รายการอ้างอิง

1. ชมทิพ พรพนมชัย. การใช้คอมพิวเตอร์ตรวจรู้อักขระภาษาไทย. วิทยานิพนธ์ปริญญามหาบัณฑิต ภาควิชาวิศวกรรมคอมพิวเตอร์ จุฬาลงกรณ์มหาวิทยาลัย, 2529.
2. เดชา รัตนาธาร. การรู้จำตัวอักษรพิมพ์ภาษาไทยโดยใช้เทคนิคแบบฟัซซีโลจิก และวิธีซินแทกติก. วิทยานิพนธ์ ปริญญามหาบัณฑิต ภาควิชาวิศวกรรมไฟฟ้า จุฬาลงกรณ์มหาวิทยาลัย, 2538.
3. นิธิพัฒน์ ชัชวาลพาณิชย์. ระบบออนไลน์สำหรับการรู้จำตัวพิมพ์อักษรไทยและตัวพิมพ์อักษรอังกฤษ. วิทยานิพนธ์ปริญญามหาบัณฑิต ภาควิชาวิศวกรรมคอมพิวเตอร์ จุฬาลงกรณ์มหาวิทยาลัย, 2537.
4. บุญธีร์ เครือตราชู และ อภิรักษ์ จิรายุสกุล. การรู้จำตัวพิมพ์อักษรไทยโดยใช้ Counterpropagation Neural Network. การประชุมวิชาการทางวิศวกรรมไฟฟ้า ครั้งที่ 16, 2538.
5. พิพัฒน์ หิรัญย์วณิชชาการ. Recognition of thai characters. บทความวิชาการ 2530 สถาบันบัณฑิตพัฒนบริหารศาสตร์. คณะสถิติประยุกต์/ศูนย์การศึกษาระบบสารสนเทศ สถาบันบัณฑิตพัฒนบริหารศาสตร์, 2530.
6. มนลดา บุญสุวรรณ. ระบบออฟไลน์สำหรับการรู้จำตัวพิมพ์อักขระไทยหลายรูปแบบ. วิทยานิพนธ์ปริญญามหาบัณฑิต ภาควิชาวิศวกรรมคอมพิวเตอร์ จุฬาลงกรณ์มหาวิทยาลัย, 2535.
7. สนธยา เมวินทร์. การศึกษาการรู้จำตัวอักษรพิมพ์ภาษาไทยโดยวิธีซินแทกติก. วิทยานิพนธ์ปริญญามหาบัณฑิต ภาควิชาวิศวกรรมไฟฟ้า จุฬาลงกรณ์มหาวิทยาลัย, 2537.
8. อภิญญา สุพรรณวรรษา. การประยุกต์ใช้การโปรแกรมตรรกะเชิงอุปนัยในการรู้จำตัวพิมพ์อักษรภาษาไทย. วิทยานิพนธ์ ปริญญามหาบัณฑิต ภาควิชาวิศวกรรมคอมพิวเตอร์ จุฬาลงกรณ์มหาวิทยาลัย, 2540.
9. Bratko, I., and Muggleton, S. Applications of Inductive Logic Programming. Communications of the ACM, 38(11), pp 65-70, 1995.
10. Califf, M. E. and Mooney, R. J. Applying ILP-based Techniques to Natural Language Information Extraction: An Experiment in Relational Learning. Workshop on Frontiers of Inductive Logic Programming, 1997.
11. Cohen, W. W. Text Categorization and Relational Learning. The Proceedings of the 12th International Conference, 1995.
12. Cohen, W. W. Learning to Classify English Text with ILP Methods. IOS Press, 1995.
13. Džeroski, S., Jacobs, N., Molina, M., Moure, C., Muggleton, S. and Laer, W. V. Detecting traffic problems with ILP. The Proceedings of the 8th international workshop on Inductive Logic Programming, pp 281-290, 1998.
14. Fayyad, U. M., Smyth, P., Weir, N. and Djorgovski, S. Automated analysis and exploration of image databases: Results, progress, and challenges. Journal of Intelligent Information Systems, 4, pp 1-19, 1995.
15. Lee, K. Automatic speech recognition: The development of the Sphinx system. Boston: Kluwer Academic Publishers, 1989.
16. Mitchell, T. M. Machine Learning. The McGraw-Hill Companies, Inc., 1997.

17. Muggleton, S. Inverse entailment and Progol. New Generation Computing, 13, pp. 245-286, 1995.

18. Muggleton, S. and Buntine, W. Machine invection of first-order predicates by inverting resolution. The Proceedings of the 5th International Machine Learning Conference, pp. 339-352, 1988.

19. Phokharatkul, P. and Kimpan, C. Recognition of Handprinted Thai Characters Using the Cavity Features of Character Based on Neural Network. The Proceedings of the 1998 IEEE Asia-Pacific Conference on Circuits and Systems, pp. 149-152. 1998.

20. Pomerleau, D. A. ALVINN: An autonomous land vehicle in a neural network. (Technical Report CMU-CS-89-107). Pittsburgh, PA: Carnegie Mellon University, 1989.

21. Roberts, S., Laer, W. V., Jacobs, N., Muggleton, S. and Broughton, J. A Comparison of ILP and Propositional Systems on Propositional Traffic Data. The Proceedings of the 8th international workshop on Inductive Logic Programming, pp 291-299, 1998.

22. Sajjapong, A., Vattanawood, W. and Covavisaruch, N. On-line Handwritten Thai Character Recognition. The Third Annual National Symposium on Computational Science and Engineering, pp. 377-382. 1999.

23. Thumwarin, P. and Chittayasothon, S. An Object-Oriented Expert System for Thai Character Recognition. The Proceedings of the 1998 IEEE Asia-Pacific Conference on Circuits and Systems, pp. 153-156. 1998.

24. Waibel, A.,Hanazawa, T., Hinton, G., Shikano, K., and Lang, K. Phoneme recognition using time-delay neural networks. IEEE Transactions on Acoustics, Speech and Signal Processing, 37(3), pp. 328-339, 1989.

ภาคผนวก ก

ตัวพิมพ์อักษรไทยที่ใช้ในการวิจัย

พยัญชนะ 44 ตัว

ก ข ฃ ค ฅ ฆ ง จ ฉ ช ฌ ญ ฎ ฏ ฐ ฑ ฒ ณ ด ต ถ ท ธ น บ ป ผ ฝ พ ฟ ภ ม ย ร ล ว ศ ษ ส ห ฬ อ ฮ

สระ วรรณยุกต์ และตัวอักษรพิเศษ 23 ตัว

ะ า ิ ี ึ ื ุ ู เ แ โ ใ ไ ็ ่ ้ ๊ ๋ ๆ ์ ๎ ๏ ๚ ๛

ตัวเลข 10 ตัว

๐ ๑ ๒ ๓ ๔ ๕ ๖ ๗ ๘ ๙

รูปแบบตัวอักษรคอร์เดีย (Cordia) และยูโครเซีย (Eucrosia)

ขนาด 20, 22, 24, 28, 32, 36 และ 48

ภาคผนวก ข

ตัวอย่างแสดงการสร้างกฎโดยระบบโปรกอล

สัญลักษณ์สำคัญที่ใช้ในระบบโปรกอลมีดังนี้

- modeh เป็นการกำหนดรูปแบบของสัญพจน์ส่วนหัว มีรูปแบบเป็น

  modeh(Recall, Head_Predicate)?

