Data Mining with Multilayer Perceptrons (MLPs)

Paulo Cortez
pcortez@dsi.uminho.pt
http://www.dsi.uminho.pt/~pcortez
Departamento de Sistemas de Informação
University of Minho - Campus de Azurém
4800-058 Guimarães - Portugal
With the advances in information technology, there has been an ever-increasing load of data in organizations (massive datasets are commonplace);

All this data (often with high complexity) holds valuable information;

Human experts are limited and may overlook relevant details;

Moreover, classical statistical analysis breaks down when such vast and/or complex data are present;

A better alternative is to use automated discovery tools to analyze the raw data and extract high-level information for the decision maker [Hand et al., 2001];
Knowledge Discovery in Databases (KDD)

“the overall process of discovering useful knowledge from data”.

Data Mining (DM)

“application of algorithms for extracting patterns (models) from data” (a particular step of the KDD process).
KDD consists in a series of iterative steps:

- Understanding the application domain;
- Acquiring or selecting a target data set;
- Data cleaning, preprocessing and transformation;
- Choosing the DM goals, learning algorithm and searching for patterns of interest;
- Result interpretation and verification; and
- Using and maintaining the discovered knowledge.
DM goals

- **classification** – labeling a data item into one of several predefined classes (e.g. diagnosing a disease according to patient's symptoms);
- **regression** – mapping a set of attributes into a real-value variable (e.g. stock market prediction);
- **clustering** – searching for natural groupings of objects based on similarity measures (e.g. segmenting clients of a database marketing study); and
- **link analysis** – identifying useful associations between data (e.g. “64% of the shoppers who bought milk also purchased bread”).
Feedforward neural network where each node outputs an activation function applied over the weighted sum of its inputs:

\[ s_i = f(w_{i,0} + \sum_{j \in I} w_{i,j} \times s_j) \]
Activation functions

- **Linear**: $y = x$
- **Tanh**: $y = \tanh(x)$
- **Logistic (or sigmoid)**: $y = \frac{1}{1+e^{-x}}$
Architecture/Topology

- Only **feedforward** connections exist;
- Nodes are organized in **layers**;

![Diagram of a feedforward neural network with layers and shortcuts]
Why Data Mining with MLPs? [Sarle, 2005]

- **Popularity** - the most used Neural Network, with several off-the-shelf packages available;

- **Universal Approximators** - general-purpose models, with a huge number of applications (e.g. classification, regression, forecasting, control or reinforcement learning);

- **Nonlinearity** - when compared to other data mining techniques (e.g. decision trees) MLPs often present a higher predictive accuracy;

- **Robustness** - good at ignoring irrelevant inputs and noise;

- **Adaptability** - can adapt its weights and/or topology in response to environment changes;

- **Explanatory Knowledge** - It is possible to extract rules from trained MLPs.
Commercial:

- SAS Enterprise Miner Software (www.sas.com);
- Clementine (http://www.spss.com/spssbi/clementine/);
- MATLAB Neural Network Toolbox (www.mathworks.com/products/neuralnet);
- STATISTICA: Neural Networks (www.statsoft.com);

Free:

- WEKA (data mining, Java source code available) (www.cs.waikato.ac.nz/~ml/weka);
- R (statistical tool) (www.r-project.org);

Build Your Own!

Better control, but take caution!
Supervised Learning – input/output mapping (e.g. classification or regression):

- **Data Collection** - learning samples must be representative, hundred/thousand of examples are required;
- **Feature Selection** - what are the relevant inputs?
- **Preprocessing** - data transformation, dealing with missing data, outliers, ...;
- **Modeling** – network design, training and performance assessment;
- **Prediction** – feed the trained MLP with new data and interpret the output;
- **Explanatory Knowledge** – input importance by sensitivity analysis and extraction of rules from trained MLPs;
Feature Selection

- Selection of the subset of relevant features. Why?
  - To reduce storage and measurement requirements;
  - To facilitate data visualization/comprehension;
  - Non relevant features/attributes will increase the MLP complexity and worst performances will be achieved.

Feature Selection methods [Blum and Langley, 1997]:

- A priori knowledge (e.g. the use of experts);
- Correlation analysis (only measures linear effects);
- Trial-and-error blind search (e.g. test some subsets and select the subset with the best performance);
- Hill-climbing search (e.g. sensitivity analysis, forward and backward selection);
- Beam search (e.g. genetic algorithms);
Handling Missing Data (?, NA, ...)[Brown and Kros, 2003]:

- Use complete data only (delete cases or variables);
- **Data Imputation**, substitute by:
  - Value given by an expert (**case substitution**);
  - Mean, median or mode;
  - Value from another database (**cold deck**);
  - Value of most similar example (**hot deck**);
  - Value estimated by a regression model (e.g. linear regression);
  - Combination of previous methods (**multiple imputation**);
Outliers

- Due to errors in data collection or rare events;
- Not related with the target variable, they prejudice the learning;
- Solution: use of experts, data visualization, statistical analysis, ...
Nonnumerical variable remapping [Pyle, 1999]

- Only numeric data can be fed into MLPs;
- **Binary** attributes can be coded into 2 values (e.g. \{-1, 1\} or \{0, 1\});
- **Ordered** attributes should be encoded by preserving the order (e.g. \{low \rightarrow -1, medium \rightarrow 0, high \rightarrow 1\});
- **Nominal** (non-ordered with 3 or more classes) attributes:
  - **1-of-C** remapping – use one binary variable per class (generic);
  - **M-of-C** remapping – requires domain knowledge (e.g. a state can be coded into 2 variables, the horizontal and vertical position in a 2D map);
Attribute **color** = \{Red, Blue, Green\};

With the linear mapping \{Red → -1, Blue → 0, Green → 1\} it is impossible to describe \(X\), which is half green and half red;

With the **1-of-C** mapping \{ Red → 1 0 0, Blue → 0 1 0, Green → 0 0 1 \}, \(X\) could be represented by: 0.5 0 0.5;
Rescaling/Normalization [Sarle, 2005]

