Soft computing practical file

Science, Computer Science

practaSAGAR INSTITUTE OF RESEARCH ANDTECHNOLOGYSOFT COMPUTING PRACTICAL FILE (CS-801) Subject Guide: Submitted by: INDEX S. No. | List of Experiments| Signature| 1. | Implement Perceptron network with binary input and output. | | 2. | Using Madaline net, generate XOR function with bipolar inputs and targets. | | 3. | Calculation of new weights for a back propagation network, given the values of input pattern, output pattern, target output, learning rate and activation function. | | 4. | Use of ART algorithm to cluster vectors. | 5. | Implement traveling salesman problem using genetic algorithm. | | 6. | Implement various laws associated with fuzzy sets. | | 7. | Implement fuzzy sets. | | 8. | Implement word matching using GA. | | Experiment 1: Implement Perceptron network with binary input and output. Program: /*PERCEPTRON*/ #include #include main() { signed int x[4][2], tar[4]; float w[2], wc[2], out= 0; int i, j, k= 0, h= 0; float s= 0, b= 0, bc= 0, alpha= 0; float theta; clrscr(); printf(" Enter the value of theta & alpha"); scanf("%f%f",θ,α); for(i= 0; i <= 3; i++) printf(" Enter the value of %d Inputrow & Target", i); for(j= 0; j <= 1; j++) { scanf("%d",&x[i][j]);} scanf("%d",&tar[i]); w[i]= 0; wc[i]= 0;} printf(" Net TargetWeight changesNew weights Bias changesBias

"); printf("----------

"); mew: printf(" ITERATION %d

", h); printf("----------

"); for(i= 0; i <= 3; i++) {for(j= 0; j <= 1; j++) {s+=(float)x[i][j]*w[j];} s+= b; printf("%. 2f", s); if(s> theta) out= 1; else if(s <-theta) ut=-1; else { out= 0;} printf("%d", tar[i]); s= 0; if(out== tar[i]) {for(j= 0; j <= 1; j++) {wc[j]= 0; bc= 0; printf("%. 2f", wc[j]);} for(j= 0; j <= 1; j++) printf("%. 2f", w[j]);

k+= 1; b+= bc; printf("%. 2f", bc); printf("%. 2f", b); } else {for(j= 0; j <= 1; j++) {wc[j]= x[i][j]*tar[i]*alpha; w[j]+= wc[j]; printf("%. 2f", wc[j]); wc[j]= 0;} for(j= 0; j <= 1; j++) printf("%. 2f", w[j]); bc= tar[i]*alpha; b+= bc; printf("%. 2f", bc); printf("%. 2f", b); } printf("
"); } if(k== 4) {printf("
Final weights
"); for(j= 0; j <= 1; j++) {printf(" w[%d]=%. 2f", j, w[j]); } printf(" Bias b=%. 2f", b); } else k= 0; h= h+1; getch(); goto mew;}getch(); } Output:

Experiment 2: Using Madaline net, generate XOR function with bipolar inputs and targets. Program: /*MADALINE*/ #include #include main() { signed int x[4][4], tar[4]; float wc[4], w[4], e= 0, er= 0, yin= 0, alp= 0. 5, b= 0, bc= 0, t= 0; int i, j, k, q= 1; clrscr(); for(i= 0; i <= 3; i++) {printf("
Enter the %d row and target", i); for(j= 0; j <= 3; j++) {scanf("%d",&x[i][j]);} scanf("%d",&tar[i]); printf("%d", tar[i]); w[i]= 0. 0; wc[i]= 0. 0;} mew: er= 0; e= 0; yin= 0; printf("
ITERATION%d", q); printf("
----------"); or(i= 0; i <= 3; i++) {t= tar[i]; for(j= 0; j <= 3; j++) {yin= yin+x[i][j]*w[j];} b= b+bc; yin= yin+b; bc= 0. 0; printf("
Net=%f", yin); e=(float)tar[i]-yin; yin= 0. 0; printf(" Error=%f", e); printf("
Target=%d
", tar[i]); er= er+e*e; for(k= 0; k <= 3; k++) {wc[k]= x[i][k]*e*alp; w[k]+= wc[k]; wc[k]= 0. 0;} printf(" Weights "); for(k= 0; k <= 3; k++) {printf("%f", w[k]);} bc= e*alp; printf(" b=%. 2f", b); getch(); printf("
Error Square=%f", er); if(er <= 1. 000) {printf("