  เมื่อ Recall คือ จำนวนครั้งมากที่สุดที่สามารถเรียกใช้เพรดิเคด Head_Predicate ได้

  Head_Predicate คือ เพรดิเคดที่จะปรากฏในส่วนหัวของอนุประโยค

- modeb เป็นการกำหนดรูปแบบของสัญพจน์ส่วนเนื้อความ มีรูปแบบเป็น

  modeb(Recall, Body_Predicate)?

  เมื่อ Body_Predicate คือ เพรดิเคดที่จะปรากฏในส่วนเนื้อความของอนุประโยค

- เครื่องหมาย + แสดงถึงอาร์กิวเมนต์ที่ใช้เป็นอินพุตของสัญพจน์ (input argument)
- เครื่องหมาย - แสดงถึงอาร์กิวเมนต์ที่ใช้เป็นเอาต์พุตของสัญพจน์ (output argument)
- เครื่องหมาย # แสดงถึงอาร์กิวเมนต์ที่เป็นค่าคงที่ (constant)

% การประกาศสัญพจน์

```
:- modeh(1,eastbound(+train))?
:- modeh(*,has_car(+train,-car))?
:- modeh(1,not open(+car))?
:- modeb(1,not long(+car))?
:- modeb(1,long(+car))?
:- modeb(1,open(+car))?
:- modeb(1,double(+car))?
:- modeb(1,jagged(+car))?
:- modeb(1,shape(+car,-shape))?
:- modeb(1,load(+car,-shape,-int))?
:- modeb(1,wheels(+car,-int))?
:- modeb(1,infront(+car,-car))?
```

```
%%%%%%%%%%%%%%%%%%%%%%%%
% ตัวอย่างบวก

eastbound(east1).
eastbound(east2).
eastbound(east3).
eastbound(east4).
eastbound(east5).

%%%%%%%%%%%%%%%%%%%%%%%%
% ตัวอย่างลบ

:- eastbound(west6).
:- eastbound(west7).
:- eastbound(west8).
:- eastbound(west9).
:- eastbound(west10).

%%%%%%%%%%%%%%%%%%%%%%%%
% ความรู้ภูมิหลัง

car(car_11). car(car_12). car(car_13). car(car_14). car(car_21). car(car_22). car(car_23).
car(car_31). car(car_32). car(car_33). car(car_41). car(car_42). car(car_43). car(car_44).
car(car_51). car(car_52). car(car_53). car(car_61). car(car_62).
car(car_71). car(car_72). car(car_73). car(car_81). car(car_82).
car(car_91). car(car_92). car(car_93). car(car_94).
car(car_101). car(car_102).

shape(elipse). shape(hexagon). shape(rectangle). shape(u_shaped).

train(east1). train(east2). train(east3). train(east4). train(east5).
train(west6). train(west7). train(west8). train(west9). train(west10).

%%%%%%%%%%%%%%%%%%%%%%%%
% Eastbound train 1

short(car_12). closed(car_12). long(car_11). long(car_13). short(car_14). open(car_11).
infront(east1,car_11). infront(car_11,car_12). infront(car_12,car_13). infront(car_13,car_14).
```

```
open(car_13). open(car_14). shape(car_11,rectangle). shape(car_12,rectangle).
shape(car_13,rectangle). shape(car_14,rectangle). load(car_11,rectangle,3).
load(car_12,triangle,1). load(car_13,hexagon,1). load(car_14,circle,1). wheels(car_11,2).
wheels(car_12,2). wheels(car_13,3). wheels(car_14,2). has_car(east1,car_11).
has_car(east1,car_12). has_car(east1,car_13). has_car(east1,car_14).

%%%%%%%%%%%%%%%%%%%%%%%%
% Eastbound train 2

has_car(east2,car_21). has_car(east2,car_22). has_car(east2,car_23).
infront(east2,car_21). infront(car_21,car_22). infront(car_22,car_23).
short(car_21). short(car_22). short(car_23). shape(car_21,u_shaped).
shape(car_22,u_shaped). shape(car_23,rectangle). open(car_21).
open(car_22). closed(car_23). load(car_21,triangle,1). load(car_22,rectangle,1).
load(car_23,circle,2). wheels(car_21,2). wheels(car_22,2). wheels(car_23,2).

%%%%%%%%%%%%%%%%%%%%%%%%
% Eastbound train 3

has_car(east3,car_31). has_car(east3,car_32). has_car(east3,car_33).
infront(east3,car_31). infront(car_31,car_32). infront(car_32,car_33).
short(car_31). short(car_32). long(car_33).
shape(car_31,rectangle). shape(car_32,hexagon). shape(car_33,rectangle).
open(car_31). closed(car_32). closed(car_33). load(car_31,circle,1).
load(car_32,triangle,1). load(car_33,triangle,1). wheels(car_31,2).
wheels(car_32,2). wheels(car_33,3).

%%%%%%%%%%%%%%%%%%%%%%%%
% Eastbound train 4

has_car(east4,car_41). has_car(east4,car_42). has_car(east4,car_43).
has_car(east4,car_44). infront(east4,car_41). infront(car_41,car_42).
infront(car_42,car_43). infront(car_43,car_44). short(car_41).
short(car_42). short(car_43). short(car_44). shape(car_41,u_shaped).
shape(car_42,rectangle). shape(car_43,elipse). shape(car_44,rectangle).
double(car_42). open(car_41). open(car_42). closed(car_43). open(car_44).
load(car_41,triangle,1). load(car_42,triangle,1). load(car_43,rectangle,1).
load(car_44,rectangle,1). wheels(car_41,2). wheels(car_42,2). wheels(car_43,2).
wheels(car_44,2).
```

```
%%%%%%%%%%%%%%%%%%%%%%%%
% Eastbound train 5

has_car(east5,car_51). has_car(east5,car_52). has_car(east5,car_53).
infront(east5,car_51). infront(car_51,car_52). infront(car_52,car_53).
short(car_51). long(car_52). short(car_53). shape(car_51,rectangle).
shape(car_52,rectangle). shape(car_53,rectangle). double(car_51).
open(car_51). closed(car_52). closed(car_53). load(car_51,triangle,1).
load(car_52,rectangle,1). load(car_53,circle,1). wheels(car_51,2).
wheels(car_52,3). wheels(car_53,2).

%%%%%%%%%%%%%%%%%%%%%%%%
% Westbound train 6

has_car(west6,car_61). has_car(west6,car_62). infront(west6,car_61).
infront(car_61,car_62). long(car_61). short(car_62). shape(car_61,rectangle).
shape(car_62,rectangle). closed(car_61). open(car_62). load(car_61,circle,3).
load(car_62,triangle,1). wheels(car_61,2). wheels(car_62,2).

%%%%%%%%%%%%%%%%%%%%%%%%
% Westbound train 7

has_car(west7,car_71). has_car(west7,car_72). has_car(west7,car_73).
infront(west7,car_71). infront(car_71,car_72). infront(car_72,car_73).
short(car_71). short(car_72). long(car_73). shape(car_71,rectangle).
shape(car_72,u_shaped). shape(car_73,rectangle). double(car_71).
open(car_71). open(car_72). jagged(car_73). load(car_71,circle,1).
load(car_72,triangle,1). load(car_73,nil,0). wheels(car_71,2).
wheels(car_72,2). wheels(car_73,2).

%%%%%%%%%%%%%%%%%%%%%%%%
% Westbound train 8

has_car(west8,car_81). has_car(west8,car_82). infront(west8,car_81).
infront(car_81,car_82). long(car_81). short(car_82). shape(car_81,rectangle).
shape(car_82,u_shaped). closed(car_81). open(car_82). load(car_81,rectangle,1).
load(car_82,circle,1). wheels(car_81,3). wheels(car_82,2).
```