- MLP learning improves if all **inputs** are rescaled into the same range with a 0 mean:
  - \[ y = \frac{x - (\text{max} + \text{min})/2}{(\text{max} - \text{min})/2} \] (linear scaling with range [-1,1])
  - \[ y = \frac{x - \overline{x}}{s} \] (standardization with mean 0 and standard deviation 1)
- **Outputs** limited to the [0,1] range if logistic ([−1,1] if tanh).
  - \[ y = \frac{(x - \text{min})}{\text{max} - \text{min}} \] (linear scaling with range [0,1])
Classification Metrics

Confusion matrix [Kohavi and Provost, 1998]

- Matches the **predicted** and **actual** values;
- The $2 \times 2$ confusion matrix:

<table>
<thead>
<tr>
<th></th>
<th>predicted</th>
<th>negative</th>
<th>positive</th>
</tr>
</thead>
<tbody>
<tr>
<td>actual\</td>
<td></td>
<td>TN</td>
<td>FP</td>
</tr>
<tr>
<td>negative</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>positive</td>
<td>FN</td>
<td></td>
<td>TP</td>
</tr>
</tbody>
</table>

Three accuracy measures can be defined:

- the **Sensitivity** (*Type II Error*) $= \frac{TP}{FN+TP} \times 100 \%$;
- the **Specificity** (*Type I Error*) $= \frac{TN}{TN+FP} \times 100 \%$;
- the **Accuracy** $= \frac{TN+TP}{TN+FP+FN+TP} \times 100 \%$;
Receiver Operating Characteristic (ROC)

- Shows the behavior of a 2 class classifier when varying a decision parameter $D$;
- The curve plots $1-\text{Specificity}$ ($x$-axis) vs the Sensitivity;
- Global performance measured by the **Area Under the Curve (AUC)**: 
  \[ AUC = \int_0^1 \text{ROC} dD; \]
- The perfect AUC value is 1.0;
The error $e$ is given by: $e = d - \hat{d}$ where $d$ denotes the desired value and the $\hat{d}$ estimated value (given by the model);
Given a dataset with the function pairs $x_1 \rightarrow d_1, \cdots, x_N \rightarrow d_N$, we can compute:

### Error metrics

- **Mean Absolute Deviation (MAD):**
  \[
  MAD = \frac{\sum_{i=1}^{N} |e_i|}{N}
  \]

- **Sum Squared Error (SSE):**
  \[
  SSE = \sum_{i=1}^{N} e_i^2
  \]

- **Mean Squared Error (MSE):**
  \[
  MSE = \frac{SSE}{N}
  \]

- **Root Mean Squared Error (RMSE):**
  \[
  RMSE = \sqrt{MSE}
  \]

- **Relative Root Mean Squared (RRMSE, scale independent):**
  \[
  RRMSE = RMSE / \sqrt{\sum_{i=1}^{N} (d_i - \bar{d}_i)^2 / N \times 100 \%}
  \]

- **Normalized Mean Square Error (NMSE, by variance, scale independent):**
  \[
  NMSE = SSE / (var(d) \times N) \times 100 \%
  \]
  where \( var(d) \) denotes the variance.
Holdout
Split the data into two exclusive sets, using random sampling:
- **training**: used to set the MLP weights (2/3);
- **test**: used to infer the MLP performance (1/3).

K-fold, works as above but uses rotation:
- data is split into K exclusive folds of equal size;
- each part is used as a test set, being the rest used for training;
- the overall performance is measured by averaging the K runs;
- **10-fold** most used if hundreds of examples;
### Gradient-descent [Riedmiller, 1994]:

- **Backpropagation (BP)** - most used, yet slow;
- **Backpropagation with Momentum** - faster than BP, requires additional parameter tuning;
- **QuickProp** - faster than BP with Momentum;
- **RPROP** - faster than QuickProp and stable in terms of its internal parameters;

### Evolutionary Computation [Rocha et al., 2003]

- May overcome local minima problems;
- Can be applied when no gradient information is available (reinforcement learning);
The MLP weights are randomly initialized within small ranges ([-1,1] or [-0.7;0.7]);

Thus, each training may converge to a different (local) minima;

**Solutions**

- Use of *multiple* trainings, selecting the *MLP* with lowest error;
- Use of *multiple* trainings, computing the average error of the MLPs;
- Use of *ensembles*, where the final output is given as the average of the MLPs;
If possible use **large datasets**: $N \gg p$ (weights);

**Model Selection**

Apply several models and then choose the best generalization MLP;

**Regularization**

Use learning penalties or restrictions:
- **Early stopping** - stop when the validation error arises;
- **Weight decay** - in each epoch slightly decrease the weights;
**MLP Capabilities**

**Linear learning when:**
- there are no hidden layers; or
- only linear activation functions are used.

**Nonlinear learning:**
- Any continuous function mapping can be learned with one hidden layer;
- Complex discontinuous functions can be learned with more hidden layers;
Some MLP design rules

Output nodes:
It is better to perform only one classification/regression task per network; i.e., use C/1 output node(s).

Activation Functions:
- Hidden Nodes: use the logistic (or tanh);
- Output Nodes: if outputs bounded, apply the same function; else use the linear function;
### Blind Search

- Only tests a small number of alternatives.
- Examples: **Grid-Search/Trial-and-error** procedures.

### Hill-climbing

- Only one solution is tested at a given time.
- Sensitivity to local minima.
- Examples: **Constructive** and **Pruning** methods.

### Beam search

- Uses a population of solutions.
- Performs global multi-point search.
- Examples: **Evolutionary Computation (EC)**.
In DM, besides obtaining a high predictive performance, it is also important to provide **explanatory knowledge**: what has the model learned?

### Measuring Input Importance [Kewley et al., 2000]

- Use of sensitivity analysis, measured as the variance ($V_a$) produced in the output ($y$) when the input attribute ($a$) is moved through its entire range:

$$V_a = \sum_{i=1}^{L} (y_i - \bar{y})/(L - 1)$$

$$R_a = V_a / \sum_{j=1}^{A} V_j$$

- $A$ denotes the number of input attributes and $R_a$ the relative importance of the $a$ attribute;
- The $y_i$ output is obtained by holding all input variables at their average values; the exception is $x_a$, which varies through its range with $L$ levels;
Extraction of rules from trained MLPs [Tickle et al., 1998]

Two main approaches:

- **Decompositional** algorithms start at the minimum level of granularity: first, rules are extracted from each individual neuron (hidden and output); then, the subsets of rules are aggregated to form a global relationship.

