Analysis of Defuzzification Methods for Network Intrusion Detection

Thuzar Hlaing
University of Computer Studies, Yangon
thuzarhlaing.ucsy@gmail.com

Abstract

Fuzzy logic is appropriated for the intrusion detection problem because many quantitative features are involved in intrusion detection. Fuzzy logic system can handle simultaneously the numerical data and linguistic knowledge. The concept of linguistic variables is used to model the state of the system which is imprecise and uncertain. The purpose of this paper is to analyze the behavior of the intrusion detection on the KDD dataset using the five defuzzification methods. The result shows that the centroid and bisector methods can detect intrusion better than the other methods for intrusion detection. The experiments and evaluations of this paper were performed with the KDD Cup 99 intrusion detection dataset. Simulation results are demonstrated by using MATLAB.

Keywords: Fuzzy Logic, Defuzzification, Centroid, Bisector, Intrusion Detection

1. Introduction

Prof. Lotfi A. Zadeh introduced the seminal paper on fuzzy sets in 1965[1]. Since then, many developments have taken place in different parts of the world. Since the 1970s Japanese researchers have been the primary force in the implementation of fuzzy theory and now have thousands of patents in the area. The uncertainties in a problem should be carefully studied by engineers prior to selecting an appropriate method to represent the uncertainty and to solve the problem. Fuzzy logic, as a robust soft computing method, has demonstrated its ability in intrusion detection systems [2]. Moreover, fuzzy systems have several important features which make them suitable for intrusion detection [3].

Fuzzy sets provide a way that is very similar to the human reasoning system. The applications of fuzzy system are information retrieval system, navigation system and robot vision. Fuzzy logic uses the reasoning of the human mind which is not always in the form of a yes or no [4]. In general, a FLS is a nonlinear mapping of an input data (feature) vector into a scalar output (the vector output case decomposes into a collection of independent multi-input/single-output systems).

This paper defines fuzzy sets for the input and output of intrusion detection data and compared the effect of using five defuzzification methods. The rest of the paper is organized as follows: Section 2 introduces about KDD dataset. Section 3 describes related work with fuzzy logic system. Section 4 explains data preprocessing step. Section 5 demonstrates a detailed fuzzy logic system and section 6 evaluations of different defuzzification methods are presented. Finally, the paper is concluded with section 7.

2. Related Work

J.Zhao and B.K.Bose [5] proposed different types of membership functions in the fuzzy control of an induction motor drive. The general membership functions under consideration are triangular, trapezoidal, gaussian, bell, sigmoidal and polynomial types. Their paper analyzed the sensitivity, evaluate and compare the effect of different types of membership functions in the fuzzy speed control of a vector-controlled induction motor drive.

J.Gomez and D.Dasgupta [6] proposed a set of fuzzy rules that used to define the normal and abnormal behavior in a computer network and a fuzzy inference engine can be applied over such rules to determine intrusions. They used a genetic algorithm to generate fuzzy classifiers for intrusion detection using datasets with patterns of the system behavior during normal and abnormal condition.

J.G.Monicka, Dr.N.O.G.Srkhar, et al [7] proposed the effect of membership functions in the fuzzy control (FC) of an ac voltage controller for speed control of induction motor drive. The different membership function evaluation is done considering seven linguistic sets for error and change in error. The simulation results showed that the triangular membership functions for fuzzifying error and change in error reduces steady state error in speed response compared to others membership functions. Use of seven linguistic variables has given a better response for fuzzifying error and change in error.

In our approach, we analyze the behavior of the intrusion detection on the KDD dataset using the five defuzzification methods: centric, bisector, mean of maximum, smallest of maximum and largest of maximum. Thus we could compare which method is better detection method for intrusion among five of them.

3. Description of KDD Dataset

The KDD Cup 1999 Intrusion Detection dataset [8] is used in this experiment. This data was prepared by the 1998 DARPA Intrusion Detection Evaluation program by MIT Lincoln Labs. In KDD99 dataset represents attribute values of class in the network data flow and each class is labeled either normal or attack. The classes in KDD99 dataset can be categorized into five main classes (one normal class and four main intrusion

classes: DOS, U2R, R2L and Probing). These four attacks are divided into 22 different attacks that tabulated in Table 1.