%%%%%%%%%%%%%%%%%%%%%%%%%

% Westbound train 9

has_car(west9,car_91). has_car(west9,car_92). has_car(west9,car_93).
has_car(west9,car_94). infront(west9,car_91). infront(car_91,car_92).
infront(car_92,car_93). infront(car_93,car_94). short(car_91). long(car_92).
short(car_93). short(car_94). shape(car_91,u_shaped). shape(car_92,rectangle).
shape(car_93,rectangle). shape(car_94,u_shaped). open(car_91). jagged(car_92).
open(car_93). open(car_94). load(car_91,circle,1). load(car_92,rectangle,1).
load(car_93,rectangle,1). load(car_93,circle,1). wheels(car_91,2). wheels(car_92,2).
wheels(car_93,2). wheels(car_94,2).

%%%%%%%%%%%%%%%%%%%%%%%%%

% Westbound train 10

has_car(west10,car_101). has_car(west10,car_102). infront(west10,car_101).
infront(car_101,car_102). short(car_101). long(car_102). shape(car_101,u_shaped).
shape(car_102,rectangle). open(car_101). open(car_102). load(car_101,rectangle,1).
load(car_102,rectangle,2). wheels(car_101,2). wheels(car_102,2).

ขั้นตอนการสร้างกฎ

ระบบทำการอ่านการกำหนดรูปแบบของสัญพจน์ส่วนหัวและส่วนเนื้อความ

[:- modeh(1,eastbound(+train))?]
[:- modeb(100,has_car(+train,-car))?]
[:- modeb(1,not open(+car))?]
[:- modeb(1,not long(+car))?]
[:- modeb(1,long(+car))?]
[:- modeb(1,open(+car))?]
[:- modeb(1,double(+car))?]
[:- modeb(1,jagged(+car))?]
[:- modeb(1,shape(+car,-shape))?]
[:- modeb(1,load(+car,-shape,-int))?]
[:- modeb(1,wheels(+car,-int))?]
[:- modeb(1,infront(+car,-car))?]

ขั้นตอนที่ 1 รับตัวอย่างบวกตัวอย่างแรกเพื่อทำการเจนเนอรัลไรซ์

[Generalising eastbound(east1).]

ขั้นตอนที่ 2 สร้างอนุประโยคที่เฉพาะมากที่สุดจากอนุประโยค eastbound(east1). ร่วมกับความรู้ภูมิหลังโดยวิธีการอาร์แอลจีจี [17]

```
eastbound(A) :- has_car(A,B), has_car(A,C), has_car(A,D), has_car(A,E),
                not open(C), not long(C), not long(E), open(B), open(D),
                open(E), long(B), long(D), shape(B,F), shape(C,F), shape(D,F), shape(E,F),
                infront(B,C), infront(C,D), infront(D,E).
```

ขั้นตอนที่ 3 สร้างอนุประโยคใหม่โดยการตัดเพรดิเคตที่ปรากฏในอนุประโยคในขั้นตอนที่ 2 เพื่อทำให้อนุประโยคสามารถครอบคลุมตัวอย่างได้มากขึ้น จากนั้นทำการค้นหาอนุประโยคที่มีการบีบอัดสูงสุดโดยวิธีการค้นหาแบบ A*

ประโยค [C:0,5,5,0 eastbound(A).] หมายความว่า อนุประโยคใหม่ eastbound(A) ซึ่งเป็นอนุประโยคที่ถูกทำการเจนเนอรัลไรซ์จากอนุประโยคในขั้นตอนที่ 2 ครอบคลุมตัวอย่างบวก (p) 5 ตัวอย่าง ตัวอย่างลบ (n) 5 ตัวอย่าง จำนวนสัญพจน์ที่จำเป็นในความสัมพันธ์ระหว่างอินพุตและเอาต์พุตในอนุประโยคส่วนหัว (h) มีค่าเท่ากับ 0 จำนวนสัญพจน์ในอนุประโยคส่วนเนื้อความ (c) มีค่าเท่ากับ 0 ทำให้ได้ค่าการบีบอัด (f) เท่ากับ 5-(5+0+0) = 0 จากนั้นทำการค้นหาอนุประโยคที่มีการบีบอัดสูงสุด

[C:0,5,5,0 eastbound(A).]
[C:-1,5,5,0 eastbound(A) :- has_car(A,B).]
[C:-1,5,5,0 eastbound(A) :- has_car(A,B).]
[C:-1,5,5,0 eastbound(A) :- has_car(A,B).]
[C:-1,5,5,0 eastbound(A) :- has_car(A,B).]
[C:-2,5,5,0 eastbound(A) :- has_car(A,B), has_car(A,C).]
[C:-2,5,5,0 eastbound(A) :- has_car(A,B), has_car(A,C).]
[C:-2,5,5,0 eastbound(A) :- has_car(A,B), has_car(A,C).]
[C:-2,5,5,0 eastbound(A) :- has_car(A,B), open(B).]
[C:-4,3,5,0 eastbound(A) :- has_car(A,B), long(B).]
[C:-2,5,5,0 eastbound(A) :- has_car(A,B), shape(B,C).]
[C:-2,5,5,0 eastbound(A) :- has_car(A,B), infront(B,C).]
[C:-2,5,5,0 eastbound(A) :- has_car(A,B), has_car(A,C).]
[C:-2,5,5,0 eastbound(A) :- has_car(A,B), open(B).]
[C:-4,3,5,0 eastbound(A) :- has_car(A,B), long(B).]