- **Pedagogical** techniques extract the direct relationships between the inputs and outputs of the MLP; By using a black-box point of view, less computation is required and a simpler set of rules may be achieved.
SVMs present **theoretical advantages** (e.g. absence of local minima) over MLPs and comparative studies have reported **better predictive performances**.

Yet:

- Few data mining packages with SVM algorithms are available;
- SVM algorithms require more computational effort;
- Under reasonable assumptions, MLPs require the search of one parameter (hidden nodes $\in \{0, \ldots, 100, \ldots\}$ or the decay $\lambda \in [0, 1]$) while SVMs require two ($C \in [0, \infty]$ and $\gamma \in [0, \infty]$);
- MLPs can be applied in real-time, control & reinforcement or dynamic/changing environments;
MLPs vs Other DM Techniques Example: Zip Code Data
[Hastie et al., 2001]

- Digit recognition task: classification of handwritten numerals;
- 60,000 samples scanned from U.S. Postal envelopes with different sizes and orientations;
- 16x16 gray scale images → 256 pixels (inputs);

Best error rates (1-Accuracy):

- 1.1% – Tangent distance with a 1-nearest neighbor classifier;
- 0.8% – Degree-9 polynomial SVM;
- 0.8% – LeNet5 (a complex MLP with shared weights);
- 0.7% – Boosted LeNet-4 (predecessor of LeNet5);
Organ Failure Diagnosis [Silva et al., 2005]

- In **Intensive Care Units (ICUs)**, scoring the severity of illness has become a routine in daily practice;
- Yet, prognostic models (**logistic regression**) are static (computed with data collected within the first 24 hours);
- Organ dysfunction is a critical **ICU** task: its rapid detection allows physicians to quickly respond with therapy; multiple organ failure will increase the death probability;

Sequential Organ Failure Assessment (SOFA)

- Costly and time consuming diary index (from 0 to 4);
- An organ is considered to fail when $SOFA \geq 3$. 
Aim

- Use **MLPs** for failure prediction (identified by high SOFA values) of six organs: respiratory, coagulation, liver, cardiovascular, central nervous system and renal.
- Several approaches will be tested, using different **feature selection**, **preprocessing** and **modeling** configurations;
- A particular focus will be given to the use of **daily intermediate adverse events**, obtained from four hourly bedside measurements.

Clinical Data

- Part of the **EURICUS II** database;
- Encompasses 5355 patients from 42 **ICUs** and 9 EU countries, during 10 months;
- One entry per day (with a total of 30570).
## Organ Failure main attributes

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
<th>Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>respirat</td>
<td>Respiratory</td>
<td>{0, 1, 2, 3, 4}</td>
</tr>
<tr>
<td>coagulat</td>
<td>Coagulation</td>
<td>{0, 1, 2, 3, 4}</td>
</tr>
<tr>
<td>liver</td>
<td>Liver</td>
<td>{0, 1, 2, 3, 4}</td>
</tr>
<tr>
<td>cardiova</td>
<td>Cardiovascular</td>
<td>{0, 1, 2, 3, 4}</td>
</tr>
<tr>
<td>cns</td>
<td>Central nervous system</td>
<td>{0, 1, 2, 3, 4}</td>
</tr>
<tr>
<td>renal</td>
<td>Renal</td>
<td>{0, 1, 2, 3, 4}</td>
</tr>
<tr>
<td>admfrom</td>
<td>Admission origin</td>
<td>{1, \ldots, 7}</td>
</tr>
<tr>
<td>admtype</td>
<td>Admission type</td>
<td>{1, 2, 3}</td>
</tr>
<tr>
<td>sapsII</td>
<td>SAPSII score</td>
<td>{0, \ldots, 160}</td>
</tr>
<tr>
<td>age</td>
<td>Patients’ age</td>
<td>{18, \ldots, 100}</td>
</tr>
<tr>
<td>nbporms</td>
<td>Number of <em>BP</em> events</td>
<td>{0, \ldots, 28}</td>
</tr>
<tr>
<td>nhroms</td>
<td>Number of <em>HR</em> events</td>
<td>{0, \ldots, 26}</td>
</tr>
<tr>
<td>no2orms</td>
<td>Number of <em>O2</em> events</td>
<td>{0, \ldots, 30}</td>
</tr>
<tr>
<td>nurorms</td>
<td>Number of <em>UR</em> events</td>
<td>{0, \ldots, 29}</td>
</tr>
</tbody>
</table>
Preprocessing

- Six new attributes created, by sliding the $SOFA_{d-1}$ values into each previous example;
- The last day of the patient's admission entries were discarded (remaining a total of 25309);
- The new attributes were transformed into binary variables: 0 if $SOFA_d < 3$; 1, else.
- Inputs rescaled into $[-1, 1]$ (1-of-C coding for the nominal attributes: $SOFA$, $admfrom$ and $admtpe$);

Performance Assessment

- **Metrics**: Sen. (sensitivity), Spe. (specificity) and Ac. (accuracy);
- **Validation:**
  - Holdout with resampling;
  - Training set uses the first 2/3 of the examples;
  - Test set uses the rest of the examples (1/3);
  - Thirty runs applied to each model;
- Fully connected MLP with bias, one hidden layer \((n_h = n/2)\) and logistic activation functions;
- One MLP per organ \(→\) one output (predicted class given by the nearest class value);
- Random weight initialization;
- RPROP algorithm, being stopped when the training error slope approaches zero or after 100 epochs;
- Implemented in a JAVA package developed by the authors.
## Feature Selection

