สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

รายงานฉบับสมบูรณ์

Final Report

การปรับปรุงระบบการเรียนรู้จำแนกประเภท สำหรับปัญหาความไม่สมดุลของชุดข้อมูล

Improving XCS for Imbalance Problems

หัวหน้าโครงการวิจัย: รศ.ดร. บุญวัฒน์ อัตชู
ที่ปรึกษาโครงการวิจัย: รศ.ดร. เอื้อน ปิ่นเงิน
นักวิจัย: นายเกรียงศักดิ์ เตมีย์
นายพรเทพ โรจนวสุ
ผู้ช่วยวิจัย: นายศรชัย อุดมธนาพงศ์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

King Mongkut's Institute of Technology Ladkrabang

กันยายน พ.ศ. 2551

September 2008

RCH
QA
278.65
ป5ท1ก

เลขหมู่.....
เลขทะเบียน 116865
วัน,เดือน,ปี 16 ต.ค. 2552

b.12328809
i.....

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กิตติกรรมประกาศ

งานวิจัยนี้จะไม่สำเร็จลุล่วงหากปราศจากแรงผลักดัน และคำแนะนำที่มีประโยชน์ของหัวหน้าโครงการและผู้ร่วมวิจัยทุกท่าน ผู้จัดทำขอขอบคุณสำนักวิจัยการสื่อสารและเทคโนโลยีสารสนเทศที่อนุมัติทุนสนับสนุนโครงการวิจัยนี้

สุดท้ายนี้คุณค่าและประโยชน์อันพึงมีจากรายงานฉบับนี้ ผู้จัดทำขอมอบให้กับผู้มีพระคุณทุกท่านหากรายงานฉบับนี้มีข้อผิดพลาดประการใดผู้จัดทำขอน้อมรับไว้เพียงผู้เดียว

ผู้จัดทำ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทคัดย่อ

ปัญหาความไม่สมดุลกันของกลุ่มข้อมูลเป็นหัวข้อที่ได้รับความสนใจเนื่องจากเป็นปัญหาสำหรับการเรียนรู้ของระบบเรียนรู้ ระบบเรียนรู้โดยส่วนใหญ่จะให้ค่าประสิทธิภาพต่ำเมื่อเรียนรู้กับปัญหาประเภทนี้ ในงานวิจัยนี้เราได้ศึกษาผลกระทบของปัญหาความไม่สมดุลกันของกลุ่มข้อมูลที่มีต่อระบบ XCS ซึ่งเป็นระบบเรียนรู้จำแนกประเภทชนิดหนึ่งที่มีประสิทธิภาพ รวมทั้งนำเสนอวิธีการพัฒนาประสิทธิภาพของระบบ โดยนำเสนอพารามิเตอร์ในการรับรู้ที่สามารถปรับเปลี่ยนได้สำหรับกฎแต่ละกฎเพื่อทำให้เกิดสมดุลในการเรียนรู้ระหว่างกลุ่มที่มีข้อมูลจำนวนสมาชิกมากและกลุ่มที่มีข้อมูลจำนวนสมาชิกน้อย ผลการทดลองแสดงให้เห็นว่าวิธีการที่นำเสนอสามารถจำแนกความไม่สมดุลได้ทุกระดับในปัญหามัลติเพล็กซ์เซอร์ ซึ่งเป็นปัญหาที่นิยมใช้ทดสอบระบบ โดยทำการทดลองทั้งการสร้างข้อมูลแบบเคลื่อนที่และแบบ 10-fold cross validation.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Abstract

In recent years, learning with imbalanced data sets receives more and more attentions. Almost learning systems perform poor when tackle with this problem. In this paper, we study the effect of class imbalance problem in XCS which is one of the most powerful learning classifier systems and proposed a method to improve the performance. The proposed method uses adaptive perception rate for each rule to provide balance learning between majority and minority class. The experiments indicate that the propose method can classify 10 levels of imbalance classes on the well-know 11-bits Boolean logic benchmark task - multiplexer problem - generated from both on-the-fly and 10-fold cross validation.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ

หน้า

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญตาราง

หน้าที่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญรูป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

ในปัจจุบันวิธีการเรียนรู้ต่างๆ ของจักรกลเรียนรู้ (Machine Learning) ได้ถูกนำมาประยุกต์ใช้งานกับปัญหาที่ใช้งานกับข้อมูลที่มีอยู่จริง เช่น การทำเหมืองข้อมูล และการค้นหาองค์ความรู้จากข้อมูล จากการทดลองใช้งานกับข้อมูลที่มีอยู่จริงทำให้พบปัญหาหลายๆ อย่าง หนึ่งในปัญหานั้นคือ ปัญหาของความไม่สมดุลกันของข้อมูล [7]

ข้อมูลที่มีความไม่สมดุล คือ ข้อมูลที่มีจำนวนของข้อมูลในแต่ละกลุ่มแตกต่างกันเป็นอย่างมาก [7] อาทิเช่น ในงานสร้างระบบป้องกันผู้บุกรุกบนเครือข่าย (Intrusion Detection System) ข้อมูลที่เก็บส่วนมากจะเป็นพฤติกรรมการใช้งานแบบปกติ ข้อมูลพฤติกรรมของผู้บุกรุกมีเพียงเล็กน้อย หรือในงานของธรณีวิทยาที่ค้นหาพื้นที่ที่มีแร่ธาตุหรือน้ำมัน ซึ่งเป็นสัดส่วนน้อยมากเมื่อเทียบกับข้อมูลที่เป็นพื้นที่ปกติ

ในปัจจุบันการทดลองส่วนใหญ่ของจักรกลเรียนรู้จะให้ประสิทธิภาพที่ดีในข้อมูลที่มีความสมดุลหรือข้อมูลที่มีความไม่สมดุลในระดับไม่มาก แต่ในชุดข้อมูลที่มีความไม่สมดุลมากนั้นยังให้ประสิทธิภาพการเรียนรู้ที่ไม่ค่อยดีนัก หรือมีแนวโน้มการเรียนรู้โน้มเอียงไปทางกลุ่มที่มีจำนวนข้อมูลมาก ดังนั้นหากชุดข้อมูลที่ใช้ในการฝึกสอนมีความไม่สมดุลมากๆ จะทำให้การเรียนรู้ของเครื่องจักรกล***ไม่สามารถ***จำแนกข้อมูลในกลุ่มที่มีจำนวนน้อยแต่มีความสำคัญมากได้

รายงานวิจัยนี้นำเสนอการปรับปรุงจักรกลเรียนรู้เพื่อให้เหมาะสมสำหรับการเรียนรู้ของข้อมูลที่มีความไม่สมดุลเพื่อเป็นพื้นฐานของการนำไปประยุกต์ใช้งานในด้านต่างๆ

1.2 ความมุ่งหมายและวัตถุประสงค์ของการวิจัย

ความมุ่งหมายและวัตถุประสงค์ของการศึกษานี้คือ การพัฒนาระบบ XCS ให้สามารถเรียนรู้กลุ่มของข้อมูลจากชุดข้อมูลที่ไม่สมดุลได้ ในลักษณะการเรียนรู้แบบออนไลน์ (online learning)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1.3 ขอบเขตของการวิจัย

ในงานวิจัยนี้มีขอบเขตในการปรับปรุงประสิทธิภาพของระบบ XCS เพื่อให้เหมาะสมกับข้อมูลที่มีความไม่สมดุล โดยจะทำการทดลองกับข้อมูลที่จำลองขึ้นเพื่อใช้ในการวิเคราะห์ และข้อมูลมาตรฐานที่นิยมใช้ในงานทางด้านนี้

1.4 ขั้นตอนการศึกษา

1. ศึกษาการทำงานของระบบการเรียนรู้ของตัวจำแนกประเภท หรือ Learning Classifier System แบบ XCS
2. ศึกษาถึงปัจจัยต่างๆ ที่ส่งผลกระทบต่อการเรียนรู้ของ XCS
3. วิเคราะห์ถึงสาเหตุที่ทำให้ระบบ XCS ไม่สามารถเรียนรู้ข้อมูลที่มีความไม่สมดุลในระดับสูง
4. ปรับปรุงแก้ไขระบบ XCS เพื่อให้เรียนรู้ข้อมูลที่มีความไม่สมดุลในระดับสูง
5. ทดสอบและปรับปรุงระบบกับข้อมูลสมมุติและข้อมูลที่ใช้งานอยู่จริงที่มีระดับความไม่สมดุลสูง
6. เผยแพร่งานวิจัย

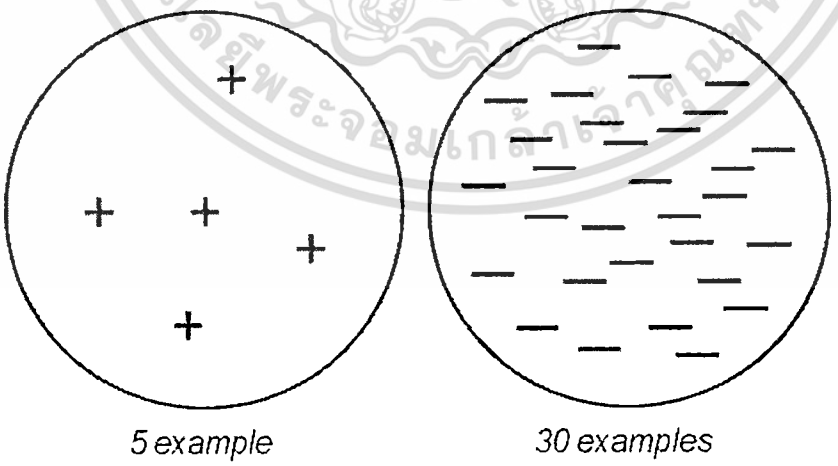
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 2
ปัญหาความไม่สมดุลของชุดข้อมูล

2.1 ลักษณะของชุดข้อมูลที่ไม่สมดุล

ชุดข้อมูลที่ไม่สมดุล คือ ชุดข้อมูลที่มีจำนวนของข้อมูลในกลุ่มใดกลุ่มหนึ่งมากหรือน้อยกว่าจำนวนข้อมูลในกลุ่มอื่นมากๆ ในปัจจุบันปัญหาความไม่สมดุลของกลุ่มข้อมูลกลายเป็นเป้าหมายในการพัฒนาของนักวิจัยหลายๆ ท่าน เนื่องจากชุดข้อมูลที่ได้จากการเก็บข้อมูลจริง (real world dataset) มีลักษณะที่ไม่สมดุล เช่น Intrusion detection[1][2], oil spill in satellite image[3], web-based clustering เป็นต้น แต่อัลกอริทึมเกี่ยวกับจักกลเรียนรู้หลายวิธีไม่สามารถเรียนรู้กลุ่มข้อมูลที่มีจำนวนน้อยๆ ได้ มีงานวิจัยหลายฉบับที่พยายามทดสอบจักกลเรียนรู้กับปัญหาความไม่สมดุลของชุดข้อมูล เช่น C4.5 [3], Support Vector Machine (SVM) [4], แต่ไม่มีอัลกอริทึมใดเลยที่สามารถแก้ไขปัญหาความไม่สมดุลได้อย่างน่าพอใจ

การใช้งานจักกลเรียนรู้ (machine learning) กับชุดข้อมูลที่ไม่สมดุลจะส่งผลให้จักกลเรียนรู้ไม่สามารถเรียนรู้กลุ่มข้อมูลที่จำนวนข้อมูลน้อยๆ ได้ โดยองค์ความรู้ที่ได้จะโน้มเอียงไปทางกลุ่มที่มีจำนวนข้อมูลมากๆ เนื่องจากจักกลเรินรู้โดยส่วนใหญ่ได้รับการพัฒนาขึ้นโดยสมมุติว่า ชุดข้อมูลที่ใช้ในการฝึกสอนระบบ จะต้องมีจำนวนข้อมูลในแต่ละกลุ่มแตกต่างกันไม่มาก



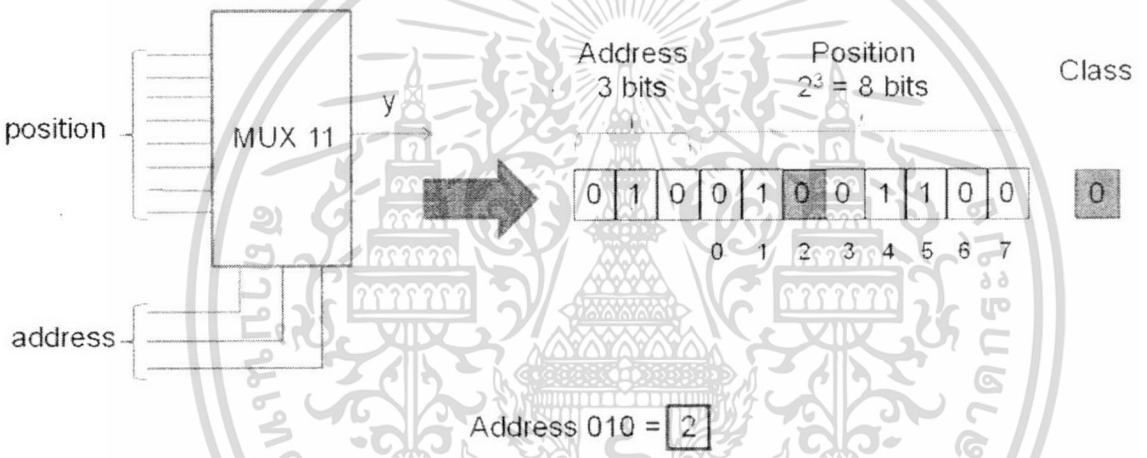
รูปที่ 2.1 ตัวอย่างชุดข้อมูลที่ไม่สมดุล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.2 ชุดข้อมูลที่ใช้ในการทดสอบและพัฒนาระบบ

2.2.1 Multiplexer problems

ชุดข้อมูลปัญหามัลติเพล็กซ์เซอร์ (Multiplexer Problem) เป็นชุดข้อมูลที่จำลองการทำงานของอุปกรณ์ผสมสัญญาณ (Multiplexer) ให้อยู่ในรูปของสายข้อมูลไบนารี่ (binary stream) โดยมี Wilson เป็นผู้คิดค้นและนำเสนอชุดข้อมุลนี้ขึ้น [5] เพื่อใช้ในการทดสอบและพัฒนาระบบ XCS ชุดข้อมูลปัญหามัลติเพล็กซ์เซอร์มีส่วนประกอบ 2 ส่วนคือ ตัวชี้ตำแหน่ง (index) เป็นส่วนต้นของสายข้อมูล มีขนาด l บิต และส่วนข้อมูล (data) เป็นส่วนท้ายของข้อมูล มีขนาด 2^l บิต เมื่อรวมทั้งสองส่วนเข้าด้วยกันแล้วข้อมูลของมัลติเพล็กซ์เซอร์จึงมีขนาด $l + 2^l$ และสามารถกำหนดกลุ่มของข้อมูลมัลติเพลกเซอร์ ได้จากข้อมูลในตำแหน่งที่ตัวชี้อ้างถึง ดังรูปที่ 2.2



รูปที่ 2.2 ตัวอย่างแสดงการจำลองการทำงานของอุปกรณ์ผสมสัญญาณขนาด 11 บิต เป็นข้อมูลปัญหามัลติเพล็กซ์เซอร์ 11 บิต

2.2.2 การสร้างชุดข้อมูลมัลติเพล็กซ์เซอร์แบบเคลื่อนที่

การสร้างข้อมูลมัลติเพล็กซ์เซอร์แบบออนไลน์สามารถทำได้โดยสุ่มสายข้อมูลไบนารี่ที่มีความยาวของสายข้อมูลเป็น $l + 2^l$ และกำหนดความระดับไม่สมดุลของชุดข้อมูล โดยกำหนดให้ค่าความน่าจะเป็นของของข้อมูลบิต '0' หรือ '1' [6] ในตำแหน่งที่ตัวชี้อ้างถึงอยู่ โดยให้แต่ละระดับมีค่าความน่าจะเป็นของกลุ่มข้อมูล '0' ดังสมการต่อไปนี้

$$p(class\_0) = \frac{2^l}{1+2^l} \tag{10}$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.2.3 การสร้างชุดข้อมูลปัญหามัลติเพล็กซ์เซอร์โดยใช้วิธี 10 fold cross validation

ในการสร้างชุดข้อมูลปัญหามัลติเพล็กซ์เซอร์ 11 บิต ให้เป็น ชุดข้อมูล โดยใช้วิธี 10 fold cross validation มีขั้นตอนการดังนี้ เริ่มจากสร้างชุดข้อมูลจากที่บันทึกรูปแบบที่เป็นไปได้ทั้งหมดของปัญหามัลติเพล็กซ์เซอร์ 11 บิต โดยแยกเก็บระหว่างกลุ่มข้อมูล '0' และ '1' ออกจากกัน จากนั้นทำการขยายจำนวนของกลุ่มข้อมูล '0' ให้มีจำนวนเป็น 2^i เท่าของจำนวนของข้อมูลในกลุ่ม '1' โดยใช้วิธีการสุ่มเลือกโดยไม่แทนที่ (random without replacement) จากนั้นทำการแบ่งกลุ่มข้อมูลออกเป็น 10 ส่วนเท่าๆ กันโดยใช้ใช้วิธีการสุ่มเลือกโดยไม่แทนที่ แล้วจึงทำการรวมกลุ่มข้อมูล '0' และ '1' ที่มีลำดับกลุ่มเดียวกันเข้าด้วยกัน เช่น รวมกลุ่มข้อมูล '0' และกลุ่มข้อมูล '1' ที่เกิดขึ้นเป็นลำดับที่ 1 เข้าด้วยกัน โดยใช้วิธีการสุ่มเลือกโดยไม่แทนที่ แต่ละกลุ่มข้อมูลที่ได้จากการรวมกันนี้เรียกว่า โฟลด (fold) จากนั้นทำการสร้างชุดข้อมูลฝึกสอน training dataset และชุดข้อมูลทดสอบ testing dataset โดยกำหนดใช้ 1 โฟลดเป็น ชุดข้อมูลทดสอบ และกำหนดให้ 9 โฟลดนำมารวมกันเป็นชุดข้อมูลฝึกสอน แล้วทำการสลับโฟลดที่นำมาเป็นชุดข้อมูลทดสอบไปเรื่อยจนครบทั้ง 10 โฟลด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 3

ระบบ XCS

3.1 การใช้งาน XCS กับปัญหาการจำแนก (Classification problems)

ระบบการเรียนรู้ของระบบตัวจำแนก (LCS) [7] มีการทำงานในลักษณะของเอเจนต์และสภาพแวดล้อม (agent-environment) ทำให้ LCS สามารถ ทำงานได้ทั้ง แบบ Multi steps และ Single step ความแตกต่างกันของการทำงานแบบ Multi step และ single step อยู่ที่การกำหนดค่าตอบแทน (reward) ที่ระบบจะได้รับจากการกระทำที่ส่งออกไปยังสภาพแวดล้อม โดยการทำงานแบบ multi step นั้นจะกำหนด reward หลายๆ ค่าแต่ single step จะมี reward เพียงค่า reward สูงสุด R_{max} และค่า reward ต่ำสุด R_{min} เท่านั้น ในงานวิจัยนี้มุ่งเน้นไปที่การแก้ปัญหาการจำแนกประเภทของข้อมูล (classification problem) ซึ่งเป็นการทำงานแบบ single step จึงกำหนดค่า reward ต่ำสุด R_{min} เป็น 0 และค่า reward สูงสุด R_{max} เป็น 1000

3.2 การนำเสนอองค์ความรู้ (Knowledge representation)

การเรียนรู้ของระบบจำแนกประเภทแบบ accuracy-based XCS [5][8] มีการทำงานอยู่ในรูปฐานข้อมูลของกฎ (rule based) ซึ่งกฎในฐานข้อมูลเรียกว่าถูกเรียกว่าตัวจำแนกประเภท ตัวจำแนกประเภทแต่ละตัวมีส่วนประกอบ 3 ส่วนดังต่อไปนี้

1) ส่วนเงื่อนไขมีลักษณะเป็นสตริงของที่ประกอบขึ้นจากตัวอักษร 0, 1 หรือ # (don't care) ตัวอย่าง เงื่อนไขของ XCS ความยาว 11 บิต 0#0100#0### ในการ mach ของเงื่อนกับ input ในปัจจุบันจะพิจารณาทีละตำแหน่ง โดยค่า '0' ของส่วนเงื่อนไขจะ mach ได้กับ input บิต 0 ที่ตำแหน่งเดียวกัน และค่า '1' ของส่วนเงื่อนไขจะ mach ได้กับ input บิต 1 ที่ตำแหน่งเดียวกัน ส่วนค่า # (don't care) จะสามารถ mach ได้กับทั้งค่า '0' และ '1' ของ input ในตำแหน่งเดียวกัน
2) ส่วนการกระทำ (a) จะต้องระบุ 1 ค่าจากค่าที่เป็นไปได้ทั้งหมด (A) ($a \in A$)
3) และส่วนการทำนาย ประกอบด้วยพารามิเตอร์ (parameter) ที่เกี่ยวข้องกับการเรียนรู้ของตัวจำแนกประเภท มีดังต่อไปนี้

3.1) Prediction payoff (p) คือ ค่าทำนายผลตอบแทนที่จะได้รับ เมื่อระบบกฎใดสอดคล้องต่ออินพุทในปัจจุบันและส่งการกระทำที่ตรงกับกฎนั้นๆ ออกไป ของตัวจำแนกประเภทนั้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.2) Prediction error (ε) เป็นค่าเฉลี่ยของความผิดพลาดในการทำนายค่าผลตอบแทนของกฎนั้น

3.3) Fitness (F) เป็นค่าความแข็งแรงของกฎนั้น ซึ่งคำนวณจากค่าถูกต้องของกฎ ในการคำนวณค่าความถูกต้องของกฎนี้จะอธิบายในหัวข้อต่อไป

3.4) Niche size estimate (as) เป็นค่าประมาณจำนวนของกฎที่สอดคล้องอินพุทเดียวกัน

นอกจากนี้ระบบจำแนกประเภทนี้ยังมีการแบ่งเซตของตัวจำแนกประเภทออกเป็น 3 เซตเพื่อใช้ในการทำงานที่แตกต่างกันออกไปดังนี้

1) เซตประชากรของตัวจำแนกประเภท (population set: [P]) คือเซตที่เก็บตัวจำแนกประเภททั้งหมดที่มีอยู่ในระบบ
2) เซตสอดคล้อง (mach set: [M]) คือเซตที่เก็บตัวจำแนกประเภทที่สอดคล้องกับอินพุทที่เข้ามาในปัจจุบัน มีวัตถุประสงค์เพื่อใช้ในตัดสินใจเลือก action ที่จะส่งออกไปยังระบบ
3) เซตการกระทำ (action set: [A]) คือเซตที่เก็บกฎที่สอดคล้องกับอินพุทในปัจจุบันและระบุ action ตรงกับ action ที่ส่งออกไปยังสภาพแวดล้อม มีวัตถุประสงค์เพื่อการปรับค่าพารามิเตอร์ของกฎที่อยู่ในเซตนี้

3.3 การปรับค่าพารามิเตอร์ต่างๆ (Parameters Update)

ในระบบ XCS ได้นำเทคนิค reinforcement learning [9] มาใช้ในการปรับค่า prediction payoff, ค่า prediction error และค่า niche size estimate ของตัวจำแนกประเภททุกตัวที่เป็นสมาชิกใน [A] โดยปรับตามรูปแบบของ Widrow-Hoff delta [5] ด้วยค่า learning rate (β) ดังนี้:

$$p_j = p_j + \beta(R - p_j) \tag{1}$$

$$\varepsilon_j = \varepsilon_j + \beta(|R - p_j| - \varepsilon_j) \tag{2}$$

$$as_j = as_j + \beta(|A| - as_j) \tag{3}$$

การปรับค่า F ในแต่ละตัวจำแนกประเภทต้องคำนวณค่า accuracy (κ_j) และคำนวณค่า relative accuracy (κ'_j) ดังสมการ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$\kappa_j = \begin{cases} \alpha(\varepsilon_j / \varepsilon_0)^{-\nu} & if\ \varepsilon_j > \varepsilon_0 \\ 1 & otherwise \end{cases} \quad (4)$$

$$\kappa_j = \begin{cases} \alpha(\varepsilon_j / \varepsilon_0)^{-\nu} & if\ \varepsilon_j > \varepsilon_0 \\ 1 & otherwise \end{cases} \quad (5)$$

และปรับค่า F ด้วยค่า κ_j' ดังสมการ

$$F_j = F_j + \beta(\kappa_j' - F_j) \quad (6)$$

โดยที่ β เป็นค่า learning rate

ε_0 เป็นค่ากำหนดระดับการยอมรับในการประมาณค่าความผิดพลาดของกฎ

ν เป็นค่าคงที่ที่มากกว่า 1

และ α เป็นค่าคงที่ในการคำนวณค่า accuracy ของกฎ มีค่าอยู่ในช่วง (0, 1)

3.4 กระบวนการค้นหากฎใหม่ (Discovery Component)

ในระบบ XCS ใช้วิธีการค้นหาตัวจำแนกประเภท 2 วิธี วิธีแรกคือการใช้ genetic algorithm (GA) [10] จะมีการใช้ GA ก็ต่อเมื่อค่าเฉลี่ยของรอบการทำงาน (time-stamp) ที่มีการกระทำกระบวนการ GA เกิดขึ้นหลังสุด มากกว่าค่า threshold (θ_{GA}) หลังจากกระทำกระบวนการ GA เรียบร้อยแล้วก็จะกำหนดรอบการทำงานใหม่ให้แต่ละตัวจำแนกประเภท ให้มีค่าเท่ากับรอบการทำงานของระบบปัจจุบัน กระบวนการทำงานของ GA จะทำการเลือกตัวจำแนกประเภทมาสองตัวจากทั้งหมดที่อยู่ใน $[A]$ โดยใช้ roulette wheel เลือกตามสัดส่วนของค่า F นำมาเป็นยีนพ่อและแม่ ตัวจำแนกประเภทใหม่ที่ได้ยีนพ่อและแม่ ต้องนำไปผ่านกระบวนการ mutation (μ) และกระบวนการ crossover (χ) โดยที่ค่าพารามิเตอร์ของตัวจำแนกประเภทใหม่ จะกำหนดจากค่าเฉลี่ยของยีนพ่อและแม่ (ถ้ามีกระบวนการ crossover เกิดขึ้น) ส่วนวิธีที่สองคือการใช้กระบวนการ covering ระบบจะใช้กระบวนการ covering ก็ต่อเมื่อไม่มีตัวจำแนกประเภทใดเลยสอดคล้องกับข้อมูลที่เข้ามา ระบบจะทำการสร้างตัวจำแนกประเภทขึ้นมาใหม่ โดยกำหนดในส่วนของ condition ให้สอดคล้องกับข้อมูล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