[C:-2,5,5,0 eastbound(A) :- has_car(A,B), shape(B,C).]

[C:-2,5,5,0 eastbound(A) :- has_car(A,B), infront(B,C).]

[C:-2,5,5,0 eastbound(A) :- has_car(A,B), has_car(A,C).]

[C:-2,5,5,0 eastbound(A) :- has_car(A,B), has_car(A,C).]

[C:-1,5,4,0 eastbound(A) :- has_car(A,B), not open(B).]

[C:-2,5,5,0 eastbound(A) :- has_car(A,B), not long(B).]

[C:-2,5,5,0 eastbound(A) :- has_car(A,B), shape(B,C).]

[C:-2,5,5,0 eastbound(A) :- has_car(A,B), infront(B,C).]

[C:-2,5,5,0 eastbound(A) :- has_car(A,B), not long(B).]

[C:-2,5,5,0 eastbound(A) :- has_car(A,B), open(B).]

[C:-2,5,5,0 eastbound(A) :- has_car(A,B), shape(B,C).]

[C:2,5,0,0 eastbound(A) :- has_car(A,B), not open(B), not long(B).]

[27 explored search nodes]

[f=2,p=5,n=0,h=0]

[Result of search is]

อนุประโยค [C:2,5,0,0 eastbound(A) :- has_car(A,B), not open(B), not long(B).] เป็นอนุประโยคที่มีค่าการบีบอัดสูงสุด

eastbound(A) :- has_car(A,B), not open(B), not long(B).

ขั้นตอนที่ 4 ลบตัวอย่างทั้งหมดที่สอดคล้องกับอนุประโยคจากขั้นตอนที่ 3

[5 redundant clauses retracted]

ขั้นตอนที่ 5 เมื่อไม่มีตัวอย่างเหลืออีก จบการทำงาน ได้อนุประโยคสุดท้ายซึ่งสามารถอธิบายแนวคิดของรถไฟที่แล่นไปทางทิศตะวันออกได้

eastbound(A) :- has_car(A,B), not open(B), not long(B).

