Sistemas Híbridos Inteligentes

Prof. Dr. André C. P. L. F de Carvalho
LABIC - ICMSC - USP

Main topics

- Introduction
- What is a Hybrid System
- A framework for Hybrid Systems
- Some approaches for Hybrid Systems
  - Evolutionary design of Neural Networks
  - Knowledge extraction from Neural Networks
  - Case Based Reasoning and Neural Networks
- Conclusion

What is a Hybrid System?

- Hybrid (latin)
  - Heterogeneous in origin or composition
- Biological hybrids
  - Many agricultural plants are in fact hybrids of other plants
- Technological hybrids
  - Engineering products combining different techniques

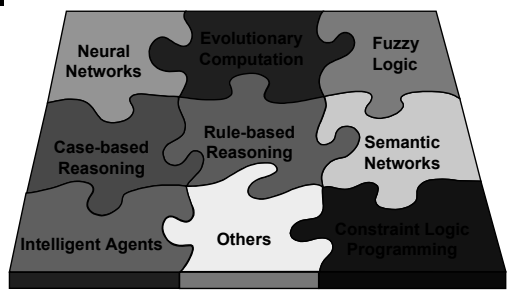
What is a Hybrid System?

- Computer science definition
  - Dynamic system integrating
    - Components of discrete (or digital) nature
    - Components of continuos (or analogue) nature
  - The system can evolve in time with different continuos profiles
    - The change from one profile to another is made in a discrete way
      - Exactly like in the state machines that characterise pure discrete systems

What is a Hybrid System?

- Hybrid does not apply only to AI
  - Other areas have “their hybrids”
    - Transgenic
      - Having genetic material introduced from another species (Oxford dictionary)
- Hybrid Intelligent Systems
  - System composed by two or more techniques, where at least one of them is based on Artificial Intelligence

AI Puzzle



Goal of Hybrid Systems

Enhancement of the capability of an AI-based system by combining the best characteristics of different Artificial Intelligence techniques.

Why to use Hybrid Systems

- For years, each new AI technique has claimed to be the most efficient technique to deal with problems requiring intelligent solutions

If the only tool you have is a hammer, than all your problems look like nails"

Why to use Hybrid Systems

- Every AI technique has its advantages and disadvantages when used to solve different problems
- Current techniques are still far away from human information processing capabilities
- There are complex AI problems which cannot be (easily) solved by current individual AI techniques

Why to use Hybrid Systems

- In the last years, the idea of mixing AI technologies has gained strength in the AI community
  - This mixing has been called Hybrid Intelligent Systems or Hybrid Systems (HIS)
- A good definition:
  - Hybrid System is a system that uses more than one problem-solving technique in order to solve a problem

Why to use Hybrid Systems

- HIS have a very good potential for solving difficult and complex problems
  - In the best situation, HIS combine all the strengths of the individual components

Why to use Hybrid Systems

Hybrid systems allows the development of intelligent systems with more strength and less weakness than the individual techniques alone

Why to use Hybrid Systems

- There are psychological evidences that humans rarely (if ever) learn purely from theory or examples
  - Theory and examples seem to interact closely during human learning
  - Although it is becoming clear that people learn from both theory and examples, there is little knowledge of how this interaction takes place

Why to use Hybrid Systems

- Hybrid Systems have achieved very good results in a large number of real-world problems
  - Credit risk analysis
  - Automatic target recognition
  - Sensor fusion
  - Stock market prediction
  - Heart condition diagnosis

When not to use Hybrid?

- Although useful in a large number of problems, hybrid systems provide more opportunities for misuse
  - As a technique becomes more complex, the number of opportunities for misuse increases
  - In the worst case, a HIS may contain none of the strengths and all the weakness of the individual components

When not to use Hybrid?

- Many HIS have been proposed only for the sake of being hybrid systems
  - No clear motivation
  - Without any clear benefit in the performance achieved
  - "*If one technique is good, then a hybrid must add something more to that and make it better*"
  - The concept that techniques have niches

When not to use Hybrid?

- A HIS may be more efficient for a large number of problems
  - But there are many problems where individual techniques can achieve a better performance

Previously Proposed Hybrids

- Neural Networks
  - Rule-Based Systems
  - Predicate Logic
  - Time Series Analysis
  - Genetic Algorithms
  - Fuzzy Logic
  - Decision trees
  - Clustering techniques

Synergy Approaches

- Unified Systems
- Transformational Systems
- Modular Systems
  - Stand-Alone Systems
  - Loosely-Coupled Systems
  - Tightly-Coupled Systems
  - Fully-Integrated Systems

Unified Systems

- Main features
  - One technique simulates the behaviour of other techniques
    - The same technique is used for all the components
  - Limited impact under real applications
  - Complex implementation
  - Low scalability

Unified Systems

- Example
  - A complex task can be divided in a set of subtasks
  - Some of the subtasks would be better handled by symbolic systems
  - But each subtask is modelled by a neural network