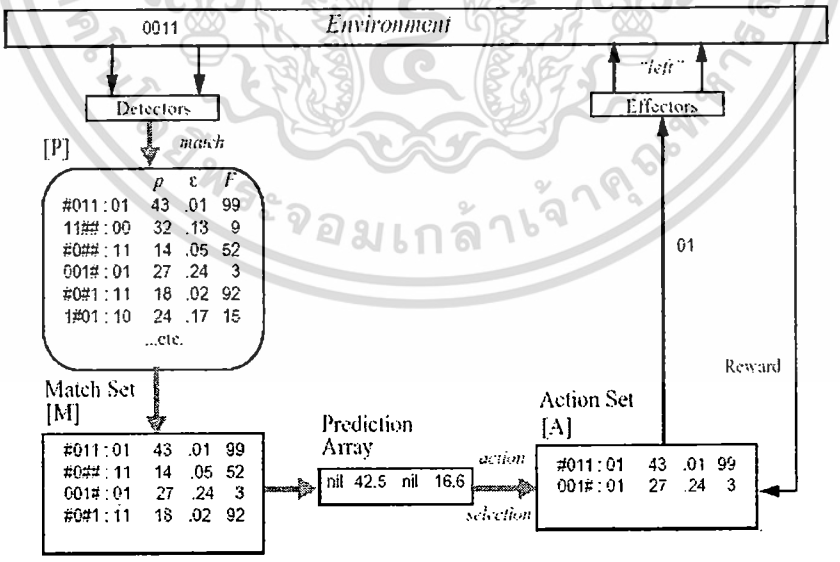
3.5 กระบวนการเลือกการกระทำ (Action Selection)

กฎที่อยู่ในเซตสอดคล้องอาจจะมีการกระทำที่ระบุไว้ไม่เหมือนกัน ดังนั้นจึงจำเป็นต้องมีวิธีการเลือกการกระทำที่จะส่งออกไปยังสภาพแวดล้อม โดยระบบ XCS จะสร้าง

ในระบบกาเรียนรู้จำแนกประเภทนี้มีวิธีการเลือกการกระทำอยู่ 2 วิธีได้แก่ การเลือกแบบสุ่ม (Random selection) และการเลือกจากค่าตอบแทนสูงสุด (maximum payoff)

3.6 การทำงานของระบบ XCS

แบบแผนการทำงานในแต่ละรอบการเรียนรู้ของระบบ XCS ดังแสดงในรูปที่ 1.1 เริ่มจากข้อมูลถูกส่งผ่านเข้าไปในระบบ ระบบจะทำการค้นหาตัวจำแนกประเภทที่มีเงื่อนไขสอดคล้อง (match) กับข้อมูลที่เข้ามาจากเซตของประชากรของตัวจำแนกประเภท $[P]$ และกำหนดกฎที่สอดคล้องกับข้อมูลที่เข้ามาให้เป็นสมาชิกของเซตสอดคล้อง $[M]$ ในการเลือกการกระทำที่จะส่งออกไปยังระบบจะสร้าง prediction array ขึ้นมาเพื่อใช้ เลือกจากการกระทำของตัวจำแนกประเภทที่อยู่ใน $[M]$ ตัวจำแนกประเภทตัวใดที่ระบุการกระทำ (a_{cl}) ตรงกับการกระทำที่เลือก (a_t) จะถูกกำหนดให้เป็นสมาชิกของ action set $[A]$ ระบบ XCS จะใช้รูปแบบของ explore/exploit สำหรับการเลือก action ระบบจะทำการเลือก action แบบสุ่มในขั้นตอนของการเรียนรู้ และในขั้นตอนการใช้งานจะทำการเลือก action โดยพิจารณาจากผลตอบแทนที่ดีที่สุด



รูปที่ 3.1 แผนภาพการทำงานของ XCS

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เมื่อได้ตัวจำแนกประเภทที่ได้จากการค้นหาทั้งสองวิธีแล้ว ระบบจะเพิ่มตัวจำแนกประเภทดังกล่าวเข้าไปในระบบ โดยในกระบวนการเพิ่มตัวจำแนกใหม่เข้าไปในระบบจะมีตัวจำแนกประเภทที่ถูกแทนที่ในฐานข้อมูลของกฎ XCS โดยใช้ roulette wheel เลือกตามสัดส่วนของ niche size estimate เพื่อกำหนดตัวจำแนกประเภทที่จะถูกลบออก ด้วยวิธีการนี้จะทำให้จำนวนตัวจำแนกประเภทที่อยู่ในแต่ละ [*A*] ให้มีขนาดใกล้เคียงกันหรือสามารถกล่าวอีกนัยหนึ่งได้ว่าจำทำให้ได้ตัวจำแนกประเภททั้งหมดที่อยู่ใน [*P*] มีคุณสมบัติไม่จำเพาะเจาะจง (Generalization)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 4
การพัฒนาระบบ XCS ให้สามารถเรียนรู้

4.1 การศึกษาขอบเขตการเรียนรู้ของ XCS บนชุดข้อมูลที่ไม่สมดุล

ในส่วนนี้อธิบายการศึกษาผลการะทบของความไม่สมดุลของชุดข้อมูลที่มีต่อการเรียนรู้ของระบบ XCS แบบปรกติ ซึ่งเป็นการศึกษาที่มีการวิจัยมาก่อนหน้าแล้วในบทความเรื่อง "Bounding XCS's Parameter for Unbalanced Datasets" ผู้แต่งคือ Albert Orriols Puig และ Ester Bernardo-Mansilla [6]

โดย Albert Orriols Puig และ Ester Bernardo-Mansilla ได้ทำการศึกษาการทำงานของ XCS แบบปรกติกับชุดข้อมูลมัลติเพล็กซ์เซอร์ที่ได้กำหนดระดับความไม่สมดุลต่างกัน 10 ระดับ ตั้งแต่ระดับสมดุล $i = 0$ จนถึงระดับไม่สมดุล $i = 9$ ในการทดลองมีการกำหนดค่าพารามิเตอร์ต่างๆ ดังนี้

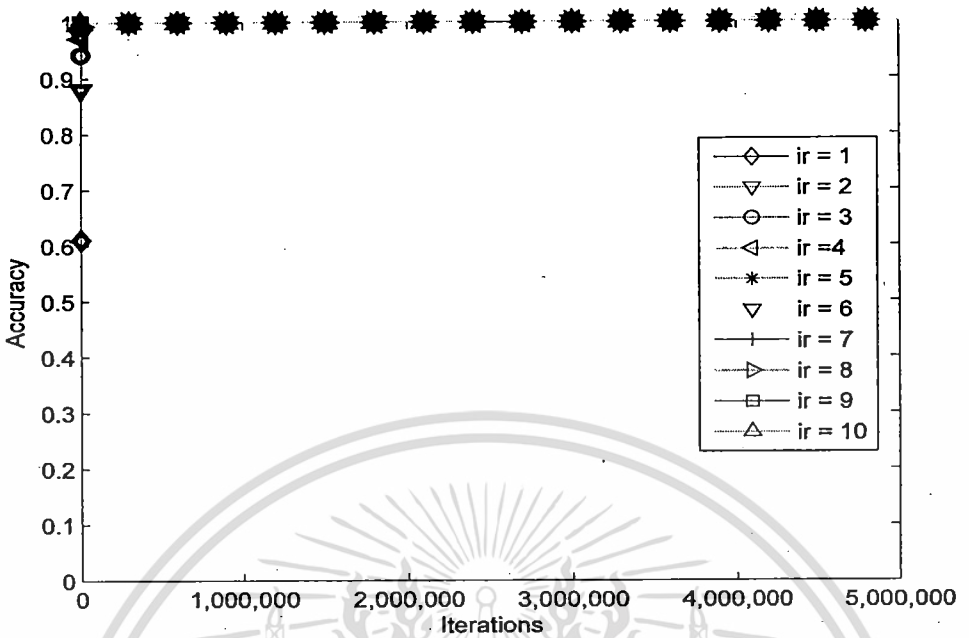
$$\text{Population size } (N) = 800,\ \beta = 0.2,\ \alpha = 0.1,\ \varepsilon_0 = 1,\ \nu = 5,\ \theta_{GA} = 25,$$
$$\chi = 0.8,\ \mu = 0.04,\ \theta_{del} = 20,\ \delta = 0.1,\ \theta_{sub} = 200 \text{ และ } P_{\#}\ 0.6$$

และเพื่อหลีกเลี่ยงผลกระทบจากการทำ generalization จึงไม่ใช้ กระบวนการ subsumption บน action set และใช้ กระบวนการ subsumption บน GA แต่ใช้กับ classifier ที่มี experienced มากๆ โดยการตั้งค่า $\theta_{sub} = 200$

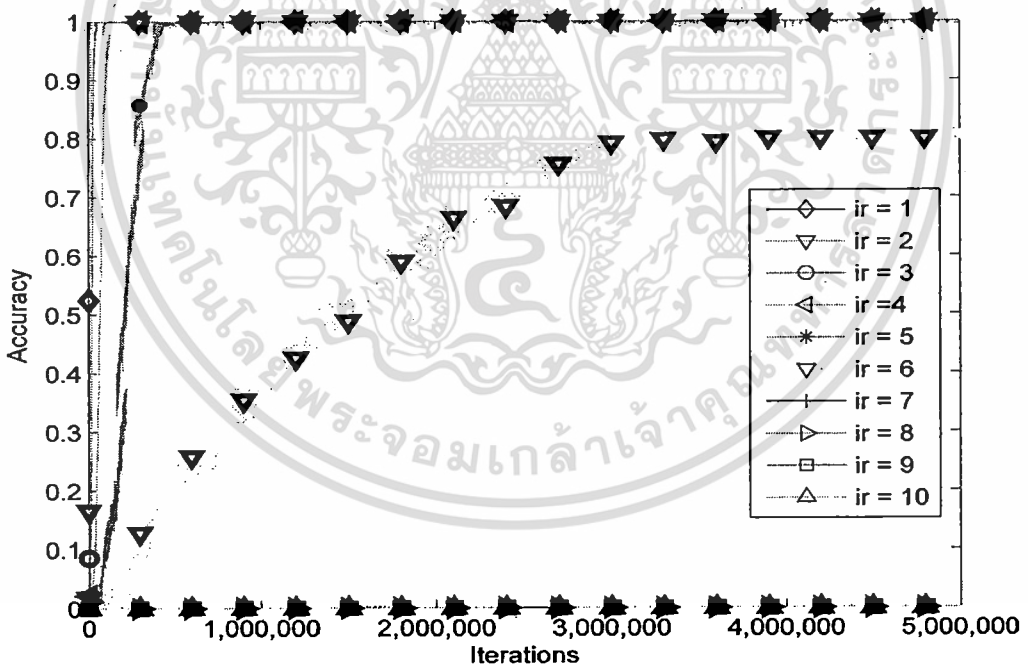
จากการทดสอบระบบ XCS แบบปรกติกับชุดข้อมูลมัลติเพล็กซ์เซอร์ 11 บิต พบว่า XCS ตามมาตราฐานไม่สามารถเรียนรู้ชุดข้อมูลที่ไม่สมดุลเมื่อระดับค่าความไม่สมดุลสูงได้ ดังรูปที่ 3.2 XCS ตามมาตราฐานสามารถเรียนรู้ชุดข้อมูลที่ไม่สมดุลได้อย่างถูกต้องจนกระทั่งระดับความไม่สมดุลเพิ่มขึ้นถึงระดับที่

$i = 5$ (ค่าความไม่สมดุลเป็น $2^5 = 32$) และเมื่อระดับความไม่สมดุลเพิ่มขึ้นความสามารถในการเรียนรู้ของระบบ XCS ตามมาตราฐานจะลดต่ำลงเรื่อยๆ จนกระทั่งไม่สามารถจำแนกกลุ่มข้อมูลที่มีจำนวนน้อยได้เลย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



