File - C:\IntelliJ-Projects\CSC685\README.md

```
#CSC685
Artificial Intelligence (AI) Volume Controller

Date: 25 April 2020

This project is for the National University Computer Science Masters Program
CSC685 - Topics in Computing / Artificial Intelligence focusing on Fuzzy Systems and Neural Networks

Project Author: Thi Dang
Project Author: Bindu Akella
Project Author: Sangeetha Hoskoti

Course Professor: Dr. Shatha Jawad, sjawad@nu.edu

Course Textbook: Fundamentals of Computational Intelligence: Neural Networks, Fuzzy Systems, and Evolutionary
                 Computation by James M. Keller, Derong Liu and David B. Fogel ( 2016) IEEE press

Course Description: Artificial Intelligence (AI) is a rapidly growing subject. AI technology is increasingly prevalent
                    in our everyday lives. It has uses in a variety of industries from gaming, finance, journalism/
                    media, to space-exploration, as well as in the state-of-the-art research fields from robotics,
                    medical diagnosis, and quantum science. In this course you will learn the basics and applications
                    of AI, including: Neural Networks, Fuzzy Systems, Evolutionary Computation, and machine learning.

```

File - C:\IntelliJ-Projects\CSC685\src\RuleSet.java

```java
public class RuleSet {

    private String code;
    private String type;
    private double value;
    private String time;
    private String decibel;

    public RuleSet(String code, String type, double value, String time, String decibel) {
        this.code = code;
        this.type = type;
        this.time = time;
        this.decibel = decibel;
    }

    public String getCode() { return code; }
    public void setCode(String code) { this.code = code; }

    public String getType() { return type; }
    public void setType(String type) { this.type = type; }

    public double getValue() { return value; }
    public void setValue(double value) { this.value = value; }

    public String getTime() { return time; }
    public void setTime(String time) { this.time = time; }

    public String getDecibel() { return decibel; }
    public void setDecibel(String decibel) { this.decibel = decibel; }

}

```

File - C:\IntelliJ-Projects\CSC685\src\RecordSet.java

```java
public class RecordSet {

    private String label;
    private int lowerBound;
    private int upperBound;
    private double valueFuzzySet1;
    private double valueFuzzySet2;
    private double valueFuzzySet3;

    public RecordSet(String label, int lowerBound, int upperBound, double valueFuzzySet1, double valueFuzzySet2,
                     double valueFuzzySet3) {
        this.label = label;
        this.lowerBound = lowerBound;
        this.upperBound = upperBound;
        this.valueFuzzySet1 = valueFuzzySet1;
        this.valueFuzzySet2 = valueFuzzySet2;
        this.valueFuzzySet3 = valueFuzzySet3;
    }

    public String getLabel() { return label; }
    public void setLabel(String label) { this.label = label; }

    public int getLowerBound() { return lowerBound; }
    public void setLowerBound(int lowerBound) { this.lowerBound = lowerBound; }

    public int getUpperBound() { return upperBound; }
    public void setUpperBound(int upperBound) { this.upperBound = upperBound; }

    public double getValueFuzzySet1() { return valueFuzzySet1; }
    public void setValueFuzzySet1(double valueFuzzySet1) { this.valueFuzzySet1 = valueFuzzySet1; }

    public double getValueFuzzySet2() { return valueFuzzySet2; }
    public void setValueFuzzySet2(double valueFuzzySet2) { this.valueFuzzySet2 = valueFuzzySet2; }

    public double getValueFuzzySet3() { return valueFuzzySet3; }
    public void setValueFuzzySet3(double valueFuzzySet3) { this.valueFuzzySet3 = valueFuzzySet3; }

}

