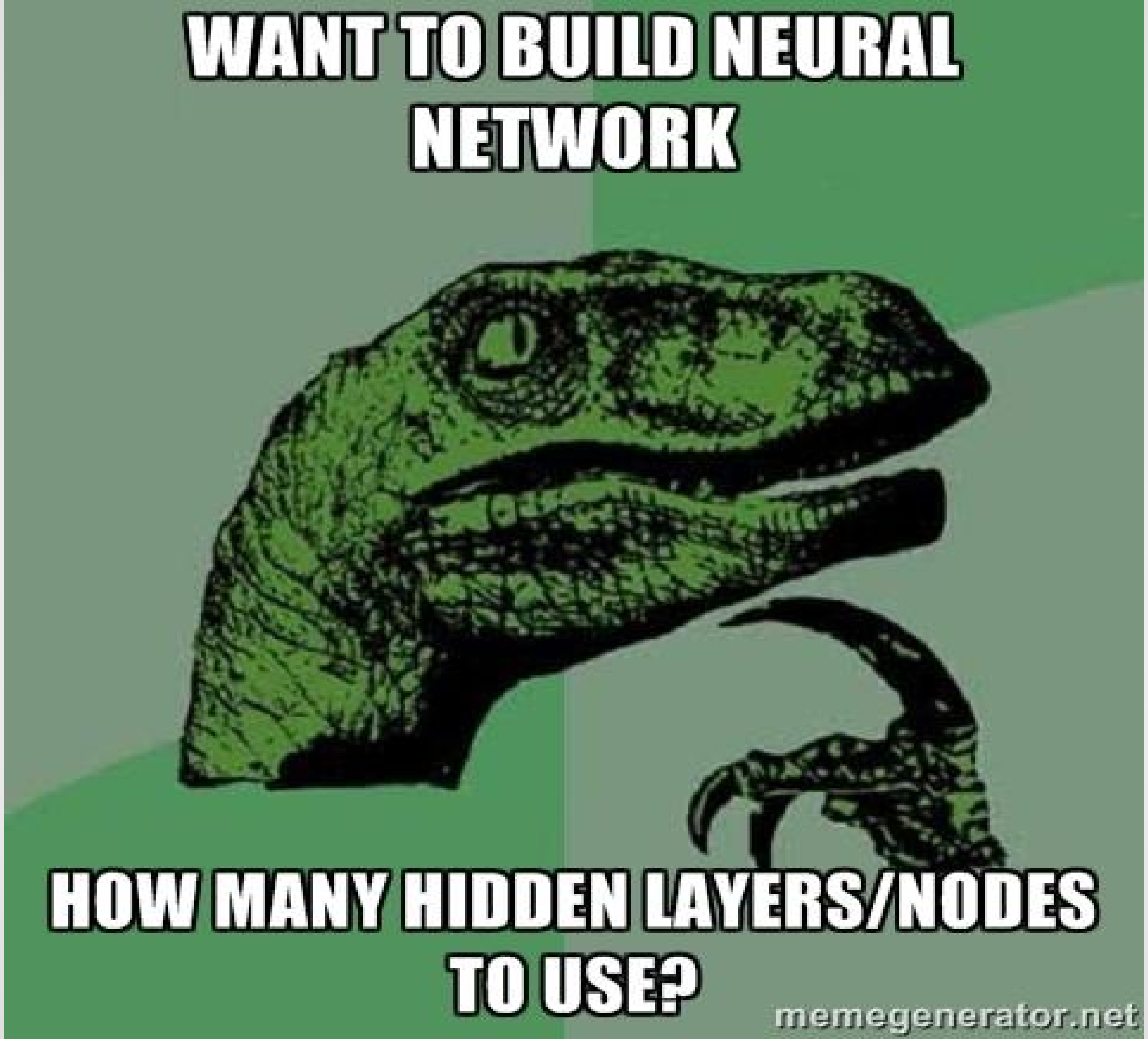
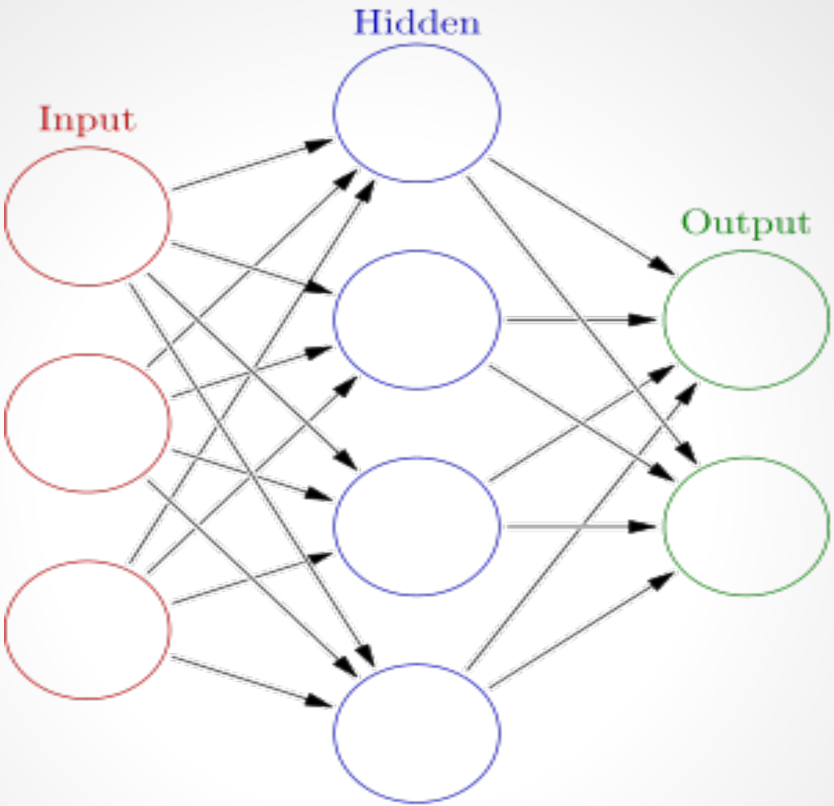
How many hidden layers and nodes ?