"); for(k= 0; k <= 1; k++) printf("%f", w[k]); getch();} else {e= 0; er= 0; yin= 0; q= q+1; goto mew;} getch();}} Output:

Experiment 3: Calculation of new weights for a back propagation network, given the values of input pattern, output pattern, target output, learning rate and activation function. Program: /*BACK PROPAGATION NETWORK*/ #include #include #include #include void main() {float v[2][4], w[4][1], vc[2][4], wc[4][1], de, del[4], bl, bia, bc[4], e= 0; float x[4][2], t[4], zin[4], delin[4], yin= 0, y, dy, dz[4], b[4], z[4], es, alp= 0. 02; int i, j, k= 0, itr= 0; v[0][0]= 0. 1970; v[0][1]= 0. 3191; v[0][2]=-0. 1448; v[0][3]= 0. 3594; v[1][0]= 0. 3099; v[1][1]= 0. 1904; v[1][2]=-0. 0347; [1][3]=-0. 4861; w[0][0]= 0. 4919; w[1][0]=-0. 2913; w[2][0]=-0. 3979; w[3][0]= 0. 3581; b[0]=-0. 3378; b[1]= 0. 2771; b[2]= 0. 2859; b[3]=-0. 3329; bl=-0. 141; x[0][0]=-1; x[0][1]=-1; x[1][0]=-1; x[1][1]= 1; x[2][0]= 1; x[2][1]=-1; x[3][0]= 1; x[3][1]= 1; t[0]= 0; t[1]= 1; t[2]= 1; t[3]= 0; clrscr(); for(itr= 0; itr <= 387; itr++) {e= 0; es= 0; for(i= 0; i <= 3; i++) {do { for(j= 0; j <= 1; j++) {zin[k] += x[i][j]*v[j][k];} zin[k]+= b[k]; k+= 1; }while(k <= 4); for(j= 0; j <= 3; j++) {z[j]=(1-exp(-zin[j]))/(1+exp(-zin[j])); dz[j]=((1+z[j])*(1-z[j]))*0. 5;} for(j= 0; j <= 3; j++) {yin+= z[j]*w[j][0];} yin+= bl; y=(1-exp(-yin))/(1+exp(-yin)); y=((1+y)*(1-y))*0. 5; de=(t[i]-y)*dy; e= t[i]-y; es+= 0. 5*(e*e); for(j= 0; j <= 3; j++) {wc[j][0]= alp*de*z[j]; delin[j]= de*w[j][0]; del[j]= delin[j]*dz[j];} bia= alp*de; for(k= 0; k <= 1; k++) {for(j= 0; j <= 3; j++) {vc[k][j]= alp*del[j]*x[i][k]; v[k][j]+= vc[k][j];}} for(j= 0; j <= 3; j++) {bc[j]= alp*del[j]; w[j][0]+= wc[j][0]; b[j]+= bc[j];} bl+= bia; for(j= 0; j <= 3; j++) {zin[j]= 0; z[j]= 0; dz[j]= 0; delin[j]= 0; del[j]= 0; bc[j]= 0;} k= 0; yin= 0; y= 0; dy= 0; bia= 0; de= 0;} printf("

Epoch %d:

", itr); for(k= 0; k <= 1; k++) {for(j= 0; j <= 3; j++) {printf("%f", v[k][j]);} printf("

");} printf("

"); for(k= 0; k <= 3; k++) {printf("%f", w[k][0]);} rintf("