ภาคผนวก ค

ความรู้ภูมิหลัง

```
primitive(1,0).  primitive(2,0).  primitive(3,0). ... primitive(1663,12).  primitive(1664,12).

endpoint(1,-1).  endpoint(2,-1).  endpoint(3,-1). ... endpoint(1663,0).  endpoint(1664,0).

startzone(1,0).  startzone(2,0).  startzone(3,0). ... startzone(1663,7).  startzone(1664,7).

endzone(1,0).  endzone(2,0).  endzone(3,0). ... endzone(1663,7).  endzone(1664,7).

iscircle(8). iscircle(9).  iscircle(10).  iscircle(11).  iscircle(12).

hrank(1,11).  hrank(2,11).  hrank(3,11) ... hrank(1663,17).  hrank(1664,17).

inc(A,B) :- B is A+1.

iscircenpt(A) :- primitive(A,B), endpoint(A,-1), iscircle(B).

member(A,[A|B]).
member(A,[B|C]) :- member(A,C).

least(X,Y,X) :- hrank(X,A), hrank(Y,B), A=<B.
least(X,Y,Y) :- hrank(X,A), hrank(Y,B), B<A.

head([A],A).
head([A|B],C) :- head(B,D), least(A,D,C).

headzone(A,B) :- head(A,C), startzone(C,B).

headprim(A,B) :- head(A,C), primitive(C,B).

endpoint_zone(A,B) :- member(C,A), endpoint(C,-1), startzone(C,B).

endpoint_primitive(A,B) :- member(C,A), endpoint(C,-1), primitive(C,B).

circle_at_endpoint_in_zone(A,B) :- member(C,A), iscircenpt(C), startzone(C,B).

circle_at_endpoint_primitive(A,B) :- member(C,A), iscircenpt(C), primitive(C,B).

count_circle_at_endpoint([],0).
count_circle_at_endpoint([A|B],C) :- iscircenpt(A), count_circle_at_endpoint(B,D), inc(D,C).
count_circle_at_endpoint([A|B],C) :- not iscircenpt(A), count_circle_at_endpoint(B,C).
```

```
count_endpoint([],0).
count_endpoint([A|B],C) :- endpoint(A,-1), count_endpoint(B,D), inc(D,C).
count_endpoint([A|B],C) :- endpoint(A,0), count_endpoint(B,C).

count_section([],0).
count_section([A],1).
count_section([A|B],C) :- count_section(B,D), inc(D,C).

count_primitive_0([],0).
count_primitive_0([A|B],C) :- primitive(A,0), count_primitive_0(B,D), inc(D,C).
count_primitive_0([A|B],C) :- not primitive(A,0), count_primitive_0(B,C).

count_primitive_1([],0).
count_primitive_1([A|B],C) :- primitive(A,1), count_primitive_1(B,D), inc(D,C).
count_primitive_1([A|B],C) :- not primitive(A,1), count_primitive_1(B,C).

count_primitive_2([],0).
count_primitive_2([A|B],C) :- primitive(A,2), count_primitive_2(B,D), inc(D,C).
count_primitive_2([A|B],C) :- not primitive(A,2), count_primitive_2(B,C).

count_primitive_3([],0).
count_primitive_3([A|B],C) :- primitive(A,3), count_primitive_3(B,D), inc(D,C).
count_primitive_3([A|B],C) :- not primitive(A,3), count_primitive_3(B,C).

count_primitive_4([],0).
count_primitive_4([A|B],C) :- primitive(A,4), count_primitive_4(B,D), inc(D,C).
count_primitive_4([A|B],C) :- not primitive(A,4), count_primitive_4(B,C).

count_primitive_5([],0).
count_primitive_5([A|B],C) :- primitive(A,5), count_primitive_5(B,D), inc(D,C).
count_primitive_5([A|B],C) :- not primitive(A,5), count_primitive_5(B,C).

count_primitive_6([],0).
count_primitive_6([A|B],C) :- primitive(A,6), count_primitive_6(B,D), inc(D,C).
count_primitive_6([A|B],C) :- not primitive(A,6), count_primitive_6(B,C).

count_primitive_7([],0).
count_primitive_7([A|B],C) :- primitive(A,7), count_primitive_7(B,D), inc(D,C).
count_primitive_7([A|B],C) :- not primitive(A,7), count_primitive_7(B,C).

count_primitive_8([],0).
count_primitive_8([A|B],C) :- primitive(A,8), count_primitive_8(B,D), inc(D,C).
count_primitive_8([A|B],C) :- not primitive(A,8), count_primitive_8(B,C).
```

```
count_primitive_9([],0).
count_primitive_9([A|B],C) :- primitive(A,9), count_primitive_9(B,D), inc(D,C).
count_primitive_9([A|B],C) :- not primitive(A,9), count_primitive_9(B,C).

count_primitive_10([],0).
count_primitive_10([A|B],C) :- primitive(A,10), count_primitive_10(B,D), inc(D,C).
count_primitive_10([A|B],C) :- not primitive(A,10), count_primitive_10(B,C).

count_primitive_11([],0).
count_primitive_11([A|B],C) :- primitive(A,11), count_primitive_11(B,D), inc(D,C).
count_primitive_11([A|B],C) :- not primitive(A,11), count_primitive_11(B,C).

count_primitive_12([],0).
count_primitive_12([A|B],C) :- primitive(A,12), count_primitive_12(B,D), inc(D,C).
count_primitive_12([A|B],C) :- not primitive(A,12), count_primitive_12(B,C).

count_line([],0).
count_line([A|B],C) :- not is_circle(A), count_line(B,D), inc(D,C).
count_line([A|B],C) :- is_circle(A), count_line(B,C).

count_circle([],0).
count_circle([A|B],C) :- is_circle(A), count_circle(B,D), inc(D,C).
count_circle([A|B],C) :- not is_circle(A), count_circle(B,C).

lastmember([A],A).
lastmember([A|B],C) :- lastmember(B,C).

begin_endzone([A|B],C,D) :- lastmember(B,E), startzone(E,C), endzone(A,D).

memberzone(A,B,C) :- member(E,A), startzone(E,B), endzone(E,C).

havemember(A,B,C,D) :- member(E,A), primitive(E,B), startzone(E,C), endzone(E,D).

count_startzone_0([],0).
count_startzone_0([A|B],C) :- startzone(A,0), count_startzone_0(B,D), inc(D,C).
count_startzone_0([A|B],C) :- not startzone(A,0), count_startzone_0(B,C).

count_startzone_1([],0).
count_startzone_1([A|B],C) :- startzone(A,1), count_startzone_1(B,D), inc(D,C).
count_startzone_1([A|B],C) :- not startzone(A,1), count_startzone_1(B,C).

count_startzone_2([],0).
count_startzone_2([A|B],C) :- startzone(A,2), count_startzone_2(B,D), inc(D,C).
count_startzone_2([A|B],C) :- not startzone(A,2), count_startzone_2(B,C).
```

```
count_startzone_3([],0).
count_startzone_3([A|B],C) :- startzone(A,3), count_startzone_3(B,D), inc(D,C).
count_startzone_3([A|B],C) :- not startzone(A,3), count_startzone_3(B,C).

count_startzone_4([],0).
count_startzone_4([A|B],C) :- startzone(A,4), count_startzone_4(B,D), inc(D,C).
count_startzone_4([A|B],C) :- not startzone(A,4), count_startzone_4(B,C).

count_startzone_5([],0).
count_startzone_5([A|B],C) :- startzone(A,5), count_startzone_5(B,D), inc(D,C).
count_startzone_5([A|B],C) :- not startzone(A,5), count_startzone_5(B,C).

count_startzone_6([],0).
count_startzone_6([A|B],C) :- startzone(A,6), count_startzone_6(B,D), inc(D,C).
count_startzone_6([A|B],C) :- not startzone(A,6), count_startzone_6(B,C).

count_startzone_7([],0).
count_startzone_7([A|B],C) :- startzone(A,7), count_startzone_7(B,D), inc(D,C).
count_startzone_7([A|B],C) :- not startzone(A,7), count_startzone_7(B,C).

firstmember([A],A).
firstmember([A|B],A).

yuk([],0).
yuk([A],0).
yuk([A|B],1) :- primitive(A,0), firstmember(B,F), primitive(F,7).
yuk([A|B],C) :- primitive(A,0), firstmember(B,F), not primitive(F,7), yuk(B,C).
yuk([A|B],C) :- not primitive(A,0), yuk(B,C).

nozone([]).

havezone(A,B) :- member(B,A).

count_zone_>=_3(A) :- count_section(A,B), 3=<B.

count_zone_<=_3(A) :- count_section(A,B), B=<3.

count_section_<_20(A) :- count_section(A,B), B<20.

count_section_>_3(A) :- count_section(A,B), 3<B.

count_endpoint_<_4(A) :- count_endpoint(A,B), B<4.

sizeless0_7(A) :- A < 0.7.

sizemore1_2(A) :- 1.2 < A.

sizemore1_45(A) :- 1.45 < A.
```

```
topright_tail(A) :- member(B,A), endpoint(B,-1), startzone(B,1), endzone(B,1), primitive(B,0).
topright_tail(A) :- member(B,A), endpoint(B,-1), startzone(B,1), endzone(B,1), primitive(B,1).

bottomright_tail(A) :- member(B,A), endpoint(B,-1), endzone(B,4), primitive(B,5).
bottomright_tail(A) :- member(B,A), endpoint(B,-1), endzone(B,4), primitive(B,6).

topleft_tail(A) :- member(B,A), endpoint(B,-1), endzone(B,2), primitive(B,4).
topleft_tail(A) :- member(B,A), endpoint(B,-1), endzone(B,2), primitive(B,5).

upleft_tail(A) :- member(B,A), endpoint(B,-1), startzone(B,6), endzone(B,6), primitive(B,0).
upleft_tail(A) :- member(B,A), endpoint(B,-1), startzone(B,6), endzone(B,6), primitive(B,2).

right_line(A) :- member(B,A), endpoint(B,-1), endzone(B,1), primitive(B,1).
right_line(A) :- member(B,A), endpoint(B,-1), endzone(B,1), primitive(B,2).

headprim_0009(A) :- headprim(A,0).
headprim_0009(A) :- headprim(A,9).

headprim_0912(A) :- headprim(A,9).
headprim_0912(A) :- headprim(A,12).

headprim_0910(A) :- headprim(A,9).
headprim_0910(A) :- headprim(A,10).

headprim_1011(A) :- headprim(A,10).
headprim_1011(A) :- headprim(A,11).

headprim_0506(A) :- headprim(A,5).
headprim_0506(A) :- headprim(A,6).

headprim_1112(A) :- headprim(A,11).
headprim_1112(A) :- headprim(A,12).

headprim_0312(A) :- headprim(A,3).
headprim_0312(A) :- headprim(A,12).

headprim_0110(A) :- headprim(A,1).
headprim_0110(A) :- headprim(A,10).

headprim_0111(A) :- headprim(A,1).
headprim_0111(A) :- headprim(A,11).

headprim_0512(A) :- headprim(A,5).
headprim_0512(A) :- headprim(A,12).

headprim_040510(A) :- headprim(A,4).
headprim_040510(A) :- headprim(A,5).
headprim_040510(A) :- headprim(A,10).
```

```
headzone_03(A) :- headzone(A,0).
headzone_03(A) :- headzone(A,3).

headzone_05(A) :- headzone(A,0).
headzone_05(A) :- headzone(A,5).

headzone_34(A) :- headzone(A,3).
headzone_34(A) :- headzone(A,4).

headyuk(A) :- member(z2,A).

upperyuk(A) :- member(z1,A).
upperyuk(A) :- member(z2,A).

nozone0(A) :- not havezone(A,z0).
nozone3(A) :- not havezone(A,z3).
nozone4(A) :- not havezone(A,z4).

have4044(A) :- member(B,A), primitive(B,4), endpoint(B,0), startzone(B,4), endzone(B,4).

have0011(A) :- member(B,A), primitive(B,0), endpoint(B,0), startzone(B,1), endzone(B,1).

topline(A) :- member(B,A), primitive(B,0), endpoint(B,0), startzone(B,1), endzone(B,1).
topline(A) :- member(B,A), primitive(B,0), endpoint(B,0), startzone(B,2), endzone(B,1).

enpt_topright(A) :- member(B,A), endpoint(B,-1), startzone(B,1), endzone(B,1).
enpt_topright(A) :- member(B,A), endpoint(B,-1), startzone(B,2), endzone(B,1).
```

