ภาคผนวก

ภาคผนวก ก

Source code MATLAB ของระบบรู้จำเสียงพูดคำโดด

ก.1 โปรแกรมบันทึกเสียง (record.m)

```
clear all; clc; close all;
dir = 'E:\thesis_speech\database_ADC\'; %บันทึกไฟล์ที่ dir
speaker = input('<< โปรแกรมบันทึกเสียงพูดจาก ADC >>\nกรุณาป้อนชื่อผู้พูด : ','s');
num = input('ครั้งที่บันทึก (01 ถึง 10) : ','s');
for I = 1:7
   repeat = 1; count = num2str(I);
   switch count
      case '1'
         subdir = 'turn_left'; fprintf('"เลี้ยวซ้าย"\n');
      case '2'
         subdir = 'turn_right'; fprintf('"เลี้ยวขวา"\n');
      case '3'
         subdir = 'forward'; fprintf('"เดินหน้า"\n');
      case '4'
         subdir = 'backward'; fprintf('"ถอยหลัง"\n');
      case '5'
         subdir = 'speed'; fprintf('"เร็วขึ้น"\n');
      case '6'
         subdir = 'slow'; fprintf('"ช้าลง"\n');
      otherwise
         subdir = 'stop'; fprintf('"หยุด"\n');
   end
   while (repeat == 1)
      s = serial('COM3'); s.BaudRate = 115200; s.InputBufferSize = 20000; fopen(s)
      fprintf('กดสวิตช์แล้วพูด...(2 วินาที)'); input('... กด Enter เพื่อเก็บข้อมูล!!!\n'); close all;
      if (s.BytesAvailable < 20000)
         fprintf('ข้อมูลไม่สมบูรณ์!!! ลองอีกครั้ง...\n\n'); repeat = 1;
      else
         data = fread(s,s.BytesAvailable,'uint8'); repeat = 0; %อ่านข้อมูล
         Speech = (data-128)/128; %แปลงข้อมูล
         speech_norm = Speech/max(Speech);
         plot(speech_norm); axis([-1000 21000 -1.1 1.1]);
      end
      fclose(s); delete(s); clear s
   end
```

```
   Dfile = [dir,speaker,'_',num2str(num),subdir,'.dat']; dlmwrite(Dfile, data) %บันทึกข้อมูล
   if I == 7
      fprintf('จบกระบวนการ !!! ขอบคุณครับ ^o^ \n'); fprintf('\n');
   end
end %for I
```

ก.2 โปรแกรมสร้างฐานข้อมูลสำหรับโครงข่ายประสาทเทียม (data.m)

```
clear all; clc; close all;
find_length = 1; %แสดงจำนวน MFCC ที่ได้จากการหาขอบเขตของคำ
fprintf('<< โปรแกรมสร้างฐานข้อมูลสำหรับ Neural Networks >>\n');
pathname = 'E:\thesis_speech\database_ADC\'; %Load ข้อมูลจาก pathname
dir = 'E:\thesis_speech\MFCC_database\'; %Save ข้อมูลที่ dir
database_f = []; %สำหรับเก็บข้อมูลลักษณะสำคัญช่วงต้น
database_b = []; %สำหรับเก็บข้อมูลลักษณะสำคัญช่วงท้าย
if find_length == 1
   mfcc_len = []; %สำหรับเก็บจำนวน MFCC ของแต่ละคำ
end
for numSpeaker = 1:10 %จำนวนผู้พูด
   switch numSpeaker
      case 1
         speaker = 'Sawitree'; display(speaker)
      case 2
         speaker = 'aon'; display(speaker)
      case 3
         speaker = 'kob'; display(speaker)
      case 4
         speaker = 'Bell'; display(speaker)
      case 5
         speaker = 'jo'; display(speaker)
      case 6
         speaker = 'ao'; display(speaker)
      case 7
         speaker = 'jaws'; display(speaker)
      case 8
         speaker = 'tea'; display(speaker)
      case 9
         speaker = 'nick'; display(speaker)
      otherwise
         speaker = 'pao'; display(speaker)
   end
