to high levels of incompleteness in the observations, but the large size of these data sets also plays a crucial role.

As shown in Sections 3.1 and 4.1, the time and memory requirements on data sets such as the WHO database, where there are many more cases than attributes, are much lower for the recurrent BCPNN than for AutoClass. This is demonstrated by the run times given in Section 5.4.

A potential disadvantage with the recurrent BCPNN is a restriction in the maximum number of classes that can be stored in the network (see Section 2.1). If the number of classes in the data exceeds this capacity, there is the risk of so called catastrophic forgetting[29]. However, this is rarely a problem since in practice the capacity often far exceeds the number of fixed points we can reasonably expect. For example, in the haloperidol data set, the capacity is $0.13 \cdot (721/\log 721)^2 \approx 1500$ which is of the same order of magnitude as the number of cases we have presented to the network.

The recurrent BCPNN has a clear benefit compared to AutoClass, in that it does not generate many false positives, and consequently does not require elaborate methods to distinguish true from spurious output patterns.

AutoClass requires an assumption of the maximum number of classes that exist in the data set, thus forcing a trade-off between computational tractability and reliability of the output. For the recurrent BCPNN there is no such constraint.

In the version of the recurrent BCPNN described in this article, there is only one node per attribute. A different architecture has been proposed where separate nodes are used to encode each attribute value (*e.g.* a binary attribute would have two nodes)[30], and the benefits of such an architecture will be investigated.

In the future we will also use this method to look for syndromes related to other drugs in the WHO database, as well as look for other types of patterns amongst the other variables in the data set. We also intend to implement the method on other data sets.

The effectiveness of the recurrent BCPNN in unsupervised pattern recognition is encouraging for its use on other data sets with a high proportion of incomplete observations. The results on the WHO data are impressive, although their generalizability to all other areas of the database is hard to guarantee, since the levels of noise and incompleteness are unknown and can be expected to vary. However, our experiments with simulated data indicate that the recurrent BCPNN is robust over a wide range of noise and incompleteness levels. As the recurrent BCPNN is already tractable for large data sets such as the WHO database, this makes it a highly promising data mining tool.

References

1. P. Chan, W. Fan, A. Prodromidis, and S. Stolfo, "Distributed data mining in credit card fraud detection," *IEEE Intelligent Systems*, vol. 14, no. 6, pp. 67–74, 1999.
2. C. Silverstein, S. Brin, and R. Motwani, "Beyond market baskets: generalizing association rules to dependence rules," *Data mining and knowledge discovery*, vol. 2, pp. 39–68, 1998.
3. D. J. Hand, "Data mining: Statistics and more?," *The American Statistician*, vol. 52, no. 2, pp. 112–118, 1998.
4. U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, eds., *Advances in Knowledge Discovery and Data Mining*. Menlo Park: MIT press, 1996.
5. B. F. Pennington, *Diagnosing Learning Disorders: A Neuropsychological Framework*. The Guilford Press, 1991. p 24.
6. R. Orre, A. Lansner, A. Bate, and M. Lindquist, "Bayesian neural networks with confidence estimations applied to data mining," *Computational Statistics and Data Analysis*, vol. 34(4), no. 4, pp. 473–493, 2000.
7. W. DuMouchel and D. Pregibon, "Empirical bayes screening for multi-item associations," in *Proceedings of the 7th ACM SIGMOD KDD international conference on knowledge discovery and data mining*, pp. 67–76, ACM Press, San Francisco, CA, 2001.
8. R. Orre, A. Bate, and M. Lindquist, "Bayesian neural networks used to find adverse drug related syndromes," in *Proc. of the ANNIMAB-1 Conference*, (Gothenburg, Sweden), pp. 215–220, Springer, April 2000.
9. A. Holst, *The Use of a Bayesian Neural Network Model for Classification Tasks*. PhD thesis, Dept. of Numerical Analysis and Computing Science, Royal Institute of Technology, Stockholm, Sweden, Sept. 1997. TRITA-NA-P9708.
10. A. Bate, M. Lindquist, I. R. Edwards, S. Olsson, R. Orre, A. Lansner, and R. M. D. Freitas, "A bayesian neural network method for adverse drug reaction signal generation," *European Journal of Clinical Pharmacology*, vol. 54, pp. 315–321, 1998.
11. A. Lansner and Ö. Ekeberg, "A one-layer feedback, artificial neural network with a Bayesian learning rule," *International Journal of Neural Systems*, vol. 1, no. 1, pp. 77–87, 1989. Also extended abstract in Proceedings from the Nordic symposium on Neural Computing, April 17–18, Hanasaari Culture Center, Espoo, Finland.
12. P. Cheeseman and J. Stutz, "Bayesian classification (AUTOCLASS): Theory and results," in *Advances in Knowledge Discovery and Data Mining* (U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, eds.), MIT Press, 1995.
13. D. J. Hand, H. Mannila, and P. Smyth, *Principles of Data Mining*. MIT Press, 2001.
14. A. Lansner and Ö. Ekeberg, "Reliability and speed of recall in an associative network," *IEEE Trans on Pattern Analysis and Machine Intelligence*, vol. 7, no. 4, pp. 490–498, 1985.
15. J. Hopfield, "Neurons with graded response have collective computational properties like those of the two-state neurons," *Proceedings of the National Academy of Science, USA*, vol. 81, pp. 3088–3092, 1984.
16. P. Cheeseman, J. Stutz, M. Self, W. Taylor, J. Goebel, K. Volk, and H. Walker, "Automatic classification of spectra from the infrared astronomical satellite (iras)," tech. rep., NASA Reference publication 1217, 1989.
17. M. Gyllenberg, T. Koski, and T. Lund, "Applying the EM-algorithm to classification of bacteria," in *Proceedings of the international ICSC congress on intelligent systems and applications* (F. Naghdy,