ภาคผนวก ง

กฎที่ได้จากการโปรแกรมตรรกะเชิงอุปนัย

```
ก(3,A,B,C,D,E)     :- headzone(B,3), headprim(B,1), count_primitive_4(B,0).
ข(3,A,B,C,D,E)     :- not headyuk(D), headzone(B,2), endpoint_primitive(B,1).
                      circle_at_endpoint_in_zone(B,2), count_circle_at_endpoint(B,1),
                      right_line(B), memberzone(B,4,1), headprim_0910(B),
                      count_startzone_5(B,0).
ฃ(3,A,B,C,D,E)     :- headzone(B,2), headprim(B,10), endpoint_zone(B,4),
                      begin_endzone(B,2,1), memberzone(B,2,1), memberzone(B,4,1),
                      havemember(B,7,2,2), headprim_0910(B), headprim_1011(B).
ค(3,A,B,[z3],C,D)  :- not upperyuk([z3]), not upperyuk(C), headzone(B,0),
                      endpoint_primitive(B,6).
ฅ(3,A,B,C,D,E)     :- not topright_tail(B), headzone(B,0), headprim(B,11), endpoint_zone(B,3),
                      havemember(B,0,2,2), upperyuk(D).
ฆ(3,A,B,C,D,E)     :- headzone(B,2), headprim(B,10), endpoint_zone(B,4), memberzone(B,0,0),
                      memberzone(B,0,4), memberzone(B,3,3).
ง(3,A,B,[],[z4],[])  :- headzone(B,1).
จ(3,A,B,C,D,E)     :- not topright_tail(B), endpoint_zone(B,2), count_primitive_9(B,0),
                      count_primitive_10(B,0), count_primitive_11(B,0), count_startzone_3(B,0),
                      memberzone(B,4,4).
ฉ(3,A,B,C,D,E)     :- endpoint_zone(B,4), count_circle_at_endpoint(B,1), topleft_tail(B),
                      havezone(D,z3).
ช(3,A,B,C,D,E)     :- not headyuk(C), not headyuk(D), headzone(B,2), endpoint_zone(B,0),
                      endpoint_zone(B,1), topright_tail(B), havemember(B,0,2,2).
ซ(3,A,B,C,D,E)     :- not nozone(C), not nozone(E), not sizemore1_2(A), headprim(B,10),
                      endpoint_zone(B,1), endpoint_zone(B,2), count_startzone_5(B,0),
                      memberzone(B,4,4), havezone(C,z1), havezone(D,z2).
ฌ(3,A,B,C,D,E)     :- headzone(B,3), headprim(B,10), endpoint_zone(B,0), endpoint_zone(B,4),
                      memberzone(B,3,0).
ญ(3,A,B,C,D,E)     :- headzone(B,3), headprim(B,10), endpoint_primitive(B,1), right_line(B),
                      begin_endzone(B,3,1), nozone0(C), nozone0(D), nozone0(E),
                      count_section_<_20(B), count_primitive_4(B,0).
ฎ(4,A,B,C,D,E)     :- headzone(B,3), headprim(B,9), count_zone_<=_3(E).
ฏ(4,A,B,C,D,E)     :- headzone(B,3), headprim(B,9), count_circle_at_endpoint(B,1),
                      count_zone_>=_3(D).
ฐ(3,A,B,C,D,E)     :- endpoint_zone(B,1), memberzone(B,2,1), memberzone(B,2,2),
```

```
                memberzone(B,4,4), havemember(B,0,1,1), headprim_0312(B).
ก(3,A,B,C,D,E)  :- not nozone(C), not nozone(D), not nozone(E), headzone(B,2),
                headprim(B,10), endpoint_zone(B,1), endpoint_primitive(B,6),
                count_primitive_4(B,0).
ข(3,A,B,C,D,E)  :- headzone(B,0), endpoint_zone(B,4), right_line(B), memberzone(B,2,2,),
                memberzone(B,3,3), memberzone(B,4,1).
ค(3,A,B,C,D,E)  :- headzone(B,3), headprim(B,10), endpoint_zone(B,4),
                endpoint_primitive(B,2), begin_endzone(B,3,1), memberzone(B,3,2).
ฆ(3,A,B,[],C,D) :- not have0011(B), headzone(B,0), headprim(B,12), endpoint_zone(B,1),
                endpoint_primitive(B,6).
ง(3,A,B,C,D,E)  :- headzone(B,0), endpoint_zone(B,1), endpoint_primitive(B,6), topline(B),
                nozone3(C), count_startzone_0(B,3).
จ(3,A,B,C,D,E)  :- not topright_tail(B), not sizemore1_2(A), headzone(B,3),
                headprim_0110(B), memberzone(B,1,1), memberzone(B,1,4),
                memberzone(B,2,1), memberzone(B,3,3), count_startzone_0(B,0).
ฉ(3,A,B,C,D,E)  :- not topright_tail(B), not nozone(C), headzone(B,2), headprim(B,12)
                endpoint_primitive(B,5), count_endpoint(B,3), memberzone(B,1,1).
ช(3,A,B,C,D,E)  :- sizeless0_7(A), enpt_topright(B), nozone0(C), nozone4(C),
                nozone4(E), headprim_040510(B).
ซ(3,A,B,[z4],C,D) :- headzone(B,2), headprim(B,12), endpoint_zone(B,4).
ญ(3,A,B,C,D,E)  :- not have4044(B), headzone(B,2), headprim(B,12),
                count_circle_at_endpoint(B,1), right_line(B), memberzone(B,2,3),
                memberzone(B,3,4).
ฌ(2,A,B,C,D,E)  :- headzone(B,2), headprim(B,12), memberzone(B,3,4).
ฝ(3,A,B,C,D,E)  :- headzone(B,2), headprim(B,11), memberzone(B,0,4).
ฟ(2,A,B,C,D,E)  :- headzone(B,2), headprim(B,11).
ฑ(3,A,B,C,D,E)  :- headzone(B,2), headprim(B,12), endpoint_zone(B,1), endpoint_zone(B,4),
                endpoint_primitive(B,1), endpoint_primitive(B,6), count_endpoint(B,5).
ฬ(2,A,B,C,D,E)  :- endpoint_zone(B,2), endpoint_zone(B,4), endpoint_primitive(B,1),
                endpoint_primitive(B,6).
ภ(3,A,B,C,D,E)  :- not topright_tail(B), headzone(B,3), headprim_0009(B).
ม(3,A,B,C,D,E)  :- headzone(B,2), headprim(B,12), endpoint_zone(B,4),
                endpoint_primitive(B,1), memberzone(B,0,4), havemember(B,1,4,1),
                havemember(B,7,0,4).
ย(3,A,B,C,D,E)  :- headzone(B,2), endpoint_zone(B,4), endpoint_primitive(B,0),
                endpoint_primitive(B,1), memberzone(B,3,4).
