Computational Intelligence Laboratory Exercise 2

Student Name: ChangXu
CCNU Student Number: 2019180034
UOW Student Number: 6643048

Central China Normal University Wollongong Joint Institute

1 Task One

```
#include<iostream>
#include<iomanip>
#include<fstream>
#include<cstdlib>
#include<cstdio>
#include<cmath>
#include<ctime>

using namespace std;
const int MAXN = 50;        // Max neurons in any layer
const int MAXPATS = 5000;   // Max training patterns

const long NumIts=1000;
const int NumIn=4;
const int NumHN=3;
const int NumOut=3;
const float r=0.9;
const float ObjErr=0.0001;


extern double Weight1[NumIn][NumHN]={0.2};
extern double Weight2[NumHN][NumOut   ]={0.2};//
extern double theta_H[NumHN]={0.1};
extern double theta_O[NumOut]={0.1};

void train(float **x,float **d,int NumIn,int NumOut,int NumPats)
{
    float *h1 = new float[NumHN]; // O/Ps of hidden layer
    float *y  = new float[NumOut]; // O/P of Net
    float *ad1= new float[NumHN]; // HN1 back prop errors
    float *ad2= new float[NumOut]; // O/P back prop errors
    float PatErr,MinErr,AveErr,MaxErr;  // Pattern errors
    int p,i,j;     // for loops indexes
    long ItCnt=0;  // Iteration counter
    long NumErr=0; // Error counter (added for spiral problem)

    for(;;)
    {
      // Main learning loop
```

```
MinErr=3.4e38; AveErr=0; MaxErr=-3.4e38; NumErr=0;
for(p=0;p<NumPats;p++)
{
    for(i=0;i<NumHN;i++)
    { // Cal O/P of hidden layer 1
        float in=0;
        for(j=0;j<NumIn;j++){
        in+=Weight1[j][i]*x[p][j];
        }
        in-=theta_H[i];
        h1[i]=(float)(1.0/(1.0+exp(double(-in))));
    }
    for(i=0;i<NumOut;i++)
    { // Cal O/P of output layer
        float in=0;
        for(j=0;j<NumHN;j++){
        in+=Weight2[j][i]*h1[j];
        }
        in-=theta_O[i];
        y[i]=(float)(1.0/(1.0+exp(double(-in))));
    }

    PatErr=0.0;
    for(i=0;i<NumOut;i++)
    {
        float err=y[i]-d[p][i]; // actual-desired O/P
        if(err>0)PatErr+=err; else PatErr-=err;
        NumErr += ((y[i]<0.5&&d[p][i]>=0.5)
                || (y[i]>=0.5&&d[p][i]<0.5));
    }
    if(PatErr<MinErr)MinErr=PatErr;
    if(PatErr>MaxErr)MaxErr=PatErr;
    AveErr+=PatErr;
}


for(i=0;i<NumOut;i++)
{
    ad2[i]=(d[p][i]-y[i])*y[i]*(1.0-y[i]);
    for(j=0;j<NumHN;j++)
    {
        Weight2[j][i]+= r*h1[j]*ad2[i];
        theta_O[j]-=  r*ad2[i];
    }
}

for(i=0;i<NumHN;i++)
{
    float err=0.0;
    for(j=0;j<NumOut;j++)
    err+=ad2[j]*Weight2[i][j];
    ad1[i]=err*h1[i]*(1.0-h1[i]);
```

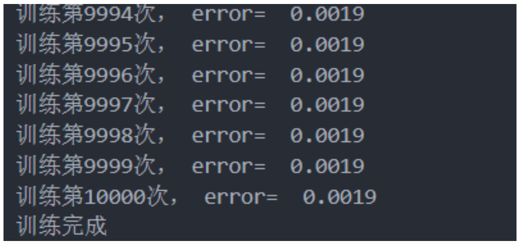
```
                for(j=0;j<NumIn;j++)
                {
                    Weight1[j][i]+=r*x[p][j]*ad1[i];
                    theta_O[j] -=  r*ad1[i];
                }
            }
        ItCnt++;
        AveErr/=NumPats;
        float PcntErr = NumErr/float(NumPats) * 100.0;
        if((AveErr<=ObjErr)||(ItCnt==NumIts)) break;
        }
    }
```