%f", bl); printf(""); for(k= 0; k <= 3; k++) {printf("%f", b[k]);} getch(); } getch(); } Output: Experiment 4: Use of ART algorithm to cluster vectors. Program: /* ART NETWORK TO CLUSTER FOUR VECTORS */ #include #include main() {float n= 4. 0, m= 3. 0, o= 0. 4, l= 2. 0; float b[4][3], t[3][4], s[4], x[4], sin= 0, y[3], xin= 0; int i, j, k= 0, J, c= 0; y[0]= 0, y[1]= 0, y[2]= 0; clrscr(); for(i= 0; i <= 3; i++) {for(j= 0; j <= 2; j++) {b[i][j]= 0. 2;}} for(i= 0; i <= 2; i++) {for(j= 0; j <= 3; j++) {t[i][j]= 1. 0;}} mew: printf(" Enter the input value:

"); for(i= 0; i <= 3; i++) {scanf("%f",&s[i]); x[i]= s[i]; in+= s[i];} for(i= 0; i <= 2; i++) {printf("

Y"); do {y[i]+= s[k]*b[k][i]; k+= 1;} while(k <= 3); if(y[0]>= y[1]) {if(y[0]>= y[2]) J= 0; else J= 2;} else {if(y[1]>= y[2]) J= 1; else J= 2;} for(i= 0; i <= 3; i++) {x[i]= s[i]*t[J][i]; xin+= x[i];} if(xin/sin>= 0. 4) {for(i= 0; i <= 3; i++) {b[i][J]=(2*x[i])/(1+xin); t[J][i]= x[i];}} else {y[J]=-1;} printf("

"); for(i= 0; i <= 3; i++) {for(j= 0; j <= 2; j++) {printf("%f", b[i][j]);} printf("

");} for(i= 0; i <= 2; i++) {for(j= 0; j <= 3; j++) {printf("%f", t[i][j]);} printf("

");} getch(); y[0]= y[1]= y[2]= 0; sin= xin= 0; c+= 1; k= 0; if(c <= 3) goto mew;} getch();} Output:

Experiment 5: Implement traveling salesman problem using genetic algorithm. Program: #include #include int tsp[10][10]={{999, 10, 3, 2, 5, 6, 7, 2, 5, 4}, {20, 999, 3, 5, 10, 2, 8, 1, 15, 6}, {10, 5, 999, 7, 8, 3, 11, 12, 3, 2}, {3, 4, 5, 999, 6, 4, 10, 6, 1, 8}, {1, 2, 3, 4, 999, 5, 10, 20, 11, 2}, {8, 5, 3, 10, 2, 999, 6, 9, 20, 1}, {3, 8, 5, 2, 20, 21, 999, 3, 5, 6}, {5, 2, 1, 25, 15, 10, 6, 999, 8, 1}, {10, 11, 6, 8, 3, 4, 2, 15, 999, 1}, {5, 10, 6, 4, 15, 1, 3, 5, 2, 999}}; int pa[1000][10]= {{0, 1, 2, 3, 4, 5, 6, 7, 8, 9}, {9, 8, 6, 3, 2, 1, 0, 4, 5, 7}, {2, 3, 5, 0, 1, 4, 9, 8, 6, 7}, {4, 8, 9, 0, 1, 3, 2, 5, 6, 7}}; int i, j, k, l, m, y, loc, flag, row, col, it, x= 3, y= 3; int count, row= 0, res[1][10], row1, col1, z; nt numoff= 4; int offspring[1000][10]; int mincost= 9999, mc; main() {int gen; clrscr(); printf(" Number of Generation : "); scanf("%d",≥n); offcal1(pa); offcal2(pa); printf("

First Generation

"); for(i= 0; i ");} for(y= 1; y <= gen-1; y++) {getch(); clrscr(); for(i= 0; i %d Generation

", y+1); for(i= 0; i ");} getch(); clrscr();} rintf("

Minimum Cost Path

"); for(z= 0; z <10; z++) printf("%d ", res[0][z]); printf("

Minimum Cost %d

Menu:

1. AUB

2. A^B

3. A~

4.