ร(3,A,B,C,D,E)  :- headprim_0009(B), headzone_34(B), sizeless0_7(A), enpt_topright(B).
ฤ(4,A,B,C,D,E)  :- headprim(B,10).
```

```
a(3,A,B,C,D,E)    :- not topright_tail(B), headzone(B,3), endpoint_primitive(B,6),
                     count_primitive_9(B,0), count_primitive_11(B,0), topleft_tail(B),
                     upperyuk(E).
b(3,A,B,[],[],C)  :- headzone(B,4), topleft_tail(B).
c(3,A,B,C,D,E)    :- endpoint_zone(B,0), endpoint_zone(B,1), endpoint_zone(B,3),
                     endpoint_primitive(B,6), enpt_topright(B), memberzone(B,1,4),
                     memberzone(B,2,1), havemember(B,6,1,4).
d(3,A,B,C,D,E)    :- headzone(B,2), headprim(B,12), endpoint_zone(B,0), endpoint_zone(B,1),
                     circle_at_endpoint_in_zone(B,0).
e(3,A,B,C,D,E)    :- headzone(B,3), headprim(B,10), endpoint_zone(B,1), memberzone(B,1,2),
                     havemember(B,0,1,1).
f(3,A,B,C,D,[z1]) :- not topright_tail(B), not nozone(C), headzone(B,2), memberzone(B,2,3).
g(3,A,B,C,D,E)    :- headprim(B,12), endpoint_zone(B,1), endpoint_primitive(B,5),
                     topright_tail(B), memberzone(B,0,4), havemember(B,3,1,1).
h(3,A,B,C,D,E)    :- headprim(B,11), count_circle_at_endpoint(B,1), topleft_tail(B),
                     memberzone(B,1,2).
i(3,A,B,C,D,E)    :- not nozone(C), not sizemore1_2(A), endpoint_zone(B,1),
                     count_primitive_10(B,0), count_primitive_12(B,0), memberzoen(B,5,5).
j(3,A,B,C,D,E)    :- headzone(B,2), headprim(B,12), endpoint_zone(B,1),
                     count_circle_at_endpoint(B,1), bottomright_tail(B), nozone0(C),
                     nozone3(C).
k(3,A,B,[],C,[])  :- headzone(B,2), count_primitive_6(B,0).
l(1,A,B,C,D,C)    :- endpoint_primitive(B,1), sizemore1_45(A), count_section_>_3(B),
                     count_primitive_2(B,0).
m(3,A,B,[],C,D)   :- headzone(B,2), endpoint_zone(B,1), count_circle_at_endpoint(B,0),
                     bottomright_tail(B).
n(1,A,B,C,C,C)    :- circle_at_endpoint_in_zone(B,6), sizemore1_2(A), count_primitive_0(B,0),
                     count_primitive_1(B,0).
o(1,A,B,C,C,C)    :- circle_at_endpoint_in_zone(B,6), count_endpoint(B,2), sizemore1_2(A),
                     havemember(B,1,6,6), count_primitive_0(B,0).
p(1,A,B,[z6,z6],C,D) :- sizemore1_2(A), havemember(B,3,6,6).
p(1,A,B,C,D,E)    :- count_circle_at_endpoint(B,0), sizemore1_2(A), havemember(B,4,6,6),
                     havemember(B,7,6,6), havezone(C,z6).
q(5,A,B,C,C,C)    :- count_section(B,2).
r(5,A,B,C,D,C)    :- endpoint_primitive(B,1).
s(3,A,B,[],[],[])  :- headzone(B,3), headprim(B,10).
t(2,A,B,C,D,[z6]) :- havemember(B,0,6,6), havemember(B,7,6,6).
t(2,A,B,C,D,E)    :- count_circle_at_endpoint(B,1), havemember(B,4,6,6).
```

```
η(3,A,B,C,D,E)    :- headzone(B,2), endpoint_zone(B,1), circle_at_endpoint_in_zone(B,2),
                     count_circle_at_endpoint(B,1), bottomright_tail(B), memberzone(B,1,4),
                     memberzone(B,2,1),
                     count_primitive_2(B,0), count_primitive_4(B,0), count_startzone_0(B,0).
(1,A,B,C,D,E)     :- not sizemore1_45(A), endpoint_primitive(B,1), count_endpoint_<_4(B),
                     begin_endzone(B,6,6), havemember(B,0,6,6), havezone(D,z6),
                     havezone(E,z6), headprim_0111(B).
(1,A,B,C,C,C)     :- count_circle_at_endpoint(B,0).
(1,A,B,C,D,[])    :- not sizemore1_45(A), headprim(B,12).
(1,A,B,C,D,E)     :- headprim_0110(B), endpoint_primitive(B,1), sizemore1_2(A),
                     havemember(B,2,6,6), havezone(E,z6).
(1,A,B,C,D,E)     :- not nozone(C), endpoint_primitive(B,0), count_primitive_5(B,0).
(1,A,B,C,C,C)     :- headprim(B,9), havemember(B,0,6,6).
(1,A,B,C,C,C)     :- not sizemore1_2(A), headprim(B,10).
ο(3,A,B,[],[],[])  :- headzone(B,0).
φ(3,A,B,[],C,D)   :- headzone(B,0), endpoint_zone(B,4).
β(3,A,B,C,D,E)    :- headzone(B,0), begin_endzone(B,0,2), memberzone(B,0,1).
α(3,A,B,C,D,E)    :- headzone(B,3), headprim(B,10), count_endpoint(B,3), bottomright_tail(B),
                     memberzone(B,1,4), havezone(E,z1), havezone(E,z2).
α(3,A,B,C,D,E)    :- headzone(B,0), headprim(B,11), endpoint_zone(B,1),
                     endpoint_primitive(B,7), count_endpoint(B,3).
α(3,A,B,C,D,E)    :- headzone(B,0), headprim(B,11), endpoint_zone(B,1),
                     begin_endzone(B,0,1), memberzone(B,0,2).
β(3,A,B,C,D,E)    :- headzone(B,3), headprim(B,12).
ω(3,A,B,C,[z4,z0],D) :- headzone(B,3).
ε(3,A,B,C,D,E)    :- headzone(B,4), endpoint_zone(B,1), begin_endzone(B,4,1),
                     havezone(D,z3).
α(3,A,B,C,D,E)    :- headzone(B,3), endpoint_zone(B,1), right_line(B), begin_endzone(B,3,1),
                     memberzone(B,0,0), memberzone(B,0,1).

primitive(1,0).  primitive(2,0).  primitive(3,0). ... primitive(1663,12).  primitive(1664,12).

endpoint(1,-1).  endpoint(2,-1).  endpoint(3,-1). ... endpoint(1663,0).  endpoint(1664,0).

startzone(1,0).  startzone(2,0).  startzone(3,0). ... startzone(1663,7).  startzone(1664,7).

endzone(1,0).  endzone(2,0).  endzone(3,0). ... endzone(1663,7).  endzone(1664,7).
```