Transformational Systems

- Main features
  - Knowledge produced by one model is transformed by another model
  - Unity
  - The transfer process can be a complete compilation of all information or may follow intermediate stages
  - Operational limitations
  - Compatibility and integrity

Transformational Systems

- Example
  - Insert, extract and refine knowledge
  - Inclusion of prior knowledge to speed up neural networks learning process (Knowledge insertion)
  - Extraction of knowledge from trained neural networks

Modular Models

- Main features
  - Components have similar status
  - Information flow can follow different module configurations
    - Parallel
    - Sequential
    - Hierarchical

Modular Models

- Main features
  - There are different degrees of coupling and integration
    - Stand Alone Models
    - Loosely-Coupled Models
    - Tightly-Coupled Models
    - Fully-Integrated Models

Stand-Alone Models

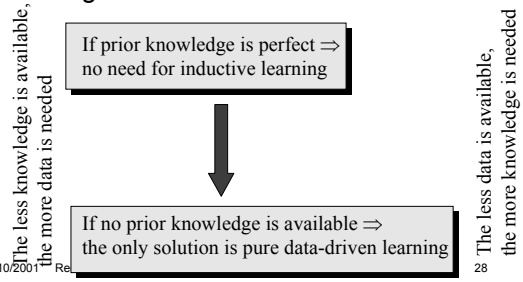
- Example of a stand-alone model
  - Different techniques are used for the same task
  - Trained and evaluated with the same dataset
  - Comparison of their performance
  - Committees

Data X Knowledge

- Definitions in the context of problem solving
  - Data: problem statement
  - Knowledge: all information that can be obtained about the application domain
- Machine learning extension
  - Data += training instances
  - Knowledge += all information additional to the data that can be obtained about the application domain

Data X Knowledge

- Knowledge/data trade-off



Data X Knowledge

- Approaches to integrate knowledge and data
  - Using knowledge to improve data-driven methods
    - Building prior knowledge in ANNs design
    - Knowledge extraction from data prior to learning
  - Using data to improve knowledge-driven methods
    - Insertion of search control heuristics for rule-based systems
    - Refine rules with data

Components of a HIS

- Most of the HIS combine data-driven techniques with knowledge-driven techniques
  - Data-driven techniques
    - Rely on data to acquire (induce) knowledge
    - Inductive learning (Ex. Neural Networks)
  - knowledge-driven techniques
    - Rely on previous knowledge
    - Deductive learning (Ex.: Fuzzy Logic)

Components of a HIS

- What are the individual techniques used in a HIS?
  - Supervised neural networks
  - Unsupervised neural networks
  - Regression techniques
  - Data reduction techniques
  - Fuzzy logic
  - Genetic algorithms
  - Predicate logic

Components of a HIS

- What are the individual techniques used in a HIS?
  - Case-based reasoning
  - Expert systems
  - Regression and decision trees
  - Clustering techniques
  - Artificial life
  - Local search
  - Statistical models and more

Evolutionary Design of ANNs

- Introduction
- Neural Networks design
- Evolutionary design
- Genetic Algorithms
- Evolutionary design of MLP networks
- Evolutionary design of RBF networks
- Examples

Introduction

- Whenever an ANN is used, ...
  - ANN model
  - ANN topology (architecture)
  - ANN training algorithm

Must be defined

Neural Networks design

- Number of possibilities is very large
  - It is not practical to manually evaluate a reasonable number of architectures
    - A large number of experiments is needed before a suitable model is found
    - Performance depends on the initial conditions (usually random)
      - Several runs with the same architecture are necessary

Neural Networks design

- Choice of the most suitable topology
  - Large networks
    - Can learn fast
    - Easier to avoid local minima
    - Can create complex decision regions
    - More fault tolerant
  - Small networks
    - Can provide good generalisation
    - Easier to extract knowledge
    - Require less computational resources

Empirical design

- State space of valid networks is:
  - Large: there are too many possibilities
  - Deceptive: similar networks may have very different behaviour
  - Multimodal: Very different networks may have similar behaviour
- Genetic algorithms can help

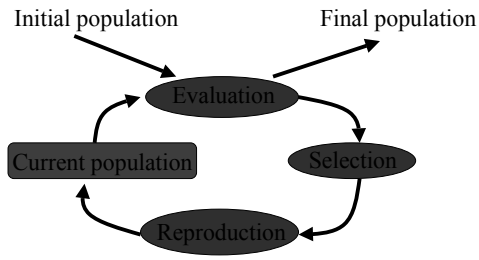
Genetic Algorithms

- Search and optimisation method
  - Based in genetics and natural selection
  - Use a population of candidates (individuals)
  - Optimisation occurs through several generations
  - At each generation
    - Selection mechanism chooses the fittest individuals (chromosomes)
    - Genetic operators produce new individuals from those selected