```

File - C:\IntelliJ-Projects\CSC685\src\VolumeController.java

```java
import java.util.Scanner;

public class VolumeController {

    public static double calculateVolume(double[] selectedVector, RecordSet[] volumeSet) {

        /*
        This includes Step 5 (calculate the scalar Volume action using the center of gravity in which the selected
        deterministic output has a vector that divides the area under a fuzzy set into equal halves.
         */

        double[] weights = new double[10];
        double[] weightedMemberships = new double[10];
        double netActionVector = 0.0;
        double totalWeightedMembership = 0.0;
        double volume = (totalWeightedMembership) / (netActionVector);

        for (int i = 0; i < volumeSet.length; i++) {

            double lower = volumeSet[i].getLowerBound();
            double upper = volumeSet[i].getUpperBound();
            double value = selectedVector[i];
            double weight = lower + (upper - lower) / 2;
            double weightedMembership = value * weight;

            weights[i] = weight;
            weightedMemberships[i] = weightedMembership;
            totalWeightedMembership = totalWeightedMembership + weightedMembership;
            netActionVector = netActionVector + value;
            volume = (totalWeightedMembership) / (netActionVector);
        }

        return volume;
    }

    public static RuleSet[] updateRelationships(RuleSet[] relationships, RecordSet[] timeSet, RecordSet[] decibelSet,
                                                int selectedTime, int selectedDecibel) {

        /*
        This includes Step 1 (find the degree membership for each input to the related fuzzy sets by using the timeSet
        and decibelSet arrays) and Step 2 (find the degree of freedom or DOF from the fulfillment of the IF part of
        all rules by ANDing the membership of both time which is variable 1 and decibel level which is variable 2).
         */

        int sTime = selectedTime;
        int sDecibel = selectedDecibel;

        // set relationship value to the value of the timeSet fuzzy set value if the selected time is within u/l bounds
```

File - C:\IntelliJ-Projects\CSC685\src\VolumeController.java

```
49          for (int i = 0; i < timeSet.length; i++) {
50
51              if (timeSet[i].getLowerBound() < sTime && timeSet[i].getUpperBound() >= sTime) {
52
53                  // get fuzzy set values of current time set
54                  double TL = timeSet[i].getValueFuzzySet1();
55                  double TM = timeSet[i].getValueFuzzySet2();
56                  double TS = timeSet[i].getValueFuzzySet3();
57
58                  // place values in the Relationships array
59                  relationships[0].setValue(TL);
60                  relationships[1].setValue(TL);
61                  relationships[2].setValue(TL);
62                  relationships[3].setValue(TM);
63                  relationships[4].setValue(TM);
64                  relationships[5].setValue(TM);
65                  relationships[6].setValue(TS);
66                  relationships[7].setValue(TS);
67                  relationships[8].setValue(TS);
68              }
69          }
70
71          // update relationship value if it is lower than current value
72          for (int j = 0; j < decibelSet.length; j++) {
73
74              if (decibelSet[j].getLowerBound() < sDecibel && decibelSet[j].getUpperBound() >= sDecibel) {
75
76                  // get fuzzy set values of current decibel set
77                  double DH = decibelSet[j].getValueFuzzySet1();
78                  double DM = decibelSet[j].getValueFuzzySet2();
79                  double DL = decibelSet[j].getValueFuzzySet3();
80
81                  // replace value if lower than current value in Relationships array
82                  if (DH < relationships[0].getValue()) { relationships[0].setValue(DH); }
83                  if (DM < relationships[1].getValue()) { relationships[1].setValue(DM); }
84                  if (DL < relationships[2].getValue()) { relationships[2].setValue(DL); }
85                  if (DH < relationships[3].getValue()) { relationships[3].setValue(DH); }
86                  if (DM < relationships[4].getValue()) { relationships[4].setValue(DM); }
87                  if (DL < relationships[5].getValue()) { relationships[5].setValue(DL); }
88                  if (DH < relationships[6].getValue()) { relationships[6].setValue(DH); }
89                  if (DM < relationships[7].getValue()) { relationships[7].setValue(DM); }
90                  if (DL < relationships[8].getValue()) { relationships[8].setValue(DL); }
91              }
92          }
93
94          return relationships;
95      }
96
```