ภาคผนวก จ

อัตราการรู้จำของวิธีการโปรแกรมตรรกะเชิงอุปนัยร่วมกับแบ็กพรอพาเกชันนิวรอลเน็ตเวิร์กจำแนกตามตัวอักษร

ตัวอักษร	ILP&BNN (1)	ILP&BNN (2)
ก	71.43	92.86
ข	85.71	85.71
ช	71.43	64.29
ค	71.43	71.43
ค	85.71	92.86
ฆ	78.57	78.57
ง	92.86	92.86
จ	100.00	100.00
ฉ	92.86	92.86
ช	85.71	92.86
ซ	57.14	85.71
ฌ	85.71	100.00
ญ	92.86	71.43
ฎ	71.43	85.71
ฏ	28.57	64.29
ฐ	85.71	92.86
ท	35.71	64.29
ฒ	78.57	92.86
ณ	78.57	78.57
ด	57.14	57.14
ต	57.14	85.71
ถ	100.00	100.00
ท	92.86	85.71
ธ	85.71	92.86
น	78.57	78.57
บ	64.29	92.86
ป	64.29	85.71

ตัวอักษร	ILP&BNN (1)	ILP&BNN (2)
ผ	78.57	71.43
ฝ	85.71	85.71
พ	78.57	100.00
ฟ	100.00	92.86
ภ	100.00	100.00
ม	85.71	100.00
ย	85.71	78.57
ร	100.00	92.86
ล	100.00	100.00
ว	100.00	100.00
ศ	100.00	100.00
ษ	64.29	71.43
ส	92.86	92.86
ห	92.86	92.86
ฬ	92.86	92.86
อ	100.00	100.00
ฮ	85.71	100.00
ฤ	100.00	100.00
ๆ	85.71	85.71
ะ	85.71	100.00
ั	85.71	85.71
า	100.00	100.00
ิ	78.57	85.71
ี	71.43	71.43
ึ	71.43	78.57
ื	71.43	64.29
่	78.57	78.57

ตัวอักษร	ILP&BNN (1)	ILP&BNN (2)
◌ั	78.57	78.57
เ	100.00	100.00
โ	100.00	100.00
ไ	92.86	92.86
ใ	100.00	100.00
ๆ	92.86	92.86
◌็	100.00	100.00
◌่	100.00	100.00
◌้	85.71	85.71
◌๊	78.57	85.71
◌๋	92.86	92.86
◌์	64.29	64.29

ตัวอักษร	ILP&BNN (1)	ILP&BNN (2)
◌	92.86	100.00
๐	100.00	100.00
๑	85.71	92.86
๒	100.00	100.00
๓	100.00	100.00
๔	78.57	92.86
๕	100.00	100.00
๖	71.43	64.29
๗	92. 86	92.86
๘	92.86	100.00
๙	92.86	100.00

หมายเหตุ

- ILP&BNN (1) แทนวิธีการโปรแกรมตรรกะเชิงอุปนัยร่วมกับแบ็กพรอพาเกชันนิวรอลเน็ตเวิร์กที่ใช้จำนวนสัญพจน์ที่ไม่ตรงและจำนวนสัญพจน์ที่ตรงกับตัวอย่างเป็นอินพุตเวกเตอร์
- ILP&BNN (2) แทนวิธีการโปรแกรมตรรกะเชิงอุปนัยร่วมกับแบ็กพรอพาเกชันนิวรอลเน็ตเวิร์กที่ใช้ค่าความจริงของแต่ละสัญพจน์เป็นอินพุตเวกเตอร์

ประวัติผู้วิจัย

นายสุกรี สินธุภิญโญ เกิดเมื่อวันพฤหัสบดีที่ 22 เดือนพฤษภาคม พุทธศักราช 2518 ที่จังหวัดกรุงเทพมหานคร ศึกษาระดับประถมศึกษาที่โรงเรียนทรงวิทย์ศึกษา ศึกษาระดับมัธยมศึกษาตอนต้นและมัธยมศึกษาตอนปลายที่โรงเรียนเตรียมอุดมศึกษาพัฒนาการ จบการศึกษาระดับมัธยมศึกษาตอนปลายจากศูนย์การศึกษานอกโรงเรียนวชิรธรรมสาธิต เข้าศึกษาต่อระดับปริญญาตรีเมื่อปีพุทธศักราช 2534 ที่คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย เข้าศึกษาในภาควิชาวิศวกรรมคอมพิวเตอร์ในปีถัดมา จบการศึกษาวิศวกรรมศาสตรบัณฑิต จากภาควิชาวิศวกรรมคอมพิวเตอร์ เมื่อปีพุทธศักราช 2538 ต่อมาในปีพุทธศักราช 2539 เข้ารับราชการเป็นอาจารย์ระดับ 3 ภาควิชาวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์และเทคโนโลยี มหาวิทยาลัยธรรมศาสตร์ และเข้าศึกษาต่อที่ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ในหลักสูตรวิศวกรรมศาสตรมหาบัณฑิต เมื่อภาคการเรียนที่ 2 ปีพุทธศักราช 2540