B~

5. Print S, A, B

Enter the %s:

", x); for(i= 0; i n; i++) {printf(" Numerator Element %d :", i+1); scanf("%f",&f); m-> nr[i]= f; fflush(stdin); printf(" Denominator Element %d:", i+1); scanf("%f",&f); m-> dr[i]= f;}} void printval(fuzzy *m, char *x) {int i; printf("

Enter the no of componets:"); scanf("%d",&a. n); b. n= a. n; getval(&a," A"); getval(&b," B"); clrscr(); printval(&a," A"); printval(&b," B"); getch(); while(1) { clrscr(); printf("

Menu:

1. AUB

2. A^B

3. A~

4. B~

5. Print S, A, B

6. Exit"); switch((ch= getch())) {case '1': ans= unionset(a, b); printval(&ans," AUB"); getch(); break; case '2': ans= intersect(a, b); printval(&ans," A^B");

getch(); break; case '3': ans= complement(a); printval(&ans," A~"); getch(); break; case '4': ans= complement(b); printval(&ans," B~"); getch(); break; ase '5': printval(&a," A"); printval(&b," B"); getch(); break; case '6': exit(0);}}} Output: Experiment 8: Implement word matching using GA. Program: #include #include #include #include char input[15], parent[50][15], child[50][15], mating_pool[105][15], mutant[05][15]; int pfit[50], cfit[50], fit[105], mfit[05], gen= 0; void get_input() {int i; clrscr(); printf("

WORD MATCHING PROBLEM - GENETIC ALGORITHMS ASSIGNMENT"); printf("

**********"); printf("

ENTER THE WORD TO BE MATCHED : "); canf("%s", input); printf("

THE ASCII EQUIVALENT OF THE LETTERS IN THE ENTERED WORD"); printf("

----------"); printf("

LETTERS :"); for(i= 0; i ASCII :"); for(i= 0; i

THE CHROMOSOMES OF PARENTS AND CHILDREN"); rintf("

"); printf("

PREVIOUS GENERATION CHILDREN CHROMOSOMES

"); for(i= 0; i <50; i++) {if(((i)%4)== 0) printf("

"); for(j= 0; j MUTANTS OF THIS GENERATION

"); for(i= 0; i <05; i++) {if (i== 3) printf("

"); for(j= 0; j

```
THE CHROMOSOMES OF PARENTS AND CHILDREN"); printf("
----------------------------------------
"); rintf("
NEXT GENERATION PARENTS CHROMOSOMES

"); for(i= 0; i <50; i++) {if(((i)%4)== 0) printf("
"); for(j= 0; j

WORD MATCHING PROBLEM - GENETIC ALGORITHM ASSIGNMENT"); printf("
**********************************************************"); printf("

THE MATCHING WORD FOR THE GIVEN INPUT WORD"); printf("

OBTAINED USING GENETIC ALGORITHM"); printf("

"); for(i= 0; i --"); for(i= 0; i

USER INPUT : %s", input); rintf("

THE FITNESS OF THE GA GENERATED WORD AND THE USER'S INPUT");
printf("

%2d/%d", pfit[0], strlen(input)); printf("

GENERATIONS COUNT : %d", gen);} int input_choice() {int choice, i; clrscr();
printf("

GENEREATION NUMBER : %d", gen); printf("
----------"); printf("

THE FITTEST INDIVIDUAL TILL THE PREVIOUS GENERATION
```

"); for(i= 0; i

WITH A FITNESS OF %d/%d", pfit[0], strlen(input)); rintf("

ENTER YOUR CHOICE (TO CONTINUE 1 TO EXIT 0) : "); scanf("%d",&choice);

return choice;} void main() {int i, choice; clrscr(); get_input();

initial_pop(); //display(); reproduction();//sorting_based_on_fitness();

display(); printf("

ENTER YOUR CHOICE (TO CONTINUE 1 TO EXIT 0) : "); scanf("%d",&choice);

while((choice== 1)&&(pfit[0]! = strlen(input))) {crossover(); gen++;

mutation(); reproduction();//sorting_based_on_fitness(); display(); choice=

input_choice();} sound(1000); delay(200); nosound(); delay(200); results();

getch(); sound(1000); delay(200); nosound();} Output: