Machine Learning: Lecture 12

Genetic Algorithms

(Based on Chapter 9 of Mitchell, T., *Machine Learning*, 1997)

Overview of Genetic Algorithms (GAs)

- GA is a learning method motivated by analogy to biological evolution.
- GAs search the hypothesis space by generating successor hypotheses which repeatedly mutate and recombine parts of the best currently known hypotheses.
- In Genetic Programming (GP), entire computer programs are evolved to certain fitness criteria.

General Operation of GAs

- ***Initialize Population***: generate ***p*** hypotheses at random.
- ***Evaluate***: for each ***p***, compute ***fitness(p)***
- While Max_h ***Fitness(h) < Threshold*** do
  - ***Select***: probabilistically select a fraction of the best ***p***'s in ***P.*** Call this new generation P_{New}
  - ***Crossover:*** probabilistically form pairs of the selected p's and produce two offsprings by applying the crossover operator. Add all offsprings to $P_{new.}$
  - ***Mutate***: Choose m% of P_{New} with uniform probability. For each, invert one randomly selected bit in its representation.
  - ***Update***: $P \leftarrow P_{new}$
  - ***Evaluate***: for each ***p*** in ***P***, compute ***fitness(p)***
- Return the hypothesis from ***P*** that has the highest fitness.

Representing Hypotheses

- In GAs, hypotheses are often represented by bit strings so that they can be easily manipulated by genetic operators such as mutation and crossover.
- ***Examples***:

(Outlook = Overcast v Rain) ^ (Wind = Strong)

<=> ***011 10***

IF Wind = Strong THEN PlayTennis = yes

<=> ***111 10 10***

where group 1 = 3-valued outlook,
group 2 = 2-valued Wind
group 3 = 2-valued PlayTennis

Genetic Operators

- **Crossover Techniques:**
  - Single-point Crossover.
    Mask example: 11111000000
  - Two-point Crossover.
    Mask example: 00111110000
  - Uniform Crossover.
    Mask example: 10011010011
- **Mutation Techniques:**
  - Point Mutation
- **Other Operators**:
  - Specialization Operator
  - Generalization Operator

Fitness Function and Selection

- A simple measure for modeling the probability that a hypothesis will be selected is given by the ***fitness proportionate selection*** (or ***roulette wheel*** selection):

$$Pr(h_i) = Fitness(h_i) / \Sigma_{j=1}^{p} Fitness(h_j)$$

- Other methods: ***Tournament Selection*** and ***Rank Selection.***
- In classification tasks, the Fitness function typically has a component that scores the classification accuracy over a set of provided training examples. Other criteria can be added (e.g., complexity or generality of the rule)

Hypothesis Space Search (I)

- GA search can move very abruptly (as compared to Backpropagation, for example), replacing a parent hypothesis by an offspring that may be radically different from the parent.
- The problem of ***Crowding***: when one individual is more fit than others, this individual and closely related ones will take up a large fraction of the population.
- ***Solutions:***
  - Use tournament or rank selection instead of roulette selection.
  - Fitness sharing
  - restrict ion on the kinds of individuals allowed to recombine to form offsprings.

Hypothesis Space Search (II): The Schema Theorem [Holland, 75]

- ***Definition:*** A schema is any string composed of 0s, 1s and *s where * means 'don't care'.
- ***Example:*** schema 0*10 represents strings 0010 and 0110.
- ***The Schema Theorem:*** More fit schemas will tend to grow in influence, especially schemas containing a small number of defined bits (i.e., containing a large number of *s), and especially when these defined bits are near one another within the bit string.

Genetic Programming: Representing Programs

- ***Example:*** $sin(x)+sqrt(x^2+y)$

Genetic Programming: Crossover Operation

- ***Example:***

Models of Evolution and Learning I: Lamarckian Evolution [Late 19th C]

- ***Proposition:*** Experiences of a single organism directly affect the genetic makeup of their offsprings.
- ***Assessment***: This proposition is wrong: the genetic makeup of an individual is unaffected by the lifetime experience of one's biological parents.
- ***However***: Lamarckian processes can sometimes improve the effectiveness of computerized genetic algorithms.

Models of Evolution and Learning II: Baldwin Effect [1896]

- If a species is evolving in a changing environment, there will be evolutionary pressure to favor individuals with the capability to learn during their lifetime.
- Those individuals who are able to learn many traits will rely less strongly on their genetic code to "hard-wire" traits. As a result, these individuals can support a more diverse gene pool, relying on individual learning of the "missing" or "sub-optimized" traits in the genetic code. This more diverse gene pool can, in turn, support more rapid evolutionary adaptation. Thus the capability of learning can accelerate the rate of evolutionary adaptation of a population.

Parallelizing Genetic Algorithms

- GAs are naturally suited to parallel implementation. Different approaches were tried:
  - Coarse Grain: subdivides the population into distinct groups of individuals (***demes***) and conducts a GA search in each deme. Transfer between demes occurs (though infrequently) by a migration process in which individuals from one deme are copied or transferred to other demes
  - Fine Grain: One processor is assigned per individual in the population and recombination takes place among neighboring individuals.