DRONACHARYA
College of Engineering

INTELLIGENT SYSTEMS (CSE-303-F)

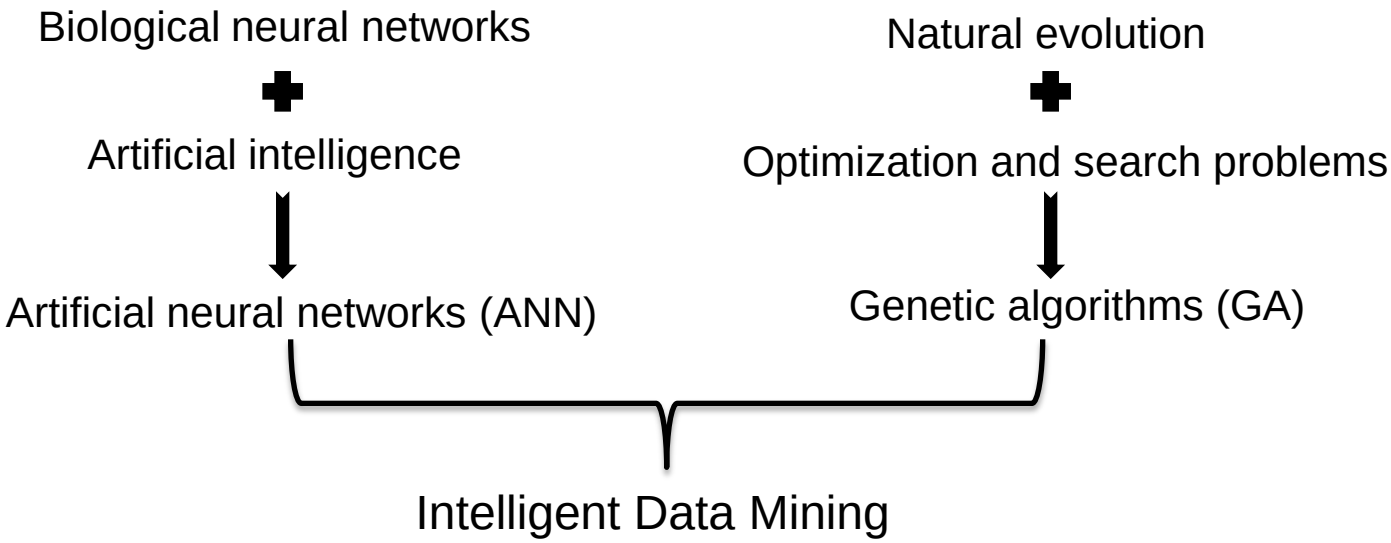
Section D

Genetic Algorithms in Artificial Neural Networks

Preface

"The human brain contains roughly 10^{11} or 100 billion neurons. That number approximates the number of stars in the Milky Way Galaxy, and the number of galaxies in the known universe. As many as 10^4 synaptic junctions may abut a single neuron. That gives roughly 10^{15} or 1 quadrillion synapses in the human brain. The brain represents an asynchronous, nonlinear, massively parallel, feedback dynamical system of cosmological proportions."

Kosko, Bart (1992)



DM Challenges and scope

- Developing a unifying theory of DM
- Scaling up for high dimensional data and high speed data streams
- DM for biological and environmental problems
- Mining complex knowledge from complex data

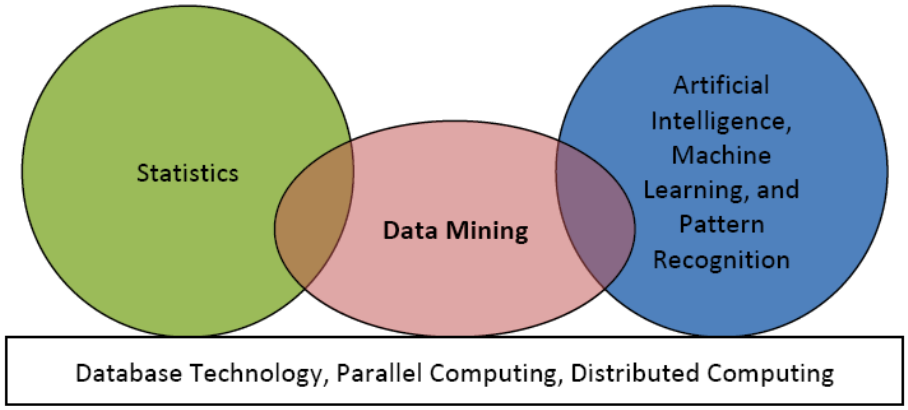


Figure 1-3: Data mining as a confluence of many disciplines (adapted from Tan, Steinbach et al. 2006)