   for round = 1:10 %จำนวนรอบของข้อมูลในกลุ่มตัวอย่าง
      fprintf('\nRound %.0f  \t',round);
```

```
        if round < 10 %ถ้ารอบที่ 1-9 ให้ใส่ 0 ข้างหน้า
           num = ['0',num2str(round)];
        else
           num = num2str(round);
        end
        for I = 1:7 %จำนวนคำ
           count = num2str(I);
           switch count
              case '1'
                 filename = [speaker,'_',num,'turn_left.dat'];
              case '2'
                 filename = [speaker,'_',num,'turn_right.dat'];
              case '3'
                 filename = [speaker,'_',num,'forward.dat'];
              case '4'
                 filename = [speaker,'_',num,'backward.dat'];
              case '5'
                 filename = [speaker,'_',num,'speed.dat'];
              case '6'
                 filename = [speaker,'_',num,'slow.dat'];
              otherwise
                 filename = [speaker,'_',num,'stop.dat'];
           end

           [MFCC,Mf,Mb] = sp2feature(filename,pathname);

           if find_length == 1
              fprintf('%d\t',length(MFCC)); mfc_len(I)=length(MFCC);
           end
           database_f = [database_f Mf'];
           database_b = [database_b Mb'];

        end %for I
        if find_length == 1
           [mfcc_len] = [mfcc_len; mfc_len];
        end
     end %for round
end %for numSpeaker
namef = [dir,'database_f','.txt'];
dlmwrite(namef,database_f); %บันทึกข้อมูลลักษณะสำคัญช่วงต้นของผู้พูดแต่ละคน
nameb = [dir,'database_b','.txt'];
dlmwrite(nameb,database_b); %บันทึกข้อมูลลักษณะสำคัญช่วงท้ายของผู้พูดแต่ละคน
if find_length == 1
   fprintf('\nค่าเฉลี่ยของจำนวน MFCC %f',mean(mfcc_len))
end
fprintf('\nจบกระบวนการ !!!\n\n');
```

ก.3 ฟังก์ชัน Preprocessing และ Feature extraction (sp2feature.m)

```
function [MFCC,Mf,Mb] = sp2feature(filename,pathname);

%<< Speech data>>
data = dlmread([pathname,filename]);
Speech = (data-128)/128; Speech = single(Speech);

%<< Preprocessing >>
Snorm = Speech/max(Speech); %Amplitude normalization
%Preemphasis
LSnorm = length(Snorm);
Sprem(1) = 0; Sprem(2:LSnorm,1) = Snorm(2:LSnorm)-(0.95*Snorm(1:LSnorm-1));

%การเพิ่มค่า Delta ให้สัญญาณ
samp = 236; %จำนวน sample ในแต่ละเฟรม
numFrame = floor(20000/(samp)); %จำนวนเฟรม
mu = 0.25;
Del2 = mu*( (-1)*Sprem(1:samp) + 1*Sprem(samp*2+1:samp*3) ...
+ 2*Sprem(samp*3+1:samp*4) );
Del1 = mu*( 1*Sprem(samp+1:samp*2) + 2*Sprem(samp*2+1:samp*3) );
Del = [Del1 Del2];
for curFrame = 3:numFrame-2
   delFrame = [curFrame-2 curFrame-1 curFrame+1 curFrame+2];
   Deln = mu*( (-2)*Sprem( delFrame(1)*samp-(samp-1):delFrame(1)*samp ) ...
   + (-1)*Sprem( delFrame(2)*samp-(samp-1):delFrame(2)*samp ) ...
   + 1*Sprem( delFrame(3)*samp-(samp-1):delFrame(3)*samp ) ...
   + 2*Sprem( delFrame(4)*samp-(samp-1):delFrame(4)*samp ) );
   Del = [Del Deln];
end
Del_lst1 = mu*( (-2)*Sprem((numFrame*samp)-samp*3+1:(numFrame*samp) ...
- samp*2) + (-1)*Sprem((numFrame*samp)-samp*2+1:(numFrame*samp)-samp) );
Del_lst2 = mu*( (-2)*Sprem((numFrame*samp)-samp*4+1:(numFrame*samp) ...
- samp*3) + (-1)*Sprem((numFrame*samp)-samp*3+1:(numFrame*samp) ...
- samp*2) + 1*Sprem((numFrame*samp)-samp+1:(numFrame*samp)) );
Del = [Del Del_lst2 Del_lst1];

Delta = Del(1:length(Del(:)))'; %ค่าความแตกต่างของเฟรมรอบข้าง
Sdel = Sprem(1:length(Delta)) + Delta;

%การวางกรอบสัญญาณ
overlap = 20; %จำนวน sample ที่เหลื่อมกันในแต่ละเฟรม
fst_frm = Sdel(1:samp+overlap); %สัญญาณในเฟรมแรก
hw = hamming(samp+overlap); %hamming window
```

```
%<< Feature extraction >>
pow_ham = (fst_frm.*hw).^2; %สัญญาณในเฟรมแรกคูณกับ hamming window
En(1) = sum(pow_ham); %พลังงานของเสียงในเฟรมแรก
pre_spec = abs(fft((fst_frm.*hw))); %Magnitude ของ spectrum
Lhalf = (length(pre_spec)/2)+1;
pow_spec = (pre_spec(1:Lhalf).^2)/(samp+overlap); %Power spectrum ในเฟรมแรก
use_compens = 1;
if use_compens == 1; %ปรับระดับ Power spectrum
   compens = ones(129,1);
   compens(4:10) = 1.2;
   compens(11:20) = 1.4;
   compens(21:40) = 1.3;
   compens(41:60) = 1.2;
   compens(61:90) = 1.1;
   pow_spec = pow_spec.*compens;
end
bnk = melbankm(8,256,11025,0,0.5,'m'); %Mel-filter bank จาก [22]
m_spec(1:8) = bnk(1:8,:)*pow_spec;
mfc = dct(log(m_spec)); %MFCC ในเฟรมแรก
for curFrame = 2:numFrame
   sig_frm = Sdel(1+(curFrame-1)*samp-overlap : samp*curFrame);
   pow_ham = (sig_frm.*hw).^2; %สัญญาณในแต่ละเฟรมคูณกับ hamming window
   En(curFrame) = sum(pow_ham); %พลังงานของเสียงในแต่ละเฟรม
   pre_spec = abs(fft(sig_frm.*hw));
   pow_spec = (pre_spec(1:Lhalf).^2)/(samp+overlap); %Power spectrum ในแต่ละเฟรม
   if use_compens == 1; %ปรับระดับ Power spectrum
      pow_spec = pow_spec.*compens;
   end
   m_spec(1:8) = bnk(1:8,:)*pow_spec;
   mfcn = dct(log(m_spec)); %MFCC ในแต่ละเฟรม
   [mfc] = [mfc mfcn];
end

%<< Euclidean distance >>
%กำหนดให้ 3 เฟรมแรกและ 3 เฟรมสุดท้ายเป็นช่วงเงียบ
mfc_sil1 = mfc(1:8);
mfc_sil2 = mfc(8+1:8*2);
mfc_sil3 = mfc(8*2+1:8*3);
mfc_sil_lst3 = mfc(end-8*3+1:end-8*2);
mfc_sil_lst2 = mfc(end-8*2+1:end-8);
mfc_sil_lst1 = mfc(end-8+1:end);
mfc_meanh = (mfc_sil1 + mfc_sil2 + mfc_sil3)/3; %ค่าเฉลี่ยช่วงต้น
mfc_meant = (mfc_sil_lst3 + mfc_sil_lst2 + mfc_sil_lst1)/3; %ค่าเฉลี่ยช่วงท้าย
mfc_mean = (mfc_meanh + mfc_meant)/2; %ค่าเฉลี่ยของ MFCC ในช่วงเงียบ
```

```
for mFrame = 1:numFrame;
   ceps = mfc(1+(mFrame-1)*8:mFrame*8);
   Ud(mFrame) = sum((ceps-mfc_mean).^2); %ค่าระยะห่างแบบ Euclidean ในแต่ละเฟรม
end

%<< Endpoint detection >>
Uav = [Ud(1) Ud(2) Ud(3) Ud(numFrame-2) Ud(numFrame-1) Ud(numFrame)];
Eav = [En(1) En(2) En(3) En(numFrame-2) En(numFrame-1) En(numFrame)];
t = 0.5; beta = 0.75; eta = 1.1; gamma = 1.3; alpha = 3;
Esil = (sum(Eav)/6) + t; Dsil = sum(Uav)/6;
D_ref1 = Dsil*alpha;    E_ref1 = Esil*beta; %เงื่อนไขที่ 1
D_ref2 = Dsil;          E_ref2 = Esil*eta;  %เงื่อนไขที่ 2
E_ref3 = Esil*gamma;                        %เงื่อนไขที่ 3
for point = 1:numFrame
   if ( En(point) > E_ref1 && Ud(point) > D_ref1 ) || ...
     ( En(point) > E_ref2 && Ud(point) > D_ref2 ) || ...
     ( En(point) > E_ref3 )
      detect(point) = 1; %เฟรมที่เป็นเสียงพูด
   else
      detect(point) = 0; %เฟรมที่ไม่ใช่เสียงพูด
   end
end
E_begin = max( [En(1),En(2),En(3)] );
E_end = max( [En(end-2),En(end-1),En(end)] );
Q = detect(2:end)-detect(1:end-1);
Qb = find(Q == 1); %ตำแหน่งเฟรมเริ่มต้น
Qe = find(Q == -1); %ตำแหน่งเฟรมสิ้นสุด
if length(Qb) > length(Qe)
   Qb(end) = [];
end
if length(Qe) > length(Qb)
   Qe(1) = [];
end
if (length(Qb) > 6) | (length(Qe) > 6 )
   fprintf('(จำนวนคู่ตำแหน่ง > 6)');
end
if ~isempty(Qb & Qe)
   s = length(Qb); %จำนวนคู่ตำแหน่ง
   while (s > 1)
      if ( Qb(s)-Qe(s-1) == 1 ) %ถ้าคู่ตำแหน่งห่างกันเพียง 1 เฟรม
         Qb(s) = []; Qe(s-1) = [];
      end
      s = s-1;
   end
   k1 = []; k2 = []; s = length(Qb);
```

```
    while s > 0
        k1(s) = 8*Qb(s); %ตำแหน่งเริ่มต้นของคู่ที่ s
        k2(s) = 8*Qe(s); %ตำแหน่งสิ้นสุดของคู่ที่ s
        s = s-1;
    end
    MFCC = [];
    for numcut = 1:length(k1)
      MFCC = [MFCC mfc(k1(numcut)+1:k2(numcut))]; %MFCC ในช่วงที่เป็นเสียงพูด
    end
else
    MFCC = [];
end

if length(MFCC) > 8
    %การปรับจำนวนค่าลักษณะสำคัญช่วงต้นและช่วงท้าย
    p = 250/2; half = floor(length(MFCC)/2); ovl = floor(half/2);
    featuref = MFCC( 1:half+ovl );
    featureb = MFCC( 1+end-half-ovl:end );
    if length(featuref) < p
        Mf = [featuref zeros(1, p-length(featuref))];
    else
        Mf = MFCC(1:p);
    end
    if length(featureb) < p
        Mb = [zeros(1, p-length(featureb)) featureb];
    else
        Mb = MFCC(end-p+1:end);
    end
else
    MFCC = []; Mf = []; Mb = [];
    fprintf('เกิดข้อผิดพลาด! ไม่สามารถหาคู่ตำแหน่งของคำได้\n');
end

%หาตำแหน่งของเสียงพูดในสัญญาณ
Sk1 = []; Sk2 = []; s = length(Qb);
while s > 0
    Sk1(s) = samp*Qb(s);
    Sk2(s) = samp*Qe(s);
    s = s-1;
end
Sdet = [];
for numdet = 1:length(Sk1)
   Sdet = [Sdet; Sprem(Sk1(numdet)+1:Sk2(numdet))];
end
%wavplay(Sdet,10000)
```

ก.4 โปรแกรมฝึกโครงข่ายประสาทเทียม (train.m)

```
clear all; clc; close all;
fprintf('<< โปรแกรมฝึก Neural Networks >>\n\n');
nfil = 10; %จำนวนรอบของข้อมูลในกลุ่มตัวอย่าง
numf = 250/2; %จำนวนค่าลักษณะสำคัญช่วงต้นและช่วงท้าย
numSpeaker = 10; %จำนวนผู้พูด
goal = 0.00001;
epochs = 250;
d_f = []; %สำหรับเก็บข้อมูลลักษณะสำคัญช่วงต้น
d_b = []; %สำหรับเก็บข้อมูลลักษณะสำคัญช่วงท้าย
for sec = 1:2
   switch sec
      case 1
         filename = ['E:\thesis_speech\MFCC_database\',...
