

Time Series Classification for the Prediction of Dialysis in Critically Ill Patients using Echo State Networks

Femke Ongenaes^{a,*}, Stijn Van Looy^b, David Verstraeten^c, Thierry Verplancke^d, Dominique Benoit^d, Filip De Turck^a, Tom Dhaene^a, Benjamin Schrauwen^c, Johan Decruyenaere^d

^a*Department of Information Technology (INTEC), Ghent University - Interdisciplinary Institute for Broadband Technology (IBBT), Gaston Crommenlaan 8 bus 201, B-9050 Ghent, Belgium*

^b*Department of Environmental Modelling, Flemish Institute for Technological Research (VITO), Boeretang 200, 2400 Mol, Belgium*

^c*Department of Electronics and Information Systems (ELIS), Ghent University, Sint-Pietersnieuwstraat 41, 9000 Ghent, Belgium*

^d*Department of Intensive Care Medicine, Ghent University Hospital, De Pintelaan 185 - 2 K12 IC, B-9000 Ghent, Belgium*

Abstract

Objective: Time series often appear in medical databases, but only few machine learning methods exist that process this kind of data properly. Most modeling techniques have been designed with a static data model in mind and are not suitable for coping with the dynamic nature of time series. Recurrent Neural Networks (RNN) are often used to process time series, but only a few training algorithms exist for RNNs which are complex and often yield poor results. Therefore, researchers often turn to traditional machine

*Corresponding author: Tel.: +32 9 331 49 38, Fax: +32 9 331 48 99

Email addresses: Femke.Ongenaes@intec.UGent.be (Femke Ongenaes), Stijn.VanLooy@vito.be (Stijn Van Looy), David.Verstraeten@UGent.be (David Verstraeten), Thierry.Verplancke@UGent.be (Thierry Verplancke), Dominique.Benoit@UGent.be (Dominique Benoit), Filip.DeTurck@intec.UGent.be (Filip De Turck), Tom.Dhaene@UGent.be (Tom Dhaene), Benjamin.Schrauwen@UGent.be (Benjamin Schrauwen), Johan.Decruyenaere@UGent.be (Johan Decruyenaere)

Preprint submitted to Engineering Applications of Artificial Intelligence July 25, 2012

learning approaches, such as support vector machines (SVM), which can easily be set up and trained and combine them with feature extraction (FE) and selection (FS) to process the high-dimensional temporal data. Recently, a new approach, called echo state networks (ESN), has been developed to simplify the training process of RNNs. This approach allows modeling the dynamics of a system based on time series data in a straightforward way.

The objective of this study is to explore the advantages of using ESN instead of other traditional classifiers combined with FE and FS in classification problems in the intensive care unit (ICU) when the input data consists of time series. While ESNs have mostly been used to predict the future course of a time series, we use the ESN model for classification instead. Although time series often appear in medical data, little medical applications of ESNs have been studied yet.

Methods and material: ESN is used to predict the need for dialysis between the fifth and tenth day after admission in the ICU. The input time series consist of measured diuresis and creatinine values during the first 3 days after admission. Data about 830 patients was used for the study, of which 82 needed dialysis between the fifth and tenth day after admission. ESN is compared to traditional classifiers, a sophisticated and a simple one, namely support vector machines and the naive Bayes (NB) classifier. Prior to the use of the SVM and NB classifier, FE and FS is required to reduce the number of input features and thus alleviate the curse dimensionality. Extensive feature extraction was applied to capture both the overall properties of the time series and the correlation between the different measurements in the times series. The feature selection method consists of a greedy hybrid filter-

wrapper method using a NB classifier, which selects in each iteration the feature that improves prediction the best and shows little multicollinearity with the already selected set. Least squares regression with noise was used to train the linear readout function of the ESN to mitigate sensitivity to noise and overfitting. Fisher labeling was used to deal with the unbalanced data set. Parameter sweeps were performed to determine the optimal parameter values for the different classifiers. The area under the curve (AUC) and maximum balanced accuracy are used as performance measures. The required execution time was also measured.

Results: The classification performance of the ESN shows significant difference at the 5% level compared to the performance of the SVM or the NB classifier combined with FE and FS. The NB + FE + FS, with an average AUC of 0.874, has the best classification performance. This classifier is followed by the ESN, which has an average AUC of 0.849. The SVM + FE + FS has the worst performance with an average AUC of 0.838. The computation time needed to pre-process the data and to train and test the classifier is significantly less for the ESN compared to the SVM and NB.

Conclusion: It can be concluded that the use of ESN has an added value in predicting the need for dialysis through the analysis of time series data. The ESN requires significantly less processing time, needs no domain knowledge, is easy to implement, and can be configured using rules of thumb.

Keywords: time series, classification, echo state network, dialysis, feature extraction and selection

1. Introduction

Time series are a special kind of input data to machine learning problems. Most modeling techniques have been designed with a static data model in mind and are not suitable for coping with the dynamic nature of time series. Most dynamic data models are very complex in both design and training algorithms. Examples of such models based on artificial neural networks are the hidden control neural network (Levin, 1993), the neural prediction model (Iso and Watanabe, 1991), the linked predictive neural network (Tebelskis et al., 1990) and the adaptive time-delay neural network (Xie et al., 2006). Recurrent Neural Networks (RNNS) are often used (Robinson, 1994) since this type of artificial neural network can represent high-dimensional nonlinear temporal data. Hidden Markov models (Rabiner, 1989) and neural network - hidden Markov model hybrids (Graves and Schmidhuber, 2005; Trentin and Gori, 2003) are also used to model time series data. An obstacle when using RNNS is that only a few training algorithms exist which are complex and often yield poor results (Haykin, 1994; Jaeger, 2002b).

More recently, three approaches to simplify the training process of RNNS were independently developed. These approaches are liquid state machines (LSM) (Maass et al., 2002), echo state networks (ESN) (Jaeger, 2001), and backpropagation decorrelation (BPDC) (Steil, 2006). The underlying idea of these three methods is similar and nowadays they are referred to as *reservoir computing* (Verstraeten et al., 2007). Reservoir computing has become a vivid research field and recently a special issue of “*Neural Networks*” was dedicated to it (Jaeger et al., 2007).

The key idea in reservoir computing is that the dynamic system producing

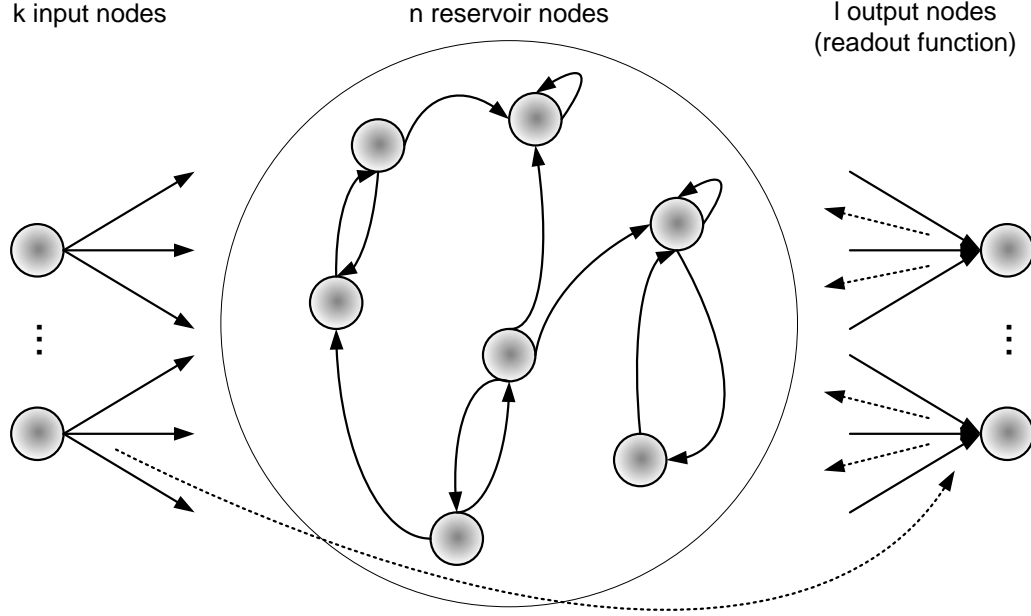


Figure 1: The general layout of an echo state network. Circles represent input, reservoir, and output nodes. Arrows represent non-zero weighted connections. Dotted arrows denote optional connections.

the time series data is modeled in a *reservoir* consisting of a RNN. The reservoir is then read by a linear readout function, which is illustrated in Figure 1. The output of this readout function can then be used to make several kinds of predictions. The training algorithm only affects the linear readout function. For training linear functions many algorithms exist such as linear regression (Fisher, 1925).

The goal of this study is to verify whether the use of reservoir computing methods is an added value in classification problems in the intensive care unit (ICU) when the input data consists of time series. We select a case study that is easily characterized by medical experts. This medical classifi-

36 cation problem is then handled using reservoir computing, which can directly
37 cope with time series data, and the performance is compared to more tra-
38 ditional machine learning approaches, which cannot directly cope with this
39 high-dimensional temporal data and thus need to be combined with feature
40 extraction (FE) and selection (FS) to process the time series.

41 LSMs and ESNs are the two pioneering reservoir computing methods.
42 However, the two methods have a very different background (Jaeger et al.,
43 2007). The initial ESN publications were framed in settings of machine learn-
44 ing and nonlinear signal processing applications (Jaeger, 2001, 2002a,b, 2003;
45 Jaeger and Haas, 2004). In contrast, LSMs were developed from a compu-
46 tational neuroscience background, aiming at elucidating principal computa-
47 tional properties of microcircuits (Maass et al., 2002, 2003, 2004; Natschläger
48 et al., 2002).

49 This difference in background also explains the main difference between
50 LSMs and ESNs (Lukoševičius and Jaeger, 2009). ESNs standardly use sim-
51 ple sigmoid neurons or leaky integrator neuron models, while LSMs use more
52 sophisticated and biologically realistic models built from a spiking neuron
53 model called the Leaky Integrate and Fire (LIF) neuron (Maass and Bishop,
54 2001) and dynamic synaptic connection models (Gerstner and Kistler, 2002)
55 in the reservoir. Since the model of both the connections and the neurons
56 themselves in LSMs is quite sophisticated, it has a large number of free pa-
57 rameters to be set, which is done manually, guided by biologically observed
58 parameter ranges. The parameters of ESNs, e.g., the warm-up drop and the
59 leak rate, are more intuitive and can easily be set by using rules of thumb or
60 performing parameter sweeps. Moreover, LSMs require pulse trains as input

61 data. Translating continuous data, of which the training data of the medical
62 problem under study in this research consists, to pulse trains is a complex
63 problem. Consequently, LSMs are usually more difficult to implement, to
64 correctly set up and tune, and typically more expensive to emulate on digital
65 computers than simple ESN-type “weighted sum and non-linearity” RNNs.
66 Thus LSMs are less widespread for engineering applications of RNNs than
67 ESNs. This makes ESNs the better choice for “simple” engineering tasks,
68 such as the medical classification problem under study in this research.