<table>
<thead>
<tr>
<th>Organ</th>
<th>A</th>
<th></th>
<th></th>
<th>B</th>
<th></th>
<th></th>
<th>C</th>
<th></th>
<th></th>
<th>D</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>respirat</td>
<td>86.3</td>
<td>72.4</td>
<td>90.2</td>
<td>86.2</td>
<td>70.0</td>
<td>90.8</td>
<td>77.9</td>
<td>4.4</td>
<td>98.8</td>
<td>77.6</td>
<td>1.8</td>
<td>99.4</td>
</tr>
<tr>
<td>coagulat</td>
<td>97.4</td>
<td>68.8</td>
<td>98.7</td>
<td>97.3</td>
<td>59.6</td>
<td>99.0</td>
<td>95.8</td>
<td>4.6</td>
<td>99.9</td>
<td>95.7</td>
<td>0.0</td>
<td>100</td>
</tr>
<tr>
<td>liver</td>
<td>98.3</td>
<td>68.6</td>
<td>99.1</td>
<td>98.3</td>
<td>60.2</td>
<td>99.4</td>
<td>97.3</td>
<td>7.6</td>
<td>99.9</td>
<td>97.3</td>
<td>0.0</td>
<td>100</td>
</tr>
<tr>
<td>cardinova</td>
<td>94.2</td>
<td>84.1</td>
<td>96.3</td>
<td>94.2</td>
<td>84.0</td>
<td>96.3</td>
<td>82.8</td>
<td>7.5</td>
<td>99.0</td>
<td>82.2</td>
<td>0.5</td>
<td>99.8</td>
</tr>
<tr>
<td>cns</td>
<td>95.7</td>
<td>92.7</td>
<td>96.4</td>
<td>95.7</td>
<td>92.3</td>
<td>96.4</td>
<td>83.5</td>
<td>23.4</td>
<td>97.1</td>
<td>81.6</td>
<td>0.4</td>
<td>99.9</td>
</tr>
<tr>
<td>renal</td>
<td>95.5</td>
<td>71.3</td>
<td>97.8</td>
<td>95.3</td>
<td>66.6</td>
<td>98.1</td>
<td>91.4</td>
<td>5.7</td>
<td>99.7</td>
<td>91.1</td>
<td>0.3</td>
<td>100</td>
</tr>
<tr>
<td>Mean</td>
<td>94.6</td>
<td>76.3</td>
<td>96.4</td>
<td>94.5</td>
<td>72.1</td>
<td>96.7</td>
<td>88.1</td>
<td>8.9</td>
<td>99.1</td>
<td>87.6</td>
<td>0.5</td>
<td>99.96</td>
</tr>
</tbody>
</table>

A - SOFA, B - All, C - Case mix and events, D - events.
Ac. - Accuracy, Sen. - Sensitivity, Spe - Specificity.
There is a higher number of false (0) than true (1) examples!
Balanced Training

- To balance the data by **over** or **under** sampling;
- In this case, the training data will have $2/3$ of true examples, plus an equal number of false examples (**under sampling**);
- Test set with other $1/3$ true entries and a number of negative ones in proportion to the original dataset;

<table>
<thead>
<tr>
<th>Organ</th>
<th>C</th>
<th></th>
<th></th>
<th>D</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>respirat</td>
<td>61.3</td>
<td>66.4</td>
<td>59.8</td>
<td>67.1</td>
<td>41.1</td>
<td>74.5</td>
</tr>
<tr>
<td>coagulat</td>
<td>67.6</td>
<td>66.8</td>
<td>67.7</td>
<td>73.7</td>
<td>41.5</td>
<td>75.1</td>
</tr>
<tr>
<td>liver</td>
<td>70.0</td>
<td>71.6</td>
<td>70.0</td>
<td>66.9</td>
<td>36.5</td>
<td>67.8</td>
</tr>
<tr>
<td>cardiova</td>
<td>65.9</td>
<td>62.5</td>
<td>66.7</td>
<td>68.2</td>
<td>37.9</td>
<td>74.8</td>
</tr>
<tr>
<td>cns</td>
<td>73.6</td>
<td>63.9</td>
<td>75.7</td>
<td>66.8</td>
<td>36.3</td>
<td>73.7</td>
</tr>
<tr>
<td>renal</td>
<td>67.8</td>
<td>65.6</td>
<td>68.0</td>
<td>73.2</td>
<td>37.6</td>
<td>76.6</td>
</tr>
<tr>
<td><strong>Mean</strong></td>
<td><strong>67.7</strong></td>
<td><strong>66.2</strong></td>
<td><strong>68.0</strong></td>
<td><strong>69.3</strong></td>
<td><strong>38.5</strong></td>
<td><strong>73.8</strong></td>
</tr>
</tbody>
</table>

Acc. - Accuracy, Sen. - Sensitivity, Spe - Specificity.
Improving Learning (C improved)

- Balanced training decreased $N$, thus reducing the computational effort;
- A trial-and-error search was performed for $n_h \in \{4, 8, 16, 32\}$ and $\text{maximum epochs} \in \{100, 500, 100\}$;
- The MLP with lowest training error was $n_h = 16$ and maximum epochs = 1000;

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>respirat</td>
<td>61.3</td>
<td>66.4</td>
<td>59.8</td>
<td>63.3</td>
<td>70.4</td>
<td>61.3</td>
</tr>
<tr>
<td>coagulat</td>
<td>67.6</td>
<td>66.8</td>
<td>67.7</td>
<td>70.0</td>
<td>72.0</td>
<td>69.9</td>
</tr>
<tr>
<td>liver</td>
<td>70.0</td>
<td>71.6</td>
<td>70.0</td>
<td>72.5</td>
<td>77.3</td>
<td>72.4</td>
</tr>
<tr>
<td>cardiova</td>
<td>65.9</td>
<td>62.5</td>
<td>66.7</td>
<td>69.1</td>
<td>66.3</td>
<td>69.8</td>
</tr>
<tr>
<td>cns</td>
<td>73.6</td>
<td>63.9</td>
<td>75.7</td>
<td>75.2</td>
<td>72.2</td>
<td>75.8</td>
</tr>
<tr>
<td>renal</td>
<td>67.8</td>
<td>65.6</td>
<td>68.0</td>
<td>71.9</td>
<td>70.5</td>
<td>72.0</td>
</tr>
<tr>
<td>Mean</td>
<td>67.7</td>
<td>66.2</td>
<td>68.0</td>
<td>70.3</td>
<td>71.5</td>
<td>70.2</td>
</tr>
</tbody>
</table>

Acc. - Accuracy, Sen. - Sensitivity, Spe - Specificity.
Comparison with Other DM Techniques

Two methods were selected from the WEKA software: Naive Bayes and JRIP (a rule base learner);