ก) **True negative rate**



ข) **True positive rate**

รูปที่ 4.1 ผลการทดสอบ XCS แบบปรกติ กับชุดข้อมูลปัญหามัลติเพล็กซ์เซอร์ 11 bit ที่ระดับความไม่สมดุลต่างกัน 10 ระดับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 4.1 แสดงค่า true negative (TN) rate คือ อัตราส่วนความถูกต้องจากการจำแนกกลุ่มข้อมูล '0' ต่อจำนวนข้อมูลในกลุ่ม '0' ที่เข้ามายังระบบ XCS ทั้งหมด และค่า true positive (TP) rate คือ อัตราส่วนความถูกต้องจากการจำแนกกลุ่มข้อมูล '1' ต่อจำนวนข้อมูลในกลุ่ม '1' ที่เข้ามายังระบบ XCS ทั้งหมด ที่ได้จากการทดสอบบนชุดข้อมูลปัญหามัลติเพล็กซ์เซอร์ 11 บิต ซึ่งได้กำหนดระดับความไม่สมดุลต่างๆ กัน 11 ระดับตั้งแต่ระดับที่สมดุล ไปจนถึงระดับที่ไม่สมดุลระดับที่ 10 รูปที่ 4.1 (ก) แสดงให้เห็นว่า XCS มาตราฐานสามารถเรียนรู้ลักษณะของกลุ่ม '0' ซึ่งเป็นกลุ่มข้อมูลที่มีจำนวนมากได้อย่างถูกต้องและรวดเร็ว โดยใช้จำนวนรอบของการเรียนรู้เพียง 10,000 รอบ แต่ทว่าที่รูปที่ 4.1 (ข) แสดงให้เห็นว่าระบบ XCS มาตราฐานสามารถเรียนรู้ลักษณะของกลุ่มข้อมูล '1' ซึ่งเป็นกลุ่มข้อมูลที่มีจำนวนน้อยได้อย่างถูกต้องจนตั้งแต่ระดับความไม่สมดุลที่ 1 ไปจนถึงระดับความไม่สมดุลที่ 5 หลังจากผ่านการเรียนรู้ไปแล้ว 10,000, 20,000, 40,000, 80,000 และ 300,000 รอบ ตามลำดับ ในขณะที่กราฟการเรียนรู้ของระดับความไม่สมดุลที่ 6 คงที่อยู่ที่ร้อยละ 80 หลังจากผ่านการเรียนรู้ไปแล้ว 250,000 รอบ และระบบ XCS มาตราฐานไม่สามารถเรียนรู้กลุ่มข้อมูล '1' ได้เลยตั้งแต่ระดับความไม่สมดุลที่ 7 ไปจนถึงระดับที่ 10

เมื่อพิจารณา classifier ที่ได้หลังจากการเรียนรู้ชุดข้อมูลมัลติเพล็กซ์เซอร์พบว่ามี classifier ที่มีลักษณะ overgeneral เกิดขึ้นเป็นจำนวนมาก คือ มี classifier ###########-0 จำนวน 385 ตัว และ classifier ###########-1 366 ตัว คิดเป็นร้อยละ (385+366)/800 = 93.88 ของ classifier ทั้งหมดที่มีอยู่ใน population set และ classifier ที่ได้โน้มเอียงไปทางกลุ่มข้อมูล '0' ซึ่งเป็นกลุ่มข้อมูลที่มีข้อมูลเป็นจำนวนมาก ดังตารางที่ 4.1

ตารางที่ 4.1 รายละเอียด classifier ที่ได้จากการเรียนรู้ของ XCS บนชุดข้อมูลมัลติเพล็กซ์เซอร์ที่ระดับความไม่สมดุลระดับที่ 7

Condition	Action	Prediction	Error	Fitness	Num.
###########	0	1000	$1.2\text{x}10^{-4}$	0.98	385
###########	1	$1.2\text{x}10^{-4}$	$7.4\text{x}10^{-5}$	0.98	366
...					

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.2 การพัฒนาประสิทธิภาพของระบบ XCS สำหรับปัญหาความไม่สมดุลของกลุ่มข้อมูล

ในส่วนนี้นำเสนอการพัฒนาระบบ XCS โดยมีสมมุติฐานว่า ค่าความไม่สมดุลที่เกิดขึ้นในแต่ละพื้นที่ๆ classifier ครอบคลุมนั้น มีค่าไม่เท่ากัน เช่น สมมุติให้ classifier A ครอบคลุมพื้นที่ในช่วง 0-2 และพบข้อมูลของกลุ่ม ก 10 ตัว และพบข้อมูลของกลุ่ม ข จำนวน 20 ตัวในพื้นที่ๆ classifier A ครอบคลุม คิดค่าความไม่สมดุลเป็น 1:2 ในขณะที่ classifier B ครอบคลุมพื้นที่ในช่วง 1-3 และและพบข้อมูลของกลุ่ม ก 9 ตัว และพบข้อมูลของกลุ่ม ข จำนวน 27 ตัวในพื้นที่ๆ classifier A ครอบคลุม คิดค่าความไม่สมดุลเป็น 1:3

จากสมมุติฐานข้างต้นเรานำเสนอการพัฒนาระบบ XCS โดยกำหนดให้มีพารามิเตอร์สำหรับประมาณค่าความไม่สมดุลที่เกิดขึ้นในแต่ละพื้นที่ๆ แต่ละ classifier ครอบคลุม เพื่อนำไปใช้ในการปรับค่าพารามิเตอร์ต่างๆ ตามสัดส่วนของความไม่สมดุลที่เกิดขึ้นในแต่ละพื้นที่ โดยเรียกพารามิเตอร์ที่ประมาณค่านี้ว่า อัตราการรับรู้ (perception rate : ψ)

เนื่องจาก classifier แต่ละตัวไม่ได้รู้จักชนิดของกลุ่มข้อมูลที่มีอยู่ทั้งหมด หากว่ารู้จักแต่เพียงว่าข้อมูลที่เข้าตรงกับกลุ่มที่ระบุไว้หรือไม่เท่านั้น ดังนั้นในการประมาณค่าความไม่สมดุลของแต่ละกลุ่มจะประมาณจากอัตราส่วน จำนวนครั้งที่ classifier ตอบได้ถูกต้อง (number of corrected classify : c) ต่อจำนวนครั้งที่ classifier นั้นตอบผิด (number of uncorrected classify: c')

$$\psi = \begin{cases} 1 & if\ c = 0, c' = 0 \\ c/c' & if\ c < c' \\ c'/c & if\ c > c' \end{cases} \tag{7}$$

และทำการเปลี่ยนสมการปรับค่าพารามิเตอร์ p และ ε ดังต่อไปนี้

$$p_j = p_j + \psi \cdot \beta(R - p_j) \tag{8}$$

$$\varepsilon_j = \varepsilon_j + \psi \cdot \beta(|\ R - p_j\ | - \varepsilon_j) \tag{9}$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 5
การทดลองและผลลัพธ์

ในส่วนนี้ของรายงานแสดงค่าประสิทธิภาพที่ได้จากการทดลอง XCS 3 รูปแบบ ได้แก่ XCS มาตรฐาน XCS ที่ Albert Orriols Puig และ Ester Bernardo-Mansilla นำเสนอ และ XCS ที่นำเสนอในรายงานฉบับนี้ มาเปรียบเทียบกัน จากการทดลอง 2 ลักษณะ ได้แก่ การทดลองกับชุดข้อมูลปัญหามัลติเพล็กซ์เซอร์แบบเคลื่อนที่ และ การทดลองกับชุดข้อมูลปัญหามัลติเพล็กซ์เซอร์แบบ 10 fold cross validation โดยทำการฝึกสอนระบบ XCS กับชุดข้อมูลปัญหามัลติเพล็กซ์เซอร์แบบออนไลน์กำหนดจำนวน 5,000,000 หน่วยข้อมูล ในแต่ละครั้งของการทดลองฝึกสอนระบบ XCS มีการกำหนดค่าของพารามิเตอร์ต่างๆ ดังนี้

Population size (N) = 800, β=0.2, α = 0.1, ε_0 = 1, v = 5, θ_{GA} = 25,

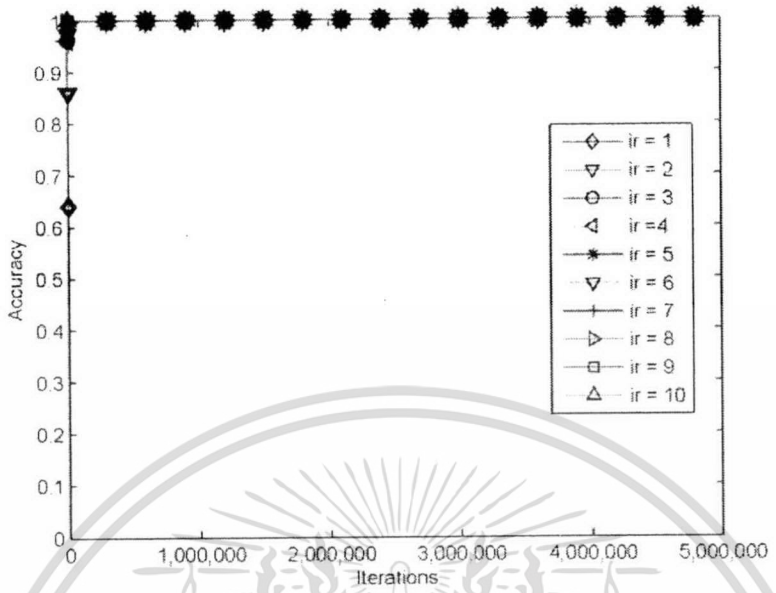
χ = 0.8, μ = 0.04, θ_{del} = 20, δ = 0.1, θ_{sub} = 200 และ $P_{\#}$ 0.6

และถ้าระบบสามารถตอบได้ถูกต้องจะได้รับค่ารางวัล (reward) เป็น 1000 แต่ถ้าตอบผิดจะได้รับค่ารางวัลเป็น 0 และค่าประสิทธิภาพที่แสดงในแต่ละจุดคิดจากการหาค่าเฉลี่ย 100 ครั้ง จากการทดลอง 10 ครั้ง ในการทดลองมี

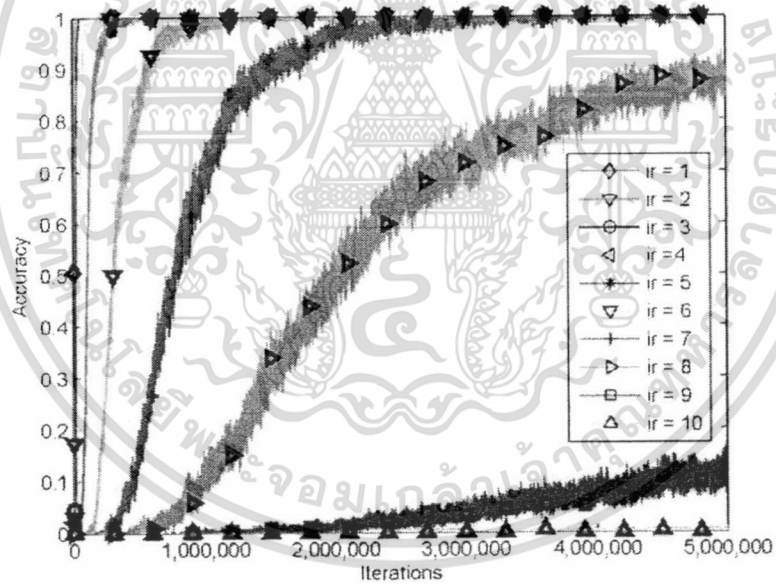
5.1 การทดลองกับชุดข้อมูลปัญหามัลติเพล็กซ์เซอร์แบบเคลื่อนที่

ในการทดลองนี้ทำการฝึกสอนระบบ XCS ด้วยชุดข้อมูลปัญหามัลติเพล็กซ์เซอร์ 11 บิต ซึ่งได้กำหนดระดับความไม่สมดุลต่างๆ กัน 10 ระดับ ตั้งแต่ระดับที่ 6 ซึ่งข้อมูลในกลุ่มข้อมูล '0' มีจำนวนเป็น 64 เท่าของจำนวนข้อมูลในกลุ่มข้อมูล '1' ไปจนถึงระดับที่ 10 ซึ่งมีจำนวนข้อมูลในกลุ่ม '0' เป็น 1024 เท่าของจำนวนข้อมูลในกลุ่ม '1' โดยที่แต่ละครั้งของการเรียนรู้ระบบ XCS จะได้รับข้อมูลมัลติเพล็กซ์เซอร์ที่สร้างขึ้นแบบเคลื่อนที่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



ก) อัตราของ True negative



ข) อัตราของ True positive

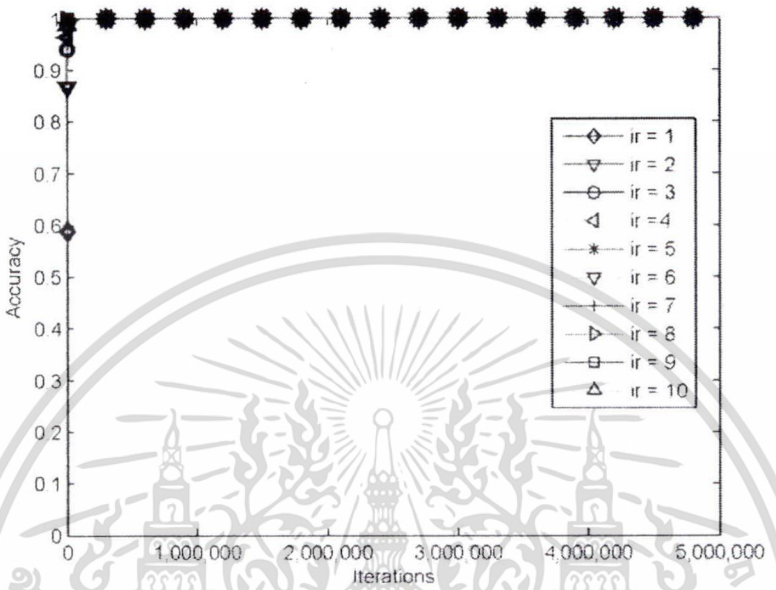
รูปที่ 5.1 ผลการทดสอบ XCS ที่พัฒนาโดย Orriols และ Bernardó กับชุดข้อมูลปัญหามัลติเพล็กซ์เซอร์ 11 bit ที่ระดับความสมดุลต่างๆ กัน 10 ระดับ