File - C:\IntelliJ-Projects\CSC685\src\VolumeController.java

```
 97     public static double[][] calculateVolumeActionVectors(RuleSet[] relationships, RecordSet[] volumeSet) {
 98
 99         /*
100         This includes Step 3 (calculate volume action vector for each rule by ANDing DOF with the Volume action subset
101         elements) and Step 4 (compute the net select action vector by ORing the vectors from Step 3).
102          */
103
104         // create a vector of ROWS = 9 rules with COLS = 10 non-fuzzy sets for volume
105         double[][] vectorArray = new double[9][10];
106
107         // i represents ROW or Rule
108         for (int i = 0; i < relationships.length; i++) {
109
110             String volumeType = relationships[i].getType();
111             double rValue = relationships[i].getValue();
112
113             // j represents COL or non-fuzzy set for Volume
114             for (int j = 0; j < volumeSet.length; j++) {
115
116                 if (volumeType == "High") {
117                     double VH = volumeSet[j].getValueFuzzySet1();
118                     vectorArray[i][j] = (rValue < VH) ? rValue : VH;
119                 } else if (volumeType == "Medium") {
120                     double VM = volumeSet[j].getValueFuzzySet2();
121                     vectorArray[i][j] = (rValue < VM) ? rValue : VM;
122                 } else if (volumeType == "Low") {
123                     double VL = volumeSet[j].getValueFuzzySet3();
124                     vectorArray[i][j] = (rValue < VL) ? rValue : VL;
125                 } else {
126                     Boolean error = true;
127                 }
128             }
129         }
130         return vectorArray;
131     }
132
133     public static double[] calculateSelectedVectors(double[][] vectorArray) {
134
135         double[] selectedVectors = new double[10];
136         double currentValue = 0.0;
137
138         for (int col = 0; col < 10; col++) {
139             for (int row = 0; row < 9; row++) {
140                 double vectorValue = vectorArray[row][col];
141                 currentValue = (vectorValue > currentValue) ? vectorValue : currentValue;
142             }
143             selectedVectors[col] = currentValue;
144             currentValue = 0.0;
```

```
145         }
146
147         return selectedVectors;
148     }
149
150     public static void main(String[] args) {
151
152         // Create arrays for Non-fuzzy & Fuzzy sets for Time (T)
153         String[] timeHeader = {"Time (T)", "Non-fuzzy Lower Limit", "Non-fuzzy Upper Limit", "Time Long (TL)",
154             "Time Medium (TM)", "Time Short (TS)"};
155         RecordSet[] timeSet = new RecordSet[10];
156         timeSet[0] = new RecordSet("T01", 0, 180, 0.0, 0.2, 1.0);
157         timeSet[1] = new RecordSet("T02", 180, 195, 0.1, 0.4, 0.9);
158         timeSet[2] = new RecordSet("T03", 195, 210, 0.2, 0.6, 0.8);
159         timeSet[3] = new RecordSet("T04", 210, 225, 0.3, 0.8, 0.7);
160         timeSet[4] = new RecordSet("T05", 225, 240, 0.4, 1.0, 0.5);
161         timeSet[5] = new RecordSet("T06", 240, 255, 0.5, 0.8, 0.4);
162         timeSet[6] = new RecordSet("T07", 255, 270, 0.7, 0.6, 0.3);
163         timeSet[7] = new RecordSet("T08", 270, 285, 0.8, 0.4, 0.2);
164         timeSet[8] = new RecordSet("T09", 285, 300, 0.9, 0.2, 0.1);
165         timeSet[9] = new RecordSet("T10", 300, 900, 1.0, 0.1, 0.0);
166
167         // Create arrays for Non-fuzzy & Fuzzy sets for Decibels (D)
168         String[] decibelHeader = {"Decibel (D)", "Non-fuzzy Lower Limit", "Non-fuzzy Upper Limit", "Decibel High (DH)",
169             "Decibel Medium (DM)", "Decibel Low (DL)"};
170         RecordSet[] decibelSet = new RecordSet[10];
171         decibelSet[0] = new RecordSet("D01", 0, 9, 0.0, 0.2, 1.0);
172         decibelSet[1] = new RecordSet("D02", 9, 18, 0.1, 0.4, 0.9);
173         decibelSet[2] = new RecordSet("D03", 18, 27, 0.2, 0.6, 0.8);
174         decibelSet[3] = new RecordSet("D04", 27, 36, 0.3, 0.8, 0.7);
175         decibelSet[4] = new RecordSet("D05", 36, 45, 0.4, 1.0, 0.5);
176         decibelSet[5] = new RecordSet("D06", 45, 54, 0.5, 0.8, 0.4);
177         decibelSet[6] = new RecordSet("D07", 54, 63, 0.7, 0.6, 0.3);
178         decibelSet[7] = new RecordSet("D08", 63, 72, 0.8, 0.4, 0.2);
179         decibelSet[8] = new RecordSet("D09", 72, 81, 0.9, 0.2, 0.1);
180         decibelSet[9] = new RecordSet("D10", 81, 140, 1.0, 0.1, 0.0);
181
182         // Create arrays for Non-fuzzy & Fuzzy sets for Volume (V)
183         String[] volumeHeader = {"Volume (V)", "Non-fuzzy Lower Limit", "Non-fuzzy Upper Limit", "Volume High (VH)",
184             "Volume Medium (VM)", "Volume Low (VL)"};
185         RecordSet[] volumeSet = new RecordSet[10];
186         volumeSet[0] = new RecordSet("V01", 0, 10, 0.0, 0.2, 1.0);
187         volumeSet[1] = new RecordSet("V02", 10, 20, 0.1, 0.4, 0.9);
188         volumeSet[2] = new RecordSet("V03", 20, 30, 0.2, 0.6, 0.8);
189         volumeSet[3] = new RecordSet("V04", 30, 40, 0.3, 0.8, 0.7);
190         volumeSet[4] = new RecordSet("V05", 40, 50, 0.4, 1.0, 0.5);
191         volumeSet[5] = new RecordSet("V06", 50, 60, 0.5, 0.8, 0.4);
192         volumeSet[6] = new RecordSet("V07", 60, 70, 0.7, 0.6, 0.3);
```