<table>
<thead>
<tr>
<th>Organ</th>
<th>Naive Bayes</th>
<th></th>
<th>JRIP</th>
<th></th>
<th>Neural Networks</th>
</tr>
</thead>
<tbody>
<tr>
<td>respirat</td>
<td>73.5</td>
<td>25.2</td>
<td>87.3</td>
<td>62.8</td>
<td>61.9</td>
</tr>
<tr>
<td>coagulat</td>
<td>83.3</td>
<td>24.8</td>
<td>85.8</td>
<td>67.8</td>
<td>62.4</td>
</tr>
<tr>
<td>liver</td>
<td>70.8</td>
<td>54.3</td>
<td>71.2</td>
<td>75.7</td>
<td>73.7</td>
</tr>
<tr>
<td>cardiova</td>
<td>73.4</td>
<td>33.4</td>
<td>82.0</td>
<td>66.6</td>
<td>70.3</td>
</tr>
<tr>
<td>cns</td>
<td>76.3</td>
<td>41.3</td>
<td>84.2</td>
<td>77.6</td>
<td>74.4</td>
</tr>
<tr>
<td>renal</td>
<td>76.8</td>
<td>45.6</td>
<td>79.9</td>
<td>69.1</td>
<td>68.5</td>
</tr>
<tr>
<td>Mean</td>
<td>75.7</td>
<td>37.4</td>
<td>81.7</td>
<td>69.9</td>
<td>68.5</td>
</tr>
</tbody>
</table>

Acc. - Accuracy, Sen. - Sensitivity, Spe - Specificity.

Conclusions

- It is possible to diagnose organ failure by using cheap and fast intermediate outcomes (overall accuracy of 70%);
- The proposed approach opens room for the development of automatic clinical decision tools.
Mortality Assessment (not published yet)

- The goal is to predict mortality in ICUs;
- Another EURICUS II derived database was adopted: 13165 records of patients from 9 EU countries, during 10 months, from 1998 until 1999;
- Each example reports over a patient’s full length of stay;
- After a consult with ICU specialists, the patients with age lower than 18, burned or bypass surgery were discarded, remaining a total of 13165 records.
## Mortality Assessment main attributes

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
<th>Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAPS II</td>
<td>SAPS II score</td>
<td>{0, 1, \ldots, 163}</td>
</tr>
<tr>
<td>age</td>
<td>Patients’ age</td>
<td>{18, 19, \ldots, 100}</td>
</tr>
<tr>
<td>admtype</td>
<td>Admission type</td>
<td>{1, 2, 3} (^a)</td>
</tr>
<tr>
<td>admfrom</td>
<td>Admission origin</td>
<td>{1, 2, \ldots, 7} (^b)</td>
</tr>
<tr>
<td>NBP</td>
<td>Daily number of blood pressure events</td>
<td>[0.0, \ldots, 33.0]</td>
</tr>
<tr>
<td>NCRBP</td>
<td>Daily number of critical blood pressure events</td>
<td>[0.0, \ldots, 6.0]</td>
</tr>
<tr>
<td>NHR</td>
<td>Daily number of heart rate events</td>
<td>[0.0, \ldots, 42.0]</td>
</tr>
<tr>
<td>NCRHR</td>
<td>Daily number of critical heart rate events</td>
<td>[0.0, \ldots, 6.0]</td>
</tr>
<tr>
<td>NO2</td>
<td>Daily number of oxygen events</td>
<td>[0.0, \ldots, 28.0]</td>
</tr>
<tr>
<td>NCRO2</td>
<td>Daily number of critical oxygen events</td>
<td>[0.0, \ldots, 6.0]</td>
</tr>
<tr>
<td>NUR</td>
<td>Daily number of urine events</td>
<td>[0.0, \ldots, 38.0]</td>
</tr>
<tr>
<td>NCRURUR</td>
<td>Daily number of critical urine events</td>
<td>[0.0, \ldots, 8.0]</td>
</tr>
<tr>
<td>TBP</td>
<td>Daily time of blood pressure events</td>
<td>[0.0, \ldots, 36.0]</td>
</tr>
<tr>
<td>THR</td>
<td>Daily time of heart rate events</td>
<td>[0.0, \ldots, 33.0]</td>
</tr>
<tr>
<td>TO2</td>
<td>Daily time of oxygen events</td>
<td>[0.0, \ldots, 33.0]</td>
</tr>
<tr>
<td>TUR</td>
<td>Daily time of urine events</td>
<td>[0.0, \ldots, 40.0]</td>
</tr>
<tr>
<td>death</td>
<td>The occurrence of death</td>
<td>{0, 1} (^c)</td>
</tr>
</tbody>
</table>

\(^a\) 1 - Non scheduled surgery, 2 - Scheduled surgery, 3 - Physician.

\(^b\) 1 - Surgery block, 2 - Recovery room, 3 - Emergency room, 4 - Nursing room, 5 - Other ICU, 6 - Other hospital, 7 - Other sources.

\(^c\) 0 - No death, 1 - Death.
SAPSII Prognostic Model

- The SAPS II is the most widely European score for mortality assessment, ranging from 0 to 163 (highest death probability);
- The model encompasses a total of 17 variables (e.g. age, previous health status or diagnosis), which are collected within the first 24 hours of the patient’s internment;
- Some of these variables imply the use of clinical tests (e.g. blood samples) which require costs and time;
- The prognostic model is given by a **Logistic Regression**.
- Implemented in the **R** statistical environment.
Preprocessing/MLP Design

- The input data was preprocessed with a max – min scaling within $[-1.0, 1.0]$ and a 1-of-C coding for the nominal attributes (e.g. admtype);
- The output was preprocessed to the values: 0 - no death, 1 - death;
- Log – application of the transform $y = \log(x + 1)$ in the event variables;
- Balanced – Balance training with under sampling;
- Fully connected MLPs with bias connections, one hidden layer (with a fixed number of hidden nodes $\text{input\_nodes}/2$) and logistic activation functions;
- The output was preprocessed to the values: 0 - no death, 1 - death;
- Predicted class given by the nearest value to the decision threshold $D$;
- RPROP training, random initialization within $[-1.0, 1.0]$, stopped after 100 epochs;
- Implemented in a JAVA package developed by the authors.
Performance Assessment

- **Metrics**: Sen. (sensitivity), Spe. (specificity), Acc. (accuracy) and AUC;
- **Validation**: Hold-out with 2/3 training and 1/3 test sizes;
- 30 runs for each model;
- \( D = 0.5 \) (the middle of the interval);