2 Task Two

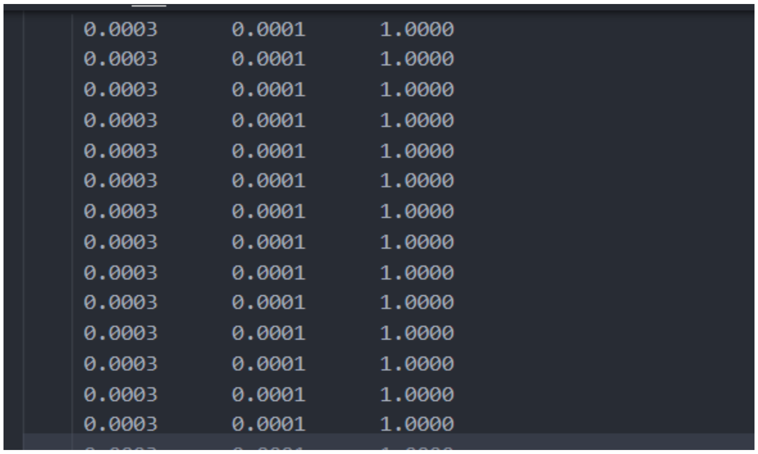
```
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#include <time.h>
#include <string>
#include <fstream>
#include <iomanip>
using namespace std;

#define innode 4
#define hidenode 10
#define outnode 3
#define trainsample 75
#define testsample 75

double trainData[trainsample][innode];
double outData[trainsample][outnode];

double testData[testsample][innode];

double w[innode][hidenode];
double w1[hidenode][outnode];
double b1[hidenode];
double b2[outnode];

double e=0.0;//
double error=1.0;//

double rate_w=0.9;//
double rate_w1=0.9;//
double rate_b1=0.9;//
double rate_b2=0.9;//

double result[outnode];//

void init(double w[], int n);
void train(double trainData[trainsample][innode],
            double label[trainsample][outnode]);
double *recognize(double *p);
void readData(std::string filename, double data[][innode], int x);
void changeData(double data[][innode], int x);

int main()
{
    int i,j;
    int trainNum=0;//
    double *r;
    int count=0;
    double maxRate = 1.0;
    //
    init((double*)w, innode*hidenode);
    init((double*)w1, hidenode*outnode);
    init(b1, hidenode);
    init(b2, outnode);
    //
    readData("./Iris-train2.txt", trainData, trainsample);
    //
    changeData(trainData, trainsample);

  for(i=0; i<trainsample; i++)
    {
        printf("%d: ",i+1);
        for(j=0; j<innode; j++)
            printf("%5.2lf",trainData[i][j]);
        printf("\n");
    }

    for(i=0; i<trainsample; i++)
    {
        if(i<25)
        {
            outData[i][0] = 1.0;
            outData[i][1] = 0.000001;
            outData[i][2] = 0.000001;
        }
        else if(i<50)
        {
            outData[i][0] = 0.000001;
            outData[i][1] = 1.0;
            outData[i][2] = 0.000001;
        }
        else
        {
            outData[i][0] = 0.000001;
            outData[i][1] = 0.000001;
            outData[i][2] = 1.0;
        }
    }
    printf ("\n");
    while(trainNum < 10000)
    {
        e = 0.0;
        trainNum++;
        train(trainData, outData);
        printf ("% d error=%8.4lf\n", trainNum, error);
    }
    printf ("\n\n");

    readData("./Iris-test.txt", testData, testsample);
    changeData(testData, testsample);
    for(i=0; i<testsample; i++)
    {
        r = recognize(testData[i]);
        for(j=0; j<outnode; j++)
            printf("\t%7.4lf\t",r[j]);
        printf("\n");
        if(i<25 && r[0]>r[1] && r[0]>r[2])
            count++;
        if(i>=25 && i<50 && r[1]>r[0] && r[1]>r[2])
            count++;
        if(i>=50 && r[2]>r[0] && r[2]>r[1])
            count++;
    }

    printf("\n\ n%    d     % d
     : %7.4lf\n\n",testsample, count, (double)count/testsample);
    system("pause");
    system("pause");
    return 0;
}


void init(double w[], int n)
{
    int i;
    srand((unsigned int)time(NULL));
    {
        w[i] = 2.0*((double)rand()/RAND_MAX)-1;
    }
}

// BP
void train(double trainData[trainsample][innode],
            double label[trainsample][outnode])
{
    double x[innode];
    double yd[outnode];

    double o1[hidenode];
    double o2[hidenode];
    double x1[hidenode];
    double x2[outnode];
    double qq[outnode];

    double pp[hidenode];


    int issamp;
    int i,j,k;
    for(issamp=0; issamp<trainsample; issamp++)
    {
        for(i=0; i<innode; i++)
            x[i] = trainData[issamp][i];