Genetic Algorithms

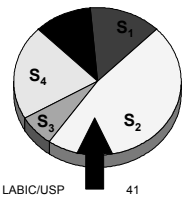
- Differ from traditional optimisation, mainly, in four aspects (Goldberg)
  - Work with a code of the group of parameters and not with the own parameters
  - Work with several possible solutions and not with a single solution
  - Use cost information or reward functions and not derivative or other auxiliary knowledge
  - Use probabilistic rules of transition instead of deterministic rules

Genetic Algorithms



Selection

- Selects the fittest individuals of the current population
  - Evaluates each individual using a fitness function
  - Select individuals according to their fitness
    - Ex.:Roulette wheel sampling

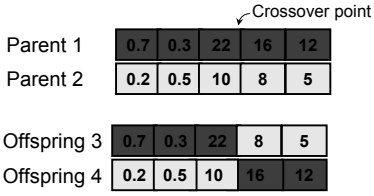


Reproduction

- Produces new generations of individuals
  - Individuals should improve their fitness at each new generation
- Genetic operators transform the population
  - Crossover
  - Mutation
  - Elitism

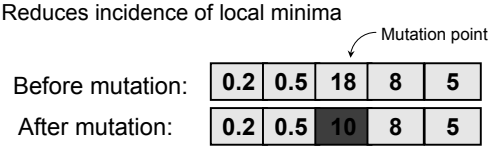
Crossover

- Recombines parents features
  - Allows heritage of desirable features by the next generations



Mutation

- Responsible for the introduction and maintenance of genetic diversity in the population
  - Modifies genes at random
  - Reduces incidence of local minima

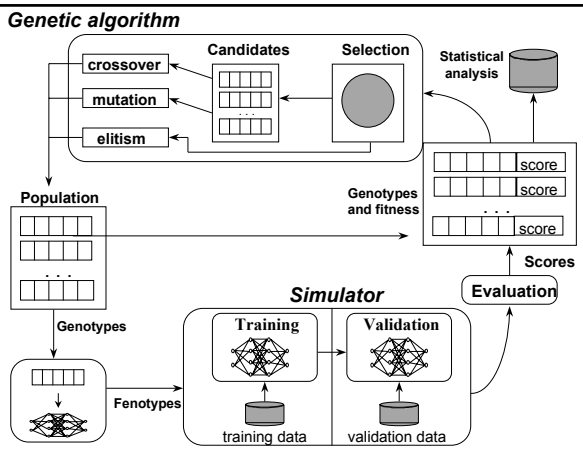


Evolutionary optimization

- Genetic Algorithms can be applied to ANN design in different manners:
  - Training of a network with a previously defined architecture
  - Optimisation of a learning algorithm
  - Definition of a network architecture and learning parameters
    - Most common

Evolutionary optimization

- Definition of a network architecture and learning parameters
  - Number of layers
  - Number of hidden units per layer
  - Activation function(s)
  - Learning rate(s)
  - Etc.



Encoding

- Several aspects must be considered before a representation is chosen:
  - If the representation allows approximately optimum solutions to be represented
  - How the reproduction operators shall perform, so that each new generation has only valid ANNs
    - How invalid structures may be excluded
  - How the representation supports the creation of new neural architectures

Encoding

- Network representation is a key aspect of evolutionary optimisation
- Good representation must:
  - Be able to generate all potentially efficient networks
  - Exclude unfeasible architectures

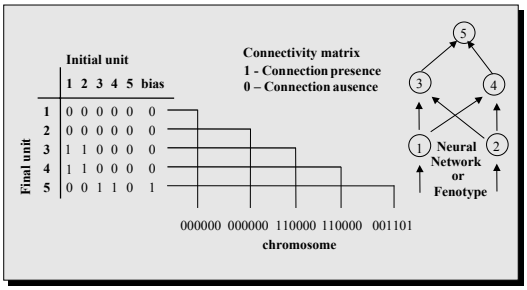
Encoding

- Ideal search space
  - Small
  - Includes the best networks
- Closed representation
  - Reproduction of feasible networks produce feasible networks
- Expressiveness power *X* creation of unfeasible or poor networks

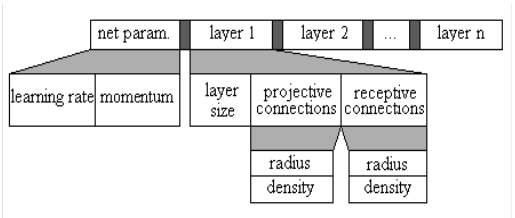
Encoding

- Encoding can be:
  - Direct:
    - Specifies each network parameter
    - Requires little encoding effort
    - Inefficient for large networks
  - Indirect:
    - Generates abstract or grammatical description
    - Better for large networks
    - Easier to exclude unfeasible networks

Direct encoding



Indirect encoding



Mandisher representation

Selection

- Fitness function
  - Combination of measures important to evaluate the networks produced
    - Learning time
    - Classification accuracy
    - Network size
  - Should lead to progressive improvement of the ANN efficiency

Reproduction

- Adaptation to genetic operators
  - Crossover
    - Limit the points where the chromosome can be broken
  - Mutation
    - Restrict the possible modifications for each gene
- Post processing
  - Elimination of unfeasible networks

MLP optimisation

- Most used ANN model $\Rightarrow$ most common network optimisation
- Parameters optimised
  - Number of layers, number of hidden units, moment term and learning rate
- Backpropagation with momentum
  - Quickprop, Rprop, Backpercolation, etc.

Experiment

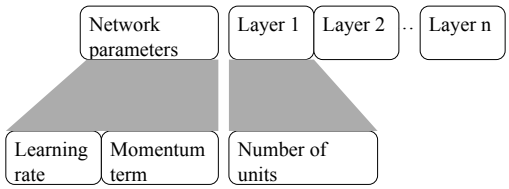
- Goal:
  - Evaluate the customer capability to pay his/her credit cards debts based on previous cases
- Data:
  - Source: Banestado Bank (FEA/RP - USP)
  - 5635 examples - 38 input attributes
    - **Good payers**: 5414 examples (96.07%)
    - **Bad payers**: 221 examples (03.93%)

Experiment

- Backpropagation with momentum
  - Uses a momentum term
    - Flat spot elimination
    - Last weight change influence current change
  - Parameters defined by trial and error
    - Extensive number of simulations
      - Involved several students

Experiment 1

- MLP network representation



 Simplification of Mandisher representation

Experiment

- MLP network
  - Parameters: η: 0.2. μ: 0.5
  - Topology: 38 - 18 - 8 - 5 - 2

Representation

0.2	0.5	18	8	5

Experiment

- GA parameters
  - Population: 20 individuals
  - Maximum number of generations: 10
  - crossover rate: 0.7
  - Mutation rate: 0.2
  - Elitism

Experiment

- Networks training
  - Stop criterion: smaller MSE (training and validation)
  - Maximum of 20 units in each hidden layer
  - At most 800 epochs

Experiments

- Best architectures
  - Empirical: 38-20-8-2 (η: 0,025 and μ: 0,025)
  - Genetic: 38-13-9-19-2 (η: 0,56538 and μ: 0,0394)

	MSE (%)		
Method	Training	Validation	Test
Genetic	2.26	0.35	0.53
Empirical	3.95	0.35	0.53

RBF Networks

- Genetic optimisation of RBF networks may handle
  - **Number of centres**
  - **Centres position**
  - **Centres width**
  - Radial activation functions
  - Weights of the output nodes

Knowledge Extraction

- Introduction
- Framework for knowledge extraction techniques
- Trepan algorithm
- EN algorithm
- Examples
- Future trends

Knowledge extraction

- ANNs have been very successfully for their classification accuracy
  - However, they cannot explain, in an understandable way, their decision process
  - End user has little or no knowledge of how a network is making its decisions
    - How did it arrive at a particular result????

Knowledge extraction

- It is desirable that an explanation capacity be part of the functionality of trained networks
  - Otherwise user might not trust the network decision
  - A great deal of effort has been made to provide the required explanation ability to ANNs

Knowledge extraction

- A substantial part of this effort has focused on the development of rule extraction techniques
  - Set of symbolic rules that represents the knowledge acquired by trained ANNs
  - Other formats are also used to represented extracted knowledge
    - Decision trees

Knowledge extraction

- Progress has been seen in other two related areas:
  - Techniques to extract finite state machine representations from recurrent networks
  - Use of ANNs to refine a previously defined rule base
    - Rules can be compiled into an ANN to incorporate knowledge from experts

Knowledge extraction

- Definition

> *Given a trained neural network and the examples used to train it, produce a concise and accurate symbolic description of the network*
>
> *Craven & Shavlik, 1994*

- Cost
  - Resources used
  - Additional effort

Knowledge extraction

- Explanation capacity is mandatory in safety-critical applications
  - Necessary to validate the network decision under every possible input
  - Examples:
    - Aircraft navigation systems
    - Nuclear and hydroelectric plants
    - Medical diagnosis

Knowledge extraction

- Main benefits
  - Allows network validation
  - Data investigation and new scientific theories induction
  - Integration of connectionist and symbolic systems
  - User feels more confident to employ network solution
  - Allows the refinement of the ANN

Taxonomy

- First taxonomy was proposed by Andrews at al
  - Based on a survey of techniques for extracting the knowledge embedded within trained ANNs
  - Focused on techniques used for extracting rules from trained feedforward networks
    - Most of the techniques found
  - Defined five classification criteria

Taxonomy

- Classification criteria
  - Expressive power (rule format) of the extracted rules
  - Quality of the extracted rules
  - Translucency of the view taken within the technique of the underlying ANN units
  - Algorithmic complexity of the technique
  - Portability of the technique

Expressive power

- Includes three categories of format:
  - Conventional symbolic rules
    - Boolean, propositional
  - Fuzzy rules
    - Based on fuzzy sets and logic
  - Rules expressed in first-order logic form
    - Rules with quantifiers and variables