Training Results

<table>
<thead>
<tr>
<th>Setup</th>
<th>Acc</th>
<th>Sen</th>
<th>Spe</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>85.75±0.10</td>
<td>41.78±0.34</td>
<td>96.43±0.08</td>
</tr>
<tr>
<td>Balanced</td>
<td>78.22±0.20</td>
<td>76.45±0.37</td>
<td>79.99±0.36</td>
</tr>
<tr>
<td>Log Balanced</td>
<td>79.87±0.19</td>
<td>78.99±0.22</td>
<td>80.76±0.26</td>
</tr>
</tbody>
</table>
## Test Set Performances

<table>
<thead>
<tr>
<th>Setup</th>
<th>Acc</th>
<th>Sen</th>
<th>Spe</th>
<th>AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>All</td>
<td>79.21±0.24</td>
<td>78.11±0.51</td>
<td>79.48±0.35</td>
<td>87.12±0.21</td>
</tr>
<tr>
<td>Case Outcomes</td>
<td>78.22±0.26</td>
<td>75.78±0.66</td>
<td>78.82±0.36</td>
<td>85.52±0.20</td>
</tr>
<tr>
<td>Outcomes</td>
<td>77.60±0.31</td>
<td>70.00±0.59</td>
<td>79.45±0.48</td>
<td>83.88±0.23</td>
</tr>
<tr>
<td>Logistic Regression</td>
<td>82.60±0.14</td>
<td>42.57±0.50</td>
<td>92.33±0.12</td>
<td>79.84±0.26</td>
</tr>
<tr>
<td>Logistic Regression B</td>
<td>69.37±0.32</td>
<td>77.57±0.64</td>
<td>67.37±0.48</td>
<td>80.04±0.25</td>
</tr>
</tbody>
</table>

![ROC Curve](https://via.placeholder.com/150)
Extraction of Rules using a Decision Tree

TCRUR > 0.51

15041928
No death
Death
Death
Death
No death
Death

Number of false cases: 1711
Number of true cases: 217

TCRO2 > 0.31

1564
147
false true

TCRBP > 1.18

1517
47
false true

TCRHR > 1.91

1345
172
false true

NBP > 0.73

139
33
false true

NO2 > 0.30

false true

No death

Death

false true

Death

Split Node

false true

Death

false true

Death

false true

Death
Input Importance by Sensitivity Analysis

<table>
<thead>
<tr>
<th>Setup</th>
<th>SAPSII</th>
<th>age</th>
<th>admtype</th>
<th>admfrom</th>
<th>BP*</th>
<th>HR*</th>
<th>O2*</th>
<th>UR*</th>
</tr>
</thead>
<tbody>
<tr>
<td>All</td>
<td>16.8</td>
<td>–</td>
<td>1.0</td>
<td>2.0</td>
<td>15.9</td>
<td>14.4</td>
<td>30.7</td>
<td>19.2</td>
</tr>
<tr>
<td>Case Outcomes</td>
<td>–</td>
<td>14.3</td>
<td>5.9</td>
<td>11.7</td>
<td>13.1</td>
<td>16.7</td>
<td>22.5</td>
<td>15.8</td>
</tr>
<tr>
<td>Outcomes</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>16.9</td>
<td>15.8</td>
<td>21.8</td>
<td><strong>45.5</strong></td>
</tr>
</tbody>
</table>

* All attributes related to the variable where included: number of events/critical events and the time.

Conclusions

- The proposed approach outperformed the SAPSII model;
- It requires less variables and the outcomes can be acquired automatically, with no extra costs, while some of the SAPSII parameters demand clinical tests;
A **Time Series** contains **time ordered** observations of an event:

![Time Series Example](image)

**Time Series Forecasting (TSF)** uses past patterns to predict the future;

Several **TSF** methods available:
- **Exponential Smoothing (ES)** (e.g. Holt-Winters);
- **Box-Jenkins (BJ)** methodology (or ARIMA).
Time Series Data

Ten series were chosen from different domains, being classified into: Seasonal and Trended (ST); Seasonal (S); Trended (T); Nonlinear (N); and Chaotic (C).

Daily IBM stock closing *prices* (left) and Annual Wolf’s *sunspots* (right).
Training cases defined by a **Sliding Time Window**. Example: for the series 1, 2, 3, 4, 5, 6, the sliding window \(< 1, 2, 4 >\) defines the training examples 1, 3, 4 → 5 and 2, 4, 5 → 6;

The number of parameters is given by: \( p_{\text{MLP}} = n(n_h + 1) + 2n_h + 1 \) (\( n_h \) – number of hidden nodes);

**RPROP** algorithm, being stopped when the training error slope approaches zero or after 1000 epochs;

Implemented in a **C++** package developed by the authors;
Performance Assessment

- **Metrics from Information Theory:**
  - **Akaike Information Criterion (AIC)** = $N \ln(SSE/N) + 2p$;
  - **Bayesian Information Criterion (BIC)** = $N \ln(SSE/N) + p \ln(N)$;

- **Training Error (RMSE);**

- **Theil’s U** = $\frac{RMSE}{\sqrt{\frac{\sum_{i=t+1}^{t+L} (x_t - x_{t-1})^2}{L}}}$ ($L$ – the number of forecasts);

- **Validation:** no sampling, examples ordered by time; training set with first 90% of the examples (test set last 10%);

- Thirty runs applied to each model;
Model Selection

- Best sliding window and **MLP** topology?
- The **sunspots** simulations.