D.Stathakis, JRC European Commission 2009

Group 21
Aditya Kumar Akash 120050046
Deependra Patel 120050032
Nishant Kumar Singh 120050043


WANT TO BUILD NEURAL NETWORK
HOW MANY HIDDEN LAYERS/NODES TO USE?
memegenerator.net

Training



How many?

Motivation

- 2 decades, no exact solution until today
- Traditionally, it has been based on trial and error, heuristics, pruning and constructive methods
- None of them gives optimal or at least near-optimal solution
- This paper describes a genetic algorithm which aims to optimize performance while minimizing network complexity

Theoretical bounds of optimal topology

- In single hidden layer, number of neurons can be as high as number of training samples

- The purpose of using a second hidden layer is to drastically reduce the total required number of hidden node

- Huang (2003) proved that in the two-hidden-layer case, with m output neurons, the number of hidden nodes that are enough to learn N samples with negligibly small error is given by -

$$2\sqrt{(m+2)N}$$

- Specifically, he suggests that the sufficient number of hidden nodes

in the first layer is $\sqrt{(m+2)N} + 2\sqrt{N/(m+2)}$

and in the second is $m\sqrt{N/(m+2)}$

CURRENT METHODS

1. Trial and error

This is the most primitive path, and it will yield severely suboptimal structures.