Rules quality

- Quality of the extracted rules can be measured according to:
  - Rule accuracy
    - The extend to which the rule set is able to correctly classify a set of previously unseen examples from the problem domain
  - Rule fidelity
    - The extend to which the rule set mimics the behaviour of the ANN from which it was extracted

Rules quality

- Quality of the extracted rules can be measured according to:
  - Rule consistency
    - The extend to which, under different training sessions, the ANN generates rule sets which produce the same classification of unseen examples
  - Rule comprehensibility
    - How compressible is the rule to the end user

Rules quality

- Rule comprehensibility
  - Difficult to measure
  - Subjective criteria
  - Possible syntactic measures:
    - The size of the rule set in terms of:
      - The number of rules
      - The number of antecedents per rule
      - The number of combinations of the antecedents per rule (m-of-n)
      - The height of a decision tree

Translucency

- Categorises a RE technique based on the granularity of the underlying ANN units
  - Gives different importance to information embedded in the nodes
    - Either explicitly or implicitly assumed within the technique
    - Divided in four categories
      - Pedagogical
      - Decompositional
      - Eclectic
      - Compositional

Translucency

- Pedagogical techniques
  - Maximum level of granularity
  - ANN is seen as a black box
  - Extracts global relationships between the inputs and the outputs of the ANN directly
    - Does not analyse the detailed characteristics of the underlying ANN solution
  - Simulates the network input-output mappings

Translucency

- Decompositional
  - Minimum level of granularity
  - Rules are first extracted at the individual units level
    - Hidden and output units
    - Subsets of extracted rules are aggregated to define global relationships

Translucency

- Decompositional
  - Decompositional techniques can be classified according to the extension of the rules antecedents
  - For a given node, rules antecedents can be originated from:
    - Nodes directly connected to it
      - If possible, makes the extraction process easier
    - Input attributes

Translucency

- Eclectic
  - Also called hybrid techniques
  - Combine the previous approaches
  - Medium level of granularity
  - Analyse the ANN at the individual unit level
    - But also extracts rules at the global level

Translucency

- Compositional
  - The granularity may also occur at the level of ensembles of nodes
    - Instead of individual nodes or whole network
    - Ex.: some techniques used for extraction of deterministic finite-state automata (DFA)
    - This technique doesn't fit in the previous categories
  - Extract rules from ensemble of nodes

Translucency

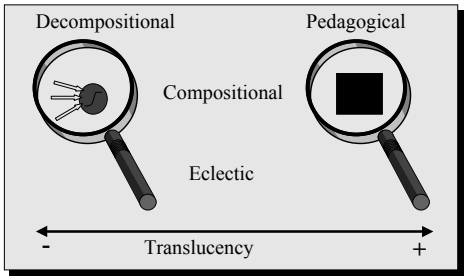


Figure adapted from Andrew et al

Algorithmic complexity

- Crucial to access computational cost of the technique
- Most authors do not comment or present data on the complexity of their technique

Portability

- Extend to which a given technique can be applied across a range of ANNs architectures and training algorithms
  - Specific purpose techniques
  - General purpose techniques

Portability

- Specific purpose techniques
  - Designed for specific architectures and/or training algorithms
  - Particular architectures and/or training algorithms are proposed together with some techniques
- General purpose techniques
  - Can be applied to any architecture and/or training algorithm

Lessons from previous works

- The main lessons from previous works can be grouped according to:
  - Results obtained
  - Portability
  - Computational complexity
  - Fidelity

Lessons from previous works

- Results obtained
  - There are several experimental results confirming the effectiveness and utility of the use of RE from ANNs as a tool for rule learning
    - Diverse range of problem domains
    - They have outperformed symbolic rule learning techniques in situations with:
      - Noisy dataset
      - Multicategoric learning

Lessons from previous works

- Portability
  - No clear difference has been reported on the performances obtained by specific and general purpose techniques
  - There is a growing demand for techniques that can be applied to situations where an ANN solution already exists

Lessons from previous works

- Computational complexity
  - May be a limiting factor
  - In many cases:
    - O(extracting rules from trained ANNs) and O(extracting rules directly from the data) are NP-hard
    - Costs (ANN learning + rule extraction >> Costs (direct rule learning techniques)

Lessons from previous works

- Fidelity
  - It is possible to have significant divergence between:
    - Rules that capture the total knowledge embedded within trained ANNs and
    - Set of rules that approximates the behaviour of the network in classifying the training set
  - Domain expert or further test data from the problem domain (validation data) should be used to validate rules extracted

Rule initialisation and refinement

- The initial knowledge about a particular problem domain can be inserted into an ANN
  - Programming the initial weights
  - Defining the ANN topology
- Other approaches:
  - Use a rule base to generate training samples
  - Initialise the ANN with a DFA

Knowledge extraction algorithms

- Several have been proposed
  - **TREPAN**
  - COMBO
  - RULEX
  - KT
  - **EN**
  - TopGen
  - KBANN
  - BRAINNE

Knowledge extraction algorithms

- Ideal algorithm
  - Produces rules easy to understand
  - Has high fidelity to ANN
  - Presents high accuracy
  - Has high portability
  - Allows the use of discrete and continuos values
  - Has polynomial computational complexity
  - Is consistent