Table 1. Different Types of attacks in KDD99 Dataset

Four Main Attack Classes	22 Different Attacks
Denial of Service (DOS)	Back, land, Neptune, pod, smurf, teardop
User to Root (U2R)	Buffer_overflow, perl, loadmodule, rootkit
Remote to User (R2L)	ftp_write, guess_passwd, imap, multihop, phf, spy, warezclient, warezmaster
Probing	Ipsweep,nmap, portsweep, satan

There are total 41 attributes in KDD99 dataset for each network connection that have either discrete or continuous values. The list of the attributes in KDD99 dataset for each network connection is shown in Table 2.

Table 2. Input attributes in KDD99 Dataset

No	Input Attribute	Type	No	Input Attribute	Type
1	duration	Con.	22	is_guest_login	Dis.
2	protocol_type	Dis.	23	count	Con.
3	service	Dis.	24	srv_count	Con.
4	flag	Dis.	25	serror_rate	Con.
5	src_bytes	Con.	26	srv_serror_rate	Con.
6	dst_bytes	Con.	27	rerror_rate	Con.
7	land	Dis.	28	srv_rerror_rate	Con.
8	wrong_fragment	Con.	29	same_srv_rate	Con.
9	urgent	Con.	30	diff_srv_rate	Con.
10	hot	Con.	31	srv_diff_host_rate	Con.
11	num_failed_logins	Con.	32	dst_host_count	Con.
12	logged_in	Dis.	33	dst_host_srv_count	Con.
13	num_compromised	Con.	34	dst_host_same_srv_rate	Con.
14	root_shell	Con.	35	dst_host_diff_srv_rate	Con.
15	su_attempted	Con.	36	dst_host_same_src_port_rate	Con.
16	num_root	Con.	37	dst_host_srv_diff_host_rate	Con.
17	num_file_creations	Con.	38	dst_host_serror_rate	Con.
18	num_shells	Con.	39	dst_host_srv_serror_rate	Con.
19	num_access_files	Con.	40	dst_host_rerror_rate	Con.
20	num_outbound_cmds	Con.	41	dst_host_srv_rerror_rate	Con.
21	is_hot_login	Dis.	-	-	-

4. Data Normalization

Data normalization is an essential step of data preprocessing for most detection algorithms that learns the statistical characters of attributes extracted from the audit data. Data normalization is to scale the values of each continuous attributes into a well-proportioned range such that the effect of one attribute cannot dominate the others [9]. In KDD Cup 1999 data, for example , the values of attribute dst_bytes(number of data bytes from destination to source) ranges from 0 to 2293370, while the attribute " dst_host_same_src_port_rate" (same_src_port _rate for destination host) only ranges from 0 to 1. If the attributes are not normalized into the same scale, one attribute (e.g., "dst_bytes ") may overwhelm all the others and this means that one attribute is considered during the detection and the statistical detection methods thus may not be effective. So, each numerical value in the data set is normalized between 0 and 1 according to the following equation:

$$x_i = \frac{v_i - \min(v_i)}{\max(v_i) - \min(v_i)} \qquad (1)$$

Where,

x is the numerical value, min is the minimum value for the attribute that x belongs to and max is the maximum value.

5. Fuzzy Logic System (FLS)

A fuzzy logic system (FLS) can be defined as the nonlinear mapping of an input data set to a scalar output data [10]. Basically a FLS consists of four main parts: fuzzification, rules processing, inference engine, and defuzzification [11]. These components and the general architecture of a FLS are as shown in Figure 1.