2. Heuristic Search

- ❖ **Objective**

  To devise a formula that estimates the number of nodes in the hidden layers as a function of the number of input and output nodes

- ❖ Could be used to estimate a single exact topology or a range of topologies that should be searched.
- ❖ These heuristics are used prior to applying trial and error

eg: grapheme to phoneme asst., we take (n+m)/2

3. Exhaustive search

- ❖ Search through all possible topologies
- ❖ Very slow process
- ❖ Testing each topology takes lots of time
- ❖ Infeasible

4. Pruning and constructive algorithms

- Aim at devising an efficient network structure by incrementally adding/removing links

  Hessian matrix :-

$$h_{ij} = \frac{\partial^2 E}{\partial u_i \partial u_j}$$

- ❖ Optimal Brain Damage
  - ➢ Progressively removes weights that causes least increase in training error
  - ➢ To simplify computation, it assumes the hessian matrix of network is diagonal
- ❖ Optimal Brain Surgeon
  - ➢ It makes no such assumption and takes more time
  - ➢ Does not demand retraining after pruning the weights
  - ➢ Outperforms the rest

PROPOSED METHOD

- Uses genetic algorithm
- A novel fitness function is introduced to evaluate each solution
- This function aims at concurrently maximizing classification accuracy and at the same time minimizing network complexity

Introduction to genetic algorithm

- Inspired from natural selection
- Involves inheritance, selection, crossover and mutation
- Algorithm is started with a set of solutions (represented by chromosomes) called population
- Solutions from one population are taken and are used to form a new population

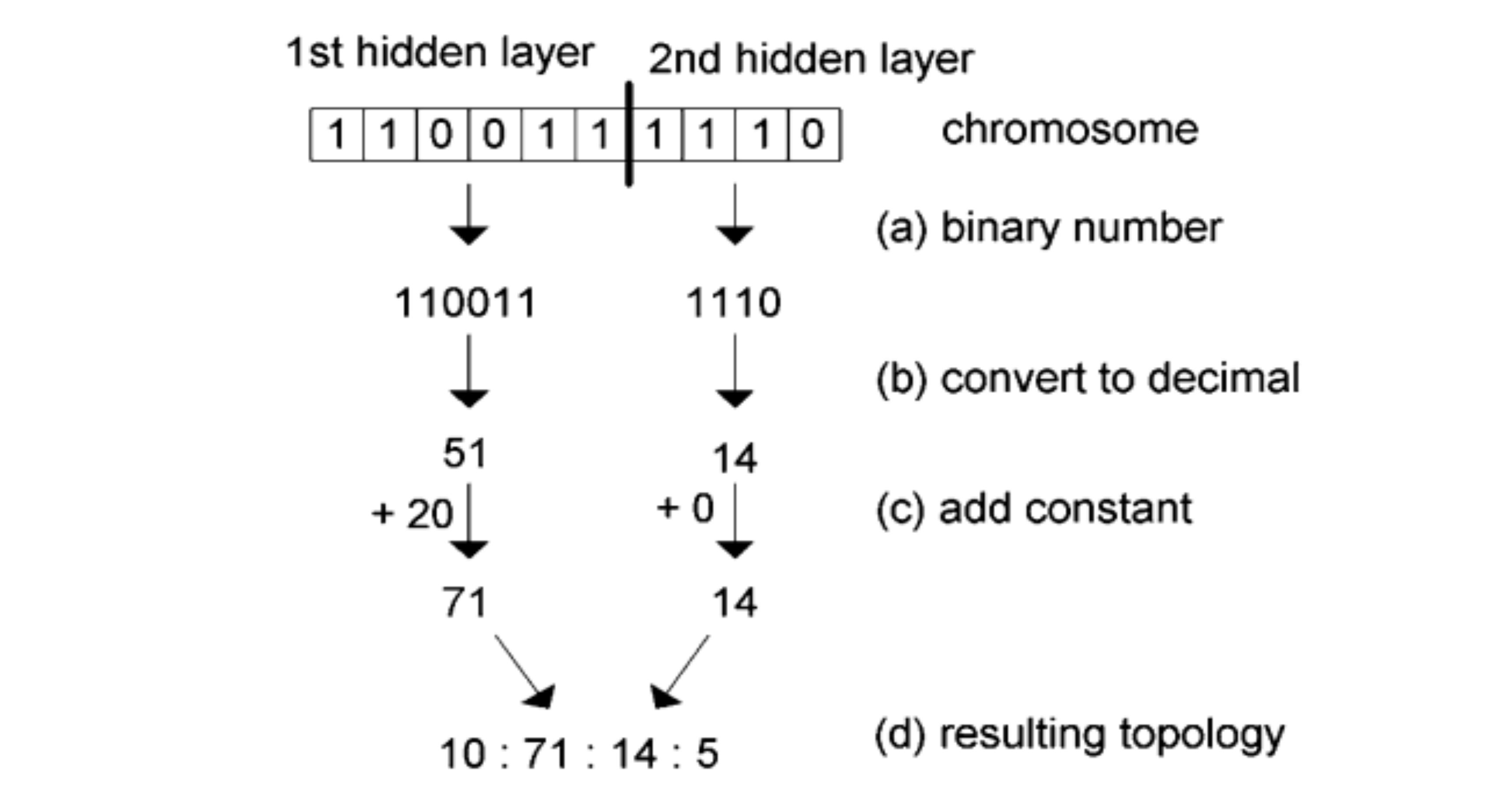
Continued..