69 The idea of separation between a reservoir and a readout function has
70 also been arrived at from the point of view of optimizing the performance of
71 the RNN training algorithms that use error backpropagation. It was found
72 that the Atiya-Parlos recurrent learning (APRL) rule (Schiller and Steil,
73 2005) restricts the adaptation of the weights to the output layer, effectively
74 splitting the RNN into a reservoir and a readout layer. The outputs weights
75 are trained and the internal weights are only globally scaled up or down a
76 bit (Schrauwen et al., 2007). This lead to a learning rule for RNNs called
77 BPDC. Here too, sigmoidal neurons are used, but a significant difference
78 between BPDC reservoirs and ESNs is the fact that feedback connections
79 from the readout layer into the reservoir and into the readout layer itself are
80 used, whereas in practice this is hardly ever the case for ESNs (Verstraeten
81 et al., 2007). As for the medical classification task under scrutiny these
82 feedback connections are not needed, ESNs were used instead of BPDC in
83 this research.

84 More information about the different reservoir computing methods and
85 their various properties and application domains can be found in Verstraeten

86 et al. (2007); Jaeger et al. (2007); Lukoševičius and Jaeger (2009).

87 Thus, the ESN was selected as reservoir computing method to handle the
88 medical classification problem studied in this research. The medical time
89 series are also classified using support vector machines (SVM) and the naive
90 Bayes (NB) classifier. This way, we can compare the performance of two
91 traditional classifiers - a sophisticated and a simple one - and the recent
92 classifier based on ESN.

93 Although medical data are often time series, little medical applications
94 of ESN have been studied yet. To our knowledge, apart from this study, of
95 which a preliminary report has been published which focusses on the clinical
96 aspect of the study (Verplancke et al., 2010), ESN have been applied to two
97 other medical use cases. An abstract reported the classification of autistic
98 and normal children (Noris et al., 2008) and a study described the detection
99 of epileptic seizures on rat data using reservoir computing (Buteneers et al.,
100 2008, 2011).

101 In time-oriented medical studies, longitudinal data analysis is a popu-
102 lar approach. However, this is only suitable for relatively short time series
103 - typically up to 10 measurements per input parameter - since longitudinal
104 data analysis focuses on the correlation of measurements within a time series,
105 which diminishes when the time series grows and measurements lie further
106 apart in time (Zeger et al., 2006). Another approach is repeatedly perform-
107 ing data analysis only in a very small interval or individual points in time.
108 However, this neglects the temporal nature of the data almost completely.

109 The remainder of this paper is structured as follows. The application data
110 is described in Section 2. In Section 3 the classification, feature extraction

111 and selection, and performance evaluation methods used in this study are
112 briefly introduced. Section 4 then summarizes the experimental setup, after
113 which the results are presented in Section 5. These are discussed in Section 6
114 after which a conclusion is formulated in Section 7.

115 **2. Application data**

116 Since we want to explore the advantages of the use of echo state networks
117 in this study, a simple problem is selected. That is, a problem that is easily
118 solved by an expert in the field. This way, we are sure that the required in-
119 formation to solve the problem is contained in the data and that the acquired
120 result is the outcome of the used method, not the used data.

121 In collaboration with the ICU department of the Ghent University we
122 selected the problem of predicting whether or not a patient will need dialysis
123 between five and ten days after admission in the ICU. The prediction is made
124 at hour 72 after submission, so only the diuresis and creatinine values of the
125 first three days after ICU admission were retrieved from the ICU database
126 for each patient included in the study. The study population consisted of an
127 observational cohort of 916 patients admitted consecutively to the ICU be-
128 tween May 31st 2003 and November 17th 2007. These patients were selected
129 from a total of 9752 medical and surgical ICU (MICU/SICU) patients admit-
130 ted in this period after application of inclusion/exclusion criteria. Namely,
131 8725 patients with a length of stay in the ICU of less than 10 days and 111
132 patients who received dialysis in the first five days of ICU admission were
133 excluded from analysis.

134 Diuresis is measured in 2 hour intervals, while creatinine is measured one,

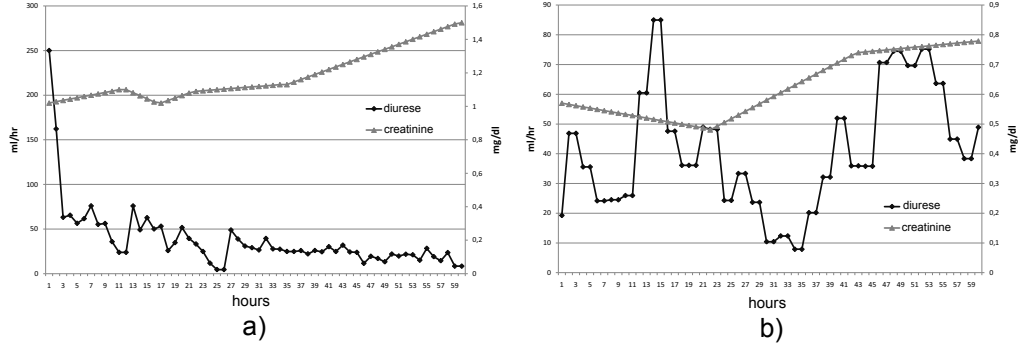


Figure 2: Interpolated creatinine and diuresis measurements for an example patient who
a) needed dialysis between five and ten days after ICU admission b) who did not need
dialysis.

two or exceptionally three times a day. These measurements are performed
by hand, so there exists some variance in the intervals between succeeding
measurements. Also the interval between creatinine measurements is larger
than the one between diuresis measurements. However, the input time series
needs to contain measurements over regular time intervals and these intervals
must be the same for both input parameters. Therefore linear interpolation
of the data is the very first preprocessing step.

The availability of both diuresis and creatinine measurements does not
fully overlap. Measurements not within the overlapping interval are excluded
from the data. Patients who do not have an overlapping interval of minimal
40 measurements are excluded from the study. After pre-processing, 830
patients are available with 60 interpolated measurements for both creatinine
and diuresis. Figure 2 visualizes these interpolated creatinine and diuresis
measurements, expressed as milliliter/hour (ml/hr), for two patients. The
patient in Figure 2a needed dialysis between five and ten days after ICU

admission and the patient in Figure 2b did not. The interval between these measurements is 1 hour, so the data consists of a 60 hour period somewhere in the first 3 days of the patient’s stay in the ICU. 62% of the patients were male and the mean age of the study population was 58.6 years. The selected population had a total mortality rate of 17% and the mean Simplified Acute Physiology Score (SAPS) II score was 37.2. 82/830 (9.9%) patients needed dialysis between the fifth and tenth day after admission, while the remaining 748/830 (90.1%) patients did not need dialysis during that period.

3. Classifiers

In this section we discuss the feature extraction and selection methods and the classifiers under study. The time series are classified using support vector machines, the naive Bayes classifier and echo state networks. Prior to the use of SVM and the NB classifier, feature extraction and selection is required to reduce the number of input features and thus alleviate the effect of curse of dimensionality (Bowden G.J., 2005). This way, we can compare the performance of two traditional classifiers - a sophisticated and a simple one - and the recent classifier based on ESN.

3.1. Feature extraction and selection

Classical classification techniques, such as the SVM and NB classifier, have been designed with a static data model in mind and are not suitable for coping with the dynamic nature of time series. The performance of the SVM and NB classifiers suffers from a large number of features if not all the features are of the same type and of equal importance (Bowden G.J., 2005). This is the case in the medical problem addressed in this research as it consists of

174 two types of features, namely diuresis and creatinine values. 60 interpolated
175 measurements for both diuresis and creatinine are used as features. Not all
176 these measurements are equally important as expert opinion reveals that the
177 tails of the time series, i.e., later measurements, contain more information
178 than the start of the series.

179 An inclusion of a large number of features in the SVM and NB classifiers
180 leads to “the curse of dimensionality” (Bowden G.J., 2005; Muttill and Chau,
181 2007), which is associated with the following shortcomings:

- 182 • As the input dimensionality increases, the computational complexity
183 and memory requirements of the model increase, which in turn increases
184 the time to build the models.
- 185 • As the input variables increase, the number of training samples required
186 also increase.
- 187 • Misconvergence and poor model accuracy may result from the inclusion
188 of irrelevant inputs due to an increase in the number of local minima
189 present in the error surface.
- 190 • Interpreting complex models is more difficult than interpreting simple
191 models that give comparable results.

192 Feature extraction, which generates additional features from the time
193 series, and feature selection, which selects the most appropriate features and
194 thus reduces the amount of input features, helps to improve the performance
195 of learning models by (Guyon and Elisseeff, 2003):

- 196 • Alleviating the effect of the curse of dimensionality

- 197 • Enhancing generalization capability
- 198 • Speeding up the learning process and
- 199 • Improving model interpretability.

200 To make sure that all the information contained in the time series is
201 captured, extensive feature extraction is applied for the SVM and the NB
202 classifier. Features are therefore extracted that capture the overall properties
203 of the time series and the correlation between the different measurements
204 in the time series. For each time series the minimum, maximum, mean,
205 median, 25th percentile, 75th percentile, standard deviation (stdev), the linear
206 regression ($y = ax + b$) coefficients a and b and the area under the curve
207 (AUC) are calculated. This results in 10 features per time serie.

208 As mentioned previously, expert opinion reveals that the tails of the time
209 series contain more information than the start of the series. We therefore
210 repeat the feature extraction multiple times for reduced time series. The
211 10 features are extracted for the full time series, the 59 last values of the
212 time series, the 58 last values of the time series, ..., and the 2 last values
213 of the time series. This results in $59 * 10 = 590$ extracted features per
214 input parameter, or 1180 extracted features in total. Finally we add the
215 measurements themselves to the extracted feature set as well, which results
216 in $1180 + 2 * 60 = 1300$ features.

217 Feature selection needs to be performed on these 1300 features to select
218 the most useful ones for the NB and SVM classifiers. Ideally, a brute-force
219 search is performed in which the classification performance of each possible
220 combination of features is tested and the best combination is selected. Brute-

force feature selection is however very resource-intensive. As the number of possible feature combinations for 1300 features is nearly endless, namely $(2^{1300} - 1)$ possible combinations, the required computation time would be virtually infinite.

To boost the performance, a greedy feature selection algorithm is used which iteratively adds the feature that improves prediction the best out of a set of features that show little multicollinearity with the already selected set of features. This approach is similar to the one used by Langley and Sage (1994), but in each iteration we filter the set of candidates so that it contains only features that are not collinear with the already selected set. This drastically reduces the size of the set of candidate features in each iteration and therefore speeds up the feature selection process. Detection of multicollinearity is done using the common rule of thumb: *variance inflation factor* > 5 (Kutner et al., 2004). The classifier used in this hybrid filter-wrapper method (Guyon and Elisseeff, 2003) is the NB classifier.

All data is globally scaled to the $[-0.9, 0.9]$ interval. Scaling features to a fixed interval is necessary to avoid favoring a feature only because it has the largest scale. The bounds -0.9 and 0.9 are chosen instead of -1 and 1 to avoid excessive weight saturation in the recurrent artificial neural network.

3.2. Support vector machines

As first discussed by Cortes and Vapnik (1995), a SVM tries to separate positive and negative examples in a multi-dimensional space by a hyperplane.

Assume that the training data is labeled as $\{\mathbf{x}_i, y_i\}, i = 1, \dots, l, y_i \in \{-1, 1\}, \mathbf{x}_i \in \mathbf{R}^d$. The points \mathbf{x} that lie on the hyperplane satisfy the equation

246 $\mathbf{w} \cdot \mathbf{x} + b = 0$, where \mathbf{w} is normal to the hyperplane. d_+ and d_- are the shortest
 247 distances from the separating hyperplane to the closest positive and negative
 248 example. The margin of the separating hyperplane is then defined as $d_+ + d_-$.

249 The SVM discussed by Cortes and Vapnik (1995) was a linear classifier.
 250 For the linearly separable case, the SVM searches for the hyperplane that
 251 separates the data from the two classes with maximal margin (Vapnik, 1995).
 252 This search can be formulated as an optimization problem, where

$$\sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j \quad (1)$$

253 is maximized, subject to

$$\sum_i \alpha_i y_i = 0, \text{ for } \alpha_i \geq 0 \quad (2)$$

254 with α_i being the Lagrangian multipliers for each training example. Given
 255 the α_i , the solution is given by

$$\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i \quad (3)$$

256 The examples for which $\alpha_i > 0$ are called support vectors. All other example
 257 have $\alpha_i = 0$.

258 When the positive and negative examples are not linearly separable, an
 259 additional condition needs to be added:

$$0 \leq \alpha_i \leq C \quad (4)$$

260 This gives the α_i an upper bound of C .

261 Switching to the non-linear case can be done by using the kernel-trick (Aiz-
 262 erman et al., 1964). Notice that the data appears in the training problems,

263 see Equation (1), only in the form of dot products $\mathbf{x}_i \cdot \mathbf{x}_j$. If the data is
 264 mapped to some other Euclidian space H , using a mapping $\Phi : \mathbf{R}^d \mapsto H$, the
 265 training problem can be solved in H by replacing $\mathbf{x}_i \cdot \mathbf{x}_j$ by $\Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$. If
 266 there is a kernel function K such that $K(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$, then only K
 267 needs to be used in the training algorithm and it never needs to be explicitly
 268 known what Φ is. An example of such a kernel function and the one which
 269 was used in this study is the Radial Basis Function (RBF) kernel function.
 270 Rüping (2001) showed that the RBF kernel performs very well on different
 271 types of time series and learning tasks. The RBF kernel function has the
 272 following definition:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|^2) \quad (5)$$

273 This results in a training algorithm with only two parameters, namely C and
 274 γ . For a more detailed introduction to SVMs, we refer to Burges (1998).
 275 SVMs have been successfully applied to perform time series prediction and
 276 prediction on real problems in different engineering fields (Lin et al., 2006;
 277 Rüping, 2001; Kampouraki et al., 2009; Zhang et al., 2010).

278 The libSVM (Chang and Lin, 2012) support vector machine implemen-
 279 tation is used in this study. The C and γ parameters were optimized using
 280 parameter sweeps during each experiment, as is further detailed in Section 4.

281 3.3. Naive Bayes classifier

282 The Naive Bayes Classifier is based on the application of Bayes' theorem,
 283 which relates the conditional and marginal probabilities of events A and B :

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \quad (6)$$

284 where $P(A)$ is the prior probability of A , $P(B)$ is the prior probability of
 285 B , $P(A|B)$ is the posterior probability of A and $P(B|A)$ is the posterior
 286 probability B .

287 A custom Java implementation of the Naive Bayes classifier is used in this
 288 study. This Naive Bayes classifier estimates the prior probability of class A
 289 as

$$P(A) \approx \frac{\text{\#items of class A in the training set}}{\text{total\#items in the training set}} \quad (7)$$

290 When a previously unseen example X is presented to the classifier, the like-
 291 lihood of class A is estimated as

$$Li(A) \approx \frac{\text{\#items of class A in the training set in the neighborhood of X}}{\text{total\#items of class A in the training set}} \quad (8)$$

292 Assuming that each feature is conditionally independent of every other fea-
 293 ture, the posterior probability that a previously unseen example X belongs
 294 to class A can be estimated as

$$P(X = A) \approx P(A) \times Li(A) \quad (9)$$

295 The number of examples in the training set that constitute the neighbor-
 296 hood of a previously unseen sample X , denoted by parameter k , is the only
 297 configurable parameter of the used Naive Bayes implementation. Parameter
 298 sweeps were performed to determine the optimal value for k per experiment,
 299 as is further detailed in Section 4.

300 It can be noted that the Naive Bayes classifier is based on applying Bayes'
 301 theorem with strong independence assumptions. However, empirical results
 302 show that it performs surprisingly well in many domains containing clear fea-
 303 ture dependencies (Domingos and Pazzani, 1997). Zhang (2004) shows that

the feature dependence distribution plays a crucial role in the explanation of this behavior and that sufficient and necessary conditions for the optimality of Naive Bayes can be formulated.

3.4. *Echo state networks*

The key idea in reservoir computing (Verstraeten et al., 2007) is to feed time series to a reservoir, thereby modeling the dynamics of the system which generates the time series. The reservoir is then read by a readout function in order to make predictions using the constructed model. When training the model, only the readout function is modified, the complex dynamic modeling behavior of the reservoir is left unchanged.

In ESN (Jaeger, 2001), the reservoir consists of a recurrent artificial neural network with sigmoid activation functions and the *echo state property* which ensures good modeling abilities. A recurrent artificial neural network is said to have the echo state property when its state is uniquely determined by the input time series. This implies the *state forgetting property*: the initial state of the reservoir has no impact on the state after feeding a - possibly infinite - time series. Although it is not yet clearly understood how it exactly works, the reservoir acts as a short-term fading memory (Jaeger, 2002a), which means in practical applications that the most recent input of the network has the largest impact on the prediction outcome. The readout function used in ESN is a linear classifier.

The general layout of an ESN is illustrated in Figure 1. It consists of k input nodes, n reservoir nodes, and l output nodes. Each node is a perceptron with a sigmoid activation function. The state of each node at a given time

328 is the weighted sum of the last fed inputs, namely

$$\mathbf{x}[t+1] = (1 - \mu)\mathbf{x}[t] + \mu f(\mathbf{W}\mathbf{x}[t] + \mathbf{W}^{in}\mathbf{u}[t]) \quad (10)$$

329 where $\mathbf{x}[t]$ denotes the network state at time t and \mathbf{u} is the input matrix.
 330 Leaky integrator neurons are used to optimize the leak rate μ of the reservoir
 331 so that it can perfectly match the timescale of the input data. For every
 332 sample, $\mathbf{x}[0]$ is initialized as 0. The weights in the ESN are represented in
 333 weight matrices. The $k \times n$ matrix \mathbf{W}^{in} contains the weights between the
 334 input and reservoir nodes and the $n \times n$ matrix \mathbf{W} contains the recurrent
 335 weights between the reservoir nodes. The spectral radius λ_{\max} is defined as
 336 the largest absolute eigenvalue of the matrix \mathbf{W} . It has been shown that
 337 reservoirs whose spectral radius is larger than one ($|\lambda_{\max}| > 1$) do not have
 338 the echo state property, but in practice the spectral radius is chosen close
 339 to one to achieve a suitable dynamic response (Jaeger, 2001). Zero weights
 340 are the equivalent of the absence of connections. Feedback connections from
 341 output nodes to reservoir nodes and connections from input nodes directly
 342 to output nodes are optional.

343 By using Equation (10) the echo state network can be recursively simu-
 344 lated using the training data D_{train} . After each sample of the training data
 345 is simulated, the $|D_{train}|$ reservoir state matrices are concatenated in a large
 346 state matrix \mathbf{A} . Because an ESN is a dynamical system, it takes some time
 347 before the full effects of the input are visible in the reservoir states. Therefore,
 348 the initial states containing the transient effects are discarded which is known
 349 as warm-up drop. The number of states that is discarded is determined by
 350 the warm-up drop parameter α .

351 Different methods can then be used to train the linear readout function,

352 and thus to determine the elements of the $(k + n + l) \times l$ output weight
 353 matrix \mathbf{W}^{out} , which contains the weights between the reservoir nodes and
 354 the output nodes. A complete overview and discussion of the different avail-
 355 able techniques reported in literature for training the readout function of
 356 the reservoir can be found in Lukoševičius and Jaeger (2009). As the med-
 357 ical problem under study does not require on-line adaptation of the model,
 358 batch learning can be performed. In batch mode, the most recommended
 359 and used method is ridge or Tikhonov Regression (Wyffels et al., 2008), as it
 360 has the lowest computational cost, while still allowing to perform regulariza-
 361 tion. Ridge regression introduces a regularization parameter λ . In addition
 362 to improving the numerical stability, the regularization in effect reduces the
 363 magnitudes of entries in \mathbf{W}^{out} , thus mitigating sensitivity to noise and over-
 364 fitting. However, because Fisher weighting is also used in this study to deal
 365 with the unbalanced data set, as further explained in the last two paragraphs
 366 of this section, ridge regression could not be used as this combination is not
 367 implemented in the Reservoir Computing Toolbox (RCToolbox) (Verstraeten
 368 and Wardermann, 2012). In this study, the RCToolbox is used to run the
 369 ESN experiments. However, using ridge regression is equivalent with using
 370 least squares regression (Björck, 1996) with noise. So, in this study, \mathbf{W}^{out}
 371 is trained by performing least squares regression on the matrix \mathbf{A} , using the
 372 desired output matrix \mathbf{y} as the right-hand side. Thus, the matrix \mathbf{W}^{out} is
 373 computed that satisfies the equation:

$$\mathbf{W}^{out} = \min_{\mathbf{W}} \|\mathbf{A} \times \mathbf{W} - \mathbf{y}\|^2. \quad (11)$$

374 In practice, this equation can be computed in a single step by using the
 375 Moore-Penrose generalized matrix inverse, or pseudo-inverse, of the matrix

376 \mathbf{A} (Penrose, 1955). This provides least squares regression with a similar nu-
 377 merical stability as ridge regression. Gaussian noise is added to the matrix
 378 \mathbf{A} in order to control the trade-off between model complexity and generaliza-
 379 tion capability (avoid overfitting). This guarantees that the model is complex
 380 enough to accurately model the underlying system, but not too complex such
 381 that it becomes sensitive to the noise in the samples. Similar to ridge re-
 382 gression, the amount of noise is determined by a regularization parameter
 383 λ .