• DM is an inter-disciplinary field of disciplines such as statistics, machine learning, Pattern Recognition (PR), Artificial Intelligence (AI), database technology …

Intelligent Data Mining

- DM techniques are all data analysis methods
  and can support/interact with each other.
- Each discipline has its own distinct attributes
  that make it particularly useful for certain types of problems and situations.
- Ex. the most fundamental difference
  between classical statistical applications and data mining
  is the size of the dataset.
- Intelligent System (IS) is all about learning rules and patterns from the data
- It is a collection of methodologies that works synergistically and provides,
  in one form or another,
  flexible information processing capability for handling real-life situations.
- It differs from conventional data analysis (e.g. statistical methods)
  in that it is tolerant of imprecision, uncertainty, partial truth, approximation and expolits it
  in order to achieve tractabillity, robustness, and low-cost solutions.

Artificial Neural Networks (ANNs)

- Biological neural systems (BNSs) can perform extraordinarily complex computations without recourse to explicit quantitative operations, and are capable of learning over time.
- Reflect the ability of large ensembles of neurons to learn through exposure to external stimuli and to generalize across related instances of the signal.
- Attractive as a model for IS methods.
- ANNs are distributed, adaptive, generally nonlinear means of learning comprised of different processing elements (PEs) called neurons.
- Based on a computing model similar to the underlying structure of the human brain, the aim being to model the brain's ability to learn and/or adapt in response to external inputs.

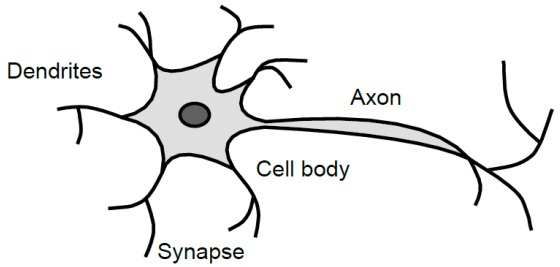


Figure 1-4: Schematic diagram of a biological neuron

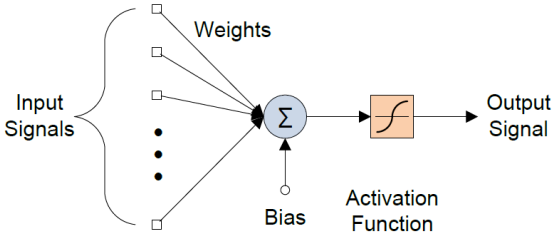


Figure 1-5: Illustration of an artificial neuron

ANNs – Advantages and Challenges

Advantages:

- Do not require a priori knowledge about the data, which is often the opposite to traditional statistical model-based methods.
- Have robustness and fault-tolerant capability.
- Can perform nonlinear modeling.
- Typically structured as parallel-processing structures.

Challenges:

- "**Black-box**" nature - even though they are successfully trained, no information is available from them in symbolic form, suitable for verification or interpretation by humans.
- Irrelevant variables may add extra noise which has consequential impact on the accuracy of the model
- As input dimensionality increases, the computational complexity and memory requirements of the model increase.

Algorithms for rule extraction

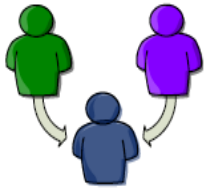
- **Decompositional approaches** - rule extraction at the level of hidden and output units, involves the extraction of rules from a network in a neuron-by-neuron series of steps.
- They can generate a complete set of rules for the trained ANNs.
- The process results in large and complex descriptions (exponential).
- **Pedagogical approaches** - map inputs directly into outputs and views ANNs as black-boxes where the aim is to extract symbolic rules which map the input-output relationship as closely as possible.
- The number of these rules and their form do not directly correspond to the number of weights or the architecture
- Sheer number of rules generated for even the simplest domains
- **Eclectic approaches -** incorporate elements of both decompositional and pedagogical techniques to complement a *symbolic learning algorithm.*
- Very little understanding for constructing and of the domains where it may outperform their traditional symbolic and ANN counterparts, and how to evaluate the results.

