Learning Classifier Systems

Andrew Cannon
Angeline Honggowarsito
Contents

- Introduction to LCS / LCS Metaphor
- The Driving Mechanism
  - Learning
  - Evolution
- Minimal Classifier System
- Michigan VS Pittsburgh

- Categories of LCS
- Optimisation
- Application: data mining
Introduction

- Our world is a Complex System
  - Interconnected parts
  - Properties exhibited by collective parts might be different from individual parts

- Adaptive
  - Capacity to change and learn from experience
Introduction

- John Holland in 1975
  - New York city, as a system that exists in a steady state of operation, made up of “buyers, sellers, administrations, streets, bridges, and buildings that are always changing. Like the standing wave in front of a rock in a fast-moving stream, a city is a pattern in time.”
Rule-Based Agents

- Represented by rule-based agents.
  - Agents - Single Components
- IF condition THEN action
- Use system’s environment information to make decision
Metaphor

- Two biological Metaphors:
  - Evolution
  - Learning
- Genetic Algorithm & Learning Mechanism
- Environment of the system
- Example
  - Robots navigating maze environment
The Driving Mechanism

Discovery - The Genetic Algorithm

- Rule Discovery
- Apply Genetic Algorithm
  - The fitness function quantifies the optimality of a given rule
- Classification Accuracy most widely used as metric of fitness
The Driving Mechanism

Learning

◦ “The improvement of performance in some environment through the acquisition of knowledge resulting from experience in that environment.”
◦ Each classifier has one or more parameters
◦ Iteratively update the parameters
Learning

- Purposes:
  - Identify useful classifiers
  - Discovery of better rules

- Different problem domains require different styles of learning.

- Learning based on the information provided
  - Batch Learning
    - Training instances presented simultaneously.
    - End result: rule set that does not change with respect to time.
Learning

- Incremental learning
  - One training instances at a time
  - End result:
    - Rule set that changes continuously

- Learning based on type of feedback
  - Supervised Learning
  - Reinforcement Learning
Minimal Classifier System

- Basic LCS Implementation
- Developed by Larry Bull
- Advancing LCS theory
- Designed to understand more complex implementations, instead of solving real world problems.
Minimal Classifier System

![Diagram of Minimal Classifier System]

**Figure 3:** MCS algorithm—an example iteration.
Minimal Classifier System

- **Input Data**
  - 4 Digit binary Number

- Learning Iteratively, one instance at a time

- **Population [N]**
  - Condition \{C\}
  - Action \{A\}
  - Fitness Parameter \{F\}

- Population is randomly initialized
Population

- **Condition**
  - String of “0, 1, #”
  - 00#1 matches 0011 or 0001

- **Action**
  - Which action is possible (0 or 1)

- **Fitness Parameter**
  - How good is the classifier

<table>
<thead>
<tr>
<th>C</th>
<th>A</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>00#1</td>
<td>1</td>
<td>88</td>
</tr>
<tr>
<td>0##1</td>
<td>0</td>
<td>17</td>
</tr>
<tr>
<td>#010</td>
<td>1</td>
<td>34</td>
</tr>
<tr>
<td>001#</td>
<td>1</td>
<td>91</td>
</tr>
<tr>
<td>0#11</td>
<td>0</td>
<td>66</td>
</tr>
<tr>
<td>11#0</td>
<td>0</td>
<td>7</td>
</tr>
</tbody>
</table>
**Match Set**

- Population Scanned
- Match Set: List of rules whose condition matches the input string at each position
- Input = 0011