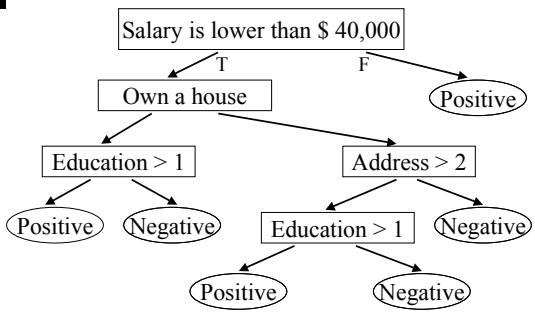
TREPAN algorithm

- TREs PArroting Networks
  - Proposed by Craven and Shavlik
  - Represents the knowledge extracted from a trained ANN in the form of a decision tree
    - Decision tree is built using a trained ANN with its training data
    - Doesn't take the network architecture into account
    - Uses the ANN as an oracle

TREPAN algorithm

Rule format:	Decision tree with simple, disjunctive or M-of-N split tests at the nodes
Quality:	Extracted decision tree shows high accuracy, fidelity and comprehensibility
Translucency:	Pedagogical
Complexity:	Polynomial in the training size, dimensionality of the input space and maximum number of values (for a discrete feature)

TREPAN algorithm



TREPAN algorithm

- Based on a principle similar to those used by the algorithms C4.5 e CART
  - Build the tree through recursive partitioning of the training set
  - Use best first search instead of depth first search

TREPAN algorithm

- Construction of the tree
  - The class associated to each sample is assigned by an oracle
    - Trained ANN is used as the oracle
      - The class assigned by the oracle may not be the same class associated in the training set
        - Trained ANN may make mistakes
      - Its main goal is to build a tree that represents the knowledge embedded in the trained ANN

TREPAN algorithm

- A common problem of decision trees based algorithms is the number of examples
  - Number of training examples available to define a node split test is reduced with the node depth
    - Tests for the nodes at high depth are based on few samples
    - Splits near the bottom of the tree are often poorly chosen

TREPAN algorithm

- Trepan reduces this problem by allowing the use of complementary samples
  - A minimum number of samples reaching a node is defined
  - When there isn't enough samples to satisfy this minimum amount
    - Add examples that were previously discharged
      - Common in sensor fusion applications
    - Create and add artificial examples

TREPAN algorithm

- Artificial samples
  - Generated at random
  - Use a marginal distribution model
    - A distribution model for each attribute
    - Restricted by the features used in the splits on the path from the root to the node
  - Oracle is used to label the samples generated

TREPAN algorithm

- According to the split tests used, there are three versions of TREPAN
  - Simple
    - Ex.: colour is blue
  - Disjunctive (or 1-of-n)
    - Ex.: colour is blue or colour is white or size > 10
  - M of N
    - Ex.: 3 of {colour is blue, colour is white, size < 5, height = 4, weight is 4}

EN Explanation Facility

- Allows the association of each active input (output) to the corresponding output (input)
- Based on ANNs internal representation
  - Node activity (active or inactive)
  - Weights value
    - Larger weights implies a higher association between nodes
  - Selection of the most significant propagation paths in the network

EN Explanation Facility

- EN provides the *How?, Why?* And *Trace* explanation options:
  - *How?* relates the ANN inputs to its outputs
    - How do I classify this application?
  - *Why?* relates the ANN outputs to its inputs
    - Why this application wasn't approved?
  - *Trace* displays the chain of reasoning which justifies the allocations

EN Explanation Facility

- The explanation facility provided by EN is based on the calculation of a general explanation degree
- Starting on the first hidden layer, the most relevant weights and nodes in each layer are chosen
- Several criteria have been proposed for the selection of weights and nodes

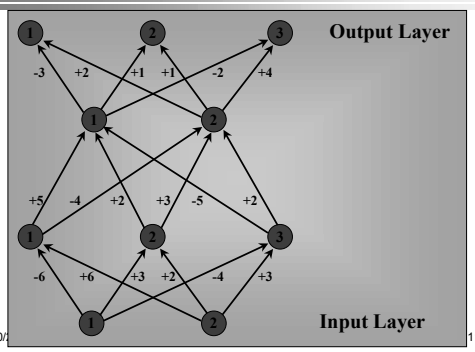
EN Explanation Facility

- Several criteria for the selection of weights and nodes
  - "PAU" (original)
    - Select nodes by their absolute weight values
  - "CCG"
    - Select nodes by their weight values
  - "SUM"
    - Select nodes by the sum of their weights