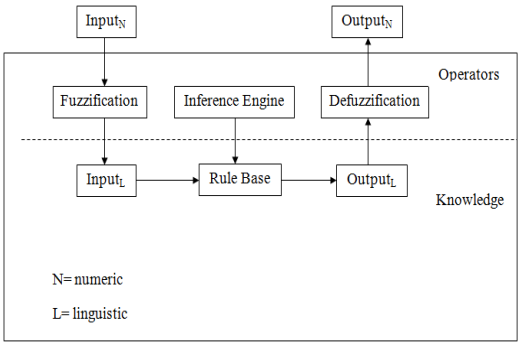

Figure 1. Fuzzy Logic System

Fuzzification: a crisp set of input data are converted to a fuzzy set using fuzzy linguistic variables, fuzzy linguistic terms and membership functions.
Rules processing: calculation the response from system status inputs according to the pre-defined rules matrix.
Inference engine: Evaluating each case for all fuzzy rules.
Defuzzificatin: the resulting fuzzy output is mapped to a crisp output using the membership functions.

5.1. Fuzzy Logic algorithm

The step by step process of fuzzy logic is explained in algorithm 1.

begin
1. Define the linguistic variables and terms (initialization)
2. Construct the membership functions (initialization)
3. Construct the rule base (initialization)
4. Convert crisp input data to fuzzy values using the membership functions (fuzzification)
5. Evaluate the rules in the rule base (inference)
6. Combine the results of each rule (inference)
7. Convert the output data to non-fuzzy values (defuzzification)

end

Figure 2. Fuzzy Logic Algorithm

5.2 Defuzzification Methods

Defuzzification is required which converts the fuzzy values into corresponding crisp values (output). These crisp values can decide the accurate detection of intrusion pattern. Therefore this paper focuses on

selection the suitable methods of defuzzification among five processing methods for intrusion detection. In this paper, the following five defuzzification methods are applied on the different defuzzification output and graph using the crisp value for intrusion detection.

I. Centroid of area z_{COA}
II. Bisector of area z_{BOA}
III. Mean of maximum z_{MOM}
IV. Smallest of maximum z_{SOM}
V. Largest of maximum z_{LOM}

I. Centroid of area

This method is also known as center of gravity or center of area defuzzification. This technique was developed by Sugeno in 1985. This is the most commonly used technique. The centroid defuzzification technique can be expressed as:

$$z_{COA} = \frac{\int \mu_A(z)\, z dz}{\int \mu_A(z) dz} \qquad (2)$$

Where z_{COA} is the crisp output, $\mu_A(z)$ is the aggregated membership function and z is the output variable.

II. Bisector Method

The bisector is the vertical line that divides the region into two sub-regions of equal area. It is sometimes, but not always coincident with the centroid line.

$$\int_{\alpha}^{Z_{BOA}} \mu_A(z) dz = \int_{Z_{BOA}}^{\beta} \mu_A(z) dz \qquad (3)$$

Where $\alpha = \min\{z/z \in Z\}$ and $\beta = \max\{z/z \in Z\}$. That is, the vertical line $z = z_{BOA}$ partitions the region between $z = \alpha$,$z = \beta$, $y = 0$ and $y = \mu_A(z)$ into two regions with the same area.

III. Mean of Maximum

Mean of maximum z_{MOM} is the average of the maximizing z at which the MF reaches a maximum μ^*. The output is computed as:

$$Z^* = (a + b)/2 \qquad (4)$$

Moreover, if $\mu_A(z)$ reaches its maximum whenever z $\in [z_{left}, z_{right}]$,then $z_{MOM} = (z_{left} + z_{right})/2$. *So,* $z_{left} = a$ and $z_{right} = b$.

IV. Largest of Maximum

Largest of maximum takes the largest amongst all z that belong to [z1, z2] as the crisp value called Z_{LOM}.

V. Smallest of Maximum

It selects the smallest output with the maximum membership function as the crisp value Z_{SOM}. In other words in Smallest of Maximum chooses smallest among all z that belong to [z1, z2].

6. Evaluation of Different Defuzzification Methods

In this section, the whole structure of the proposed solution described. It's consisting of input, reasoning rules, and output. Complete Intrusion Detection testing was developed with the help of MATLAB fuzzy logic Toolbox [12] shown in Figure 3.