ในขั้นแรกเราได้ทำการทดสอบระบบที่พัฒนาโดย Orrials and Bernadó เพื่อแก้ไขปัญหาความไม่สมดุลโดยเฉพาะ จากผลการทดลองในรูปที่ 5.1 เมื่อเปรียบเทียบกับระบบ XCS แบบดั้งเดิมในรูปที่ 4.1 จะเห็นได้ว่าระบบของ Orriols และ Bernardó สามารถให้ประสิทธิภาพเพิ่มขึ้นอย่างชัดเจนตั้งแต่

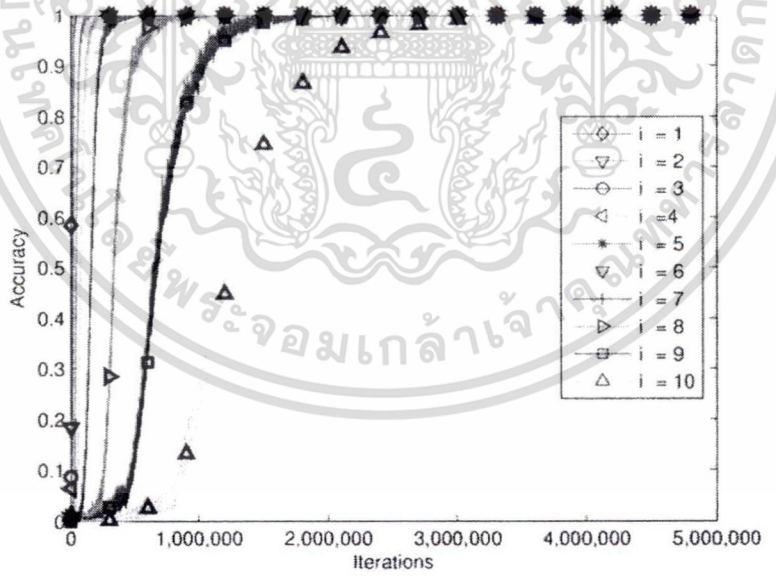
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

ระดับความไม่สมดุลที่ 6 ไปจนถึงระดับความไม่สมดุลที่ 8 เมื่อผ่านการเรียนรู้ประมาณ 5,000,000 รอบ แต่ในระดับความไม่สมดุลที่ 9 และ 10 ระบบ XCS ยังไม่สามารถจัดการกับปัญหาได้



ก) อัตราของ True negative



ข) อัตราของ True positive

รูปที่ 5.2 ผลการทดสอบ XCS ที่นำเสนอกับชุดข้อมูลปัญหามัลติเพล็กซ์เซอร์ 11 bit ที่ระดับความไม่สมดุลต่างกัน 10 ระดับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

116865

รูปที่ 5.2 แสดงประสิทธิภาพของระบบ XCS ที่ได้ทำการปรับปรุงใหม่ จากผลการทดลองแสดงให้เห็นว่าระบบสามารถที่จะจัดการกับปัญหาความไม่สมดุลได้ตั้งแต่ระดับที่ 1 จนถึงระดับที่ 10 หลักจากผ่านการเรียนรู้ไปเพียงแค่ 2,400,000 รอบ

5.2 การทดลองกับชุดข้อมูลปัญหามัลติเพล็กซ์เซอร์แบบ 10 fold cross validation

ในการเรียนรู้ชุดข้อมูลปัญหามัลติเพล็กซ์เซอร์ 11 บิต ซึ่งได้กำหนดระดับความไม่สมดุลต่างๆ กัน 10 ระดับ ตั้งแต่ระดับที่ 6 ซึ่งข้อมูลในกลุ่มข้อมูล '0' มีจำนวนเป็น 64 เท่าของจำนวนข้อมูลในกลุ่มข้อมูล '1' ไปจนถึงระดับที่ 10 ซึ่งมีจำนวนข้อมูลในกลุ่ม '0' เป็น 1024 เท่าของจำนวนข้อมูลในกลุ่ม '1' โดยชุดข้อมูลที่นำมาใช้ในการฝึกสอนและทดสอบประสิทธิภาพได้ทำการสร้างขึ้นไว้ก่อนจากรูปแบบข้อมูลที่เป็นไปได้ทั้งหมด $2^{11} = 2,048$ รูปแบบ จากนั้นเราจะทำการแบ่งข้อมูลออกเป็นชุดสำหรับเรียนรู้และชุดข้อมูลสำหรับวัดประสิทธิภาพ โดยอาศัยวิธีการ 10-fold cross validation ในการแบ่งข้อมูล

ตารางที่ 5.1 เปรียบเทียบความสามารถระบบ XCS 3 ชนิด ในการเรียนรู้ของกลุ่มข้อมูล '1' ของชุดข้อมูลปัญหามัลติเพล็กซ์เซอร์ 11 บิตแบบออฟไลน์

ชนิดของ XCS	ระดับความไม่สมดุล	ร้อยละความถูกต้องในการจำแนกข้อมูลกลุ่ม '1'
XCS มาตรฐาน	$i = 6$	0 ± 0.00
	$i = 7$	0 ± 0.00
	$i = 8$	0 ± 0.00
	$i = 9$	0 ± 0.00
	$i = 10$	0 ± 0.00
XCS ที่ Orriols และ Bernardó นำเสนอ	$I = 6$	1.00 ± 0.00 ↑
	$i = 7$	1.00 ± 0.00 ↑
	$i = 8$	0.98 ± 0.13 ↑
	$i = 9$	0.54 ± 0.17 ↑
	$i = 10$	0.02 ± 0.09

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ชนิดของ XCS	ระดับความไม่สมดุล	ร้อยละความถูกต้องในการจำแนกข้อมูลกลุ่ม '1'
XCS ที่นำเสนอในรายงานฉบับนี้	$i = 6$	$1.00 \pm 0.00 \uparrow$
	$i = 7$	$1.00 \pm 0.00 \uparrow$
	$i = 8$	$1.00 \pm 0.00 \uparrow$ ♦
	$i = 9$	$1.00 \pm 0.00 \uparrow$ ♦
	$i = 10$	$0.90 \pm 0.12 \uparrow$ ♦

ตารางที่ 5.1 แสดงค่าเฉลี่ยและส่วนเบี่ยงเบนมาตรฐานของค่าประสิทธิภาพที่ได้จากการทดสอบระบบ XCS ด้วยชุดข้อมูลฝึกสอนและชุดข้อมูลทดสอบที่แตกต่างกัน 10 ชุด ถ้าระบบ XCS มีค่าความถูกต้องเพิ่มขึ้นอย่างมีนัยยะเมื่อเทียบกับระบบ XCS มาตราฐานจะใช้สัญลักษณ์ $\uparrow$ ถ้าระบบ XCS มีค่าความถูกต้องลดลงอย่างมีนัยยะเมื่อเทียบกับระบบ XCS มาตราฐานจะใช้สัญลักษณ์ $\downarrow$ ถ้าระบบ XCS มีค่าความถูกต้องเพิ่มขึ้นอย่างมีนัยยะเมื่อเทียบกับระบบ XCS ของ Orriols และ Bernardó จะใช้สัญลักษณ์ ♦ ถ้าระบบ XCS มีค่าความถูกต้องลดลงอย่างมีนัยยะเมื่อเทียบกับระบบ XCS มาตราฐานจะใช้สัญลักษณ์ ◊

ผลลัพธ์ที่ได้แสดงให้เห็นว่า ระบบ XCS ที่ Orriols และ Bernardó นำเสนอ ดีกว่าระบบ XCS มาตรฐานอย่างมีนัยยะ ตั้งแต่ระดับความไม่สมดุลที่ 6 ไปจนถึงระดับความไม่สมดุลที่ 9 แต่ระบบ XCS ที่นำเสนอในรายงานฉบับนี้แสดงประสิทธิภาพดีกว่า XCS มาตรฐานอย่างมีนัยยะ ตั้งแต่ระดับความไม่สมดุลที่ 6 ไปจนถึงระดับความไม่สมดุลที่ 10 และก็ยังดีกว่า ระบบ XCS ที่ Orriols และ Bernardó นำเสนอไว้ อย่างมีนัยยะ ตั้งแต่ระดับที่ 8 ไปจะถึงระดับที่ 10

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 6
สรุปและข้อเสนอแนะ

ในงานวิจัยนี้นำเสนอปัญหาของข้อมูลที่ไม่มีความสมดุลกล่าวคือ เมื่อข้อมูลแต่ละชนิดมีจำนวนมากน้อยแตกต่างกันจะทำให้ในการเรียนรู้โดยทั่วไปของระบบต่างๆ ในสาขาจักกลเรียนรู้นั้นมีแนวโน้มในการเรียนรู้โน้มเอียงไปทางชนิดข้อมูลที่มีจำนวนมาก ทำให้ไม่สามารถจำแนกข้อมูลชนิดที่มีจำนวนน้อยได้ ส่งผลให้ค่าความถูกต้องของระบบมีค่าไม่ดีเมื่อคิดแยกตามชนิดข้อมูล

ในงานวิจัยนี้ได้นำเสนอพารามิเตอร์ ψ เป็นอัตราการรับรู้ของแต่ละกฎ สำหรับปรมาณค่าความไม่สมดุลซึ่งหาค่าได้จากอัตราส่วนของการตอบถูกกับอัตราส่วนในการตอบผิด โดยเพิ่ม ψ เข้าไปในส่วนของการปรับค่า prediction p และ error prediction ε จากผลการทดลองแสดงให้เห็นว่าพารามิเตอร์ที่นำเสนอสามารถเพิ่มค่าความถูกต้องทั้งการเรียนรู้ได้มากกว่าระบบ XCS แบบดั้งเดิม และระบบ XCS ที่พัฒนาโดย Orriols และ Bernardó ทั้งชุดข้อมูลมัลติเพล็กซ์เซอร์แบบเคลื่อที่และชุดข้อมูลแบบ 10-fold cross validation

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เอกสารอ้างอิง

[1] Shafi K., Kovacs T., Abbass H.A., and Zhu W. *Intrusion Detection with Evolutionary Learning Classifier Systems*, Natural Computing, Springer, August 2007.

[2] D. A Cieslak, N. V Chawla and A. "Striegel Combating Imbalance in Network Intrusion Datasets", *2006 IEEE International Conference on Granular Computing*, 10-12 May 2006

[3] N. V Chawla , "C4.5 and Imbalanced Data sets: Investigating the effect of sampling method, probabilistic estimate, and decision tree structure", *Workshop on Learning from Imbalanced Datasets II: ICML*, Washington DC, 2003.

[4] Q. Tao, et al, "Posterior Probability Support Vector Machines for Unbalanced Data", *IEEE Transactions on Neural Network*, Vol. 16, No. 6, page 1561-1573, 2006

[5] M. V. Butz and S. W. Wilson, "An Algorithmic Description of XCS", *Journal of Soft Computing 6*, page 144-153, 2002.

[6] Orriols and E. Bernadó, "Bounding XCS's Parameters for Unbalanced Datasets", *GECCO*, Washington, USA, July 8-12, 2006.

[7] J. H. Holland, "Adaptation", In Rosen & Snell (eds), *Progress in Theoretical Biology*, 4. Plenum, 1976.

[8] S. W. Wilson, "Classifier Fitness Based on Accuracy", *Evolutionary Computation*, Vol. 3, No. 2, page 149-176, 1995.

[9] R. S. Sutton and A. G. Barto, "Reinforcement Learning: an Introduction", *MIT Press*, 1998

[10] Martin V. Butz. "Rule-based Evolutionary Online Learning Systems: Learning Bounds, Classification, and Prediction". PhD thesis, University of Illinois at Urbana-Champaign, 2004

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

งานวิจัยที่ได้รับการตีพิมพ์

1. ศรชัย อุดมธนาพงศ์ พรเทพ โรจนวสุ ไพฑูรย์ ศรีนิล เกรียงศักดิ์ เตมีย์ เอื้อน ปิ่นเงิน "การพัฒนาประสิทธิภาพของระบบ XCS สำหรับปัญหาความไม่สมดุลของชุดข้อมูล" *The 12th National Computer Science and Engineering Conference*, Pattaya, Thailand, pages 272-277, 20-21 November 2008,

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ก

โครงสร้าง และ อัลกอริทึมของ XCS

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ก.1 อัลกอริธึม (Algorithm) ของ XCS

ใน Main Loop ของโปรแกรม มีขั้นตอนการทำงานตามลำดับดังต่อไปนี้ ขั้นตอนแรกเริ่มจากการสุ่มเลือกข้อมูลจากข้อมูลทั้งหมด กำหนดให้เป็นอินพุตเวคเตอร์ให้กับระบบ ขั้นตอนที่สองทำการสร้าง match set (M) จาก classifier ที่อยู่ในฐานข้อมูลซึ่งสอดคล้องกับอินพุตเวคเตอร์ ขั้นตอนที่สามคำนวณ PA ซึ่งจะบ่งบอกถึงโอกาสของแต่ละ action จะถูกเลือกใช้สำหรับกระทำ และสร้าง action set (A) โดยนำทุก classifier ที่มี action เดียวกันกับ action ที่ถูกเลือกจาก match set ลำดับต่อไประบบจะทำการ update ค่า parameter ต่างๆ ในทุก classifier ที่อยู่ใน action set ตามค่า reward ที่ได้รับหลังจากที่มีการกระทำตาม action ที่เลือก และในบางรอบของ time-step อาจจะมีกระบวนการ GA ขึ้นในส่วนของ action set การทำงานใน main loop จะวนรอบกระทำจนถึง termination criterion