- New population is better than the old one
- Solutions which are selected to form new solutions(offsprings) are selected according to their fitness
- The more suitable they are, more the chances they have to reproduce
- This process is repeated until either the maximum number of generations has been produced or a fitness level has been reached

Symbiosis of genetic algorithm and neural networks

- Topologies with upto two hidden layers are searched
- The number of nodes of the two hidden layers is coded into binary chromosome
- The length l of the chromosome is 10 bits
- First six bits are reserved for first layer and next four for the second layer

Continued..



1st hidden layer
2nd hidden layer
1 1 0 0 1 1 1 1 1 0
chromosome
(a) binary number
110011
1110
(b) convert to decimal
51
14
+ 20
+ 0
(c) add constant
71
14
(d) resulting topology
10 : 71 : 14 : 5

Continued..

- ❖ Convert binary numbers corresponding to each hidden layer into decimal numbers
- ❖ A constant is then added to each of those decimals
- ❖ For the second layer, constant is kept as 0 to account for the question - ‘How many layers?’. Possibility of network with one hidden layer

Continued..

- The number of bits required is decided by the two equations shown earlier
- In our example with 10 inputs and 5 outputs, topologies between 10:[20-83]:[0-15]:5 are searched
- The network is built based on the number of nodes dictated by the genetic algorithm
- It is then trained and its performance is evaluated according to the fitness function

Parameters of the Genetic Algorithm

- ❖ Two point crossover is performed with a probability of 0.75. Example:

  Parent 1: 1100|010|100

  Parent 2: 0101|001|011

  After crossover:

  Offspring 1: 1100|001|100

  Offspring 2: 0101|010|011

- ❖ Tournament selection is performed with tournament size = 4

Tournament selection

- A few individuals chosen at random from the population
- The winner of each tournament is selected with probability p
- Second best selected with probability p*(1-p), third p*((1-p)^2) and so on
- Many such tournaments are performed

Parameters of GA continued..

- Uniform mutation is performed with mutation probability of 0.01
- Mutation important for genetic diversity
- Arbitrary bit in the genetic sequence changed from its original state
- Fitness scaling is performed to maintain even selection pressure throughout the GA

Parameters of GA continued..

- The formulas are set according to Goldberg

(let n denote the population size and l the length of chromosome)

  - $n = \ell^2 \log^2_{10}(\ell)$
  - max function evaluations $= \ell \log_{10} \ell$
  - max generations $= \ell^2 \log^2_{10}(\ell) / n$

Fitness Function

- Novel Fitness function is used in the current genetic algorithm to rank the population

- Assumption Made : Compactness an additional benefit and hence the fitness function incorporates it

Formula for Novel Fitness fn

$$f = e + s\frac{c - c_{\min}}{c_{\max} - c_{\min}}.$$

- e - overall verification error for current topology
- c - complexity factor , measured as number of weights
- c_min, c_max depend on the given chromosome length
- s - accuracy sacrifice %

Novel Fitness fn conti.