                  'database_f','.txt'];
         d_f = dlmread(filename); %อ่านข้อมูลลักษณะสำคัญช่วงต้น
      otherwise
         filename = ['E:\thesis_speech\MFCC_database\',...
                  'database_b','.txt'];
         d_b = dlmread(filename); %อ่านข้อมูลลักษณะสำคัญช่วงท้าย
   end
end
%ต้องการสอนให้ผลลัพท์ของ Neural networks ออกมาใกล้เคียงกับ desire
desire = [1 0 0 0 0 0 0 ;
          0 1 0 0 0 0 0 ;
          0 0 1 0 0 0 0 ;
          0 0 0 1 0 0 0 ;
          0 0 0 0 1 0 0 ;
          0 0 0 0 0 1 0 ;
          0 0 0 0 0 0 1];
output = []; %สำหรับเก็บ desire
for o = 1:nfil*numSpeaker
   output = [output desire];
end

tic %เริ่มจับเวลา
%เริ่มการ train ช่วงต้น (Network 1)
%สร้าง Hopfield Network เพื่อปรับสมดุลของค่า d_f (ได้ค่า dd เพื่อนำไปใช้แทน d_f)
Hnet = newhop(d_f);
HW = Hnet.LW{1,1};
Hb = Hnet.b{1,1};
de = HW*d_f;
```

```
for H = 1:nfil*7*numSpeaker;
   dd(:,H) = de(:,H)+Hb;
end
mnmx = [];
for i = 1:numf
   m = minmax(dd(i,:));
   mnmx = [mnmx; m];
end
msEr = 1; %Mean squared error
while msEr > goal
   %สร้าง feed-forward backpropagation network
   Snet = newff(mnmx,[25,18,16,7],{'logsig','logsig','logsig',...
   'logsig'},'trainscg');
   Snet.trainParam.epochs = epochs;
   Snet.trainParam.goal = goal;
   Snet.trainParam.show = 100;
   [Snet,tr,Y,E] = train(Snet,dd,output);
   msEr = mse(E);
end
iwf = Snet.IW{1}; iwft = iwf'; iwf_bin = iwft(1:length(iwft(:)));
lw21f = Snet.LW{2,1}; lw21ft = lw21f'; lw21f_bin = lw21ft(1:length(lw21ft(:)));
lw32f = Snet.LW{3,2}; lw32ft = lw32f'; lw32f_bin = lw32ft(1:length(lw32ft(:)));
lw43f = Snet.LW{4,3}; lw43ft = lw43f'; lw43f_bin = lw43ft(1:length(lw43ft(:)));
b1f = Snet.b{1}; b1f_bin = b1f';
b2f = Snet.b{2}; b2f_bin = b2f';
b3f = Snet.b{3}; b3f_bin = b3f';
b4f = Snet.b{4}; b4f_bin = b4f';
save('weight_bias_f.mat','iwf','lw21f','lw32f','lw43f',...