RUN EXPERIMENT ():

```
1   do{
2       x <- env : randomize input data
3       [M] <- GENERATE MATCH SET out of [P] using x
4       PA <- GENERATE PREDICTION ARRAY out of M
5       act <- SELECT ACTION according to PA
6       [A] <- GENERATE ACTION SET out of M according to act
7       env: execute action act
8       P <- get reward
9       UPDATE SET [A] using P
10      RUN GA in [M]
11  }while (termination criteria are not met)
```

Formation of the Match Set

GENERATE MATCH SET procedure ส่วนอินพุตประกอบด้วยอินพุตเวคเตอร์ และฐานข้อมูลกฎ [P] ในกระบวนการทำงานจะทำการตรวจสอบ classifier ที่อยู่ใน [P] แต่ละตัวถ้าสอดคล้องกับอินพุตเวคเตอร์ ก็ทำการ add classifier ไปไว้ใน match set หลังจากนั้นทำการ covering ตาม sub-procedures GENERATE COVERING CLASSIFIER ถ้าใน match set เป็น empty set ตัว classifier ที่ได้จากการทำ covering จะต้องถูก add ไปไว้ใน match set และ replace แทนที่ classifier ใน [P] (ทำการ DELETE FROM POPULATION [P] และตามด้วย add ไปไว้ใน [P])

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

GENERATE MATCH SET ([*P*], *x*):

```
1  Initialize empty set [M]
2  For each classifier cl in [P]
3      If ( DOES MATCH classifier cl in situation x)
4              add classifier cl to set [M]
5  if ([M] is empty)
6      clc <- GENERATE COVERING CLASSIFIER considering x
7      add classier cl to set [M]
8      DELETE FROM POPULATION [P]
9      add classier cl to set [P]
10 return [M]
```

DOES MATCH procedure จะทำการตรวจสอบแต่ละ attributes ในส่วน condition ของ classifier เทียบกับแต่ละ attributes ของอินพุตเวคเตอร์ ว่าตรงกันหรือไม่ (ในส่วนของ condition # แทนได้ทั้ง 0 และ 1) ถ้าตรงกันหมดแสดงว่า classifier นั้นสอดคล้องกับอินพุตเวคเตอร์ก็จะ return true แต่ถ้าไม่ก็จะ return false

DOES MATCH (*cl*, *x*):

```
1  for each attribute x in cl.C
2     if (x ≠ # and x ≠ the corresponding attribute in condition)
3          return false
4  return true
```

Covering operator จะกระทำก็ต่อเมื่อ ถ้าจำนวน action ต่างๆ ที่อยู่ใน [*M*] ไม่ครบทุก action โดยจะทำการสร้าง classifier ขึ้นมาใหม่หนึ่งตัว กำหนดให้ส่วนของ condition เป็นค่าเดียวกับอินพุตเวคเตอร์ในแต่ละ attributes อาจจะมี # ด้วยตามความน่าจะเป็น *P*# สำหรับ action ก็ทำการสุ่มเลือกตามค่า action ที่ไม่มีใน [*M*] พร้อมทั้งกำหนดค่าตามค่าเริ่มต้นให้กับ parameter ทั้งหมด

GENERATE COVERING CLASSIFIER (x):

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
1   initialize classifier cl
2   initialize condition cl.C with the length of condition
3   for each attribute x in cl.C
4       if (RandomNumber[0,1) < P#)
5           x <- #
6       else
7           x <- the corresponding attribute in x
8   cl.A <- random action not present in [M]
9   cl.p <- pI ; cl.e <-eI; cl.F <- FI ; cl.exp <- 0; cl.ts <- actual time t; cl.as <- 1; cl.n <- 1
10  return cl
```

The Prediction Array

ระบบจะคำนวณค่าทำนาย payoff ที่จะได้รับจากการกระทำตามแต่ละ action เมื่อมีอินพุตเวคเตอร์เข้ามา โดยจะเก็บค่าทำนายแต่ละ action ไว้ใน Prediction Array *PA* โดยจะคำนวณค่าทำนายแต่ละ action จากค่า fitness คูณด้วยค่าน้ำหนักที่ได้จากค่าเฉลี่ยของค่าทำนายของทุกๆ classifiers ใน [*M*] ที่มี *action* เดียวกัน ตาม GENERATE PREDICTION ARRAY procedure

GENERATE PREDICTION ARRAY ([*M*]):

```
1   initialize prediction array PA to all null
2   initialize fitness sum array FSA to all 0.0
3   for each classifier cl in [M]
4       if (PA[cl.A] = null )
5           PA[cl.A] <- cl_p * cl_F
6       else
7           PA[cl.A] <- PA[cl.A] + cl.p * cl.F
8           FSA[cl.A] <- FSA[cl.A] + cl.F
9   for each possible action act
10      if (FSA[act] is not zero)
11          PA[act] <- PA[act] / FSA[act]
12  return PA
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Choosing an Action

XCS มีวิธีการเลือก action หลายรูปแบบ ในระบบนี้ใช้วิธี e-greedy โดยกำหนดให้ทั้ง exploration และ exploitation ตามสัดส่วนเท่ากันคือ 0.5 นั้นก็คือในบ้างครั้งเราจะทำการเลือก action จากค่าทำนายที่ดีที่สุดใน PA หรือในบ้างครั้งเราจะทำการเลือก action ด้วยการใช้เทคนิค roulette wheel เลือกตามสัดส่วนค่าทำนาย payoff ของแต่ละ action จาก PA ตาม SELECT ACTION procedure ดังนี้

SELECT ACTION (*PA*):

```
1  if (RandomNumber[0,1) < 0.5
2      //Do pure exploration here
3      return a randomly chosen action from those not null in PA
4  else
5      //Do pure exploitation here
6      return the best action in PA
```

Formation of the Action Set

หลังจากที่เรามี [M] และทำการเลือก action ที่จะกระทำแล้ว ในส่วนของ GENERATE ACTION SET procedure จะทำการสร้าง action set [A] โดยการดึง classifier ทุกตัวที่มี action ตรงกับที่ระบบเลือกจาก match set

GENERATE ACTION SET ([*M*], *act*):

```
1  initialize empty set [A]
2  for each classifier cl in [M]
3      if (cl.A = act )
4          add classifier cl to set [A]
```

Updating Classifier Parameters

ทุกๆ time-step ตัว classifier แต่ละตัวที่อยู่ใน action set จะต้องทำการปรับค่า parameter: *exp*, *p*, ε , *as*, และ *F* ในส่วนของ UPDATE SET procedure ค่า prediction payoff ทำการปรับค่าตามค่า payoff ที่ได้รับหลังจากที่กระทำ action ที่ระบบเลือกไปแล้ว ส่วนของการปรับค่าของ prediction error นั้นจะปรับตามค่า error จากการ prediction payoff สำหรับค่า prediction action set size ทำการ update โดยขนาดของ match set ปัจจุบัน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

UPDATE SET ([*A*], *P*, [*P*]):

```
1   for each classifier cl in [A]
2       cl.exp++
3       //update prediction cl.p
4       If (cl.exp < 1/ε)
5           cl.p <- cl.p + (P – cl.p) / cl.exp
6       else
7           cl.p <- cl.p  + ε*(P – cl.p)
8       //update prediction error cl. ε
9       If (cl.exp < 1/ε)
10          cl. ε <- cl. ε + (|P – cl.p | - cl. ε) / cl.exp
11      else
12          cl. ε <- cl. ε+ ε* (|P – cl.p | - cl.ε)
13      //update action set size estimate cl.as
14      If (cl.exp < 1/ε)
15          cl.ac <- cl.ac + (||M|| - cl.ac) / cl.exp
16      else
17          cl.ac <- cl.ac + ε* (||M|| - cl.ac)
18  UPDATE FITNESS in set [A]
```

สำหรับการปรับค่า F มีความซับซ้อนกว่า เราจึงแยกไว้ใน UPDATE FITNESS procedure เริ่มจากการคำนวณค่าความ accuracy *K* จากค่า error ของแต่ละ classifier หลังจากนั้นก็ทำการปรับค่า *F* จากค่า normalized accuracy

UPDATE FITNESS ([*A*]):

```
1   accuracySum = 0
2   initialize accuracy vector K
3   for each classifier cl in [A]
4       if (cl.ε < ε0)
5           K(cl) <- 1
6       else
7
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

8	$K(cl) <- \alpha\left(\frac{\varepsilon}{\varepsilon_0}\right)^{-\nu}$
9	
10	accuracySum <- accuracySum + $K(cl)$ * $cl.n$
11	for each classifier cl in $[A]$
	$cl.F <- cl.F + \varepsilon$ * ($K(cl)$ * $cl.n$ / accuracySum – $cl.F$

The Genetic Algorithm in XCS

sub-procedure สุดท้ายใน main loop คือ RUN GA มีความซับซ้อนมากพอสมควร ในขั้นตอนแรกจะทำการตรวจสอบว่าใน time-step นี้ จะมีการทำ GA หรือไม่ GA จะกระทำก็ต่อเมื่อค่า average time ในการทำ GA ก่อนหน้านี้ ของ classifier ที่อยู่ใน match set มากกว่าค่า threshold θ_{GA} ในขั้นตอนต่อไปถ้ามีการกระทำ GA เกิดขึ้น ก็คือทำงาน selection ตัว classifier ที่อยู่ใน match set มาสองตัว กำหนดให้เป็น parents หลังจากนั้นก็ผ่านขบวนการ crossover และ mutation จะได้ classifier ใหม่เป็น offspring ถ้า offspring ผ่านการทำ crossover มา ค่า parameter จะถูกกำหนดจากค่า average parameter จาก parents แต่ถ้า offspring ไม่ผ่านการทำ crossover ค่า parameter จะถูกกำหนดตามค่า parameter initial สุดท้าย ตัว offspring ทั้งสองจะต้องนำไป replace แทนที่ classifier ใน $[P]$ (ทำการ DELETE FROM POPULATION $[P]$ 2 ตัว และตามด้วย add offspring ทั้งสองตัวไปไว้ใน $[P]$)

RUN GA ($[A]$, x):

1	If (actual time t - $\sum_{cl\in[M]} cl.ts / \|[A]\| > \theta_{GA}$)
2	for each classifier cl in $[A]$
3	$cl.ts$ <- actual time t
4	$parent1$ <- SELECT OFFSPRING in $[A]$
5	$parent2$ <- SELECT OFFSPRING in $[A]$
6	$child1$ <- copy classifier $parent1$
7	$child2$ <- copy classifier $parent2$
8	if (RandomNumber[0,1) < χ)
9	APPLY CROSSOVER on $child1$ and $child2$
10	$child1.e = child2$ <- ($parent1.e$ + $parent2.e$) / 2
11	$child1.ms = child2$ <- ($parent1.ms$ + $parent2.ms$) / 2
12	for both chidren $child$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
13          APPLY MUTATION on child with
14      add child in [P]
15      DELETE FROM POPULATION [P]
```

ในการ selection classifier ที่อยู่ใน action set จะใช้เทคนิค roulette wheel เลือกตามสัดส่วนของค่า fitness โดยตอนแรกจะทำการหาผลรวมค่า fitness ทั้งหมดที่อยู่ใน action set แล้วก็ทำการหมุนวงล้อ สุดท้ายก็ได้ classifier ที่ถูกเลือกตามผลลัพธ์ที่ได้จาก roulette-wheel

SELECT OFFSPRING ([*M*]):

```
1   fitnessSum <- 0
2   for each classifier cl in [A]
3       F <- 1 / (cl.ev + 1)
4       fitnessSum <- fitnessSum + F
5   choicePoint <- RandomNumber[0,1) * fitnessSum
6   fitnessSum <- 0
7   for each classifier cl in [M]
8       F <- 1 / ( cl.ev + 1 )
9       fitnessSum <- fitnessSum + F
10      if ( fitnessSum > choicePoint )
11              return cl
```

Crossover ตาม procedure นี้มีหลักการทำงานเหมือนกับในการ crossover ตามกระบวนการของ GA ทั่วๆ ไป โดยจะใช้ two point เป็นตำแหน่ง cross แล้วทำการสลับค่าตรงระหว่างตำแหน่งที่เป็นจุด cross แรก จนถึงจุด cross ที่สอง

APPLY CROSSOVER (*cl1*, *cl2*):

```
1   x <- RandomNumber[0,1) * (length of cl1.C + 1)
2   y <- RandomNumber[0,1) * (length of cl1.C + 1)
3   if (x > y)
4       switch x and y
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
5   i <- 0
6   do{
7       if (x <= i and i < y)
8           switch cl1.C[i] and cl2.C[i]
9       i++
10  }while (i < y)
```

Mutation ก็มีหลักการทำงานเหมือนกันในกระบวนการ mutation ตามกระบวนการของ GA ทั่วๆ ไป จะทำการปรับเปลี่ยนค่าเป็นค่าตรงข้าม หรือค่า # เฉพาะตำแหน่งที่เกิด mutate

APPLY MUTATION(*cl*):

```
1   i <- 0
2   do{
3       if (RandomNumber[0,1) < χ )
4           if (cl.C[i] = # )
5               cl.C[i] <- x[i]
6           else
7               cl.C[i] <- #
8           i++
9   }while ( i < length of cl.C )
10  if ( RandomNumber[0,1) < χ )
11  cl.A <- a randomly chosen other possible action
```

การเลือก classifier ที่จะทำการ delete ออกจาก [P] ใช้หลักการ balance จำนวน classifier ที่อยู่ใน match set มีขนาดเท่าๆ กัน โดยใช้เทคนิค roulette wheel เลือกตามสัดส่วนของค่า prediction match set size

