Meta Learning

Sharan Vaswani

Learning Goals

- Explain the concept of bias for machine learning algorithms and formally define Meta Learning in its terms
- State at least two variations of Meta Learning
- Explain the basic concept of ensemble methods like Bagging, Boosting, Stacked Generalization
- Explain the basic idea behind Genetic Algorithms
- Explain how genetic algorithms can be used to learn learning rules and optimize hyper-parameters for simple neural networks

Outline

- Introduction
- Bias for learning algorithms
- Definition
- Variations of Meta Learning
- Ensemble Methods
- Neural Networks (introduction)
- Genetic Algorithms (introduction)
- Genetic Connectionism
- Genetic Algorithms for neural network hyper-parameter optimization

Introduction

- Meta Learning (social psychology) definition:
  "being aware of and taking control of one's own learning"

- Meta Learning (computer science) definition:
  "automatic learning algorithms are applied on meta-data about machine learning experiments "

- "Learning to Learn"

Bias for learning algorithms

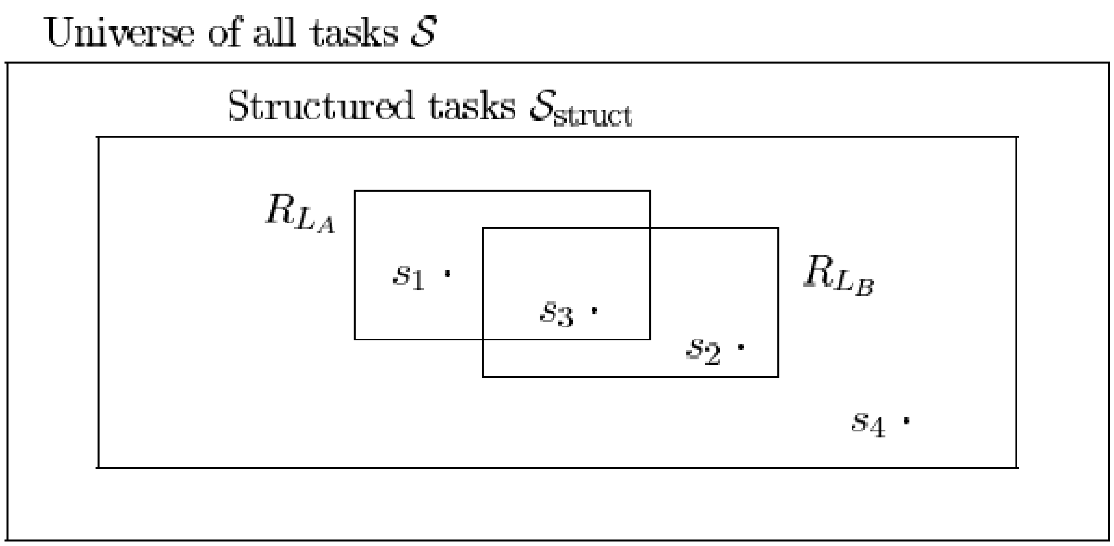


Figure 1. Each learning algorithm covers a region of (structured) tasks favored by its bias. Task s_1 is best learned by algorithm L_A, s_2 is best learned by algorithm L_B, whereas s_3 is best learned by both L_A and L_B. Task s_4 lies outside the scope of L_A and L_B.