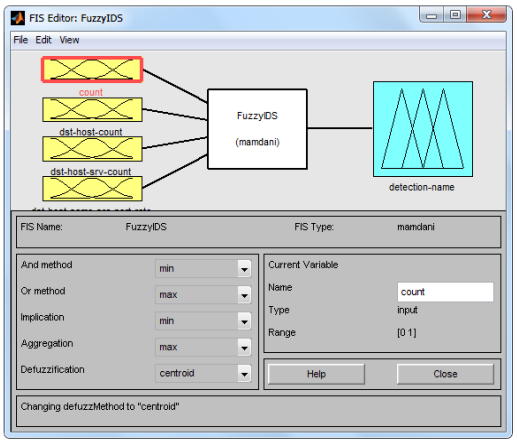

Figure 3. FIS Editor for IDS

The four input parameters: count, dst_host_count, dst_host_srv_count and dst_host_same_src_port_rate are selected from the dataset as input. Because they are important attributes for detecting the intrusion pattern. The output values are used to determine the accurate detection of intrusion pattern in the dataset.

6.1. Membership Function

Membership function is a graphical representation of the magnitude of participation of each input. There are different forms of membership functions such as triangular, trapezoidal, piecewise linear, Gaussian and bell-shaped. The most commonly used shapes for membership functions are triangular, trapezoidal, and Gaussian. Membership functions were chosen by the user arbitrarily, user's experience and perspective**.** In this paper, triangular membership function is used for fuzzification and defuzzification of a FLS because of its simplicity, easy comprehension and computational efficiency.

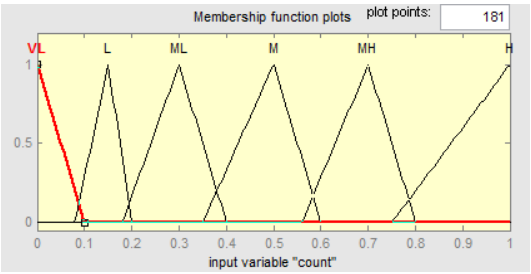

Figure 4.Membership function for T (count) = {VL (verylow), L (low), ML (mediumlow), M (medium), MH (mediumhigh), H (high)}

In Figure 4, the input parameters are ranged into VL (verylow), L (low), ML (mediumlow), M (medium), MH (mediumhigh) and H (high). All four attributes use the same range of six membership functions values.

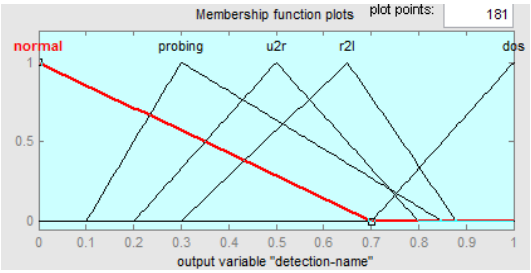

Figure.5. Membership function for output variable "detection-name"

The membership function (detection range) of the output variable "detection-name" is defined as following range shown in Figure 5.

1. normal : It ranges [-0.25,…, 0,…, 0.7]
2. probing Attack: It ranges [0.1,…, 0.3,…, 0.85]
3. User to Root (u2r) Attack: It ranges [0.2,…, 0.5,…, 0.8]
4. Remote to User (r2l) Attack: It ranges [0.3,…, 0.65,…, 0.88]
5. Denial of Service (dos) Attack: It ranges [0.7,…, 1,…, 1.2]

6.2 Rules of Attack Decision

The decision making is an important part of the entire system. The fuzzy inference system formulates suitable rules and based on these rules the decisions are made. Each input parameters have six membership functions. Therefore 6^4=1296 rules were learned respectively for four input parameters. After fuzzification process, these rules are obtained. The following are some of the fuzzy decision rules that were evolved. Some rules are as follows:

1. If (count is H) and (dst-host-count is H) and (dst-host-srv-count is ML) and (dst-host-same-src-port-rate is ML) then (detection-name is dos)

2. If (count is H) and (dst-host-count is H) and (dst-host-srv-count is M) and (dst-host-same-src-port-rate is M) then (detection-name is dos)