DELETE FROM POPULATION ([P]):

```
1   voteSum <- 0
2   for each classifier cl in [A]
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
3       voieSum <- voieSum + cl.ms
4   choicePoint <- RandomNumber[0,1) * voieSum
5   voieSum <- 0
6   for each classifier cl in [M]
7       voieSum <- voieSum + cl.ms
8       if(voieSum > choicePoint)
9           remove classifier c from set [P]
10          return
```

ก.2 Class Diagram ของ XCS

จากอัลกอริทึมข้างต้นสามารถออกแบบ class diagram สำหรับพัฒนาโปรแกรมได้ดังนี้

อินเตอร์เฟสหลัก

- Environment เป็น interface ของคลาส DataFromFile ใช้สำหรับอ่านข้อมูลจากไฟล์เพื่อใช้ในการเรียนรู้ และคลาส DataFromFly ใช้สร้างข้อมูลจากฟังก์ชั่นที่กำหนดเพื่อใช้ในการเรียนรู้
- LCS เป็น interface ของคลาส XCS ซึ่งเป็น LCS ประเภทหนึ่ง

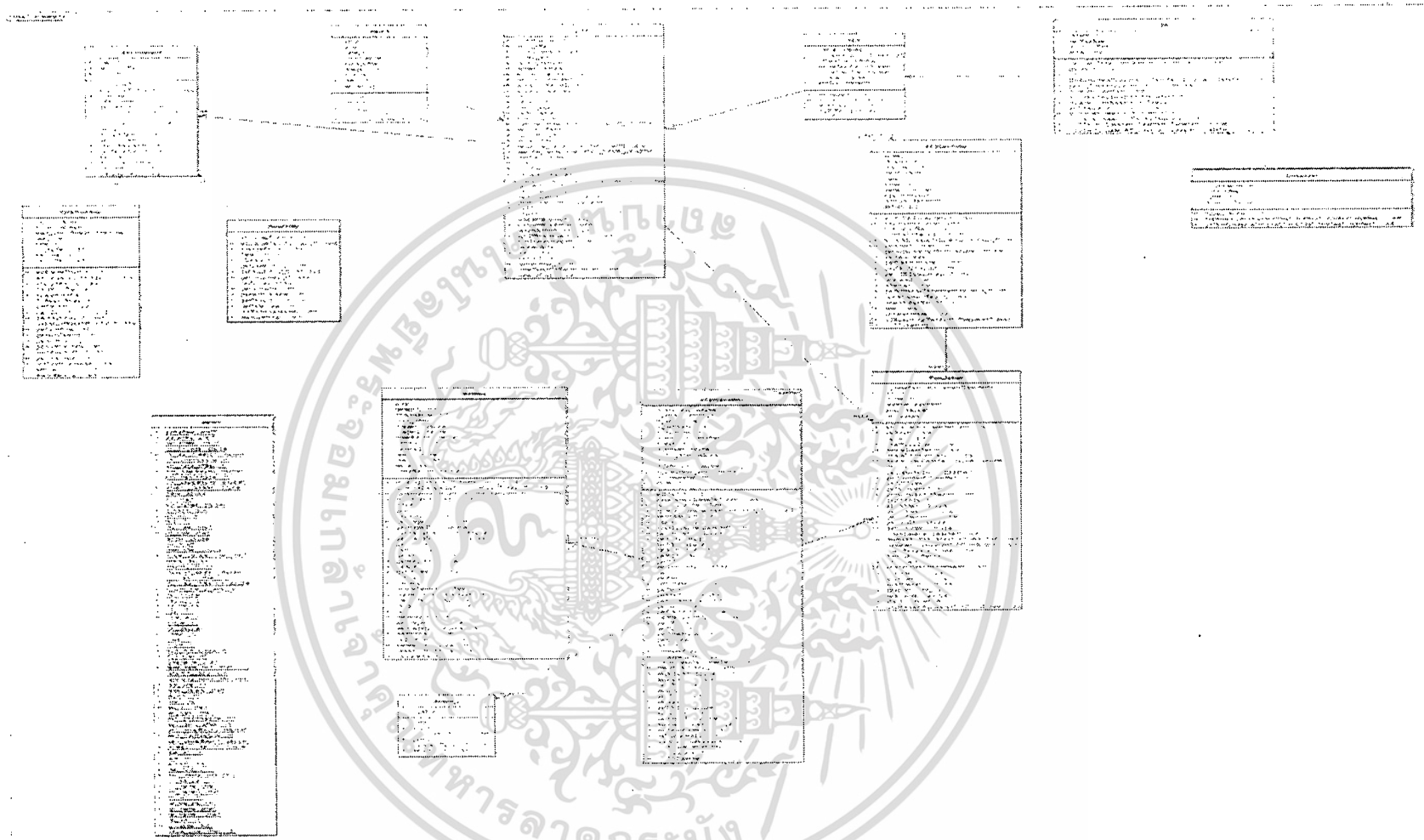
คลาสหลัก

- คลาส input_t เป็นคลาสสำหรับเก็บข้อมูลที่อ่านมาจากไฟล์ หรือ สร้างจากฟังก์ชั่น
- คลาส XCS เป็นคลาสหลักของระบบทำหน้าที่ควบคุมการเรียนรู้
- คลาส XCSLearning เป็นคลาสที่ใช้ในการเรียนรู้ของ XCS ซึ่งเป็นส่วนประกอบของคลาส XCS อีกทีหนึ่ง
- คลาส Population เป็นคลาสที่ใช้เก็บกฎทั้งหมดในระบบ XCS
- คลาส XCSClassifier เป็นคลาสที่เก็บกฎและค่าพารามิเตอร์ประจำแต่ละกฎ
- คลาส HybRep เป็นคลาสที่เก็บส่วนเงื่อนไขของกฎ
- คลาส Action_t เป็นคลาสที่เก็บส่วนแอคชั่นของกฎ
- คลาส GA เป็นคลาสที่ใช้ในกระบวนการวิวัฒนาการของกฎ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- คลาส Crossover เป็นคลาสที่ใช้ในกระบวนการแลกเปลี่ยนโครโมโซมของกฎถูกเรียนใช้ในคลาส GA อีกทีหนึ่ง
- คลาส param เป็นคลาสที่เก็บค่าคงที่ต่างๆ ของระบบ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ ก.1 Class diagram ของ XCS

ภาคผนวก ข

คู่มือการใช้งานโปรแกรม XCS

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ข.1 การคอมไพล์โปรแกรม

ในการการ compile บนระบบปฏิบัติการ Unix นั้นจะใช้คำสั่งดังนี้

icc –o <ชื่อไฟล์ที่ต้องการใช้ run> *.cpp

รูปที่ ข.1 การ compile โปรแกรม XCS

ข.2 คำสั่งในการใช้งาน

ในการใช้งานโปรแกรมจะเรียกผ่าน command โดยมีพารามิเตอร์ต่างๆ ดังนี้

1. จำนวนของกฎใน Population
2. Crossover rate
3. Mutation rate
4. Covering spread (สำหรับ parameter ที่เป็น real value)
5. mutation spread (สำหรับ parameter ที่เป็น real value)
6. representation type 2 เป็น hyper re
7. condition noise rate
8. ation noise rate
9. มี noise หรือไม่ (0 ไม่มี noise และ 1 มี noise)
10. รูปแบบของการเรียนรู้ เป็นการเรียนรู้แบบ online หรือ off-line
11. รูปแบบของปัญหา (onfly เป็นการ generate ปัญหาจากฟังก์ชัน. files เป็นการอ่านข้อมูลจากไฟล์ที่มีอยู่แล้ว)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

12. จำนวนรอบการเรียนรู้
13. ชนิดของปัญหาในที่นี้คือ Multiplexer
14. ลักษณะของปัญหา เป็น binary หรือ real
15. จำนวน address bit ของ Multiplexer
16. ไฟล์ที่กำหนดพารามิเตอร์ต่าง

รูปที่ ข.2 การใช้งานโปรแกรม XCS

ข.3 ผลลัพท์ที่ได้จากโปรแกรม

จากตัวอย่าง จะแสดง

ลำดับของการเรียนรู้ <Rd: >, ค่าความถูกต้องโดยรวมของระบบ <Sum: >,

จำนวนของกฎที่มีโดยไม่ซ้ำ, จำนวนของกฎที่มีอยู่จริง,

ค่าความถูกต้องของ class i <Cl[i]: > (i เป็นจำนวนนับที่แสดงถึงลำดับของ class ต่างๆ ของปัญหา)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ค
ผลงานวิจัยที่ได้รับการตีพิมพ์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

The 13th National Computer Science and Engineering Conference (NCSEC 2008)

การพัฒนาประสิทธิภาพของระบบ XCS สำหรับปัญหาความไม่สมดุลของกลุ่มข้อมูล

Improving XCS performance for Imbalance Problem

ศรชัย อุลมธนาพงศ์ พรเทพ โรจนวสุ ไพฑูรย์ ศรีนิล เกรียงศักดิ์ เตมีย์ เยื่อน ปิ่นเงิน
ห้องวิจัยศาสตร์ข้อมูล สำนักวิจัยการสื่อสารและเทคโนโลยีสารสนเทศ
ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ถ. ฉลองกรุง เขตลาดกระบัง กรุงเทพฯ 10520 โทรศัพท์ : 0-2737-3000
ภาควิชาวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ มหาวิทยาลัยรามคำแหง
ถ. รามคำแหง แขวงหัวหมาก เขตบางกะปิ กรุงเทพฯ 10240 โทรศัพท์ : 0-2310-8022
E-mail: s9060705@kmitl.ac.th, s8060022@kmitl.ac.th, s8060020@kmitl.ac.th, kreangsakt@hotmail.com, oaen@ru.ac.th

บทคัดย่อ

ปัญหาความไม่สมดุลกันของกลุ่มข้อมูลเป็นหัวข้อที่ได้รับความสนใจเนื่องจากเป็นปัญหาสำหรับการเรียนรู้ของระบบเรียนรู้ ระบบเรียนรู้โดยส่วนใหญ่จะให้ค่าประสิทธิภาพต่ำเมื่อเรียนรู้กับปัญหาประเภทนี้ ในงานวิจัยนี้เราได้ศึกษาผลกระทบของปัญหาความไม่สมดุลกันของกลุ่มข้อมูลที่มีต่อระบบ XCS ซึ่งเป็นระบบเรียนรู้จำแนกประเภทชนิดหนึ่งที่มีประสิทธิภาพ รวมทั้งนำเสนอวิธีการพัฒนาประสิทธิภาพของระบบ โดยนำเสนอพารามิเตอร์ในการรับรู้ที่สามารถปรับเปลี่ยนได้สำหรับกฎแต่ละกฎเพื่อทำให้เกิดสมดุลในการเรียนรู้ระหว่างกลุ่มที่มีข้อมูลจำนวนสมาชิกมากและกลุ่มที่มีข้อมูลจำนวนสมาชิกน้อย ผลการทดลองแสดงให้เห็นว่าวิธีการที่นำเสนอสามารถจำแนกความไม่สมดุลได้ทุกระดับในปัญหามัลติเพล็กซ์เซอร์ ซึ่งเป็นปัญหาที่นิยมใช้ทดสอบระบบ โดยทำการทดลองทั้งการสร้างข้อมูลแบบเกิดขึ้นและแบบ 10-fold cross validation.

Abstract

In recent years, learning with imbalanced data sets receives more and more attentions. Almost learning systems perform poor when tackle with this problem. In this paper, we study the effect of class imbalance problem in XCS which is one of the most powerful learning classifier systems and proposed a method to improve the performance. The proposed method uses adaptive perception rate for each rule to provide balance learning between majority and minority class. The experiments indicate that the propose method can classify 10 levels of imbalance classes on the well-know 11-bits Boolean logic benchmark task - multiplexer problem - generated from both on-the-fly and 10-fold cross validation.

Keywords: learning classifier systems, imbalance, classification, font style, blank line

1. Introduction

A Learning Classifier System (LCS) [1] is a rule-based machine learning system that combines between Reinforcement Learning and Genetic Algorithm. The reinforcement learning component is responsible for adjusting the strength of rules in the system according to some reward obtained from the environment. The genetic algorithm acts as an innovation discovery component which plays a key role in discovering novel and potentially useful rules. The advantages of this approach are rule-based representation, which make it easy to interpret by human; online learning ability, where the system is updated after the presentation of each data instance; robustness due to the use of genetic algorithm.

In 1995, XCS [8] emerged as a type of LCS where a classifier's fitness depends on the prediction of its expected payoff instead of the actual predicting itself as in traditional LCS which worked better on both reinforcement and supervised learning problems. Recently, XCS has been wildly applied to classification and data mining problems [2]. An investigation [9] on 15 UCI datasets showed that XCS's performance is at least as good as traditional machine learning techniques. In [10] study compared XCS with C4.5, the Naive Bayes classifier, PART, the instance based learning algorithm with one and three nearest neighbor settings, and the support vector machine on 42

NCSEC 2008

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

datasets. Statistical tests of significance showed that XCS outperforms other algorithms in some datasets.