384 Other methods that are sometimes used in literature to train the linear
 385 readout function are weighted regression and evolutionary search (Jiang et al.,
 386 2008). The first uses weights to emphasize some time steps t over others. As
 387 this study wanted to evaluate how well the ESN performed on the time series
 388 without using domain expert knowledge, this method was not used. State-
 389 of-art evolutionary methods are able to achieve the same level of precision for
 390 supervised tasks as with the best application of linear regression. However,
 391 their computational cost is much higher.

392 Finally, the output of the reservoir can be computed as follows:

$$\hat{\mathbf{y}}[k] = \mathbf{W}^{out} \mathbf{x}[k] \quad (12)$$

393 where $\hat{\mathbf{y}}$ is the actual output of the reservoir system.

394 As mentioned previously, the RCToolbox is used to run the ESN experi-
 395 ments. As the original time series, and thus not the extracted features, are
 396 used as input for the ESN, a reservoir with $k = 2$ input nodes and $l = 1$
 397 output nodes is initialized. The elements of the input weight matrix \mathbf{W}^{in} are
 398 drawn from the discrete set $\{-0.1, 0.1\}$ with equal probabilities. The density
 399 of the input weight matrix is 10%, which means that 10% of the weights are

400 non-zero. The elements of the reservoir weight matrix \mathbf{W} are drawn from
 401 a Gaussian distribution. The density D of the reservoir weight matrix is
 402 chosen as $d = 20\%$. The optimal value for the regularization parameter λ is
 403 determined by performing a brute-force grid search of the parameter space
 404 with cross-validation.

405 Output-to-output connections are not used. Input-to-output connections
 406 are used to enable a direct linear mapping of the input.

407 The RCToolbox allows performing parameter sweeps to find the optimal
 408 values for the various parameters of an ESN, namely the leak rate μ , the
 409 number of reservoir nodes n , the spectral radius λ_{\max} and the warm-up drop
 410 parameter α . These optimal values are found by performing a sensitivity
 411 analysis for each parameter. This means that the values for this parameter
 412 are varied while all other parameter settings of the ESN are left unchanged.
 413 The parameter value which results in the best average performance of the
 414 ESN is chosen.

415 The sensitivity analysis of the leak rate μ is visualized in Figure 3. This
 416 figure shows the observed average performance and its standard deviation in
 417 30 runs for leak rate values between $\mu = 0.01$ and $\mu = 1$. Higher performance
 418 values are better. For a more detailed explanation of the performance mea-
 419 sure, see Section 3.5. Different runs with the same settings result in different
 420 performance results because the data is randomly divided among the folds
 421 and the reservoir is randomly initialized. In theory, performance should not
 422 depend on these random circumstances. In practice, the dependence should
 423 be minimized. For example because of the limited amount of available data,
 424 there will always be a certain amount of dependence on how exactly the data

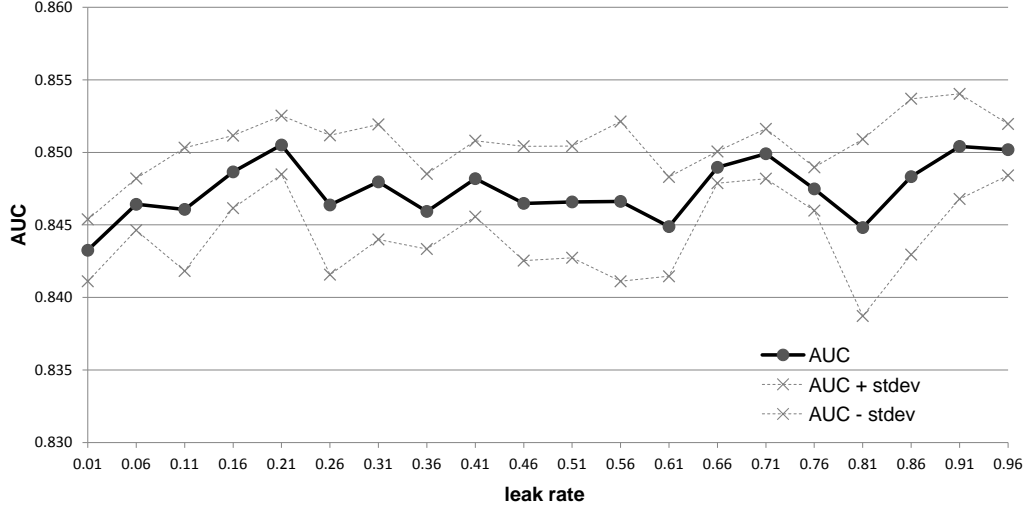


Figure 3: Sensitivity analysis of the leak rate of the reservoir. Dots and crosses are measured values. Lines are interpolated values. The area under the receiver operating characteristic (ROC) curve (AUC) is a performance measure. The solid line is the average performance in 30 runs. The dotted line denotes the observed standard deviation.

is divided among the folds. In this problem setting, the best average performance and the smallest deviation in performance is aimed at. From Figure 3 it is clear that adjusting the leak rate does not boost the performance of the ESN significantly. Likely this is due to the fact that in this case, the optimal parameters of the ESN are outside the usual range: for the high total input to the reservoir used here, the reservoir acts more like a static kernel rather than a dynamical system. As a consequence, the leak rate is chosen to be the value $\mu = 0.01$. This is the default value for the leak rate in the RCToolbox. This means that the reservoir will work very slowly, implementing a low-pass filter.

The sensitivity analyses of the number of reservoir nodes n and the spec-

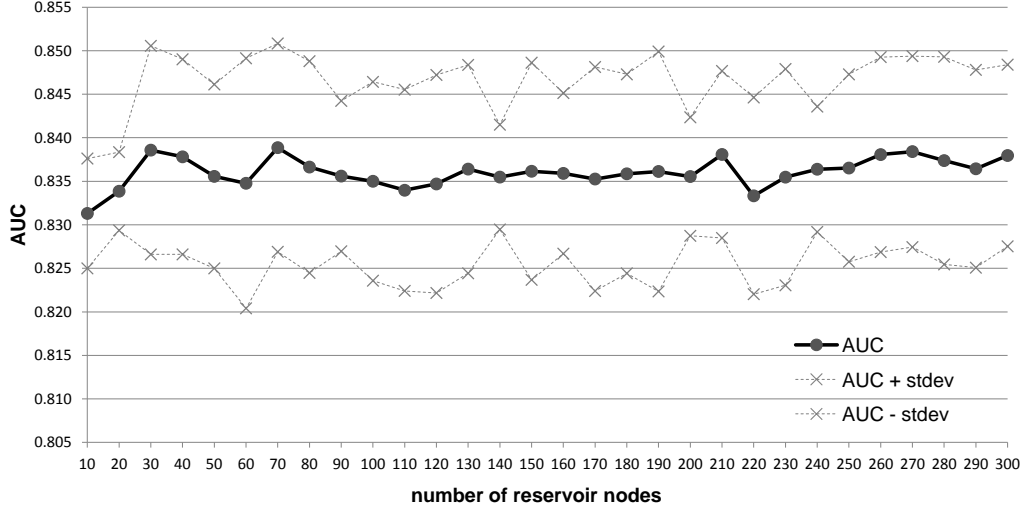


Figure 4: Sensitivity analysis of the number of reservoir nodes. Dots and crosses are measured values. Lines are interpolated values. The area under the receiver operating characteristic (ROC) curve (AUC) is a performance measure. The solid line is the average performance in 30 runs. The dotted line denotes the observed standard deviation.

436 tral radius λ_{\max} are shown in Figures 4 and 5. These figures show the ob-
 437 served average performance and its standard deviation in 30 runs for number
 438 of reservoir nodes between $n = 10$ and $n = 300$ and for spectral radius val-
 439 ues between $\lambda_{\max} = 0.1$ and $\lambda_{\max} = 1.5$. From Figures 4 and 5 it can be
 440 derived that adjusting the number of reservoir nodes n or the spectral radius
 441 λ_{\max} also had minimal effects on the performance of the ESN. As mentioned
 442 previously, a spectral radius close to one should be chosen to achieve a suit-
 443 able dynamic response and to guarantee that the echo state property holds.
 444 Therefore, the spectral radius is chosen to be the value $\lambda_{\max} = 0.99$. The
 445 weights are rescaled so that the spectral radius λ_{\max} is set to this value. The
 446 number of reservoir nodes was chosen to be $n = 70$, as this was the parameter

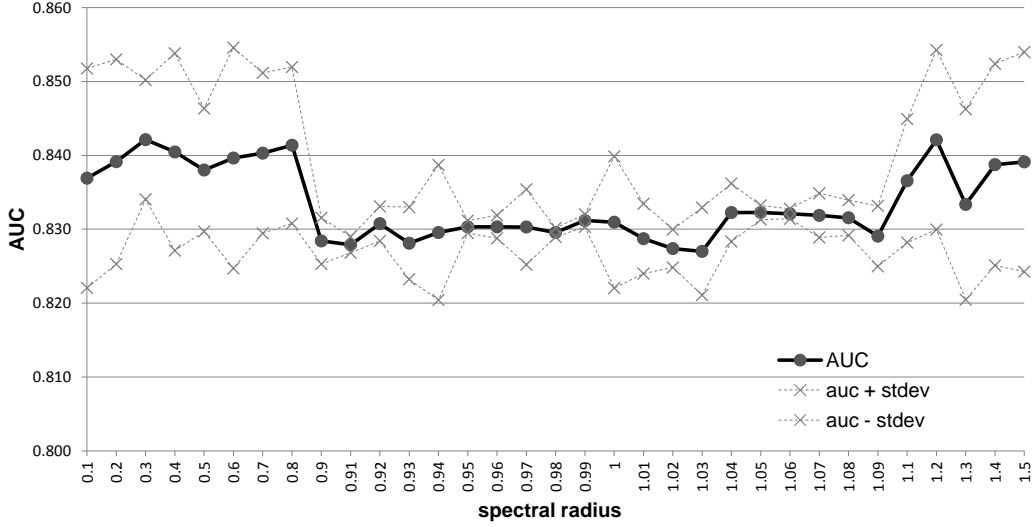


Figure 5: Sensitivity analysis of the spectral radius. Dots and crosses are measured values. Lines are interpolated values. The area under the receiver operating characteristic (ROC) curve (AUC) is a performance measure. The solid line is the average performance in 30 runs. The dotted line denotes the observed standard deviation.

value with the highest average AUC across all the runs.