- Need for Bias

Bias for learning algorithms

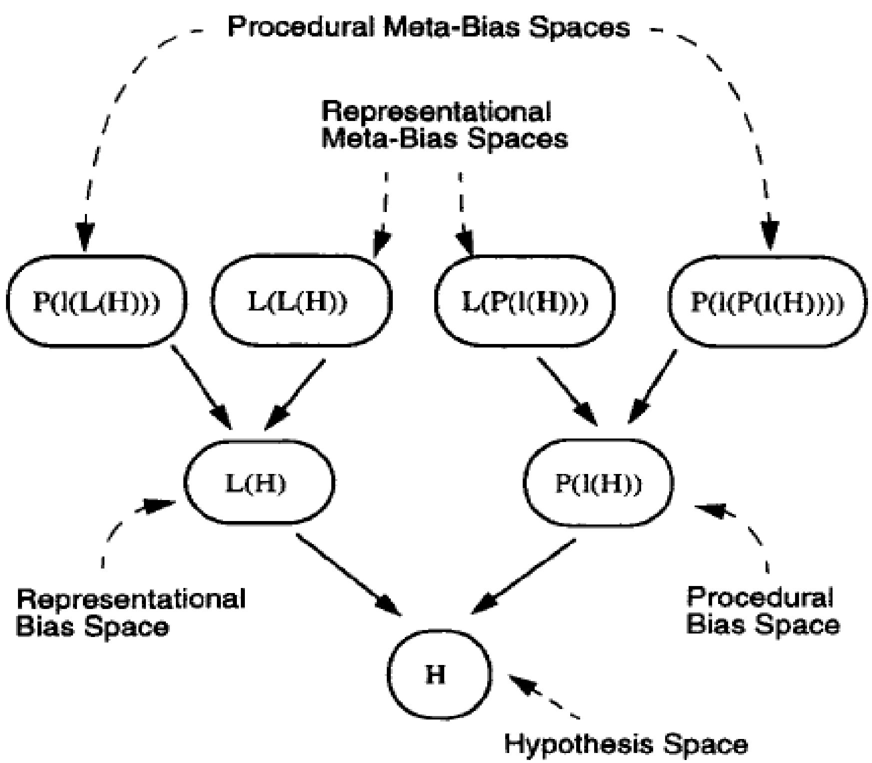


Figure 1. Multi-tiered bias search space.

Notation:
H :
Hypothesis space (learning algorithm runs in this space)

L(H) :
Each state is a language for describing the hypotheses in that space

P(L(H) :
Each state is search strategy for searching in the hypothesis spaces

Example

Exercise:

Features: size, shape, color

Dataset:

Small, cube, blue +1

Small , sphere, red -1

Example

Selected features in this case determine the hypothesis space

Case 1:

Features selected: size, color

h1: Small, blue - positive and small ; small, red – negative

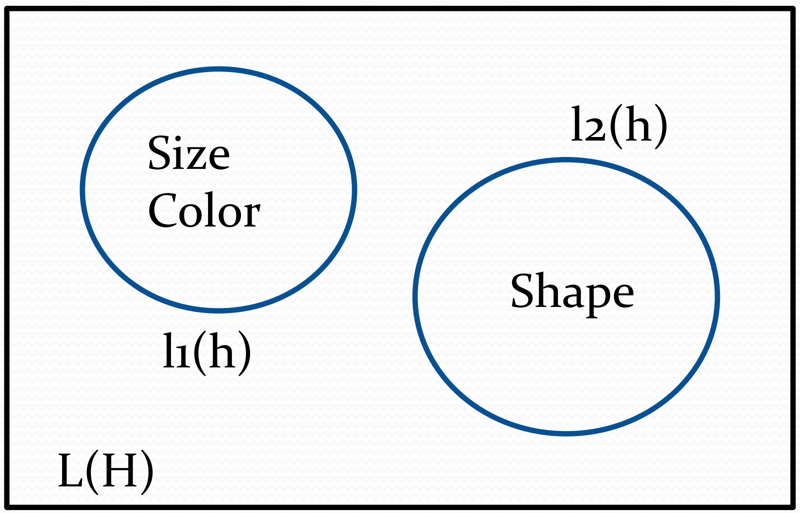
Case 2:

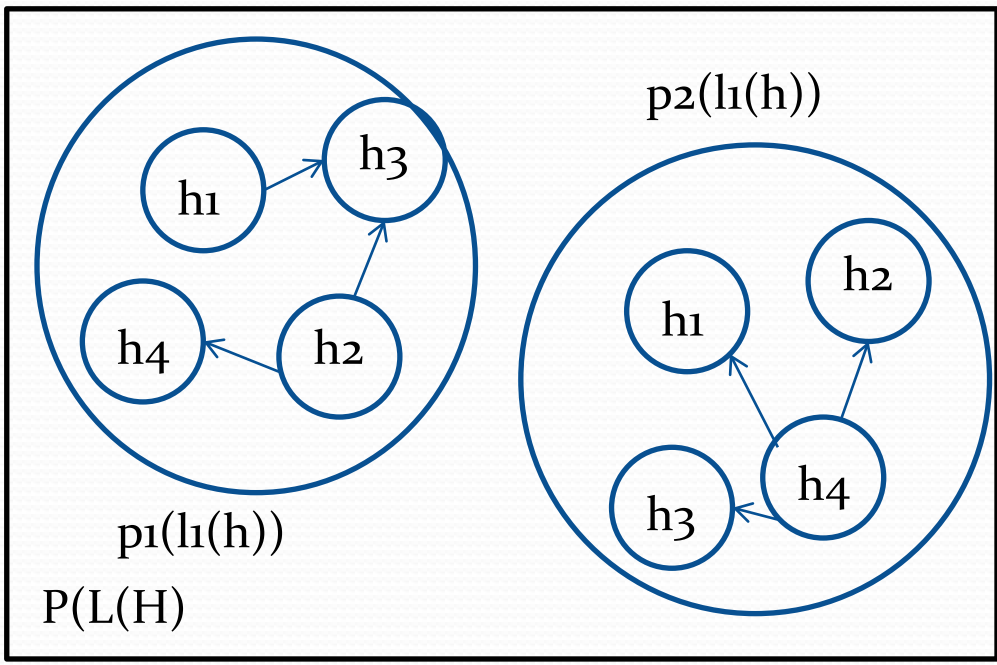
Features selected: shape

h2: Cube – positive ; sphere - negative

Both hypotheses are consistent with the training data

Example


Size
Color
l1(h)
l2(h)
Shape
L(H)


h1
h3
h4
h2
p1(l1(h))
p2(l1(h))
h1
h2
h3
h4
P(L(H))

Definition

Meta Learning affects the hypothesis space for the learning algorithm by either:

- Changing the hypothesis space of the learning algorithms (hyper-parameter tuning, feature selection)
- Changing the way the hypothesis space is searched by the learning algorithms (learning rules)

Variations of Meta Learning

- Algorithm Learning (selection)
  - Select learning algorithms according to the characteristics of the instance
- Hyper-parameter Optimization
  - Select hyper-parameters for learning algorithms. The choice of the hyper-parameters influences how well you learn.
- Ensemble Methods
  - Learn to learn "collectively" – Bagging, Boosting, Stacked Generalization

Variations of Meta Learning

- Dynamic bias selection
  - Adjust the bias of the learning algorithm dynamically to suit the new problem instance.
- Inductive Transfer
  - Learn to learn using previous knowledge from related tasks
- Learning to learn
  - In the sense of learning the learning rules for algorithms
- Fully self referential learners

Ensemble Methods

- Bagging
  - Randomly drawn (with replacement) subsets from the training data
  - Majority vote
- Adaboost
  - iteratively train classifiers
  - for each iteration, assign higher weights to training examples which were misclassified
- Random Forests
  - Bagging with random selection of features

Ensemble Methods

- Stacked generalization:
  - Given: dataset of N instances, k classifiers (level 0 classifiers)
  - Divide the training data set into J partitions – train each classifier k on J-1 folds and test it on the remaining partition
  - Prediction on the n^{th} instance by the k^{th} classifier = z_{kn}
  - Dataset to level 1 classifier = $\{y_n, z_{1n}, z_{2n} \dots z_{kn}\}$ for n = 1:N
  - Train on this dataset using meta classifier (level 1 classifier)