3. If (count is H) and (dst-host-count is H) and (dst-host-srv-count is MH) and (dst-host-same-src-port-rate is MH) then (detection-name is dos)

4. If (count is VL) and (dst-host-count is VL) and (dst-host-srv-count is VL) and (dst-host-same-src-port-rate is VL) then (detection-name is normal)

5. If (count is VL) and (dst-host-count is L) and (dst-host-srv-count is L) and (dst-host-same-src-port-rate is VL) then (detection-name is normal)

6. If (count is VL) and (dst-host-count is H) and (dst-host-srv-count is H) and (dst-host-same-src-port-rate is VL) then (detection-name is normal)

7. If (count is VL) and (dst-host-count is VL) and (dst-host-srv-count is VL) and (dst-host-same-src-port-rate is H) then (detection-name is u2r)

8. If (count is VL) and (dst-host-count is VL) and (dst-host-srv-count is VL) and (dst-host-same-src-port-rate is M) then (detection-name is u2r)

9. If (count is VL) and (dst-host-count is VL) and (dst-host-srv-count is VL) and (dst-host-same-src-port-rate is ML) then (detection-name is u2r)

10. If (count is VL) and (dst-host-count is ML) and (dst-host-srv-count is ML) and (dst-host-same-src-port-rate is VL) then (detection-name is r2l)

11. If (count is VL) and (dst-host-count is VL) and (dst-host-srv-count is L) and (dst-host-same-src-port-rate is H) then (detection-name is r2l)

12. If (count is VL) and (dst-host-count is H) and (dst-host-srv-count is VL) and (dst-host-same-src-port-rate is ML) then (detection-name is probing)

13. If (count is VL) and (dst-host-count is VL) and (dst-host-srv-count is H) and (dst-host-same-src-port-rate is H) then (detection-name is probing).

etc.

6.3 Rule Viewer

The Rule viewer displays the end of the last section in a MATLAB technical computing environment. The Rule viewer combines the results of each rule as shown in Figure 6. The first four columns (yellow plots) show the membership functions referenced by the antecedent, or the if-part of each rule. The fifth column (blue plot) shows the membership functions referenced by the consequent, or the then-part of each rule. The rule viewer shows one calculation at a time and in great detail. The Rule viewer is needed for the different defuzzification methods calculation.

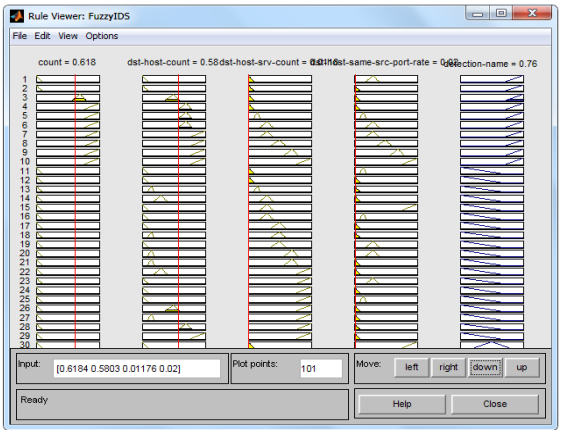

Figure 6.Rule Viewer for IDS

6.4 Defuzzification Outputs

After the calculation with different defuzzification methods from Rule viewer, the defuzzification outputs are described in Table 3. According to the experiment, the four inputs values 0.6184, 0.5804, 0.01176 and 0.02, the centroid, bisector, MOM, LOM and SOM methods produce the values of 0.867, 0.87, 0.88,1 and 0.76 respectively. According to the results, centroid, bisector and MOM methods have approximately the same results.