Genetic algorithms - biology

- Population – A set of individual solutions

- Fitness – A measure of an individual's ability to survive/reproduce

- Inheritance – Offspring resemble their parents



- Generation – Successively created generations

Figure 1-8: Evolutionary Computation components

- Organism has a set of rules (a blueprint) defining how it is built up from the tiny building blocks of life.
- Rules are encoded in the *genes*, which are connected together into long strings called *chromosomes*.
- Each gene represents a specific trait of the organism and has several different settings.
- Genes and their settings are usually referred to as an organism's *genotype*.
  The physical expression of the genotype(the organism itself) is called the *phenotype*.
- When two organisms mate, their resultant offspring ends up having shared genes - *recombination*.
- Occasionally a gene may be *mutated*.
- Life on earth has evolved to be as it is through the processes of natural selection, recombination and mutation.

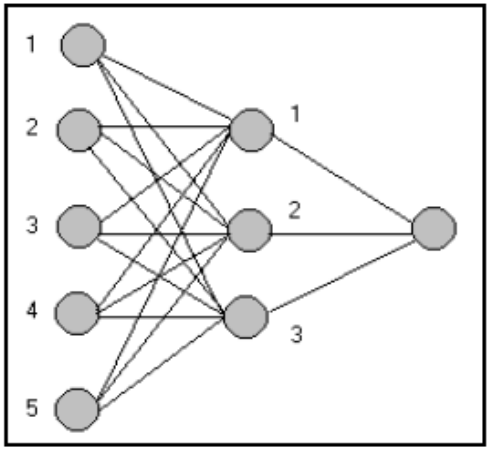
Genetic algorithm (GA)

- Before using a GA to solve a problem, a way must be found of *encoding* any potential solution to the problem. This could be as a string of real numbers or, more typically, a binary bit string. It is referred to as the chromosome. A typical chromosome may look like this: 10010101110101001010011101101
- At the beginning of a run a large population of random chromosomes is created. Each one, when decoded will represent a different solution to the problem at hand. Let's say there are N chromosomes in the initial population.
- The following steps are repeated until a solution is found:

1. Test each chromosome to see how good it is at solving the problem at hand and assign a fitness score accordingly. The fitness score is a measure of how good that chromosome is at solving the problem to hand.
2. Select two members from the current population. The chance of being selected is proportional to the chromosomes fitness. Roulette wheel selection is a commonly used method.
3. Dependent on the crossover rate crossover the bits from each chosen chromosome at a randomly chosen point.
4. Step through the chosen chromosomes bits and flip dependent on the mutation rate.
5. Repeat steps 2, 3, 4 until a new population of N members has been created

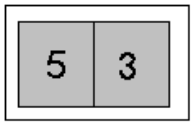
The Genetic Algorithm/Neural Network System (1/3)

- The starting point of any rule-extraction system is firstly to train the network on the data required,
  i.e. the ANN is trained so that a satisfactory error level is reached.
- For classification problems, each input unit typically corresponds to a single feature in the real world, and each output unit to a class value or class.
- The first objective of this approach is to encode the network in such a way that a genetic algorithm can be run over the top of it
  which is achieved by creating an n-dimensional weight space
  where n is the number of layers of weights.
- The network can be represented by simply
  enumerating each of the nodes and/or connections.
- Typically, there will be more than one output class
  or class value and therefore more than one output node.



GA/NN System (2/3)

- From encoded network, genes can be created which are used to construct chromosomes where there is at least one gene representing a node at the input layer and at least one for a node at the hidden layer.



A typical chromosome

  - This chromosome corresponds to the fifth unit in the input layer and the third unit in the hidden layer.
  - The first gene contains the weight connecting input node 5 to hidden unit 3, and the second gene contains the weight connecting hidden unit 3 to the output class.

- Fitness is computed as a direct function of the weights which the chromosome represents. For this chromosome the fitness function is:

$$Fitness = Weight(5 \rightarrow 3) * Weight(3 \rightarrow Output)$$

- This fitness is computed for an initial set of random chromosomes, and the population is sorted according to fitness.