Example - 1 1 0 0 1 1 | 1 1 1 0

- 1st layer, Max Nodes = (2^6 - 1) + 20 = 83, Min = 20
- 2nd layer, Max Nodes = (2^4 - 1) + 0 = 15, Min = 0
- Highest allowed structure = 10 : 83 : 15 : 5
- Lowest allowed structure = 10 : 20 : 5

Thus,

C_max = 10*83 + 83*15 + 15*5 = 2150

C_min = 10*20 + 20*5 = 300

Parameters of Fitness Function

- C_min, C_max used for normalization to smoothly incorporate the fitness function, Ideally C_min = 1, C_max = inf.
- ‘s’ - Denotes the sacrifice in accuracy for compactness, s = 1 denotes 1% sacrifice, s = 0 denoted compactness is no more an objective
- With s = 0 we exert on pressure on algorithm and hence the more accurate individuals prevail
- For easy classification s is relaxed while for difficult ones s is kept low

Properties of Fitness fn

- Only user defined parameter is ‘s’, intuitive meaning, Hence applicable in other areas
- Logarithmic scaling of complexity ?? *No need here since rank selection is adopted*. Relieves from setting the curvature of logarithmic function used in case of proportional functions
- Wide scaling - Highest scaled take over the population gene pool too quickly thus preventing algorithm from searching other areas
- On the other hand, if the scaled values vary little, all individuals have approximately the same chance of reproduction and the search progresses very slowly.

Experimental Results

Experimental data refers to ***Lefkas Island.***

Input Characteristics:

- 7 LANDSAT Bands
- Elevation
- Slope
- Aspect

Output Classes:

5 landuse classes : artificial surfaces, agricultural areas, forest and semi-natural areas, wetlands, and water bodies

Method Comparison

Heuristics

Method name	Reference	Range	Topology	Mean	Max	Min	σ
Kanellopoulos–Wilkinson rule also Hush rule	Kanellopoulos and Wilkinson (1997)	Low	10 : 20 : 5	70.54	71.87	68.48	0.69
		Medium	10 : 30 : 5	73.42	74.98	71.19	0.71
	Hush (1989); Kanellopoulos and Wilkinson (1997)	High	10 : 40 : 5	73.91	75.21	71.77	0.75

Obs : Larger topology estimate better, stated another way heuristics tend to underestimate the complexity of the network,

None suggest use of second hidden layer

Pruning

Method name	Reference	Range	Topology	Mean	Max	Min	σ
Optimal Brain Surgeon (OBD)	LeCun *et al.* (1990)			–	75.48	–	–
Optimal Brain Damage (OBS)	Hassibi and Stork (1993)			–	75.26	–	–

The maximum topology proposed among all heuristics, a 10 : 40 : 5 structure, is adopted as the starting point.