Table 3.Output values obtained for different defuzzification methods

	Inputs				Output				
No.	**count**	**Dst_host_ count**	**Dst_host_ srv_count**	**Dst_host_same_ src_port_rate**	**Centroid**	**Bisector**	**MOM**	**LOM**	**SOM**
1	0.6184	0.5804	0.01176	0.02	0.867	0.87	0.88	1	0.76
2	0.998	0.698	0.1294	0.19	0.867	0.87	0.88	1	0.76
3	1	0.9725	0.4039	0.42	0.878	0.88	0.905	1	0.81
4	1	0.8941	0.325	0.36	0.88	0.88	0.91	1	0.82
5	0.0215	0.149	0.5059	0.03	0.247	0.23	0.235	0.3	0
6	0.007828	0.0159	0.5451	0.25	0.263	0.25	0.261	0.31	0
7	0.001957	0.00392	0.01176	1	0.484	0.48	0.472	0.67	0.28
8	0.001957	0.03921	0.13725	1	0.486	0.49	0.485	0.74	0.23
9	0.001957	1	0.25490	0.35	0.603	0.6	0.62	0.76	0.48
10	0.00195	0.18823	0.003921	0.02	0.455	0.45	0.435	0.72	0.15

.

6.4.1 Graph representation of Defuzzification Methods

The Surface Viewer is used to display the dependency of one of the outputs on two of the inputs. The surface plots shown in Figure 7 to 11 and used the data from Table 3. Dependency of attack pattern on two input values (dat-host-count) and (count) are used for all the graphs.

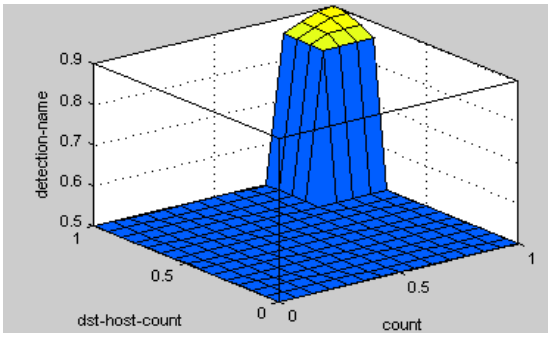

Figure 7. Centroid Method

The centroid method produces the value that is exact defuzzification ouput as shown in Figure 7.

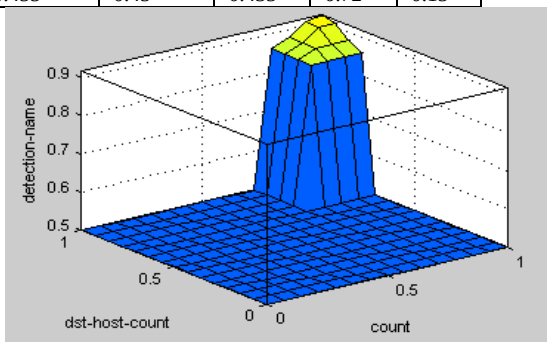

Figure 8. Bisector Method

According to the surface viewer graph in Figure 8 , the bisector method gives out the good output result because this method output is nearly the same of centroid method but the output surface is not united.

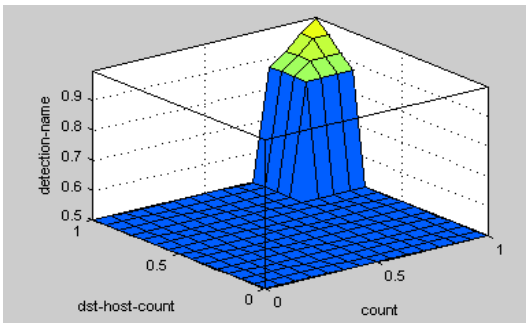

Figure 9. Mean of Maximum

The MOM method is not good defuzzification output the compare to the centroid and bisector methods .

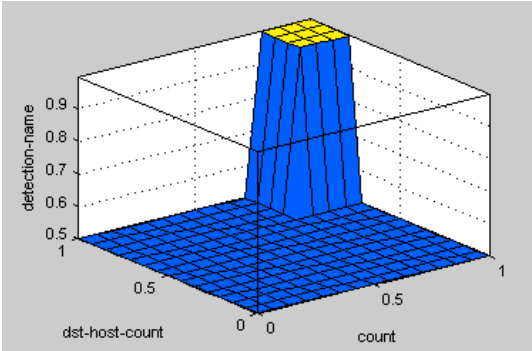

Figure 10. Largest of Maximum

The LOM and SOM see the obvious bias output as shown in Figure 10 and 11.