    'b1f','b2f','b3f','b4f'); %บันทึกค่า weight & bias ช่วงต้น
fprintf('สิ้นสุดการฝึกช่วงต้น\n\n');

%เริ่มการ train ช่วงท้าย (Network 2)
%สร้าง Hopfield Network เพื่อปรับสมดุลของค่า d_f (ได้ค่า dd เพื่อนำไปใช้แทน d_b)
Hnet = newhop(d_b);
HW = Hnet.LW{1,1};
Hb = Hnet.b{1,1};
de = HW*d_b;
for H = 1:nfil*7*numSpeaker
   dd(:,H) = de(:,H)+Hb;
end
mnmx = [];
for i = 1:numf
   m = minmax(dd(i,:));
   mnmx = [mnmx; m];
end
msEr = 1; %Mean squared error
```

```
while msEr > goal
   %สร้าง feed-forward backpropagation network
   Snet = newff(mnmx,[25,18,16,7],{'logsig','logsig','logsig',...
   'logsig'},'trainscg');
   Snet.trainParam.epochs = epochs;
   Snet.trainParam.goal = goal;
   Snet.trainParam.show = 100;
   [Snet,tr,Y,E] = train(Snet,dd,output);
   msEr = mse(E);
end
iwb = Snet.IW{1}; iwbt = iwb'; iwb_bin = iwbt(1:length(iwbt(:)));
lw21b = Snet.LW{2,1}; lw21bt = lw21b'; lw21b_bin = lw21bt(1:length(lw21bt(:)));
lw32b = Snet.LW{3,2}; lw32bt = lw32b'; lw32b_bin = lw32bt(1:length(lw32bt(:)));
lw43b = Snet.LW{4,3}; lw43bt = lw43b'; lw43b_bin = lw43bt(1:length(lw43bt(:)));
b1b = Snet.b{1}; b1b_bin = b1b';
b2b = Snet.b{2}; b2b_bin = b2b';
b3b = Snet.b{3}; b3b_bin = b3b';
b4b = Snet.b{4}; b4b_bin = b4b';
save('weight_bias_b.mat','iwb','lw21b','lw32b','lw43b',...
    'b1b','b2b','b3b','b4b'); %บันทึกค่า weight & bias ช่วงท้าย
fprintf('สิ้นสุดการฝึกช่วงท้าย\n\n');
toc %สิ้นสุดการจับเวลา

weight_bias = [iwf_bin iwb_bin lw21f_bin lw21b_bin lw32f_bin lw32b_bin ...
        lw43f_bin lw43b_bin b1f_bin b1b_bin b2f_bin b2b_bin ...
        b3f_bin b3b_bin b4f_bin b4b_bin]; %นำข้อมูลมาเรียงต่อกัน
fwriteid = fopen(['e:\thesis_speech\weight_bias.bin'],'w');
count = fwrite(fwriteid,weight_bias,'float32'); %บันทึกข้อมูลแบบ binary
status = fclose(fwriteid);
```

ก.5 โปรแกรมรู้จำเสียงพูดคำโดด (speech.m)

```
clear all; clc; close all; format long g;
sp_depend = 1; %1 = รู้จำแบบขึ้นกับผู้พูด, 0 = รู้จำแบบไม่ขึ้นกับผู้พูด
decision = 0.01;
load weight_bias_f; load weight_bias_b; %Load ค่าถ่วงน้ำหนักของ Neural networks
fprintf('<< โปรแกรมรู้จำเสียงพูดคำโดด >>\n');
pathname = 'E:\thesis_speech\database_ADC\'; %Load ข้อมูลเสียงจาก pathname
if sp_depend == 1 %รู้จำแบบขึ้นกับผู้พูด
   i = 10; fault = [0 0 0 0 0 0 0 0 0 0];
else %รู้จำแบบไม่ขึ้นกับผู้พูด
   i = 5; fault = [0 0 0 0 0];
end
```

```
for numSpeaker = 1:i; %จำนวนผู้พูด
   if sp_depend == 1 %รู้จำแบบขึ้นกับผู้พูด
      switch numSpeaker
         case 1
            speaker = 'Sawitree'; display(speaker)
         case 2
            speaker = 'aon'; display(speaker)
         case 3
            speaker = 'kob'; display(speaker)
         case 4
            speaker = 'Bell'; display(speaker)
         case 5
            speaker = 'jo'; display(speaker)
         case 6
            speaker = 'ao'; display(speaker)
         case 7
            speaker = 'jaws'; display(speaker)
         case 8
            speaker = 'tea'; display(speaker)