<table>
<thead>
<tr>
<th>Window</th>
<th>( n_h )</th>
<th>( p )</th>
<th>( RMSE_t )</th>
<th>( AIC )</th>
<th>( BIC )</th>
<th>( RMSE_f )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>14</td>
<td>14.7</td>
<td>1355</td>
<td>1404</td>
<td>18.2±0.1</td>
</tr>
<tr>
<td>(&lt; 1, 2, \ldots, 13 &gt;)</td>
<td>6</td>
<td>104</td>
<td>11.6</td>
<td>1419</td>
<td>1784</td>
<td>20.5±0.7</td>
</tr>
<tr>
<td></td>
<td>12</td>
<td>194</td>
<td>9.0</td>
<td>1474</td>
<td>2155</td>
<td>20.2±0.8</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>5</td>
<td>14.8</td>
<td>1345</td>
<td>1369</td>
<td>17.8±0.3</td>
</tr>
<tr>
<td>(&lt; 1, 2, 9, 10, 11, 12 &gt;)</td>
<td>5</td>
<td>47</td>
<td>11.5</td>
<td>1302</td>
<td>1467</td>
<td>17.0±0.6</td>
</tr>
<tr>
<td></td>
<td>13</td>
<td>111</td>
<td>9.4</td>
<td>1328</td>
<td>1717</td>
<td>19.0±0.8</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>5</td>
<td>15.1</td>
<td>1352</td>
<td>1369</td>
<td>18.1±0.0</td>
</tr>
<tr>
<td>(&lt; 1, 2, 10, 11 &gt;)</td>
<td>1</td>
<td>11</td>
<td>14.1</td>
<td>1329</td>
<td><strong>1368</strong></td>
<td>17.8±0.3</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>53</td>
<td>10.7</td>
<td><strong>1278</strong></td>
<td>1464</td>
<td>19.4±0.5</td>
</tr>
</tbody>
</table>
Evolutionary Neural Network (ENN) encoding

Weight Matrix

Pruning:
Uses a **Genetic Algorithm**, each bit codes a connection:

- Fitness measured by the **BIC** value;
- **Genetic Operators**: 2-point crossover and bit mutation;
- Maximum number of nodes set to 13 input and 6 hidden ones;
Series: passengers, paper, deaths, melbourne, chemical and prices:
Series: **sunspots**, **kobe**, **quadratic** and **henon**
Sunspots forecasts

The graph shows the comparison between sunspots and MLP forecasts over time. The solid line represents sunspots, while the dashed line represents the MLP forecasts. The x-axis represents time, and the y-axis represents the number of sunspots. The data is plotted from time 265 to 285.
## Overall Comparison (Theil’s U values)

<table>
<thead>
<tr>
<th>Series</th>
<th>ES</th>
<th>BJ</th>
<th>ENN</th>
</tr>
</thead>
<tbody>
<tr>
<td>passengers (ST)</td>
<td>0.104</td>
<td>0.118</td>
<td>0.123</td>
</tr>
<tr>
<td>paper (ST)</td>
<td>0.035</td>
<td>0.076</td>
<td>0.057</td>
</tr>
<tr>
<td>deaths (S)</td>
<td>0.501</td>
<td>0.496</td>
<td>0.420</td>
</tr>
<tr>
<td>melbourne (S)</td>
<td>0.137</td>
<td>0.186</td>
<td>0.125</td>
</tr>
<tr>
<td>chemical (T)</td>
<td>0.830</td>
<td>0.861</td>
<td>0.873</td>
</tr>
<tr>
<td>prices (T)</td>
<td>1.00</td>
<td>1.008</td>
<td>0.997</td>
</tr>
<tr>
<td>sunspots (N)</td>
<td>0.762</td>
<td>0.434</td>
<td>0.287</td>
</tr>
<tr>
<td>kobe (N)</td>
<td>0.823</td>
<td>0.027</td>
<td>0.020</td>
</tr>
<tr>
<td>quadratic (C)</td>
<td>0.424</td>
<td>0.424</td>
<td>0.000</td>
</tr>
<tr>
<td>henon (C)</td>
<td>0.417</td>
<td>0.326</td>
<td>0.053</td>
</tr>
</tbody>
</table>

### Conclusions:
- ENN works automatically and is the best method in 7 (of 10) series;
- The drawback is the increase of computational effort;
Tenderness is the most important characteristic affecting meat quality;

In the past, two major approaches have been proposed: instrumental and sensory analysis;

Both approaches require laboratory work, being expensive, time demanding and invasive;

An alternative is to use cheap and non invasive carcass measurements, which can be collected within the first 24 hours after slaughtering (e.g. color readings);

The classic animal science approach is based in the use of Multiple Regression (MR) models (will fail if nonlinear relationships are present);
Lamb Meat data

- This study considered animals from the northeast region of Portugal, known as *Trás-os-Montes*;
- The data were collected during one year period, from November/2002 until November/2003;
- Each instance of the dataset denotes the readings obtained from a slaughtered animal.
- The database is quite small (81 examples) since each animal slaughter presents considerable costs.
- **WBS** – force is the major objective index for measuring meat tenderness;
- **STP** – blind taste panel with 12 individuals;
## Lamb Meat attributes

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
<th>Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>Breed</td>
<td>Breed type</td>
<td>{1, 2}$^a$</td>
</tr>
<tr>
<td>Sex</td>
<td>Lamb sex</td>
<td>{1, 2}$^b$</td>
</tr>
<tr>
<td>HCW</td>
<td>Hot Carcass Weight ((kg))</td>
<td>[4.1, 14.8]</td>
</tr>
<tr>
<td>STF2</td>
<td>Sternal Fat Thickness at 2nd sternebra level ((mm))</td>
<td>[6.0, 27.8]</td>
</tr>
<tr>
<td>C</td>
<td>Subcutaneous fat depth above \textit{longissimus} muscle ((mm))</td>
<td>[0.3, 5.1]</td>
</tr>
<tr>
<td>pH1</td>
<td>The pH measured 1 hour after slaughtering</td>
<td>[5.5, 6.8]</td>
</tr>
<tr>
<td>pH24</td>
<td>The pH measured 24 hours after slaughtering</td>
<td>[5.5, 5.9]</td>
</tr>
<tr>
<td>a*</td>
<td>Color red index</td>
<td>[11.5, 22.2]</td>
</tr>
<tr>
<td>b*</td>
<td>Color yellow index</td>
<td>[6.5, 12.5]</td>
</tr>
<tr>
<td>dE</td>
<td>Total color difference (encompasses \textit{L}, \textit{a*} and \textit{b*})</td>
<td>[46.5, 60.9]</td>
</tr>
<tr>
<td>dL</td>
<td>Luminosity differential when compared to a standard</td>
<td>([-56.1, -39.1])</td>
</tr>
<tr>
<td>dB*</td>
<td>Yellow differential index</td>
<td>[15.3, 22.5]</td>
</tr>
<tr>
<td>WBS</td>
<td>Warner-Bratzler Shear force ((kg/cm^2))</td>
<td>[9.5, 57.0]$^c$</td>
</tr>
<tr>
<td>STP</td>
<td>Sensory Taste Panel</td>
<td>[0.7, \ldots, 7.1]$^c$</td>
</tr>
</tbody>
</table>