GA/NN System (3/3)

- An elitist strategy is then used whereby a subset of the top chromosomes is selected for inclusion in the next generation. Crossover and mutation are then performed on these chromosomes to create the rest of the next population.
- The chromosome is then converted into IF…THEN rules with an attached weighting and is achieved by using the template: 'IF <gene1> THEN output is <class – output unit> (weighting - fitness)'
- The weighting is a major part of the rule generation procedure because the value of this is a direct measure of how the network interprets the data.
- The rule template above therefore allows the extraction of single-condition rules.
- The number of extracted rules in each population can be set by the user, according to the complexity of the network and/or the data. A larger number of rules will yield less fit chromosomes and thus less important rules.
- This property is essential in extracting rules which represent knowledge at the periphery of expertise.

Experiment 1

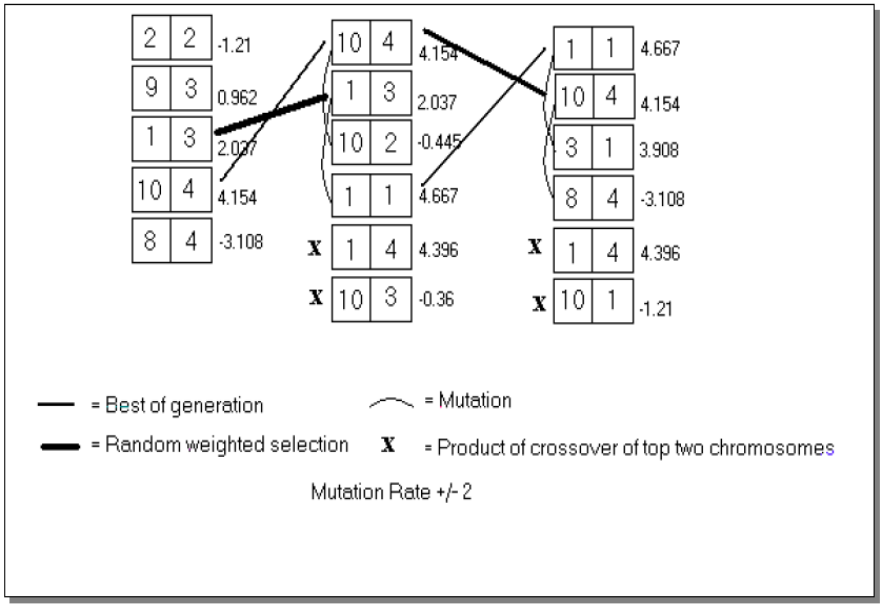
Name	Hair	Height	Weight	Lotion	Result
Sarah	Blonde	Average	Light	No	Sunburned
Dana	Blonde	Tall	Average	Yes	Not sunburned
Alex	Brown	Short	Average	Yes	Not sunburned
Annie	Blonde	Short	Average	No	Sunburned
Emily	Red	Average	Heavy	No	Sunburned
Pete	Brown	Tall	Heavy	No	Not sunburned
John	Brown	Average	Average	No	Not sunburned
Katie	Blonde	Short	Light	Yes	Not sunburned

- A random number generator was used to create the initial population of five chromosomes for the detection of rules, where an extra gene is added to the end of the chromosome to represent one of the two output class values.
- The alleles for this gene are either 1 or 2 (to represent the output node values of 10 (sunburned) and 01 (not sunburned).

- Example of input is: 10001010010
- NN with 11 input, 5 hidden and 2 output units was created.
- It was then trained (using back-propagation) until a mean square error of 0.001 was achieved.
- NN weights were then recorded and the genetic algorithm process started.

Hair	Blonde	100
	Brown	010
	Red	001
Height	Short	100
	Average	010
	Tall	001
Weight	Light	100
	Average	010
	Heavy	001
Lotion	No	10
	Yes	01
Class	Sunburned	10
	Not sunburned	01

Experiment 1 - result



- A traditional symbolic learning algorithm finds following four rules:
- (a) If person has red hair then person is sunburned;
- (b) If person is brown haired then person is not sunburned;
- (c) If person has blonde hair and no lotion used then person is sunburned;
- (d) If person has blonde hair and lotion used then person is not sunburned

GA rule findings:

1. IF unit1 is 1 THEN output is 1 (fitness 4.667)
2. IF unit 3 is 1 THEN output is 1 (fitness 3.908)
3. IF unit 10 is 1 then output is 1 (fitness 4.154)
4. IF unit 2 is 1 THEN output is 2 (fitness 8.43)
5. IF unit 11 is 1 THEN output is 2 (fitness 10.12)