File - C:\IntelliJ-Projects\CSC685\src\VolumeController.java

```
193         volumeSet[7] = new RecordSet("V08", 70, 80, 0.8, 0.4, 0.2);
194         volumeSet[8] = new RecordSet("V09", 80, 90, 0.9, 0.2, 0.1);
195         volumeSet[9] = new RecordSet("V10", 90, 100, 1.0, 0.1, 0.0);
196
197         // Create array for Variable Relationships
198         RuleSet[] relationshipSet = new RuleSet[9];
199         relationshipSet[0] = new RuleSet("R1", "Low", 0.0, "TL", "DH");
200         relationshipSet[1] = new RuleSet("R2","Medium", 0.0, "TL", "DM");
201         relationshipSet[2] = new RuleSet("R3","Medium", 0.0, "TL", "DL");
202         relationshipSet[3] = new RuleSet("R4","Low", 0.0, "TM", "DH");
203         relationshipSet[4] = new RuleSet("R5","Medium", 0.0, "TM", "DM");
204         relationshipSet[5] = new RuleSet("R6","Medium", 0.0, "TM", "DL");
205         relationshipSet[6] = new RuleSet("R7","Medium", 0.0, "TS", "DH");
206         relationshipSet[7] = new RuleSet("R8","Medium", 0.0, "TS", "DM");
207         relationshipSet[8] = new RuleSet("R9","High", 0.0, "TS", "DL");
208
209         System.out.println("What is the average time between commercials/ads (180 to 300 seconds)?");
210         Scanner inputTime = new Scanner(System.in);
211         int selectedTime = inputTime.nextInt();
212
213         // if selected time value is invalid, ask again with more detail
214         if (selectedTime < 180 || selectedTime > 300) {
215             System.out.println("Incorrect time value, please enter a number between 180 and 300, for seconds");
216             selectedTime = inputTime.nextInt();
217         }
218
219         System.out.println("What is the current decibel level (0 to 90 dB)?");
220         Scanner inputDecibel = new Scanner(System.in);
221         int selectedDecibel = inputDecibel.nextInt();
222
223         // if selected decibel value is invalid, ask again with more detail
224         if (selectedDecibel < 0 || selectedDecibel > 90) {
225             System.out.println("Incorrect decibel value, please select a number between 0 and 90, for dB");
226             selectedDecibel = inputDecibel.nextInt();
227         }
228
229         // Steps 1 and 2: update the relationships table based on the selected time and decibel level
230         RuleSet[] updatedRelationships = updateRelationships(relationshipSet, timeSet, decibelSet, selectedTime,
231             selectedDecibel);
232
233         // Step 3:
234         double[][] calculatedActionVectors = calculateVolumeActionVectors(updatedRelationships, volumeSet);
235
236         // Step 4:
237         double[] calculatedSelectedVectors = calculateSelectedVectors(calculatedActionVectors);
238
239         // Step 5:
240         double calculatedVolume = calculateVolume(calculatedSelectedVectors, volumeSet);
```

File - C:\IntelliJ-Projects\CSC685\src\VolumeController.java

```
241         int integerVolume = (int) calculatedVolume;
242
243         System.out.println("With commercials or ads occurring on average, every " + selectedTime + " seconds, and " +
244                                 "the current decibel reading of " + selectedDecibel + ", the volume should be " +
245                                 "adjusted to " + integerVolume + ".");
246
247     }
248 }
249
```