Among them, mining imbalance data has been emphasized as one of most interested topic in XCS. A class imbalance problem is the datasets that contain one of the classes is represented by a very small number of examples compared to the other classes. This case occurs in several domains such as fraud detection, oil spills in satellite images, failures in manufacturing process, etc. Many machine learners assume a uniform distribution of classes, so that they may suffer from biases toward the majority class when they are exposed to high levels of class imbalance. Researches in the field of XCS have few studies influence of imbalance. Orriols and Bernadó analyzed behavior of XCS with imbalance datasets [5]. They showed XCS with standard setting parameter, it implemented as described in [6], cannot learn and classify high imbalance datasets and presented guidelines to set XCS's parameters based on imbalance ratio of datasets.

In this paper we proposed a method to develop XCS for imbalance datasets based on a guideline of Orriols and Bernadó. The developed XCS is able to classify 100% at imbalance level from $i = 1$ to $i = 10$ of datasets, under bounding that Orriols and Bernado had proposed.

The rest of this paper is organized as follow. Section 2 provide introduction of XCS for data mining. Section 3 describes the well-know Boolean logic benchmark task : multiplexer problem. Section 4 illustrates how to improve XCS for imbalance problem. Section 5 shows the experimental result. Conclusions are drawn and future work is discussed in Section 6.

2. Description of Accuracy-based Learning Classifier System: XCS

An Accuracy-based Learning Classifier System (XCS) was introduced by Wilson[7]. It was developed from the Holland's traditional system. Moreover, today it has been recognized as the best and most popular LCS. The most importance components of XCS are knowledge representation, parameter updates and discovery component.

XCS represents the agent knowledge as a population set $[P]$ of rule or classifier. Each rule consists of condition, action and prediction parameter. In condition part $C \in \{0, 1, \#\}^L$ and action $A \in \{a_1, a_2, \ldots, a_k\}$ specifies the action proposes. In prediction parameters, there are three principal parameters: prediction payoff (p) which estimates the payoff for that classifier if classifier is selected to action, prediction error (ε) which estimates the average error made in the payoff predictions, fitness (F) which can calculate based on error.

The XCS work as the follow: First, XCS finds a match set $[M]$ with all classifiers in the population $[P]$ whose condition matches the incoming instance. If the match set $[M]$ is empty then the covering operator is used to generate the classifiers that match current input. After that, the system generate prediction array from the match set $[M]$ by averaging of prediction for every possible action using the following equation:

$$p_j = p_j + \beta(R - p_j) \tag{1}$$

In action selection, XCS selects action by using roulette wheel of average the prediction payoff for exploration phase and selects maximum prediction payoff for exploitation phase. After that an action set $[A]$ is generated from classifier in match set $[M]$ that have the same action as the chosen action.

After received reward (R) form the environment, all parameters of classifiers in the action set $[A]$ are updated using Q-learning technique of reinforcement learning. The updated equation is computed as:

$$p_j = p_j + \beta(R - p_j) \tag{2}$$

$$\varepsilon_j = \varepsilon_j + \beta(|R - p_j| - \varepsilon_j) \tag{3}$$

$$as_j = as_j + \beta(|A| - as_j) \tag{4}$$

where β is a learning rate, $\beta \in (0, 1)$. Fitness updating is based on accuracy that we can calculate form the following equation:

$$\kappa_j = \begin{cases} \alpha(\varepsilon_j / \varepsilon_0)^{-n} & \text{if } \varepsilon_j > \varepsilon_0 \\ 1 & \text{otherwise} \end{cases} \tag{5}$$

$$\kappa'_j = \frac{\kappa_j}{\sum_{\mu \in [A]} \kappa_j} \tag{6}$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

where parameter ε_0 ($\varepsilon_0 > 0$) is a threshold for accepted error. If some classifiers have ε less than ε then this classifier predicts true and set k equal 1. For α ($\alpha \in (0, 1)$ and ν ($\nu > 1$) is constant for calculate accuracy. After that we can calculate fitness using the following equation:

$$F_j = F_j + \beta(\kappa_j' - F_j) \tag{7}$$

Discovery component in XCS has two parts. First covering operator, when match set [*M*] is empty it will generate new classifier for current message input. Every condition bit in rule will randomly select between current input bit and '#' and random one of possible action. Then the system initials prediction parameter's values by default value. Second, GA is invoke when time stamp (ts) more than threshold (θ_{GA}). If GA is activated, XCS will select two classifiers from action set [*A*] by using roulette wheel on their fitness. If the population size reaches a predefined limit, some classifiers are removed by voting within the population.

3. Multiplexer Problem

The multiplexer problem is traditionally studied in the LCS literature due to its interesting function properties by Wilson in 1985 [7]. Multiplexer data are defined as binary strings of length $l = k + 2^k$ that consist of 2 part: the first k bits as an address and last 2^k bits as data. The class value is determined by the value of data that refer by address. For example, in the 6-multiplexer ($l = 2+2^2 = 6$), the class value for the input string 100010 is 1, because the "address", 10, indexes bit 2 of the remaining four bits. 11-multiplexer and 20-multiplexer are the next more complicated multiplexers. The corresponding expression for 11-multiplexer has eight terms each consisting of four factors; for 20-multiplexer there are 16 terms of five factors each.

We used 11-multiplexer in our experiment. We divided our experiment into two parts. Firstly, on-the-fly experiment, the train and test data is directly generated by function. The imbalance multiplexer [6] permits to control the imbalance complexity of the multiplexer by undersampling the class labeled as "1". Secondly, validation experiment, we create offline 11-multiplexer by generate all of possible value and the imbalance complexity is controlled by oversampling the class labeled as '0' contain the proportion of imbalance ratio which is a ratio between number of instance from major class N_{maj} and number of instance from minor class N_{min}. Then training set is divided into 10 subsets in a stratified sample. A 10-fold cross validation is used with a 9:1 train/validation proportion. The system is trained on each training set in batch mode and then tested on validation set. Each experiment was run 10 times with different random seeds.

4. XCS for Imbalance Datasets

Orriols and Bernadó proposed that there are some parameters that have an effect with XCS's learning. They analyzed that if we run XCS with parameter ε_0=0 and maximum reward R_{max} = 1000 then it should classify minority class correctly when $1 < fr < 1998$. When they tested conventional XCS on imbalance datasets that created from multiplexer problem at level i=0 to i=9. They can classify minority class at level i = 0 to i = 4 result show as figure 1. So that useless ε_0 and β there are some problems to have effect with XCS's learning.

We improve XCS for imbalance datasets based on assumption that the imbalance ratio in each classifier is not equal. From the assumption, we propose a parameter to expect an imbalance ratio that classifier cover to update equation for p and ε_j as following:

$$p_j = p_j + \psi \cdot \beta(R - p_j) \tag{8}$$

$$\varepsilon_j = \varepsilon_j + \psi \cdot \beta(|R - p_j| - \varepsilon_j) \tag{9}$$

where ψ is a parameter for estimate imbalance ratio, it called adaptive perception rate. To estimate an imbalance ratio in each classifier, we calculate by ratio between number of correct classify and number of incorrect classify as follow equation:

$$\psi = \begin{cases} 1 & \text{if } c = 0, c' = 0 \\ c/c' & \text{if } c > c' \\ c'/c & \text{if } c < c' \end{cases} \tag{10}$$

where c is number of correct classify and c' is number of incorrect classify.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Figure 2 shows the true negative rate and the true positive rate, Orriols and Bernadó's XCS. We can see that they can improve TP rate for level $i = 5$, However, the TP rate at $i = 9$ still perform poor.

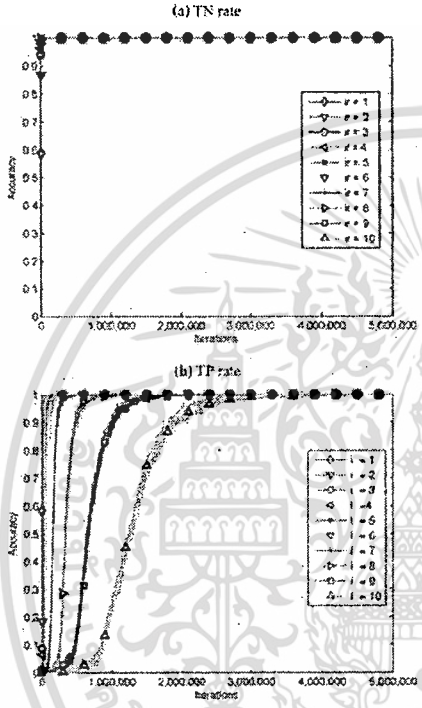


Fig. 3. TN rate (a) and TP rate (b) of XCS with propose method for 11-bit multiplexer problem from imbalance levels 1-10.

Figure 3, XCS with adaptive perception rate. We can see that perform significantly better than conventional XCS and Orriols and Bernadó's XCS both on imbalance levels $i = 9$ and $i = 10$.

In term of learning speed, if we focus on imbalance datasets level $i = 5$. The conventional XCS is able to classify a positive 100% after 330,000 iterations learning and Orriols and BernadÓ's XCS can classify a positive 100% after learn 240,000 iterations and by XCS with adaptive perception rate can classify a positive 100% after only 120,000 iterations. It means that XCS with adaptive perception rate can improve leaning speed of XCS.

5.2 Experiment on 10-fold cross validation 11-bit multiplexer

In this section, we compare the three versions of XCSs on offline 11-multiplexer. Differences that are statistically significant at a significance level of .05 are denoted by a ↑ if other propose is better than conventional XCS and by a ↓ if other propose is worse than conventional XCS. A ♦ if other propose is better than Orriols and Bernadó's and by a ◊ if other propose is worse than Orriols and Bernadó's XCS.

Table 1 Comparison TP rate between three XCSs on 11-bits multiplexer dataset from imbalance levels 6-10.

Method	Level	TP rate
Conventional XCS	$i = 6$	0±0.00
	$i = 7$	0±0.00
	$i = 8$	0±0.00
	$i = 9$	0±0.00
	$i = 10$	0±0.00
Orriols and Bernadó's XCS	$i = 6$	1.00±0.00 ↑
	$i = 7$	1.00±0.00 ↑
	$i = 8$	0.98±0.13 ↑
	$i = 9$	0.54±0.17 ↑
	$i = 10$	0.02±0.09
XCS with propose method	$i = 6$	1.00±0.00 ↑
	$i = 7$	1.00±0.00 ↑
	$i = 8$	1.00±0.00 ↑ ♦
	$i = 9$	1.00±0.00 ↑ ♦
	$i = 10$	0.90±0.12 ↑ ♦

Table 1 presents the mean and the standard deviation (over 10 runs) of the predictive accuracy in different system on the offline 11-multiplexer data sets with imbalance dataset from level $i = 6$ to $i = 10$. The result shows that the accuracy of Orriols and Bernadó's XCS is significantly better than conventional XCS in imbalance level $i = 6$ to $i = 9$. However, our proposed XCS is significantly better than conventional XCS in imbalance level $i = 6$ to $i = 10$ and also

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

significantly better than Orriols and Bernadó's XCS in imbalance level $i = 8$ to $i = 10$.

6. Conclusions

This paper proposed an improved version of XCS. We used adaptive perception rate for balance leaning between major and minor classes

The proposed method, XCS with adaptive perception rate, was test on multiplexer problem with difference imbalance levels from $i = 1$ to $i = 10$ both in on-the-fly and 10-fold cross validation. It is shown that the true positive rate (TP) of the proposed method is better than conventional XCS and Orriols and Bernadó's XCS on tested imbalance levels of 11-bit multiplexer problem. In term of learning speed, we showed that our proposed method can improve learning speed of XCS. For further research, we would like to study effects of noise to our system and also we would like to analyze on other artificial problems and real-world datasets.

7. Acknowledgements

Work reported in this paper was funded by the Research center Communication and Information Technology.

References

[1] J. H. Holland, "Adaptation", In Rosen & Snell (eds), *Progress in Theoretical Biology*, 4. Plenum, 1976.

[2] L. Bull, E-Bernadó-Mansilla and J. Holmes, "Learning Classifier System in Data Mining", *Springer*, 2008.

[3] Q. Tao, et al, "Posterior Probability Support Vector Machines for Unbalanced Data", *IEEE Transactions on Neural Network*, Vol. 16, No. 6, pp. 1561-1573, 2006.

[4] J. Auer and R. Hall, "Investigating ID3-Induced Rules from Low-Dimensional Data Cleaned by Complete Case Analysis", *AI 2004: Advances in Artificial Intelligence*, Springer Berlin, pp. 414-424, 2004.

[5] A. Orriols and E. Bernadó, "The Class Imbalance Problem in Learning Classifier Systems: A Preliminary Study", *Proceedings of 2005 workshops on Genetic and Evolutionary Computing*, Washington D.C., USA, pp. 74-78, 2005.

[6] A. Orriols and E. Bernadó, "Bounding XCS's Parameters for Unbalanced Datasets", *GECCO*, Washington, USA, July 8-12, 2006.

[7] M. V. Butz and S. W. Wilson, "An Algorithmic Description of XCS", *Journal of Soft Computing 6*, pp. 144-153, 2002.

[8] S. W. Wilson, "Classifier Fitness Based on Accuracy", *Evolutionary Computation*, Vol. 3, No. 2, page 149-176, 1995.

[9] Ester Bernad´o-Mansilla, Xavier Llor´a, and Josep M. Garrell-Guiu, XCS and GALE: a comparative study of two learning classifier systems with six other learning algorithms on classification tasks. IWLCS-2001, pages 337-341, 2001.

[10] Martin V. Butz, Rule-based Evolutionary Online Learning Systems: Learning Bounds, Classification, and Prediction. PhD thesis, University of Illinois at Urbana-Champaign, 2004.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้