F. Kurfess, H. Ogata, E. Szczerbicki, H. Bother, and H. Tlanfield, eds.), vol. 2, pp. 65–71, 2000.

18. Bayesian Learning Group, "The autoclass project." http://ic-www.arc.nasa.gov/ic/projects/bayes-group/autoclass/. Accessed 16 June 2003.
19. P. Cheeseman, J. Stutz, and W. Taylor, "AUTOCLASS C (version 3.3.3unx)." [Unix manual pages], Dec 2001.
20. S. Olsson and I. R. Edwards, "The WHO international drug monitoring programme," in *Side Effects of Drugs Annual 23* (J. Aronsson, ed.), pp. 524–529, Elsevier, 2000.
21. K. Parfitt, ed., *Martindale: the complete drug reference*. Pharmaceutical press, 32 ed., 1999.
22. M. Dukes and J. Aronson, *Meyler's Side Effects of drugs*, vol. 14. Elsevier, 2000.
23. M. Bristow and D. Kohen, "How "malignant" is the neuroleptic malignant syndrome?," *BMJ*, vol. 307, no. 6914, pp. 1223–1226, 1993.
24. L. Sokoloff and R. Pavlakovic, "Neuroleptic-induced dysphagia," *Dysphagia*, vol. 12, no. 4, pp. 177–186, 1997.
25. T. Hayashi, T. N. I. Koga, Y. Uchida, and S. Yamawaki, "Life-threatening dysphagia following prolonged neuroleptic therapy," *Clin. Neuropharmacol*, vol. 20, no. 1, pp. 77–81, 1997.
26. T. A. T. Hughes, G. Shone, G. Lindsay, and C. M. Wiles, "Severe dysphagia associated with major tranquillizer treatment," *Postgrad Med J*, vol. 70, no. 826, pp. 581–584, 1994.
27. F. Cruz, D. Thiagarajan, and J. Harney, "Neuroleptic malignant syndrome after haloperidol therapy," *South Med J*, vol. 76, no. 5, pp. 684–690, 1983.
28. R. A. Hanel, M. C. Sandmann, M. Kranich, and P. D. Bittencourt, "Neuroleptic malignant syndrome: case report with recurrence associated with the use of olanzapine," *Arq Neuropsiquiatr*, vol. 56, no. 4, pp. 833–840, 1998.
29. A. Sandberg, *Bayesian attractor neural network models of memory.* PhD thesis, Royal Institute of Technology, Stockholm, Sweden, 2003.
30. A. Holst and A. Lansner, "A higher order bayesian neural network for classification and diagnosis," in *Proceedings of the International Workshop on Applications of Neural Networks to Telecommunications 2* (J. Alspector, R. Goodman, and T. X. Brown, eds.), pp. 347–354, Lawrence Erlbaum Associates, Hillsdale, 1995. Stockholm, Sweden, May 22-24.