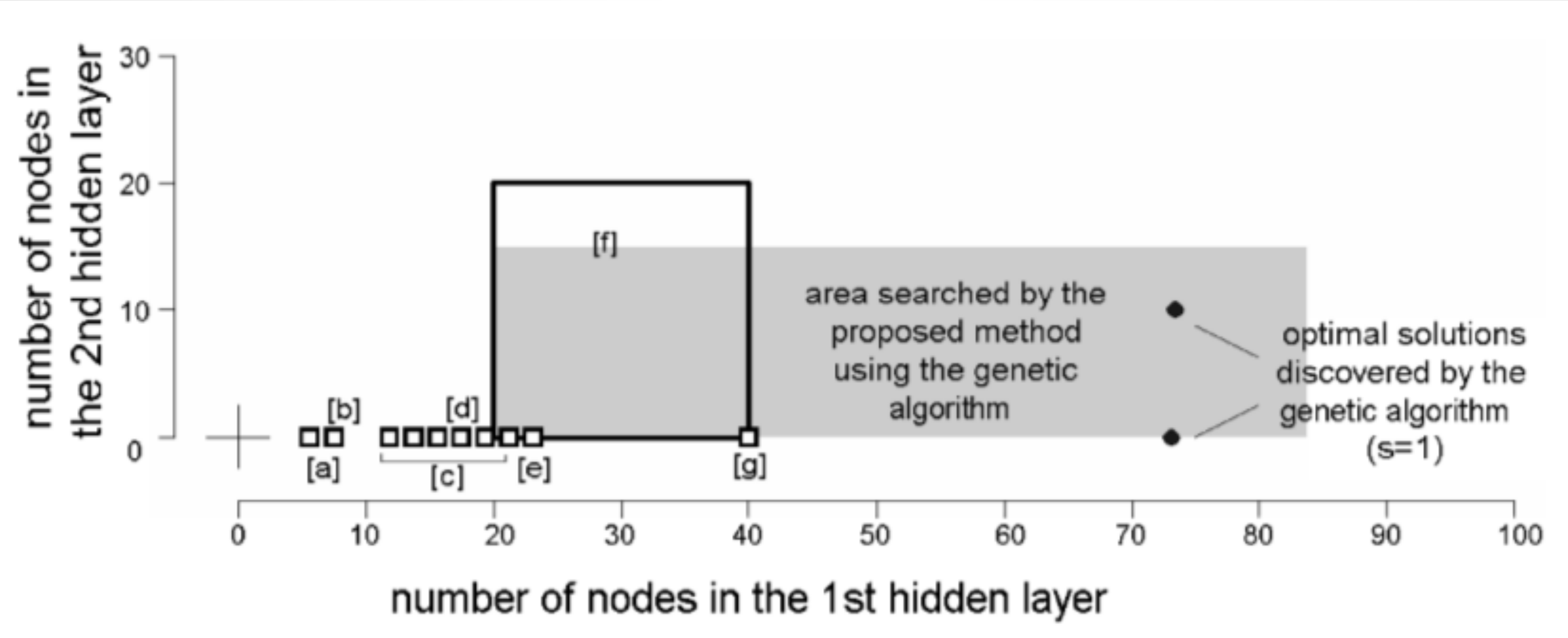
The computation resources required for both algorithms are high, with Optimal Brain Surgeon being more demanding.