\(^a\) 1 – \textit{Churra Galega Bragançana}, 2 – \textit{Churra Galega Mirandesa}

\(^b\) 1 – \textit{Male}, 2 – \textit{Female}

\(^c\) Low values suggest tender meat.
Preprocessing/MLP Design

- The dataset contained 2 missing WBS and 10 missing STP values, which were discarded (leading to 2 datasets with 79 and 71 samples);
- Attributes standardized to a zero mean and one standard deviation;
- MLP with a fixed number of hidden nodes ($n_h = 24$);
- The decay parameter is tuned by a grid-search $\lambda \in [0.0, 0.2]$;
- One MLP per task (WBS and STP) with logistic functions in the hidden nodes and linear function in the output node;
- **Multiple Neural Network (MNN)**: 5 trainings, being selected the MLP with the lowest penalized error;
- **Neural Network Ensemble (NNE)**: 5 MLPs, being the final prediction is given as the average of the individual predictions;
- **NNESA** - NNE with the most important features given by a Sensitivity Analysis procedure;
- Implemented in the R statistical environment;
Performance Assessment

- **Metrics:** MAD, RMSE, Relative MAD
  \[ RMAD = \frac{1}{N} \times \frac{MAD}{\overline{MAD}_Y} \times 100 \% \], RRMSE;

- **Validation:** 10-fold cross-validation;

- 5 runs for each model;

Results

<table>
<thead>
<tr>
<th>Task</th>
<th>Method</th>
<th>MAD</th>
<th>RMAD</th>
<th>RMSE</th>
<th>RRMSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>WBS</td>
<td>MR</td>
<td>9.17</td>
<td>134.7%</td>
<td>11.64</td>
<td>130.4%</td>
</tr>
<tr>
<td></td>
<td>MNN</td>
<td>6.14</td>
<td>90.1%</td>
<td>8.14</td>
<td>91.2%</td>
</tr>
<tr>
<td></td>
<td>NNE</td>
<td>5.89</td>
<td>86.6%</td>
<td>7.79</td>
<td>87.3%</td>
</tr>
<tr>
<td></td>
<td>NNESA</td>
<td>5.54</td>
<td>81.4%</td>
<td>7.46</td>
<td>83.7%</td>
</tr>
<tr>
<td></td>
<td>STP</td>
<td>1.64</td>
<td>119.3%</td>
<td>2.13</td>
<td>131.7%</td>
</tr>
<tr>
<td></td>
<td>MR</td>
<td>1.37</td>
<td>99.9%</td>
<td>1.68</td>
<td>104.1%</td>
</tr>
<tr>
<td></td>
<td>MNN</td>
<td>1.27</td>
<td>92.4%</td>
<td>1.56</td>
<td>96.7%</td>
</tr>
<tr>
<td></td>
<td>NNE</td>
<td>1.17</td>
<td>84.9%</td>
<td>1.45</td>
<td>89.5%</td>
</tr>
</tbody>
</table>
### Input Importance by Sensitivity Analysis

<table>
<thead>
<tr>
<th>Task</th>
<th>Method</th>
<th>Breed</th>
<th>HCW</th>
<th>STF2</th>
<th>pH1</th>
<th>a*</th>
<th>dE</th>
<th>dL</th>
<th>dB*</th>
</tr>
</thead>
<tbody>
<tr>
<td>WBS</td>
<td>NNE</td>
<td>4.3</td>
<td>5.8</td>
<td>7.6</td>
<td>–</td>
<td>50.3</td>
<td>11.1</td>
<td>5.5</td>
<td>8.5</td>
</tr>
<tr>
<td></td>
<td>NNESA</td>
<td>2.8</td>
<td>21.4</td>
<td>7.7</td>
<td>–</td>
<td>41.7</td>
<td>11.7</td>
<td>6.2</td>
<td>8.5</td>
</tr>
<tr>
<td>STP</td>
<td>NNE</td>
<td>41.0</td>
<td>–</td>
<td>5.1</td>
<td>6.6</td>
<td>22.6</td>
<td>2.9</td>
<td>–</td>
<td>3.8</td>
</tr>
<tr>
<td></td>
<td>NNESA</td>
<td>36.8</td>
<td>–</td>
<td>20.1</td>
<td>9.3</td>
<td>22.4</td>
<td>9.7</td>
<td>–</td>
<td>1.7</td>
</tr>
</tbody>
</table>

### Example of MR vs NNESA

![Graph showing comparison between MR and NNESA](image)
Conclusions

- The **NNESA** method outperformed other MLP approaches, as well as the **Multiple Regression**;
- The **NNESA** method is non-invasive, much cheaper than the **WBS** or **STP** procedures, and can be computed 24 hours after slaughter;
- Input importance differences between **WBS** and **STP** may be due to psychological factors. It was found that **Mirandesa** lambs were considered less stringy and more odor intense than **Bragançana** lambs. Due to animal stress during slaughter? Further research is required.
- In future work: the **NNESA** will be tested in real-time environments and other nonlinear techniques (e.g. **Support Vector Machines**) will also be considered;
At last, a nice demo...

Artificial Life Environment (reinforcement learning)

- There is energy (e.g. grass) all over the field;
- Two populations of beings: preys and predators;
- Each artificial being is modeled by a MLP:
  - Inputs: vision within a certain range and angle (nothing, prey or predator);
  - Outputs: actions (nothing, move forward, rotate left or right);
  - Weights are directly coded into chromosomes, using real-valued representations;
- Evolutionary learning;


Glossary of Terms.

*Data Preparation for Data Mining.*
Morgan Kaufmann, S. Francisco CA, USA.

Supervised Learning in Multilayer Perceptrons - from Backpropagation to Adaptive Learning Techniques.
*Computer Standards and Interfaces*, 16.

Evolutionary Neural Network Learning.

Simultaneous Evolution of Neural Network Topologies and Weights for Classification and Regression.