         case 9
            speaker = 'nick'; display(speaker)
         otherwise
            speaker = 'pao'; display(speaker)
      end
      i1 = 11; i2 = 20;
   else %รู้จำแบบไม่ขึ้นกับผู้พูด
      switch numSpeaker
         case 1
            speaker = 'pik';
            display(speaker)
         case 2
            speaker = 'poonperm';
            display(speaker)
         case 3
            speaker = 'Yong';
            display(speaker)
         case 4
            speaker = 'PENG';
            display(speaker)
         otherwise
            speaker = 'pop';
            display(speaker)
      end
      i1 = 1; i2 = 10;
   end
   for round = i1:i2 %จำนวนรอบของข้อมูลในกลุ่มทดสอบ
      fprintf('\nRound %.0f  \t',round);
```

```
if round < 10 %ถ้ารอบที่ 1-9 ให้ใส่ 0 ข้างหน้า
   num = ['0',num2str(round)];
else
   num = num2str(round);
end
for I = 1:7 %จำนวนคำ
   count = num2str(I);
   switch count
      case '1'
         filename = [speaker,'_',num,'turn_left.dat'];
      case '2'
         filename = [speaker,'_',num,'turn_right.dat'];
      case '3'
         filename = [speaker,'_',num,'forward.dat'];
      case '4'
         filename = [speaker,'_',num,'backward.dat'];
      case '5'
         filename = [speaker,'_',num,'speed.dat'];
      case '6'
         filename = [speaker,'_',num,'slow.dat'];
      otherwise
         filename = [speaker,'_',num,'stop.dat'];
   end

   [MFCC,Mf,Mb] = sp2feature(filename,pathname);

   if length(MFCC) > 8
      %<<< Recognition using Neural Networks >>>
      hid1 = 1./(1+exp(-((iwf*Mf') + b1f)));
      hid2 = 1./(1+exp(-((lw21f*hid1) + b2f)));
      hid3 = 1./(1+exp(-((lw32f*hid2) + b3f)));
      resultf = 1./(1+exp(-((lw43f*hid3) + b4f))); %ผลลัพธ์ของ Network 1
      hid1 = 1./(1+exp(-((iwb*Mb') + b1b)));
      hid2 = 1./(1+exp(-((lw21b*hid1) + b2b)));
      hid3 = 1./(1+exp(-((lw32b*hid2) + b3b)));
      resultb = 1./(1+exp(-((lw43b*hid3) + b4b))); %ผลลัพธ์ของ Network 2
      result = resultf.*resultb; %ผลลัพธ์ของ Neural Networks

      if max(result) < decision
         fprintf('x');
      end
      if result(1) == max(result)
         fprintf('เลี้ยวซ้าย<- \t'); output = 1;
         if I ~= output | result(1) < decision
            fault(numSpeaker) = fault(numSpeaker)+1;
         end
```

```
            elseif result(2) == max(result)
               fprintf('เลี้ยวขวา-> \t'); output = 2;
               if I ~= output | result(2) < decision
                  fault(numSpeaker) = fault(numSpeaker)+1;
               end
            elseif result(3) == max(result)
               fprintf('เดินหน้า^^  \t'); output = 3;
               if I ~= output | result(3) < decision
                  fault(numSpeaker) = fault(numSpeaker)+1;
               end
            elseif result(4) == max(result)
               fprintf('ถอยหลัง**  \t'); output = 4;
               if I ~= output | result(4) < decision