Exercise

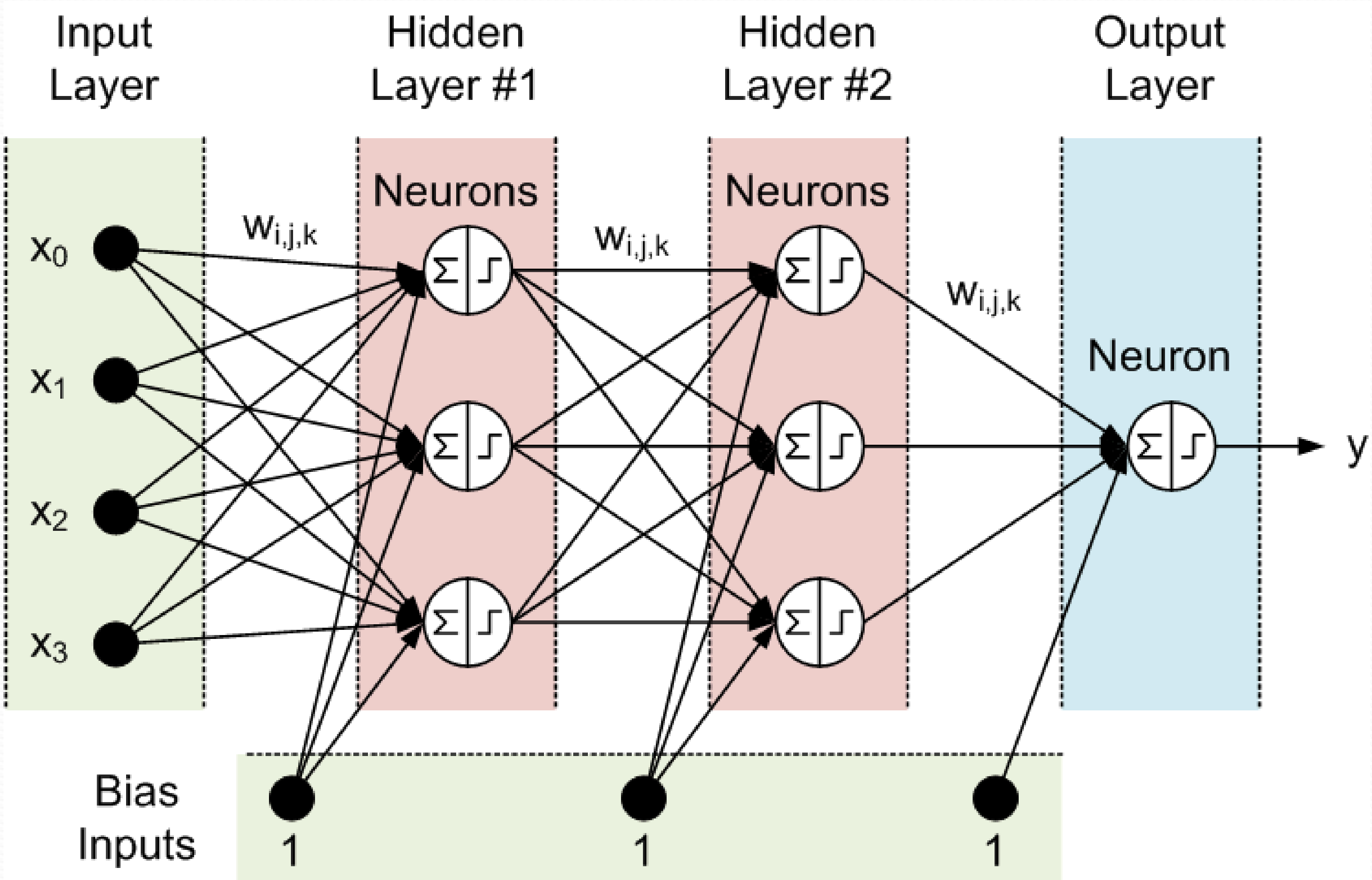
What if the meta classifier just chooses the feature which best correlates with the actual label for each test instance ?