<table>
<thead>
<tr>
<th>C</th>
<th>A</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>00#1</td>
<td>1</td>
<td>88</td>
</tr>
<tr>
<td>0##1</td>
<td>0</td>
<td>17</td>
</tr>
<tr>
<td>#010</td>
<td>1</td>
<td>34</td>
</tr>
<tr>
<td>001#</td>
<td>1</td>
<td>91</td>
</tr>
<tr>
<td>0#11</td>
<td>0</td>
<td>66</td>
</tr>
<tr>
<td>11#0</td>
<td>0</td>
<td>7</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>C</th>
<th>A</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>00#1</td>
<td>1</td>
<td>88</td>
</tr>
<tr>
<td>0##1</td>
<td>0</td>
<td>17</td>
</tr>
<tr>
<td>#010</td>
<td>1</td>
<td>34</td>
</tr>
<tr>
<td>001#</td>
<td>1</td>
<td>91</td>
</tr>
<tr>
<td>0#11</td>
<td>0</td>
<td>66</td>
</tr>
</tbody>
</table>

Population | Match set
Action Set

- Action Set established using explore/exploit scheme by alternating between:
  - Select action found in M (Explore)
  - Select deterministically with prediction array

<table>
<thead>
<tr>
<th>C</th>
<th>A</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>00#1</td>
<td>1</td>
<td>88</td>
</tr>
<tr>
<td>0##1</td>
<td>0</td>
<td>17</td>
</tr>
<tr>
<td>001#</td>
<td>1</td>
<td>91</td>
</tr>
<tr>
<td>001#</td>
<td>0</td>
<td>66</td>
</tr>
</tbody>
</table>

Action set:
- 00#1 1 88
- 001# 1 91

Prediction array:
- Action 1 179
- Action 0 83
Minimal Classifier System

- Prediction array: List of prediction values calculated for each action
- Prediction value: sum of fitness values found in the subset of M advocating the same action
- Learning starts when the reward is received
Michigan VS Pittsburgh

- Holland proposed Michigan style
- Pittsburgh was proposed by Kenneth Dejong and his student
- Main Distinction between two approaches:
  - Individuals structure
  - Problem solving structure
  - Individuals competition/competitors
  - Online VS offline learning
Michigan VS Pittsburgh

- **Structure of the individual**
  - Michigan, each individual is a classifier, entire population is the problem solution
  - Pittsburgh, each individual is a set of classifiers representing a solution

- **Individual competition / cooperation**
  - Michigan, apply individual competition
  - Pittsburgh, apply individual cooperation
Michigan VS Pittsburgh

- Offline / Online Learning
  - Michigan apply online or offline learning, while Pittsburgh apply offline learning

- Problem solution
  - Michigan, distribute problem solution
  - Pittsburgh, compact problem solution
Sources

Categories of LCS
- Strength based (ZCS)
- Accuracy based (XCS)
- Anticipation based (ALCS)

Optimisation

Application: data mining
Strength based (ZCS)

- Zeroth level classifier systems (ZCS)
- Introduced by Wilson in 1994
Zeroth level classifier systems (ZCS)  
Introduced by Wilson in 1994 

Fixed size population of rules
Strength based (ZCS)

- Rules maps conditions to actions

Condition → Action
Strength based (ZCS)

- Rules map conditions to actions

  Condition → Action

- Fitness is the predicted accumulated reward (initialised to some value $S_0$)
Strength based (ZCS)

Stimuli

Match set M
Strength based (ZCS)

- Select an action from $M$ via roulette wheel selection

$$P(\text{rule}) = \frac{\text{Fitness(\text{rule})}}{(\text{sum of fitnesses of all rules in } M)}$$

Match set $M$
All the rules in $M$ that advocated the action from the selected rule become the action set $A$.
Strength based (ZCS)

- Distribute rewards

- First reward classifiers from the previous step
  1. Take the sum of fitnesses of rules in the current action set A
  2. Multiple by discount factor $\gamma$ and learning factor $\alpha$
  3. Distribute equally among the members of the action set from the previous step
Strength based (ZCS)

- Distribute rewards

- Then receive a reward from the environment as a result of executing the current action
  1. Multiply the reward received from the environment by learning rate $\alpha$
  2. Distribute equally among the members of the current action set
Strength based (ZCS)

- Penalise other members of the match set
- Multiple the fitness of each member of the current match set that is not contained in the current action set \((M\setminus A)\) by \(\tau\)
Strength based (ZCS)

- At each step, a genetic algorithm is run with probability $p$
- Two members of the global rule population are selected via roulette wheel selection
- Two offspring are produced via one point crossover and mutation with fitnesses given by the average fitness of their parents
- Two members of the global population are deleted via roulette wheel selection based on inverse fitnesses
Run a covering operator if no rules match the environmental stimuli (match set $M$ is empty) or if every rule in the match set has fitness equal to or less than some fraction $\Phi$ of the population average.