Genetic algorithm - accuracy over compactness

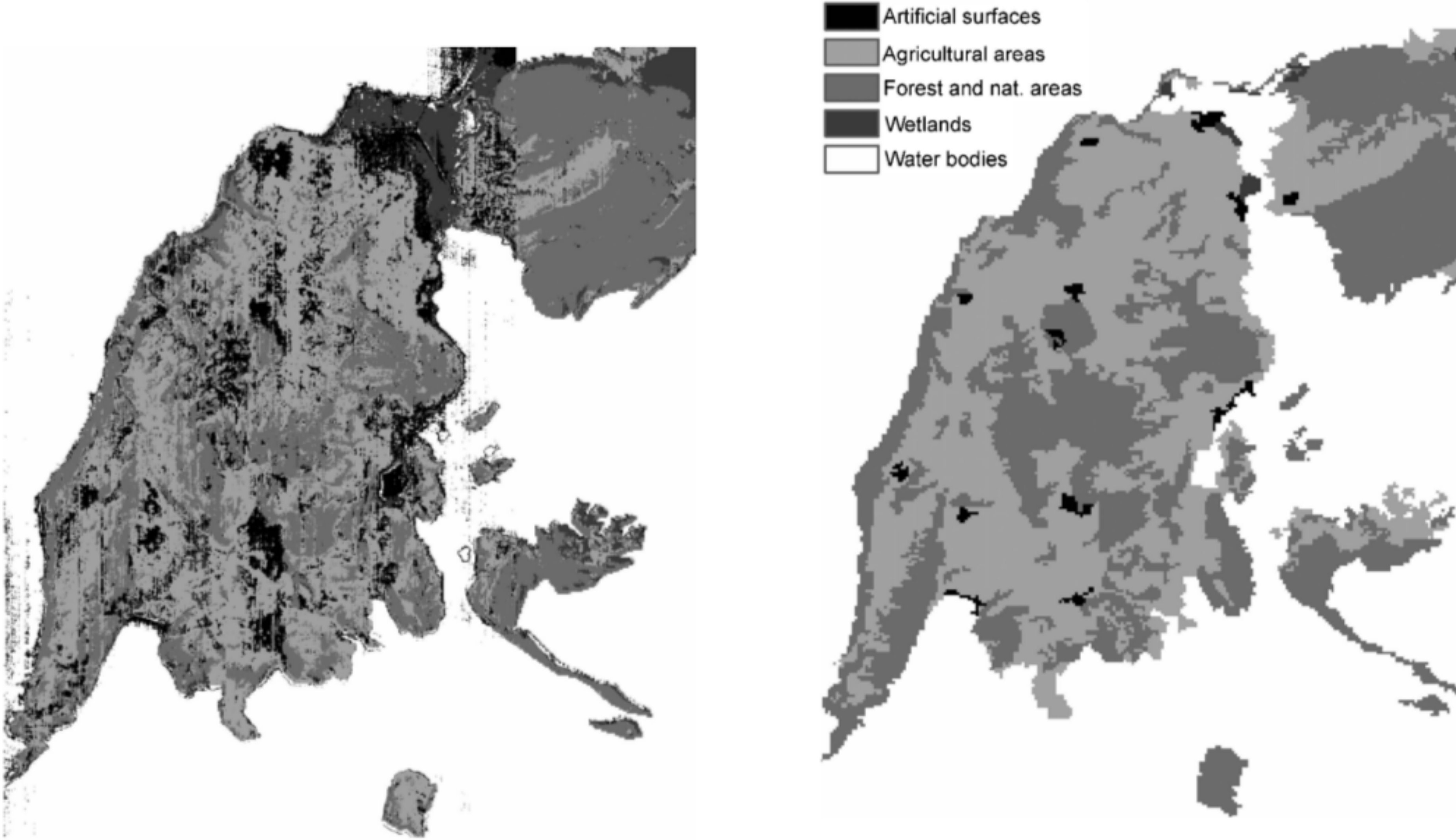
Method name	Reference	Range	Topology	Mean	Max	Min	σ
Genetic algorithm (s=1)	Proposed method	10 : [20–83] : [0–15] : 5	10 : 73 : 10 : 5	70.53	78.34	19.96	15.29
			10 : 73 : 5	71.23	77.03	19.96	15.29
Genetic algorithm (s=0)		10 : [20–83] : [0–15] : 5	10 : 74 : 14 : 5	71.62	79.63	24.33	13.44

- For s = 1, Two complexity solutions evolve, One of them has only 1 hidden layer
- The accuracy difference for s = 0, s = 1 is not much since the classification problem is difficult , so here it makes less sense to use s > 1

Solution Space Searched



The proposed method reveals several superior solutions


Artificial surfaces
Agricultural areas
Forest and nat. areas
Wetlands
Water bodies

CONCLUSION

- The *heuristics tend to underestimate complexity*.
- On the number of hidden layers, when seeking to optimize accuracy the *use of a second layer is desirable*
- *Ideally searching with the genetic algorithm* should *cover for the theoretical bounds of the number of hidden nodes* per layer

Questions?

Thank You

References

1. http://www.tandfonline.com/doi/pdf/10.1080/01431160802549278
2. http://en.wikipedia.org/wiki/Genetic_algorithm
3. http://www.obitko.com/tutorials/genetic-algorithms/parameters.php
4. http://www.obitko.com/tutorials/genetic-algorithms/ga-basic-description.php
5. http://en.wikipedia.org/wiki/Tournament_selection
6. http://en.wikipedia.org/wiki/Mutation_%28genetic_algorithm%29
7. http://www.cse.unr.edu/~sushil/class/gas/notes/scaling/index.html
8. http://www.nd.com/products/genetic/crossover.htm
9. http://yann.lecun.com/exdb/publis/pdf/lecun-90b.pdf
10. http://en.wikipedia.org/wiki/Hessian_matrix