Experiment 2

Run	Supervisor	Overtime	Operator	Output
1	Sally	Yes	Joe	High
2	John	No	Samantha	High
3	Sally	Yes	Joe	High
4	John	No	Joe	Low
5	Sally	Yes	Samantha	High
6	Patrick	No	Samantha	Low
7	Sally	Yes	Joe	High
8	Patrick	No	Samantha	Low

- Symbolic algorithms do not produce good results over this data set.
- *See5* creates the ruleset:

1. IF overtime = Yes THEN output = High [0.833]
2. IF overtime = No THEN output = Low [0.667]

- *CN2* creates these single-condition rules:

1. IF supervisor = Sally THEN output = High [0 4]
2. IF supervisor = Patrick THEN output = Low [2 0]

- The genetic algorithm was started with a population of 10 and run for just 20 generations.
- The top rules for each classification were as follows:
- IF Supervisor = John THEN output = High (12.948)
- IF Supervisor = Sally THEN output = High (10.966)
- IF Operator = Samantha THEN output = High (7.847)
- IF Overtime = No THEN output = Low (11.498)
- IF Operator = Joe THEN output = Low (10.706)
- IF Supervisor = Patrick THEN output = Low (7.120)

* The ANN with 7 input, 4 hidden and 2 output units was trained over a series of 1522 epochs to achieve a mean squared error of 0.040.

Supervisor	**Sally**	100
	John	010
	Patrick	001
Overtime	**Yes**	10
	No	01
Operator	**Joe**	10
	Samantha	01
Output	**High**	10
	Low	01

Experiment 3

- The dataset used was the mushroom dataset - a well-known collection of data used for classifying mushrooms into an edible or poisonous class.
- The data contains 125 categories spanning 23 attributes.
- Some categories were eliminated from the data and a smaller network with 30 hidden units was trained on the smaller 62 category data set for 69 epochs. The error was 0.03 but testing was.
- The genetic algorithm was run for 100 iterations with a population of 20. There were 7 operations per population, 4 crossover and 3 mutation.
  The mutation rate was randomly set between –40 to +40.
- Found rules:

1. IF odour=p THEN poisonous. (max 2.23) (found by CN2 and See5)
2. IF gill-size=n THEN poisonous. (max 1.13) (exclusive)
3. IF stalk-root = e THEN poisonous (max 1.13) (exclusive)
4. IF gill-size=b THEN edible. (max 2.3) (found by CN2)
5. IF odour=n THEN edible (max 1.58) (exclusive)
6. IF cap-surface=f THEN edible (max 1.58) (found by CN2)

Discussion – advantages

- This system essentially finds a collection of paths (rules) through the trained network to determine the optimal ones for a particular classification
- The preliminary results provide evidence of the feasibility of integrating GAs with trained neural networks, both technically and in terms of efficiency.
- The approach can be scaled up easily, with the major constraint on scale being the accuracy of the trained neural network when dealing with large datasets.
- Particularly interesting was the extraction of rules not captured by traditional symbolic learning techniques which lie at the periphery of domain expertise or which capture exceptions (which can then be further analysed to identify reasons for being exceptions).
- May be required in commercial applications of data mining, where the task is not to mine the data to extract rules which are already known to domain experts
- In short it utillises the best aspects of neural network learning in noisy domains with the best aspects of symbolic rules through the application of GAs.

Discussion - issues

- It is possible that one input unit can exert both a negative and a positive influence over the same classification. When fired, this unit could contribute in a large way towards the classification through one hidden unit, but it might also have another set of heavily negative connections to other hidden units which would negate that classification. In that case, the genetic algorithm will find the large positive and negative connections and interpret their effect separately, thereby creating erroneous and perhaps contradictory rules.
- If the network determines that a certain attribute is not contributing to a classification, it is far more likely to reduce the effect that that unit has on the network rather than increase two sets of weights. This is largely how backpropagation works, but it shows up a possible weakness in this approach if used on networks which have been trained using a different learning algorithm from backpropagation.
- Further experiments are required on ANNs of different types (e.g. competitive, non-supervised learning networks) and different architectures (e.g. of more than one hidden layer of neurons).