Finally, the warm-up drop parameter α is optimized by performing a sensitivity analysis. The observed average performance and its standard deviation in 30 runs for warm-up drop values between $\alpha = 0$ (no warm-up drop) and $\alpha = 59$ (only the last time point remains) are plotted in Figure 6. From Figure 6 it is clear that adjusting the warm-up drop parameter significantly boosts the performance of the ESN. A warm-up drop of $\alpha = 56$ first time steps of the time series leads to the best performance results. This corresponds with the opinion of the domain experts that the tail of the time series contains more information than the start of the series.

As can be noted from Section 2, the dataset is unbalanced. There are a

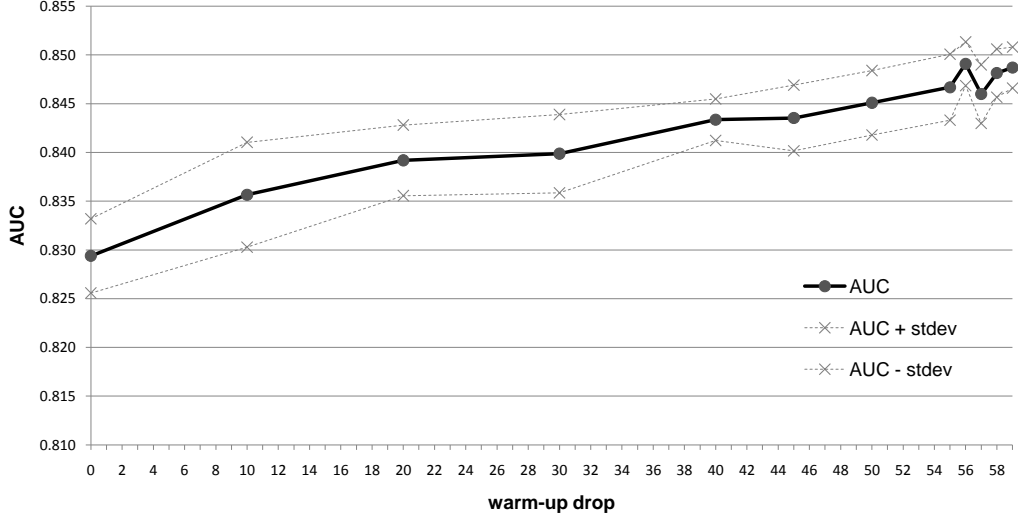


Figure 6: Sensitivity analysis of the warm-up drop parameter of the echo state network. Dots and crosses are measured values. Lines are interpolated values. The area under the receiver operating characteristic (ROC) curve (AUC) is a performance measure. The solid line is the average performance in 30 runs. The dotted line denotes the observed standard deviation.

lot more examples of patients who did not receive dialysis between the fifth and tenth day after admission than there are patients that did (748 vs. 82 of the 830 patients). This unbalance will have an effect on the generalization capabilities of the classifiers. Since the read-out is trained using regression, the separating hyperplane will shift towards the class centers that are most present in the dataset (the threshold will not be zero). This is undesirable as one wants the hyperplane to lie in the middle between the two classes (threshold equal to zero). To achieve this, Fisher labeling is applied (Duda et al., 2001).

Assume, that the positive class has n_1 examples and the negative class

has n_2 examples, then Fisher labeling relabels these classes from the usual $[-1, 1]$ for positive and negative examples respectively to $[(n_1 + n_2)/n_1, (n_1 + n_2)/n_2]$. In this way, the class labels reflect the unbalance of the number of examples in each class. This guarantees that the shifting of the hyperplane is undone. Thus for this dataset, the Fisher labeling relabels the classes to $[830/82, 830/748]$.

3.5. Performance evaluation

Each of the 3 used methods outputs a prediction score. The SVM and the NB output a prediction score per sample. The ESN, on the other hand, outputs a prediction score per time point in the time series. As the warm-up drop parameter α is set to 56, only 4 time points remain and thus 4 prediction scores are outputted by the ESN per sample. These are summarized to one prediction score per sample by taken the mean of these 4 values. In contrast with a categorical prediction - class A versus class B - a prediction score is a value $x \in \mathbb{R}$ in the interval $] - \infty, +\infty[$. The sign of x corresponds to a class while the magnitude of x reflects the estimated probability of actually belonging to that class. By varying the prediction threshold, different classifiers can be constructed. These classifiers vary from one that classifies all patients into one class to one that classifies all patients into the other class.

The correctness of a classification can be evaluated by computing the number of true positives (TP , positive examples classified as positive), true negatives (TN , negative examples classified as negative), false positives (FP , negative examples classified as positive), and false negatives (FN , positive example classified as negative) respectively. The most often used measures for binary classification based on these values are Accuracy, Precision, Sen-

493 sitivity (Recall), Specificity, F-Score and the area under the ROC curve
494 (AUC) (Sokolova and Lapalme, 2009). These measures differ in their ability
495 to preserve their value under a change of the number TP , TN , FP , and/or
496 FN . A measure is invariant if its value does not change when one or more
497 of the TP , TN , FP , or FN values change. This inability can be beneficial
498 or adverse, depending on the goal of the classification task. More informa-
499 tion about the different performance measures for classification can be found
500 in Sokolova and Lapalme (2009).

501 For the medical problem under scrutiny, we are interested in the overall
502 performance of the classifier, i.e., interested in the performance of the clas-
503 sifier on both identifying and correctly classifying positive and negative ex-
504 amples. In other words, it is equally important to correctly identify whether
505 a patient will receive dialysis or not between five and ten days after admis-
506 sion in the ICU. Precision, Recall and F-Score are invariant to changes in
507 the number of TN . These measures thus do not acknowledge the ability of
508 the classifiers to correctly identify negative examples. In contrast, Specificity
509 is invariant to changes in the number of TP . This measure thus does not
510 acknowledge the ability of the classifiers to correctly identify positive exam-
511 ples. Consequently, two measures remain that are non-invariant to changes
512 to the number of TN and TP , namely AUC and accuracy. However, the
513 accuracy is invariant to the distribution of classification results because it
514 does not distinguish TP from TN and FN from FP . This measure is thus
515 not trustworthy when using unbalanced data sets. The AUC is non-invariant
516 to the distribution of classification results, which makes it a good measure
517 for comparing classifiers on unbalanced data sets, such as the one used in

518 this study.

519 The *AUC* is calculated based on the Specificity and Sensitivity perfor-
520 mance measures of the classifier. Sensitivity measures the proportion of
521 actual positive examples, i.e., patients needing dialysis, which are correctly
522 identified by the classifier as follows:

$$\text{Sensitivity} = \frac{TP}{TP + FN}. \quad (13)$$

523 In contrast, Specificity measures the proportion of actual negative examples,
524 which are correctly identified by the classifier as follows:

$$\text{Specificity} = \frac{TN}{TN + FP}. \quad (14)$$

525 Plotting *Sensitivity* versus $(1 - \text{Specificity})$ for all these classifiers, results in
526 the so called receiver operating characteristic (ROC) curve (Zweig and Camp-
527 bell, 1993). The area under this ROC curve (*AUC*) is an estimation of the
528 probability that a positive patient receives a higher prediction score than a
529 negative patient by the classification method under study. An *AUC* value
530 of 1.0 indicates a classifier that perfectly separates positives from negatives,
531 while a classifier that randomly classifies patients as positive or negative cor-
532 responds to $AUC = 0.5$. All other classifiers will result in $0.5 < AUC < 1.0$.

533 A two-sample t-test is used to determine whether an observed difference
534 in *AUC* is random or real. A *p*-value expresses the probability of having a test
535 statistic at least as extreme as the one that was actually observed, assuming
536 that the null-hypothesis is true. The lower the *p*-value, the less likely the
537 result, and consequently the more statistically significant the result is. A
538 result is statistically significant if it is unlikely that it occurred by chance.

539 Generally, the null hypothesis is rejected if the p-value is smaller than or
 540 equal to the significance level, α .

541 In this paper, the test statistics are the average *AUCs* across the 30 runs
 542 for each classifier. The null-hypothesis in these tests is that both average
 543 *AUCs* are equal. The significance level α is chosen to be 0.05, which expresses
 544 that results that are only 5% likely or less are deemed extraordinary, given
 545 that the null hypothesis is true.

546 Since we test 3 average *AUCs* for equality, the significance level α must
 547 be corrected for multiple testing. This can be done by applying Dunn-Šidák
 548 correction (Abdi, 2007), that is

$$\alpha_{\text{cor}} = 1 - (1 - \alpha)^{1/C}, \quad (15)$$

549 where α is the chosen significance level, α_{cor} is the corrected α -value, and C
 550 is the number of tests. The null-hypothesis in this test is that both average
 551 *AUCs* are equal. Thus the corrected significance level, with whom the p-
 552 values are compared, is

$$\alpha_{\text{cor}} = 1 - (1 - 0.05)^{1/3} = 0.016952. \quad (16)$$

553 When choosing a prediction threshold, we can select the value where
 554 the balanced accuracy of the classifier is the highest. We define balanced
 555 accuracy as follows:

$$\text{balanced accuracy} = \frac{\textit{Sensitivity} + \textit{Specificity}}{2}. \quad (17)$$

556 Using maximum balanced accuracy prevents favoring a classifier that always
 557 outputs the majority class in the case of heavily unbalanced data sets such
 558 as the one used in this study. If the classifier performs equally well on either

class, this term reduces to the conventional accuracy, i.e., the number of correct predictions divided by the total number of predictions. In contrast, if the conventional accuracy is above chance only because the classifier takes advantage of an imbalanced test set, then the maximum balanced accuracy will drop to chance.