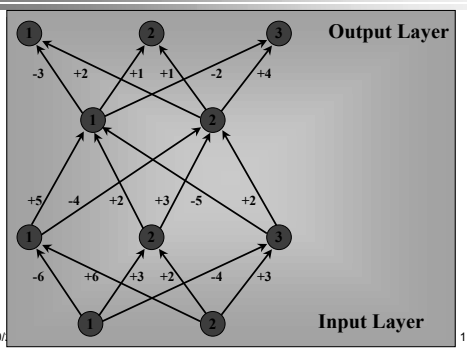
EN algorithm

- Selection criterion "PAU":
  - Rank the weights according to their absolute values
  - A fraction *d* of these weights is selected
  - Nodes with input connections associated to the selected weights are chosen for the weights selection in the next layer

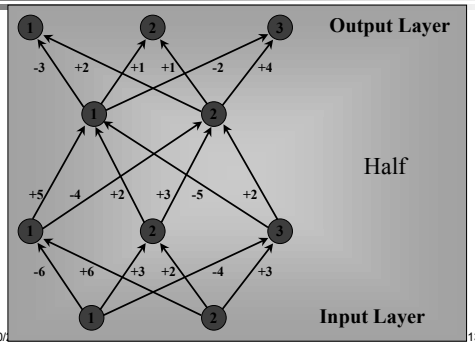
Example



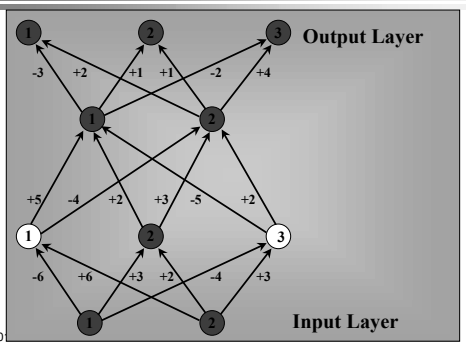
Method PAU (*How?*)



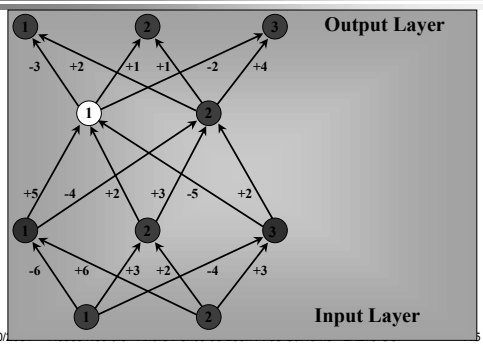
Method PAU (*How?*)



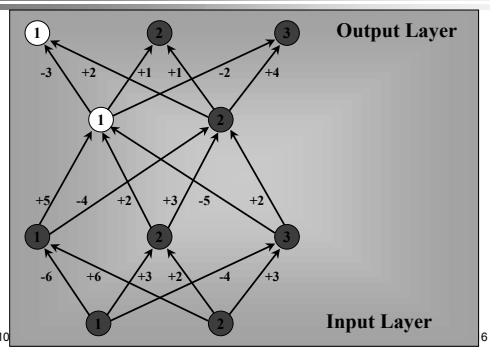
Method PAU (*How?*)



Method PAU (*How?*)



Method PAU (*How?*)



CBR AND ANN

- Introduction
- Case Based Reasoning
- Hybrid CBR
  - CBR + ANN
- Some case studies
- Final remarks

Why CBR ?

- Most of the time, the trouble in building Expert Systems comes from trying to fit experience into rules

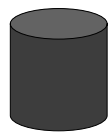
If ....
Then ...
Else...

EXPERIENCE **RULES**

Why CBR ?

- CBR offers a cost-effective solution to the 'knowledge acquisition bottleneck' problem
- CBR systems can learn from experience and so can be self-maintained
- Rule-based systems are better when it is hard to gather case data

Why CBR ?



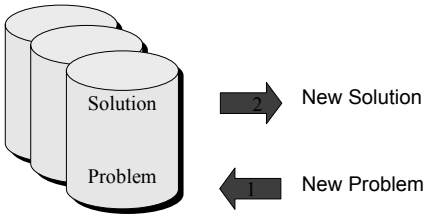
EXPERIENCE

BASE OF EXPERIENCES

But not a DB!

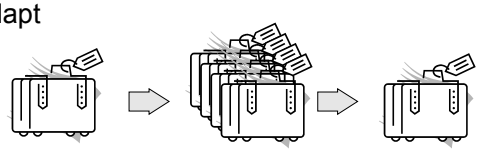
How it works?

- Solves new problems by adapting the solutions of previous similar problems



CBR Processing

- Current Situation
- Retrieve similar cases from the library
- Adapt



Travel package? Similar cases Adaptation

Case-Based Reasoning

".... transferring knowledge from past problem solving episodes to new problems that share significant aspects with corresponding past experience and using the transferred knowledge to construct solutions to new problems." (Carbonell, 1986)