Create a new rule with random action and average fitness that fires under the current stimuli (possibly generalised). Replace an existing rule selected via roulette wheel selection based on inverse fitness.
Strength based (ZCS)

- Suggested parameters (Bull and Hurst 2002)
  - Population size of 400
  - Initial rule fitness $S_0 = 20.0$
  - Learning rate $\alpha = 0.2$
  - Discount factor $\gamma = 0.71$
  - Tax $\tau = 0.1$
  - Genetic algorithm run with probability $p = 0.25$
  - Cover operating firing fraction $\Phi = 0.5$
ZCS Disadvantages

- May not fully represent problem space
Accuracy based (XCS)

- Extended classifier system
- Most studied and widely used family of LCS

- Each rule predicts a particular reward (and error)
- Each rule has a particular fitness
- Retain rules that predict lower rewards as long as those predictions are accurate
Accuracy based (XCS)

- Population of rules (initially empty but bounded to some size $P$) specifying actions in response to conditions
- Match set formed in response to stimuli from environment
- Action selected from match set
  - Highest fitness
  - Roulette wheel selection
  - Alternation between exploration and exploitation
- Rules advocating the same action form the action set
Receive a reward $r$ from the environment for executing the specified action

Update the predicted reward for each rule in the action set
  $\circ p \leftarrow p + \beta (r-p)$

Update the predicted error for each rule in the action set
  $\circ \varepsilon \leftarrow \varepsilon + \beta (|r-p| - \varepsilon)$

$\circ \beta = \text{estimation rate}$
Accuracy based (XCS)

- If $\varepsilon < \varepsilon_0$, set prediction accuracy $k=1$
- Otherwise, set prediction accuracy
  - $k = \alpha (\varepsilon_0 / \varepsilon)^\nu$ for some $\alpha, \nu > 0$
- Calculate relative prediction accuracy
  - $k' = k(\text{rule}) / (\text{sum of } k \text{ for all rules in action set})$
- Update the fitness of each rule
  - $f \leftarrow f + \beta (k' - f)$
  - $\alpha = \text{learning rate}$
  - $\beta = \text{estimation rate}$
Accuracy based (XCS)

- Run genetic algorithm to introduce diversity and increase fitness of population
- Run every $\theta_{GA}$ time steps
- Run on members of action set (rather than members of global population)
- Favours accurate classifiers
- Two parents selected via roulette wheel selection (based on fitness) produce two offspring via mutation and crossover
Accuracy based (XCS)

- Covering operator adds new rules when no rules match the current environmental condition (possibly generalised)
- If the population exceeds its bounded size, the requisite number of rules are deleted via roulette wheel selection based on the average size of the action sets containing each rule
Sometimes, when a new rule is added, it is checked whether or not a more general rule already exists – if it does, another copy of the more general rule is added instead of the more specific rule.