As can be seen, the *AUC* gives us a global view on the quality of the constructed classifiers, while the maximum balanced accuracy is an indication of the best prediction accuracy we can expect. Moreover, the *AUC* is well-known and much used performance measure of binary classification tasks within the medical domain (Sokolova et al., 2006). Both the *AUC* and maximum balanced accuracy are invariant to a uniform change of positive and negative examples in the data set. This means that these measures are stable with the respect to the uniform increase of the data size. As in our medical problem, the proportion of representatives for the positive and negative class will remain stable across different data sizes, these measures are a good choice. We will also look at the required execution time, which is a measure for the computational complexity of the methods under study.

4. Problem setting

To summarize, we compare the classification performance of 3 methods on the given problem. The performance measures are *AUC* and maximum balanced accuracy, which are determined for each of the methods using cross-validation. The computational complexity of the methods is compared through their required execution times. All these tests were performed on the same machine - Advanced Micro Devices (AMD) Athlon 64 X 2 Dual

583 Core Processor, 3000 megahertz (MHz) Central Processing Unit (CPU), 2
584 Gigabyte (GB) of Random-Access Memory (RAM) - under exactly the same
585 conditions.

586 The input data consists of 2 time series per patient. Each time series
587 consists of 60 linear interpolated values, which are constructed out of the
588 original patient data. For the ESN method, no further preprocessing of the
589 data is necessary. Prior to the use of SVM and the NB classifier, feature
590 extraction and selection and global rescaling of the data is required.

591 Several parts of the algorithms under study have a stochastic nature.
592 Examples are the random division of the available data into folds and the
593 random initialization of the reservoir weights in the ESN. To avoid faulty
594 interpretation of results that origin from a coincidental odd configuration,
595 the experiments are repeated 30 times, each time using another random
596 initialization.

597 The pre-processing phase of the SVM and NB classifier, consisting of
598 the feature extraction and selection process and global rescaling of the data,
599 is also subject to random factors, for example, the random division of the
600 available data into folds. Moreover, there are several multicollinear features.
601 In each iteration of the feature selection, the set of candidates is filtered so
602 that it contains only features that are not collinear with the already selected
603 set. Which of the multicollinear features thus ends up in the selected set is
604 also subject to the random initialization of the feature selection. Therefore,
605 this pre-processing phase is also repeated for each run of the SVM and NB
606 classifier.

607 Consequently, the data set that is used as input for the NB and SVM

608 classifiers is different in every run. To determine the optimal values for the
 609 parameter k of the NB classifier and parameters C and γ of the SVM classifier
 610 parameter sweeps thus need to be performed for each of the 30 runs. For
 611 each parameter, the value is selected that achieves the highest performance
 612 for the classifier in that run. Consequently, different parameter values are
 613 obtained for the NB and SVM classifiers in each run. The optimal value of
 614 the parameter k of the NB classifier across the 30 runs ranges from $k = 29$
 615 to $k = 47$ and is on average $k = 40$. The optimal value of the parameters
 616 C and γ of the SVM classifier across the 30 runs range from $C = -4.12$ to
 617 $C = 23.65$ and $\gamma = -22.05$ to $\gamma = -10.57$ and are on average $C = 17.66$ and
 618 $\gamma = -17.58$.

619 **5. Results**

620 Table 1 and Table 2 show respectively the observed AUC and maximum
 621 balanced accuracy performance measures. The best maximum balanced ac-
 622 curacy and best AUC achieved across the 30 runs for each classification
 623 method are shown as well as the average value and its accompanying stan-
 624 dard deviation (stdev) and Confidence Intervals (CI) at 95% and 99%. The
 625 performance measures for the ESN classifier are shown for both the configu-
 626 ration for which all the parameter values of the ESN were optimized through
 627 parameter sweeps and the default configuration which uses the default set-
 628 tings of the RCToolbox for the ESN. The default settings are a reservoir size
 629 $n = 100$, a leak rate $\mu = 0.01$, a scale factor $\lambda_{\max} = 0.9$ and a warm-up
 630 drop $\alpha = 0$. Table 3 contains the p -values that are obtained while testing
 631 the average AUC s for equality.

Table 1: Observed area under the curve (AUC) in 30 runs using 3 different classification methods: the echo state network (ESN), the support vector machine (SVM) and the naive Bayes classifier (NB). The latter two are preceded by a pre-processing phase, consisting of the feature extraction (FE) and feature selection (FS) process and global rescaling of the data.

	best	average	stdev	CI 95%	CI 99%
ESN - optimized	<i>0.854</i>	<i>0.849</i>	<i>0.002</i>	<i>0.001</i>	<i>0.001</i>
ESN - default	<i>0.804</i>	<i>0.799</i>	<i>0.003</i>	<i>0.001</i>	<i>0.001</i>
SVM + FE + FS	<i>0.857</i>	<i>0.838</i>	<i>0.021</i>	<i>0.007</i>	<i>0.010</i>
NB + FE + FS	<i>0.885</i>	<i>0.874</i>	<i>0.006</i>	<i>0.002</i>	<i>0.003</i>

Table 2: Observed maximum balanced accuracy in 30 runs using 3 different classification methods: the echo state network (ESN), the support vector machine (SVM), and the naive Bayes classifier (NB). The latter two are preceded by a pre-processing phase, consisting of the feature extraction (FE) and feature selection (FS) process and global rescaling of the data.

	best	average	stdev	CI 95%	CI 99%
ESN - optimized	<i>0.803</i>	<i>0.795</i>	<i>0.002</i>	<i>0.001</i>	<i>0.001</i>
ESN - default	<i>0.746</i>	<i>0.742</i>	<i>0.003</i>	<i>0.001</i>	<i>0.001</i>
SVM + FE + FS	<i>0.812</i>	<i>0.784</i>	<i>0.019</i>	<i>0.007</i>	<i>0.009</i>
NB + FE + FS	<i>0.826</i>	<i>0.809</i>	<i>0.009</i>	<i>0.003</i>	<i>0.004</i>

632 Figure 7 shows the obtained ROC curves in run 1. The obtained ROC
633 curves in the other runs are very similar.

Table 3: P -values resulting from the tests for equality between the AUC s.

	SVM + FE + FS	NB + FE + FS
ESN - optimized	<i>0.0097</i>	<i>< 0.001</i>
SVM + FE + FS		<i>< 0.001</i>

Table 4: Required computation time for the data pre-processing phase for the support vector machine (SVM), the naive Bayes classifier (NB) and the echo state network (ESN). SVM and NB share the same pre-processing phase, consisting of the feature extraction (FE) and feature selection (FS) process and global rescaling of the data.

	average	stdev	CI 95%	CI 99%
ESN	<i>253.87ms</i>	<i>6.86ms</i>	<i>2.45ms</i>	<i>3.22ms</i>
SVM & NB	<i>3h 59m 55s 245.93ms</i>	<i>47m 6s 299.70ms</i>	<i>16m 51s 359.77ms</i>	<i>22m 9s 152.05ms</i>

634 As Table 4 shows, the pre-processing phase preceding the support vector
635 machines and naive Bayes classifier approach, which includes the loading
636 and interpolating the data and performing feature extraction and selection,
637 requires on average 3 hours (h) 59 minutes (m) 55 seconds (s) and 245.93
638 milliseconds (ms) of computation time. The pre-processing phase for the
639 recurrent reservoir, which only includes loading and interpolating the data
640 as no feature extraction and selection is needed, requires on average only
641 253.87 ms of computation time.

642 Table 5 shows the computation time needed to train the three classifiers.
643 The reported train time includes finding the optimal value for the size of
644 the neighborhood k , see Equation (8), for the NB classifier, for the C and γ
645 parameters, see Equations (4) and (5), of the SVM classifier and the regular-
646 ization parameter λ of the ESN classifier with default configuration. To reach

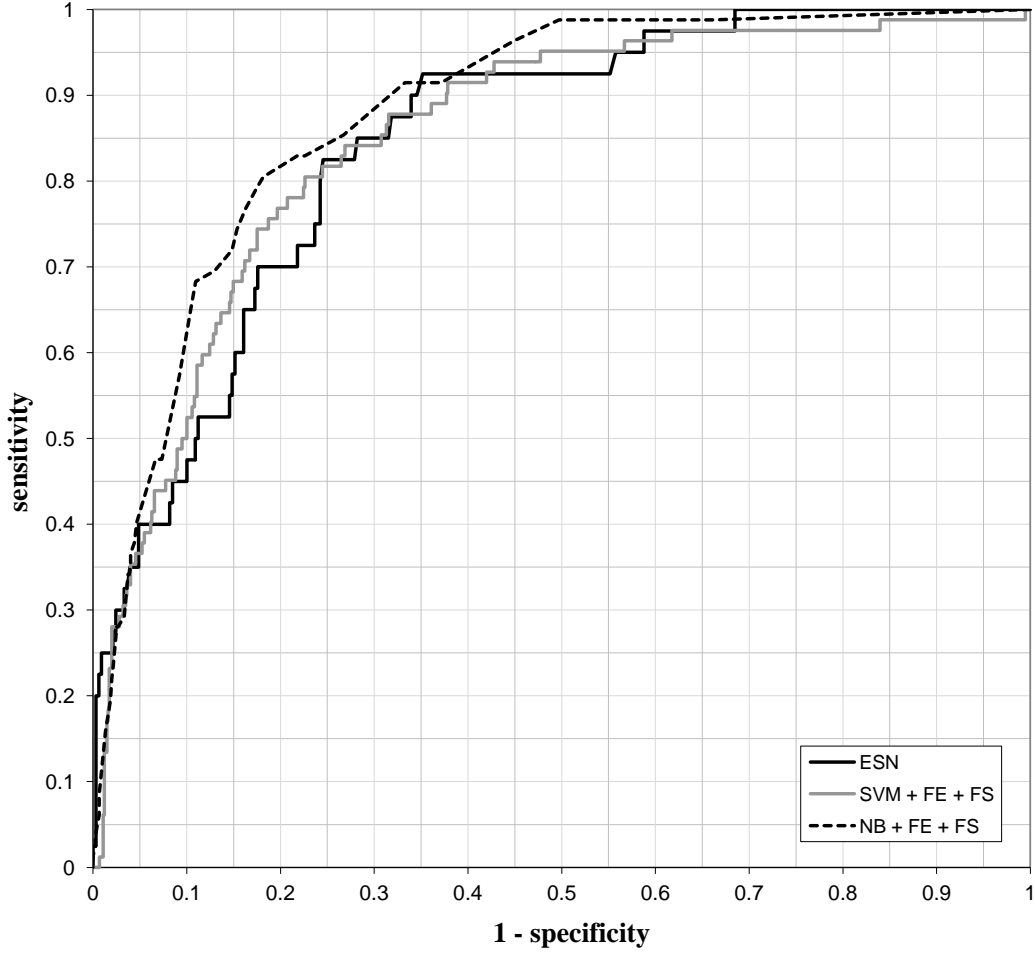


Figure 7: The obtained ROC curves in run 2 for the echo state network (ESN), the support vector machine (SVM), and the naive Bayes classifier (NB). The latter two are preceded by a pre-processing phase, consisting of the feature extraction (FE) and feature selection (FS) process and global rescaling of the data.