        for(i=0; i<outnode; i++)
            yd[i] = label[issamp][i];

        for(i=0; i<hidenode; i++)
        {
            o1[i] = 0.0;
            for(j=0; j<innode; j++)
                o1[i] = o1[i]+w[j][i]*x[j];
            x1[i] = 1.0/(1.0+exp(-o1[i]-b1[i]));
        }

        for(i=0; i<outnode; i++)
        {
            o2[i] = 0.0;
            for(j=0; j<hidenode; j++)
                o2[i] = o2[i]+w1[j][i]*x1[j];
            x2[i] = 1.0/(1.0+exp(-o2[i]-b2[i]));
        }


        for(i=0; i<outnode; i++)
        {
            qq[i] = (yd[i]-x2[i]) * x2[i] * (1-x2[i]);
            for(j=0; j<hidenode; j++)
                w1[j][i] = w1[j][i]+rate_w1*qq[i]*x1[j];
        }

        for(i=0; i<hidenode; i++)
        {
            pp[i] = 0.0;
            for(j=0; j<outnode; j++)
                pp[i] = pp[i]+qq[j]*w1[i][j];
            pp[i] = pp[i]*x1[i]*(1.0 - x1[i]);

            for(k=0; k<innode; k++)
                w[k][i] = w[k][i] + rate_w*pp[i]*x[k];
        }
```

```
        //
        for(k=0; k<outnode; k++)
        {
            e+=fabs(yd[k]-x2[k])*fabs(yd[k]-x2[k]);
        }
        error=e/2.0;

        for(k=0; k<outnode; k++)
            b2[k]=b2[k]+rate_b2*qq[k];

        for(j=0; j<hidenode; j++)
            b1[j]=b1[j]+rate_b1*pp[j];
    }
}

// Bp
double *recognize(double *p)
{
    double x[innode];
    double o1[hidenode];
    double o2[hidenode];
    double x1[hidenode];
    double x2[outnode];

    int i,j,k;

    for(i=0;i<innode;i++)
        x[i]=p[i];

    for(j=0;j<hidenode;j++)
    {
        o1[j]=0.0;
        for(i=0;i<innode;i++)
            o1[j]=o1[j]+w[i][j]*x[i];
        x1[j]=1.0/(1.0+exp(-o1[j]-b1[j]));
    }

    for(k=0;k<outnode;k++)
    {
        o2[k]=0.0;
        for(j=0;j<hidenode;j++)
            o2[k]=o2[k]+w1[j][k]*x1[j];
        x2[k]=1.0/(1.0+exp(-o2[k]-b2[k]));
    }

    for(k=0;k<outnode;k++)
    {
        result[k]=x2[k];
    }
    return result;
}
```

```
void readData(std::string filename, double data[][innode], int x)
{
    ifstream inData(filename, std::ios::in);
    int i,j;
    double dataLabel;
    for(i=0; i<x; i++)
    {
        for(j=0; j<innode; j++)
        {
            inData >>data[i][j];
        }
        inData >>dataLabel;
    }
    inData.close();
}

void changeData(double data[][innode], int x)
{
    double minNum,maxNum;
    int i,j;
    minNum = data[0][0];
    maxNum = data[0][0];
    for(i=0; i<x; i++)
    {
        for(j=0; j<innode; j++)
        {
            if(minNum > data[i][j])
                minNum = data[i][j];
            if(maxNum < data[i][j])
                maxNum = data[i][j];
        }
    }
    for(i=0; i<x; i++)
    {
        for(j=0; j<innode; j++)
            data[i][j] = (data[i][j]-minNum)/(maxNum-minNum);
    }
}
```



```
训练第9994次，  error=  0.0019
训练第9995次，  error=  0.0019
训练第9996次，  error=  0.0019
训练第9997次，  error=  0.0019
训练第9998次，  error=  0.0019
训练第9999次，  error=  0.0019
训练第10000次，  error=  0.0019
训练完成
```



```
0.0003      0.0001      1.0000
0.0003      0.0001      1.0000
0.0003      0.0001      1.0000
0.0003      0.0001      1.0000
0.0003      0.0001      1.0000
0.0003      0.0001      1.0000
0.0003      0.0001      1.0000
0.0003      0.0001      1.0000
0.0003      0.0001      1.0000
0.0003      0.0001      1.0000
0.0003      0.0001      1.0000
0.0003      0.0001      1.0000
0.0003      0.0001      1.0000
0.0003      0.0001      1.0000
```