Ans: Winner takes all

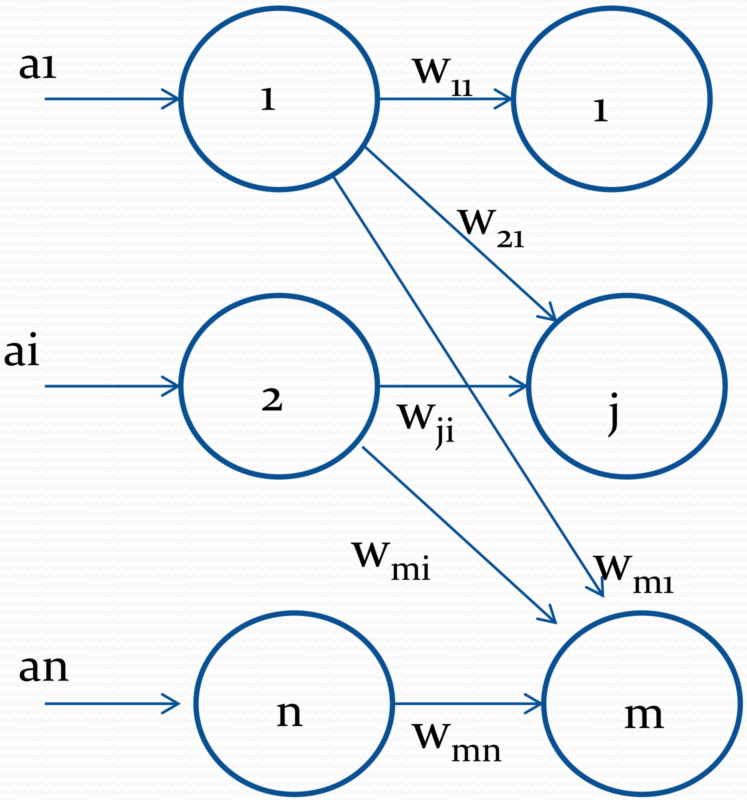
What if the instead of the k classifiers, there are actually k hyper-parameter values ?