647 the performance results of the Optimized ESN classifier, parameter sweeps
648 need to be performed. The train time for performing one parameter sweep of
649 the reservoir size, leak rate, scale factor or warm-up drop parameters of the
650 ESN are also reported. Performing one sweep means that this parameter is

Table 5: Required train time for the support vector machine (SVM), the naive Bayes classifier (NB) and the echo state network (ESN).

	average	stdev	CI 95%	CI 99%
ESN - default	6m 35s 209.35ms	1s 247.55ms	446.42ms	586.70ms
SVM	14m 19s 936.23ms	5m 37s 934.51ms	2m 0s 926.09ms	2m 38s 923.82ms
NB	35m 37s 258.40ms	39s 154.80ms	14s 11.11ms	18s 413.72ms
ESN - 1 parameter sweep	6m 35s 117.86ms	2s 647.05ms	947.22ms	1s 244.86ms

Table 6: Required test time for the support vector machine (SVM), the naive Bayes classifier (NB) and the echo state network (ESN).

	average	stdev	CI 95%	CI 99%
ESN	0.030ms	0.009ms	0.003ms	0.004ms
SVM	0.033ms	0.183ms	0.065ms	0.086ms
NB	0.300ms	0.466ms	0.167ms	0.219ms

set to 1 value (e.g. reservoir size = 300) and the ESN is trained. In practice, mainly the warm-up drop parameter needed to be swept to obtain the improved performance results of the Optimized ESN classifier.

Finally, Table 6 visualizes the computation time needed to test the three classifiers with data about one patient.

6. Discussion

The p -values comparing the naive Bayes (NB) classifier combined with feature extraction (FE) & selection (FS), the support vector machine (SVM) combined with FE & FS and the echo state network (ESN) classifier are

660 smaller than the Dunn-Šidák corrected significance level $\alpha = 0.016952$, see
 661 Equation (16). We therefore conclude that there is a significant difference
 662 between the average *AUC*s of the used methods observed at the 5% level.
 663 This means that the SVM + FE + FS, with an average *AUC* of 0.838, is
 664 the worst classifier. The NB + FE + FS has an average *AUC* of 0.874 and
 665 is thus the best classifier. The ESN classifier lies somewhere in the middle
 666 with an average *AUC* of 0.849. Inspection of Figure 7, which shows the ROC
 667 curves, and Table 2, which shows the observed maximum balanced accuracy,
 668 see Equation (17), confirms this conclusion. However, the results of the
 669 NB classifier combined with FE & FS are biased as the feature selection
 670 method is a hybrid filter-wrapper method which also uses a NB classifier
 671 as classifier. Consequently, features selected by this hybrid filter-wrapper
 672 method are optimal for and best recognized by the NB classifier used in
 673 this feature selection method. If we then again apply a NB classifier on the
 674 selected features, the achieved results are slightly biased towards the NB
 675 classifier, since the selected feature set favors this type of classifier.

676 Based on the observed values of the performance measures we cannot
 677 definitely favor the ESN classifier. The picture changes when we look at
 678 the procedure followed for each method. The SVM and the NB classifier
 679 are designed for datasets where the data resides in an n -dimensional space as
 680 such. The longitudinal correlation along the different dimensions/parameters
 681 is not taken into account in any way. Therefore SVM and NB perform rather
 682 poorly when time series data is used unprocessed. To get satisfying results,
 683 we first must extract useful features based on the time series. This can be
 684 done in an automated way or by using domain knowledge of the problem

685 at hand. Extracting features in an automated way often results in missing
 686 important characteristics of the data, while acquiring domain knowledge is a
 687 time consuming and often cumbersome activity. In this study we used a com-
 688 bined approach, exploiting the time saving properties of automated feature
 689 extraction and limiting the domain knowledge gathering to acquiring general
 690 properties of the data. The latter allows to steer the automated procedure,
 691 which avoids exploring useless regions in the search space. This approach
 692 still results in an enormous amount of candidate features, which makes a
 693 feature selection phase necessary as well. Furthermore, both feature extrac-
 694 tion and feature selection phases combined require a considerable amount of
 695 computation time, namely on average approximately 4 hours (see Table 4).

696 In the ESN approach, no feature extraction and selection is needed. The
 697 reservoir stores features from the input data and actually adds features to it,
 698 as we go from an input space from $k = 2$ dimensions to a reservoir space of
 699 $n = 70$ dimensions. Thus, by putting a reservoir between the input data and
 700 the readout, a lot more features are available to build the estimation on. The
 701 ESN consequently succeeds nicely in modeling the information contained in
 702 the time series data. It therefore needs on average less than a second of pre-
 703 processing time (see Table 4) and no domain knowledge. Additionally, the
 704 reservoir algorithms are easy to implement, and existing rules of thumb suffice
 705 for acquiring a good performing configuration of the reservoir, as can be
 706 noted from the performance of the ESN with default configuration in Table 1.
 707 Moreover, a simple linear regression classification suffices for determining the
 708 final classification results, where complex non-linear methods are required in
 709 the traditional approach.

710 Note that expert opinion states that in the data used the required in-
711 formation is mostly contained in the tail of the time series. This was ex-
712 plicitly taken into account during the pre-processing phase of the SVM and
713 NB classifiers by extracting features from an increasingly shorter time series.
714 Namely, the 10 features were extracted for the full time series, the 59 last
715 values of the time series, the 58 last values of the time series, ..., and the 2
716 last values of the time series. If we study the features, which were selected
717 during the feature selection phase, we see that mainly features of the shorter
718 time series and linear regression coefficients were selected. However, for the
719 ESN classifier this domain knowledge does not need to be taken explicitly
720 into account. The ESN classifier takes it implicitly into account because
721 of the fading short-term memory (Jaeger, 2002a) characteristic of the ESN.
722 This means that the most recent input of the network has the largest impact
723 on the prediction outcome, which matches the domain knowledge that the
724 most important information is contained in the tail of the time series. This
725 explains the successful results.

726 The computation time for training the ESN classifier is also better than
727 the other classifiers, as shown in Table 6. However, additional time is needed
728 to optimize the values of the various parameters of the ESN classifier through
729 parameter sweeps. Optimizing the value of the warm-up drop parameter re-
730 sulted in significant performance improvements. In practice, about 5 sweeps
731 would have to be performed to obtain the optimal value for the warm-up
732 drop parameter. Therefore, the train time for the different classifiers is com-
733 parable.

734 As can be derived from Table 6, the test time of the SVM and ESN is also

comparable. The computation time for testing the NB classifier is slightly higher on average, because the NB classifier takes into account each training sample when calculating the neighborhood of the testing sample. Since the data set used in this study is relatively small, the difference in test time between the NB classifier and the SVM and ESN classifiers is still negligible.

Since the ESN allows complex non-linear modeling in a simpler and computationally much more efficient way compared to the traditional approach while yielding a comparable classification performance, the authors believe that the ESN will play an important role in future analysis of medical time series data.

7. Conclusion

Medical data often consists of time series. This kind of data should be analyzed by specialized methods. The echo state network (ESN) is a recent method that was optimized to handle time series data. ESNs are easy to implement and to use, and do not require that feature extraction and selection is performed on the time series data before using it as input. We show the usefulness of ESN by using it to predict the need for dialysis between the fifth and tenth day after admission in ICU patients, and comparing the results to those acquired by using support vector machines (SVM) and the naive Bayes (NB) classifier combined with feature extraction (FE) and selection (FS). A hybrid filter-wrapper feature selection method is used with an NB classifier as classifier. Performance is measured by the area under the ROC curve and the maximum balanced accuracy.

Limitations of this study are that no extensive comparative study was per-

759 formed between different feature selection methods that could be combined
760 with the SVM and NB classifiers and the lack of comparison of the ESN to
761 other classification methods which can directly process time series. Future
762 work will further investigate these limitations by studying if the choice of the
763 feature selection method significantly improves the performance of the SVM
764 and NB classifiers on this medical classification task. Moreover, the perfor-
765 mance of the ESN will be compared to other reservoir computing methods,
766 such as liquid state machines and backpropagation decorrelation.

767 The results of this study showed statistically significant difference at the
768 5% level between the performance of ESN and the other two methods. The
769 SVM + FE + FS had the worst performance, the NB classifier + FE + FS
770 the best and the performance of the ESN lies in the middle. However, the
771 results of the NB classifier + FE + FS are biased as the feature selection
772 method is a hybrid filter-wrapper method which also uses a NB classifier.
773 Moreover, its simplicity in usage, its ability to model and extract features
774 without the need of domain knowledge, and its limited usage of computing
775 time, make ESN the most suitable method for predicting the need for dialysis
776 when using measured time series as input.

777 Future work will focus on applying the reservoir computing methods on a
778 medical classification task which is not trivial for the medical experts, namely
779 detecting whether a patient who has been admitted to the ICU has sepsis.
780 Sepsis is the number one cause of death in the ICU.

781 8. Acknowledgments

782 Femke Ongenae would like to thank the IWT, Institute for the promotion
783 of Innovation through Science and Technology in Flanders, for supporting
784 this work through her PhD grant.

