

Evolutionary Learning

Hans

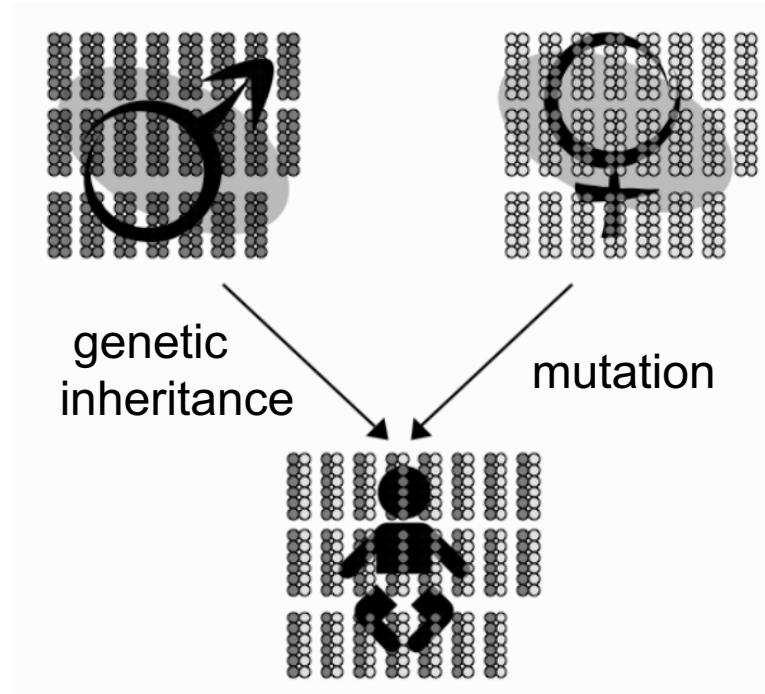
21.01.2021

Evolution

- Objective: live longer → e.g. {

the ability to defence bacteria and viruses

smart brain to do the Ph.D. work
- When does the evolution take place?



Genetic Algorithm

- Knapsack Problem:

Suppose that you are packing for your holidays. You've bought the biggest and best rucksack, but there is still no way that you are going to fit in everything you want to take and the things that your mum is insisting you take. You decide to measure how much space it takes up and then write a program to work out how to fill as much of the bag as possible.

Genetic Algorithm

Step 1:

- Seek for a method for representing problems as chromosomes
- String Representation
- Knapsack Problem: take the item or not (BINARY string)
e.g. one possible case: (0, 1, 1, 0)
- How to judge which solution is better?

Genetic Algorithm

Step 2:

- Find a way to calculate the fitness of a solution
- **Fitness function:** take a string as an argument and return a value for that string
 - The best solution holds the highest fitness.
 - The measure of fitness changes over time, while we'll ignore it in the genetic algorithm.
- Knapsack Problem:
 - The volume of the bag, e.g. 500
 - The volume of each item, e.g. 300, 200, 150, 250
 - Total volume for each string, e.g. (0, 1, 1, 0) -> 350
 - If string volume is less than bag volume, keep it.
 - If string volume is more than bag volume, subtract twice the over amount,
e.g. (0, 1, 1, 1) -> 600(100) -> 400
- **Population:** all the candidate strings

Genetic Algorithm

Step 3:

- Identify a selection method to choose parents:
 - **Tournament Selection:** Repeatedly pick four strings from the population, with replacement and put the fittest two of them into the mating pool.
 - **Truncation Selection:** Pick some fraction of the best strings and ignore the rest.
 - **Fitness Proportional Selection:** Select strings with a probability based on its fitness.

$$p^{\alpha} = \frac{\exp(sF^{\alpha})}{\sum_{\alpha'} \exp(sF^{\alpha'})}.$$

Genetic Algorithm

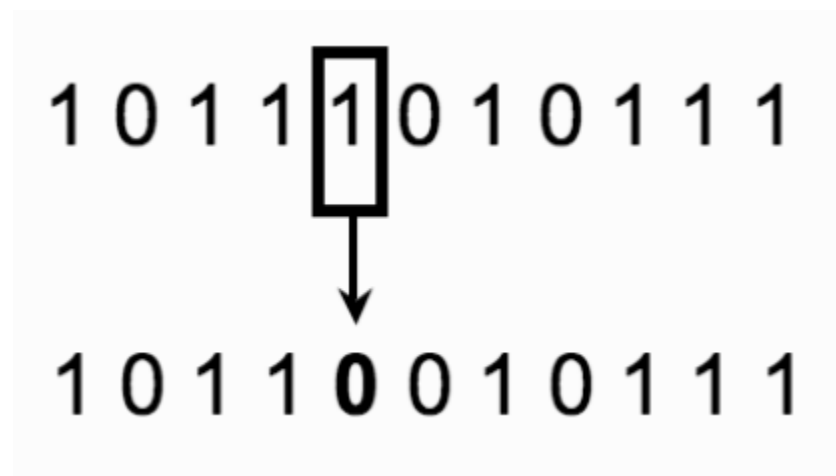
Step 4:

- Look for a way to generate offspring by breeding the parents
 - **Single point crossover:** A position in the string is chosen randomly, and the offspring is made up of the first part of parent 1 and the second part of parent 2.
 - **Multi-point crossover:** Multiple points are chosen, with the offspring being made in the same way.
 - **Uniform crossover:** Random binary numbers are used to select which parent to take each element from.


(a)		(b)		(c)	
<div>1 0 0 1 1 0 0 0 1 0 1</div> <div>0 1 1 1 1 0 1 0 1 1 0</div> <div>1 0 0 1 1 0 1 0 1 1 0</div> <div>0 1 1 1 1 0 0 0 1 0 1</div>		<div>1 0 0 1 1 0 0 0 1 0 1</div> <div>0 1 1 1 1 0 1 0 1 1 0</div> <div>1 0 0 1 1 0 1 0 1 0 1</div> <div>0 1 1 1 1 0 0 0 1 1 0</div>		<div>Random Samples 0 0 1 1 0 1 1 0 1 1 0</div> <div>String 0 1 0 0 1 1 0 0 0 1 0 1</div> <div>String 1 0 1 1 1 1 0 1 0 1 1 0</div> <div>1 0 1 1 1 0 1 0 1 1 1</div> <div>0 1 0 1 1 0 0 0 1 0 0</div>	

Mutataion

- Probability $p \approx 1/L$ where L is the string length
- Approximately one mutation in each string
- The effects of mutation on a string:



Offspring Population

- **Elitism:** The top N strings in the current generation are selected to replace the worst N strings in the next generation.
 - **Tournaments:** The two parents and their two offsprings compete, with the two fittest out of the four being put into the new population.
 - **Niching(Island Populations):** The population is separated into several subpopulations, which all evolve independently for some period of time, and a few members of one subpopulation are occasionally injected into another subpopulation.
- 
- Local Optimum

Thank You!