Ans: Cross validation

Neural Networks

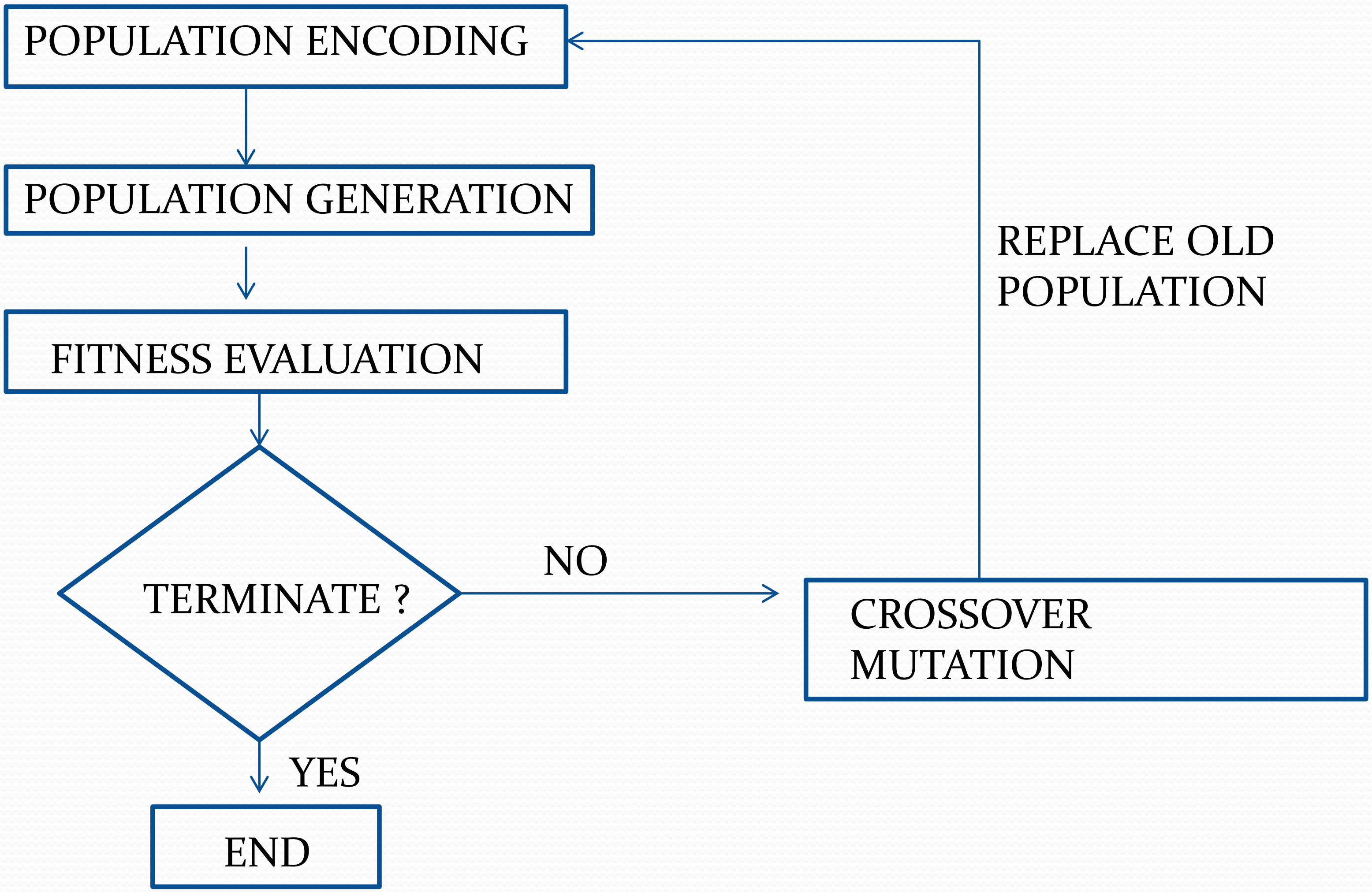


SINGLE LAYER NEURAL NETWORK AND THE DELTA RULE



- Independent of the neuron activation function
- Minimizes the squared error between the desired output value and the output neuron's activation
- $\Delta w_{ij} = c\ (t_i - o_i)\ a_j$

Genetic Algorithms



Exercise

Minimize $f(x) = (x-4)^2$ for integer x in the range [0,15]

- Population encoding:
  - 4 bit strings (0000,0010,1111)
- Population generation
  - Generate random 4 bit strings
- Fitness evaluation for string 1100
  - x = 8 => fitness = -16
- Crossover: 1 1 | 0 0

  0 0 | 1 0
  - children - 1110 and 0000
- Mutation on 0000: 0010

Genetic Connectionism

- Population Encoding - Encode possible learning rules as a binary string (pairwise products)

  length(chromosome) = 35 bits
- Population Generation – Generate random bit strings each encoding a different learning rule (Pop = 40)
- Fitness Evaluation – Use the learning rule to train the neural network for a variety of learning problems, use error on training instances as a measure of the fitness (for 20 different tasks)

Genetic Connectionism

- Crossover – 2 point crossover (crossover rate = 0.8)
- Mutation – Random bit flip (mutation rate = 0.01)
- Elitist selection strategy

Genetic Connectionism

- Learns the delta rule !