- Favours generalisation
- Computationally expensive
Accuracy based (XCS)

- Suggested parameters (Butz and Wilson 2002)
  - Maximum population size $P=800$ or $P=2000$
  - Learning rate $\alpha = 0.1$
  - Estimation rate $\beta = 0.2$
  - Genetic algorithm run every $\theta_{GA} = 25$ time steps
Anticipation based (ALCS)

- Anticipatory learning classifier systems
- Anticipate the effect of taking a particular action under a particular condition
  
  \[(\text{Condition, Action}) \rightarrow \text{Effect}\]
- Optimise accuracy of predicted effects
Anticipation based (ALCS)

- Effects might specify that after taking an action (in a specific state) that particular environmental variables might stay the same (=), adopt a particular value or cannot be predicted (?).
Anticipation based (ALCS)

- For example, the rule
  \[\text{[##1][a]} \rightarrow [1?=]\]
  would predict that after taking action \text{a} in a state matching the condition \text{##1}, that the first environment variable is 1, the second cannot be predicted while the third does not change.
Anticipation based (ALCS)

For example, the rule

\[\text{[##1][a] } \rightarrow [1?=]\]

would predict that after taking action \(a\) in a state matching the condition \(##1\), that the first environment variable is 1, the second cannot be predicted while the third does not change.
Anticipation based (ALCS)

For example, the rule

```
[##1][a] → [1?=]
```

would predict that after taking action a in a state matching the condition ##1, that the first environment variable is 1, the second cannot be predicted while the third does not change.
Anticipation based (ALCS)

- For example, the rule
  
  \[
  \text{[##1][a]} \rightarrow \text{[1?=]}
  \]

  would predict that after taking action a in a state matching the condition ##1, that the first environment variable is 1, the second cannot be predicted while the third does not change
For example, the rule

[##1][a] → [1?=]

would predict that after taking action a in a state matching the condition ##1, that the first environment variable is 1, the second cannot be predicted while the third does not change.

Anticipation based (ALCS)
Anticipation based (ALCS)

- In a grid based maze, if there is a wall to the North of an agent, moving North will result in no environmental variables changing.
- If there is a wall to the North of an agent and no wall to the East of an agent, moving East will result in there being a wall to the North West of the agent.
Anticipation based (ALCS)

- Apply heuristics to specialise or generalise classifiers
- May favour exploration over exploitation to be able to efficiently explore the problem space
In practice, seek to optimise
- Performance (quality of solution)
- Scalability
- Adaptability
- Speed

Ideal rule set is
- Correct
- Complete
- Compact/minimal
- Non-overlapping
Application: data mining

- Kharbat, Odeh and Bull 2008
- Perform data mining from a breast cancer data set to aid in diagnosis
UK health trust had a data set on breast cancer patients
Early diagnosis is very important
Seek to find patterns to aid in diagnosis
Each patient represented by 45 attributes that may be binary, categorical or real valued
Three grades of cancer aggressiveness (G1, G2, G3)
Seek to find patterns of data corresponding to each grade
Application: data mining

- Selected a sample of 1150 patients from the data set
- Data needed to be pre-processed
  - Normalise real-valued attributes to the range [0,1]
  - Balance the three grades of cancer
Application: data mining

- Accuracy based LCS (XCS) used
- Performance of XCS was compared to C4.5 (decision tree inductive learning technique)
- Train XCS to match patterns in collections of attributes to grades of cancer aggressiveness (G1, G2, G3)
- Parameters of XCS determined empirically – maximum population size of 10,000 found to be optimal
After training, rule set was compacted to make it more manageable using techniques such as removing low accuracy rules and clustering similar rules together.

Domain experts asked to comment on the quality and usefulness of rules found and whether they revealed interesting or new information that could aid in diagnosis.
2,901 rules were randomly selected from XCS and compacted to 300 to be examined by a domain expert.

- 9 considered new or interesting.
- Some of the rules from the compacted set matched patterns that were already well-known.
- None contradicted existing knowledge.
- Not all rules were found to be useful.

Application: data mining
Performance of XCS was superior to C4.5 in terms of originality, quality, richness and descriptiveness of rules

More complicated rules

Very large rule set (300 when compacted) makes it more tedious to find interesting new results
Application: data mining

- XCS system used as a means to an end in this case

- Alayón et al. 2006 describe a system used to recognise patterns in medical images used for cancer diagnosis

- Stone and Bull 2008 describe a system intended for foreign exchange trading
Sources

Questions?