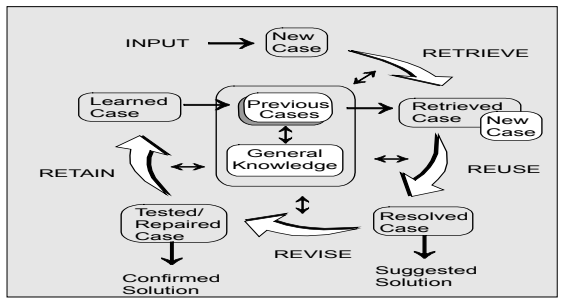
Examples of CBR

- **Classification:** "The patient's ear problems are like this prototypical case of otitis media"
- **Compiling solutions:** "Patient N's heart symptoms can be explained in the same way as previous patient D's"
- **Assessing values:** My house is like the one that sold down the street for $250,000 but has a better view"
- **Justifying with precedents:** "This Missouri case should be decided just like Roe v. Wade where the court held that a state's limitations on abortion are illegal"
- **Evaluating options:** "If we attack Cuban/Russian missile installations, it would be just like Pearl Harbor"

Case Based Reasoning

- Processes involved in the development of a CBR system:
  - Case representation
  - Case indexing
  - Storage and retrieval of cases
  - Adaptation of cases
  - Evaluation and repairing of cases

CBR Cycle (Aamodt, 1993)



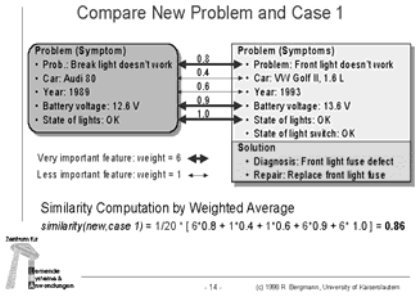
What is a case?

- There are two types of cases (D. Leake)
  - Input cases:
    - Description of specific problem situations
  - Stored cases
    - Encapsulate previous specific problem situations with solutions and outcomes
    - Contain a lesson, and a specific context in which the lesson was applied
      - Context is used to indicate when the lesson may be applied again

What is a case?

- A case usually has two parts
  - A case part
    - Symptoms
    - Used to identify the case
      - Index and recovery
  - A solution case
    - Explain how this case was (un)successfully solved previously
    - Adapted when the case is retrieved

What is a case?



Hybrid CBR

- CBR systems have been combined to several intelligent techniques
  - Ex.: Agents, Genetic Algorithms, Neural Networks, Rule-based Systems
- This integration occurs through:
  - Division of tasks between the CBR system and another intelligent technique
  - Design of an intelligent system combining features belonging to CBR and another intelligent technique

Hybrid CBR

- Four integration approaches:
  - Central Control
    - The CBR and the other technique are controlled by a central device
  - Distributed Control
    - The control is divided between the two techniques
  - Case based reasoning dominant
    - The control is biased towards the CBR component
  - Case based reasoning non-dominant
    - The control is biased towards the other component

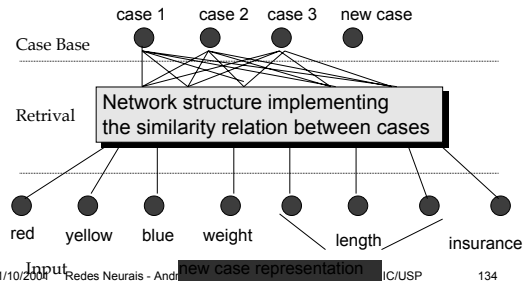
Hybrid CBR

- Integration of CBR and ANNs has been investigated by a number of researchers
  - In most of the proposals, the ANN has been used for case indexing and recovering
    - ANN Module looks for patterns of similarity between cases
  - Other proposals involve the ANN in the reasoning process
    - By extracting its knowledge to support the reasoning

Hybrid CBR

- Using the network in the index system has disadvantages
  - The network may need to be retrained every time a new case is added to the Case Base
    - Incremental learning can be used in order to train the network with the new cases
  - ANN can also be used for case acquisition

ANN measuring similarity



ANN and Fuzzy Systems

- Several approaches have been proposed
  - One of the most common hybrid system
    - If not the most common
  - Three main classes
    - Use of ANN to simulate a Fuzzy System
      - Allows insertion and extraction of fuzzy rules
    - Use of ANN to adapt parameters of a Fuzzy System
    - Systems composed by ANNs and classical fuzzy systems working independently

ANN and Statistics

- Statistical techniques have been used to:
  - pre-process input data
  - Define size of input window in temporal series analysis
  - Initialise weights
  - Control generalisation
  - Committees

ANN and Statistics

- There is considerable overlap between the ANNs and statistics
  - Statistics is concerned with data analysis
  - In ANNs terminology, statistical inference means learning to generalise from noisy data
    - Most ANNs that can learn to generalise effectively from noisy data are similar or identical to statistical methods

Future trends

- Fundamental research
  - Formal methods to deal with validation and verification of complex systems
  - Methods for the integration of different computation paradigms
  - Representation methods for various types of knowledge
  - Connections to cognitive phenomena
  - Tools for efficient implementation
  - Real-world applications to gain experience

Conclusion

- Defined HIS
- Described a general framework for HIS
- Presented some approaches for HIS
  - Rule extraction
  - Hybrid CBR
  - Evolutionary design of ANN