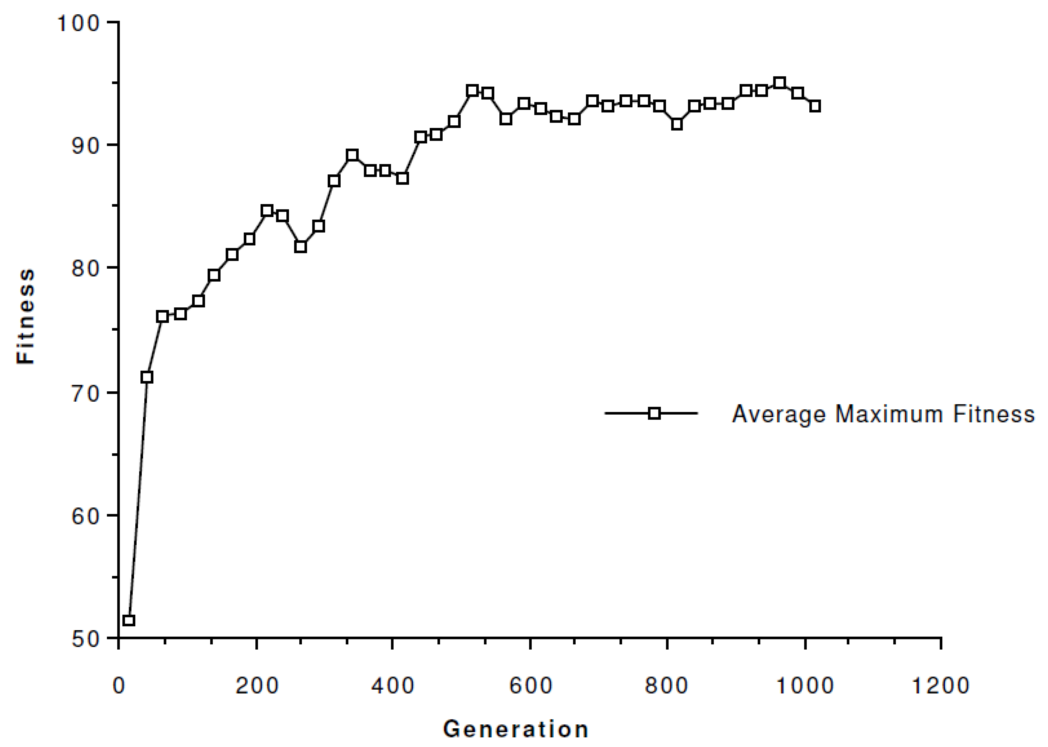


Figure 1. The evolution of maximum fitness in the population.