                  fault(numSpeaker) = fault(numSpeaker)+1;
               end
            elseif result(5) == max(result)
               fprintf('เร็วขึ้น!!  \t'); output = 5;
               if I ~= output | result(5) < decision
                  fault(numSpeaker) = fault(numSpeaker)+1;
               end
            elseif result(6) == max(result)
               fprintf('ช้าลง~~   \t'); output = 6;
               if I ~= output | result(6) < decision
                  fault(numSpeaker) = fault(numSpeaker)+1;
               end
            else
               fprintf('หยุด@@    \t'); output = 7;
               if I ~= output | result(7) < decision
                  fault(numSpeaker) = fault(numSpeaker)+1;
               end
            end
         else
            fprintf('เกิดข้อผิดพลาด! ไม่สามารถหาตำแหน่งของคำได้\n');
         end
      end %for I
   end %for round
   fprintf('\nจำนวนครั้งที่รู้จำผิด = %d\n',fault(numSpeaker));
end %for numSpeaker

fault
if sp_depend == 1
   performance = (700-sum(fault))/700
else
   performance = (350-sum(fault))/350
end
```

ก.6 โปรแกรมแปลงค่าตัวเลขเป็นเลขฐาน 16 สำหรับสร้าง Look-up Table (coef2hex.m)

```
clear all; clc; close all;
hw = single(hamming(256)); hhw = hw(1:128);
hex_hamming = num2hex(hhw); %ค่า Hex ของ hamming window
N = 256;
for k = 1:N
   cs(k) = single(cos(2*pi*k/N));
   sn(k) = single(sin(2*pi*k/N));
end
sin = sn(1:64); hex_sin = num2hex(sin); %ค่า Hex ของ sin
cos = cs(1:64); hex_cos = num2hex(cos); %ค่า Hex ของ cos
bnk = melbankm(8,256,11025,0,0.5,'m'); %จาก [22]
%plot(melbankm(8,256,11025,0,0.5,'m')')
bnkm = single(full(bnk));
bnk1 = bnkm(1,:); hex_bnk1 = num2hex(bnk1); %ค่า Hex ของ filter bank 1
bnk2 = bnkm(2,:); hex_bnk2 = num2hex(bnk2); %ค่า Hex ของ filter bank 2
bnk3 = bnkm(3,:); hex_bnk3 = num2hex(bnk3); %ค่า Hex ของ filter bank 3
bnk4 = bnkm(4,:); hex_bnk4 = num2hex(bnk4); %ค่า Hex ของ filter bank 4
bnk5 = bnkm(5,:); hex_bnk5 = num2hex(bnk5); %ค่า Hex ของ filter bank 5
bnk6 = bnkm(6,:); hex_bnk6 = num2hex(bnk6); %ค่า Hex ของ filter bank 6
bnk7 = bnkm(7,:); hex_bnk7 = num2hex(bnk7); %ค่า Hex ของ filter bank 7
bnk8 = bnkm(8,:); hex_bnk8 = num2hex(bnk8); %ค่า Hex ของ filter bank 8
```

ก.7 โปรแกรมรับ-ส่งข้อมูลกับ MicroBlaze (compare.m)

```
clear all; clc; close all;
[file,path] = uigetfile('E:\thesis_speech\database_ADC\*.dat','Speech file');
data = dlmread([path,file]); fprintf('===== ส่งข้อมูล binary =====\n');
s = serial('COM3'); s.BaudRate = 115200; s.OutputBufferSize = length(data);
fopen(s); fwrite(s,data); fclose(s);
fprintf('รับข้อมูลจาก MicroBlaze...'); recv = input('กด Enter... ');
if (isempty(recv))
   fprintf('===== รับข้อมูล binary =====\n');
   s.InputBufferSize = 20000; fopen(s); fwrite(s,13)
   fprintf('รอรับข้อมูลจนครบ...\n'); input('กด Enter เพื่อเก็บข้อมูล!!!')
   data = fread(s,s.BytesAvailable,'uint8'); fclose(s);
end
delete(s); clear s;
[MFCC,Mf,Mb] = sp2feature(file,path);
```