References

- Abdi, H., 2007. Encyclopedia of Measurement and Statistics. Sage, Thousand Oaks, CA, Ch. The Bonferroni and Šidák corrections for multiple comparisons, pp. 1–9.
- Aizerman, M., Braverman, E., Rozonoer, L., 1964. Theoretical foundations of the potential function method in pattern recognition learning. Automation and Remote Control 25, 821–837.
- Björck, A., 1996. Numerical Method for Least Squares Problems. SIAM, Philadelphia, PA, USA.
- Bowden G.J., Dandy G.C., M. H., 2005. Input determination for neural network models in water resources applications. Part 1 - Background and methodology. Journal of Hydrology 301, 75–92.
- Burges, C., 1998. A tutorial on support vector machines for pattern recognition. Data Mining and Knowledge Discovery journal 2, 121 – 167.
- Buteneers, P., Schrauwen, B., Verstraeten, D., Stroobandt, D., 2008. Real-time epileptic seizure detection on intra-cranial rat data using reservoir computing. In: Köppen, M., Kasabov, N., Coghill, G. (Eds.), 15th International Conference on Neural Information Processing of the Asia-Pacific

- Neural Network Assembly (ICONIP 2008). Springer-Verlag, Berlin, pp. 56–63.
- Buteneers, P., Verstraeten, D., van Mierlo, P., Wyckhuys, T., Stroobandt, D., Raedt, R., Hallez, H., Schrauwen, B., 2011. Automatic detection of epileptic seizures on the intra-cranial electroencephalogram of rats using reservoir computing. *Artificial Intelligence in Medicine* 53 (3), 215–223.
- Chang, C., Lin, C., 2012. LIBSVM: a library for support vector machines. Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm> (Published: 2001, Accessed: 25 July 2012).
- Cortes, C., Vapnik, V., 1995. Support-vector networks. *Machine Learning* 20 (3), 273–297.
- Domingos, P., Pazzani, M., 1997. On the optimality of the simple bayesian classifier under zero-one loss. *Machine Learning* 29, 103–130.
- Duda, R., Hart, P., Stork, D., 2001. *Pattern Classification* (2nd ed.). Wiley Interscience, New York, NY, USA, Ch. Relation to Fisher’s Linear Discriminant, p. 654.
- Fisher, R., 1925. *Statistical methods for research workers*. Oliver and Boyd, Edinburgh.
- Gerstner, W., Kistler, W. M., 2002. *Spiking neuron models: Single Neurons, Populations, Plasticity*. Cambridge University Press, Cambridge, UK.
- Graves, A., Schmidhuber, J., 2005. Framewise phoneme classification with

- bidirectional LSTM and other neural network architectures. *Neural Networks* 18 (5-6), 602–610.
- Guyon, I., Elisseeff, A., 2003. An introduction to variable and feature selection. *Journal of Machine Learning Research* 3, 1157–1182.
- Haykin, S., 1994. *Neural networks: a comprehensive foundation*. Prentice Hall, New Jersey.
- Iso, K., Watanabe, T., 1991. Speaker-independent speech recognition using a neural prediction model. *Electronics and Communications in Japan (Part III: Fundamental Electronic Science)* 74 (8), 22–30.
- Jaeger, H., 2001. The “echo state” approach to analysing and training recurrent neural networks. Tech. Rep. GMD 148, German National Research Institute for Computer Science.
- Jaeger, H., 2002a. Short term memory in echo state networks. Tech. Rep. GMD 152, German National Research Institute for Computer Science.
- Jaeger, H., 2002b. A tutorial on training recurrent neural networks, covering BPTT, RTRL, EKF, and the “echo state network”. Tech. Rep. GMD 159, German National Research Institute for Computer Science.
- Jaeger, H., 2003. Adaptive nonlinear system identification with echo state networks. In: Becker, S., Thrun, S., Obermayer, K. (Eds.), *Advances in Neural Information Processing Systems (NIPS)*. MIT Press, Cambridge, MA, pp. 593–600.

- Jaeger, H., Haas, H., 2004. Harnessing nonlinearity: predicting chaotic systems and saving energy in wireless communication. *Science* 304 (5667), 78–80.
- Jaeger, H., Maass, W., Principe, J., 2007. Special issue on echo state networks and liquid state machines: Editorial. *Neural Networks* 20 (3), 287–289.
- Jiang, F., Berry, H., Schoenauer, M., 2008. Supervised and evolutionary learning of echo state networks. In: Rudolph, G., Jansen, T., Lucas, S., Poloni, C., Beume, N. (Eds.), 10th International Conference on Parallel Problem Solving from Nature (PPSN). Springer-Verlag, Berlin, Heidelberg, pp. 215–224.
- Kampouraki, A., Manis, G., Nikou, C., 2009. Heartbeat time series classification with support vector machines. *IEEE Transactions on Information Technology in Biomedicine* 13 (4), 512–518.
- Kutner, M., Neter, J., Nachtsheim, C., Li, W., 2004. Applied Linear Regression Models. McGraw-Hill, New York.
- Langley, P., Sage, S., 1994. Induction of selective bayesian classifiers. In: de Mántaras, R., Poole, D. (Eds.), Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence (UAI-94). Morgan Kaufmann, San Mateo, pp. 399–406.
- Levin, E., 1993. Hidden control neural architecture modeling of nonlinear time varying systems and its applications. *IEEE Transactions on Neural Networks* 4 (1), 109–116.

- Lin, J.-Y., Cheng, C.-T., Chau, K.-W., 2006. Using support vector machines for long-term discharge prediction. *Hydrological Sciences Journal* 51 (4), 599–612.
- Lukoševičius, M., Jaeger, H., 2009. Reservoir computing approaches to recurrent neural network training. *Computer Science Review* 3 (3), 127–149.
- Maass, W., Bishop, C., 2001. Pulsed neural networks. Bradford Books/MIT Press, Cambridge, MA, USA.
- Maass, W., Natschläger, T., Markram, H., 2002. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Computation* 14 (11), 2531–2560.
- Maass, W., Natschläger, T., Markram, H., 2003. A model for real-time computation in generic neural microcircuits. In: Becker, S., Thrun, S., Obermayer, K. (Eds.), *Advances in Neural Information Processing Systems* (NIPS). MIT Press, Cambridge, MA, pp. 213–220.
- Maass, W., Natschläger, T., Markram, H., 2004. Computational Neuroscience: A Comprehensive Approach. CRC-Press, Boca Raton, Florida, USA, Ch. Computational models for generic cortical microcircuits, pp. 575–605.
- Muttil, N., Chau, K.-W., 2007. Machine learning paradigms for selecting ecologically significant input variables. *Engineering Applications of Artificial Intelligence* 20 (6), 735–744.
- Natschläger, T., Markram, H., Maass, W., 2002. *A Practical Guide to Neuroscience Databases and Associated Tools*. Kluwer Academic Publishers,

- Boston, Ch. 9: Computer models and analysis tools for neural microcircuits, pp. 123–128.
- Noris, B., Nobile, M., Piccini, L., Berti, M., Mani, E., Molteni, M., et al., 2008. Gait analysis of autistic children with echo state networks. *Parkinsonism & Related Disorders* 14 (Suppl. 1), S70.
- Penrose, R., 1955. A generalized inverse for matrices. *Mathematical proceedings of the Cambridge Philosophical Society* 51, 406–413.
- Rabiner, L., 1989. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE* 77 (2), 257–286.
- Robinson, A., 1994. An application of recurrent nets to phone probability estimation. *IEEE Transactions on Neural Networks* 5 (2), 298–305.
- Rüping, S., 2001. SVM kernels for time series analysis. In: Klinkenberg, R., Rüping, S., Fick, A., Henze, N., Herzog, C., Molitor, R., Schröder, O. (Eds.), *Tagungsband der GI-Workshop-Woche Lernen - Lehren - Wissen - Adaptivität (LLWA-01)*. University of Dortmund, Dortmund, Germany, pp. 43–50.
- Schiller, U. D., Steil, J. J., 2005. Analyzing the weight dynamics of recurrent learning algorithms. *Neurocomputing* 63, 5–23.
- Schrauwen, B., Verstraeten, D., Campenhout, J. V., 2007. An overview of reservoir computing: theory, applications and implementations. In: *15th European Symposium on Artificial Neural Networks (ESANN 2007)*. pp. 471–482.

- Sokolova, M., Japkowicz, N., Szpakowicz, S., 2006. Beyond Accuracy, F-Score and ROC: A family of discriminant measures for performance evaluation. In: Sattar, A., Kang, B.-h. (Eds.), 19th Australian joint conference on Artificial Intelligence: advances in Artificial Intelligence (AI'06). Springer-Verlag, Berlin, Heidelberg, pp. 1015–1021.
- Sokolova, M., Lapalme, G., 2009. A systematic analysis of performance measures for classification tasks. *Information Processing and Management* 45 (4), 427–437.
- Steil, J., 2006. Online stability of backpropagation-decorrelation recurrent learning. *Neurocomputing* 69 (7-9), 642–650.
- Tebelskis, J., Waibel, A., Petek, B., Schmidbauer, O., 1990. Continuous speech recognition by linked predictive neural networks. In: Lippmann, R. P., Moody, J. E., Touretzky, D. S. (Eds.), *Proceedings of the conference on Advances in Neural Information Processing Systems (NIPS)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 199–205.
- Trentin, E., Gori, M., 2003. Robust combination of neural networks and hidden markov models for speech recognition. *Neural Networks* 14 (6), 1519–1531.
- Vapnik, V., 1995. *The nature of statistical learning theory*. Springer-Verlag, Berlin.
- Verplancke, T., Van Looy, S., Steurbaut, K., Benoit, T., De Turck, F., De Moor, G., et al., 2010. A novel time series analysis approach for pre-

- diction of dialysis in critically ill patients using echo-state networks. *BMC Medical Informatics and Decision Making* 4.
- Verstraeten, D., Schrauwen, B., D’Haene, M., Stroobandt, D., 2007. An experimental unification of reservoir computing methods. *Neural Networks* 20 (3), 414–423.
- Verstraeten, D., Wardermann, M., 2012. The Reservoir Computing Toolbox v2.0. Software available at <http://snn.elis.ugent.be/rctoolbox> (Published: 2009, Accessed: 25 July 2012).
- Wyffels, F., Schrauwen, B., Dirk, S., 2008. Stable output feedback in reservoir computing using ridge regression. In: Kurkova-Pohlova, V., Koutnik, J. (Eds.), 18th International Conference on Artificial Neural Networks (ICANN 2008). Springer-Verlag, Berlin, Heidelberg, pp. 807–818.
- Xie, J.-X., Cheng, C.-T., Chau, K.-W., Pei, Y.-Z., 2006. A hybrid adaptive time-delay neural network model for multi-step-ahead prediction of sunspot activity. *International Journal of Environment and Pollution* 28 (3-4), 364–381.
- Zeger, S., Irizarry, R., Peng, R., 2006. On time series analysis of public health and biomedical data. *Annual Review of Public Health* 27, 57–79.
- Zhang, D., Zuo, W., Zhang, D., Zhang, H., 2010. Time series classification using support vector machine with Gaussian elastic metric kernel. In: 20th International Conference on Pattern Recognition (ICPR). IEEE Computer Society, Piscataway, NJ, USA, pp. 29–32.

- Zhang, H., 2004. The optimality of naive bayes. In: Barr, V., Zdravko, M. (Eds.), Proceedings of the 17th International Florida Artificial Intelligence Research Society Conference. AAAI Press, Menlo Park, CA, pp. 562–567.
- Zweig, M., Campbell, G., 1993. Receiver-operating characteristic (ROC) plots: a fundamental evaluation tool in clinical medicine. *Clinical chemistry* 39 (4), 561–577.