Fitness % =
Total Error for different tasks / No of examples

Genetic Connectionism

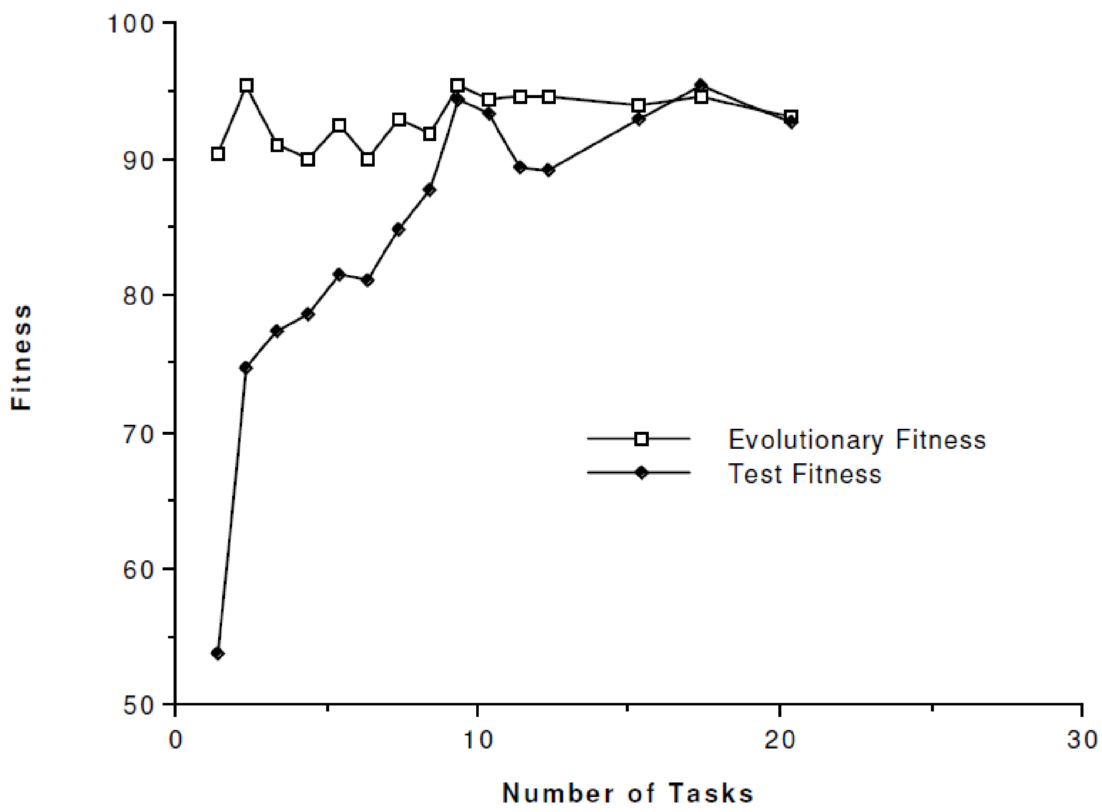


Figure 2. Evolutionary fitness and test fitness, versus number of tasks.

Fitness vs Number of tasks used in training

Neural Network Hyper-parameter Optimization

- Can learn weights of the neural network too (is this meta learning ?)
- Can learn genetic algorithms to learn hyper-parameters like number of hidden neurons, number of hidden layers, activation functions
- Encode the neural network parameters in a chromosome and train the NN using back-prop
- Can do all of the above simultaneously with different rates of evolution

Neural Network Hyper-parameter Optimization

- Genetic algorithms become ineffective when the chromosome becomes too long
- Can we still learn learning rules for neural networks ?

Learning learning rules for complex neural networks

Fixed weight recurrent neural networks

- Parameters specific to the task to be learnt is encoded in the self loops of the RNN
- Weights encode the learning algorithm
- Changing Weights => Using a different learning algorithm
- Can change the weights using gradient descent => use backprop to meta-learn !

References

- Chalmers, David J. "The evolution of learning: An experiment in genetic connectionism." *Proceedings of the 1990 connectionist models summer school. San Mateo, CA, 1990.*
- Vilalta, Ricardo, and Youssef Drissi. "A perspective view and survey of meta-learning." *Artificial Intelligence Review 18.2 (2002): 77-95.*
- http://www.scholarpedia.org/article/Ensemble_learning
- Wolpert, David H. "Stacked generalization." *Neural networks 5.2 (1992): 241-259*

References

- Younger, A. Steven, Peter R. Conwell, and Neil E. Cotter. "Fixed-weight on-line learning." *Neural Networks, IEEE Transactions on 10.2 (1999): 272-283.*
- Younger, A. Steven, Sepp Hochreiter, and Peter R. Conwell. "Meta-learning with backpropagation." *Neural Networks, 2001. Proceedings. IJCNN'01. International Joint Conference on. Vol. 3. IEEE, 2001.*
- Abraham, Ajith. "Meta learning evolutionary artificial neural networks."*Neurocomputing 56 (2004): 1-38*
- Gödel machines: Fully self-referential optimal universal self-improvers. *Artificial general intelligence. Springer Berlin Heidelberg, 2007. 199-226.*

References

- Gordon, Diana F., and Marie Desjardins. "Evaluation and selection of biases in machine learning." *Machine Learning 20.1-2 (1995): 5-22.*
- Mitchell, Tom M. *The need for biases in learning generalizations. Department of Computer Science, Laboratory for Computer Science Research, Rutgers Univ., 1980.*