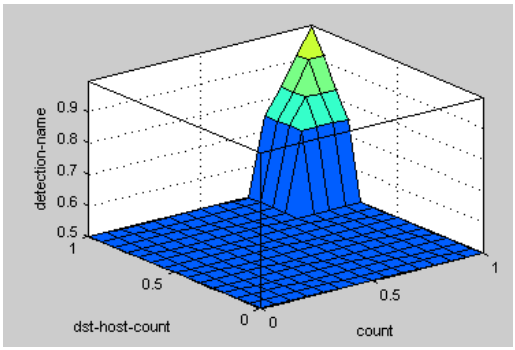

Figure 11. Smallest of Maximum

This paper measured the input parameters (X-axis and Y-axis) and the output parameter (Z-axis) on a scale of 0 to 1. In Figures 7 to 11 describe two inputs (count and dst_host_count) which have the values 0.6184 and 0.5804 applied by five different methods of defuzzification.By looking these graphs, centroid and bisector methods have the same graph representation and MOM, LOM and SOM have wide variation graph results.

6.4.2 Analysis of Defuzzification Methods

Defuzzification is the last step of the fuzzy logic system. Defuzzification is required to convert the fuzzy values into corresponding crisp values (output). To get an accurate result (output) and to choose the right defuzzification methods is very important for intrusion detection. Therefore, the different defuzzification methods need to be compared and analyzed on the data set.

According to the defuzzification output values in Table 3, the centroid, bisector and MOM methods produced the same results for intrusion detection. The output values of these three methods enter within the defined range of detection value shown in Figure 5. Therefore, these three methods can highlight on the attack data with the defined range. The three methods are appropriate for intrusion detection. The other two methods, LOM and SOM values are so high or so low that do not enter in the attack pattern range. Thus these two methods cannot detect the intrusion pattern. The graph output shows that the centroid and bisector methods have the good decision making and nearly the same graph representation. The MOM, LOM and SOM have wide variation results. So, the centroid and bisector methods are more appropriate than the other remaining methods for intrusion detection.

7. Conclusion

Fuzzy logic is a tool and can become useful and powerful when combined with Analytical Methodologies and Machine Reasoning Techniques. And then fuzzy logic is improved handling of uncertain and possibilities by using the concept of linguistic variables. Therefore fuzzy rules are human understandable. This paper has presented the different defuzzification methods to determine the optimal defuzzification method for intrusion detection.

According to the results of experiment, it shows that centroid method, bisector method and mean of maximum method have approximately the same results in the intrusion detection application. The smallest of maximum (SOM) and largest of maximum (LOM) methods are wide variations in the results. The centroid and bisector methods have the perfect of graph representation. Thus, it concludes that centroid, bisector methods are better than MOM, LOM and SOM methods. Therefore, for the future intrusion detection process, the centroid and bisector methods are the best suit amomg the different defuzzification methods.

References

[1] L.A.Zadeh,"Fuzzy sets", Information and Control, vol.8, pp.338-353, 1965.

[2] M.s.Abadeh, J.Habibi, C.Lucas, "Intrusion detection using a fuzzy genetics-based learning algorithm", Journal of Network and Computer Applications 2005.

[3] J.E.Dickerson, J. Juslin, O. Koukousoula, J.A. Dickerson, "Fuzzy intrusion detection", Proceeding of IFSA World Congress and 20th North American Fuzzy Information Processing Society Conference, NAFIPS2001, Vancouver, British Columbia, pp.1506–1510, July 2001.

[4] P.Ponce-Cruz, F.D.Ramirez-Figueroa, Intelligent Control Systems with Lab VIEW™©Springer2010

[5] J.Zhao and B.K.Bose, "Evaluation of Membership Functions for Fuzzy Logic Controlled Induction Motor Drive", IEEE 2002.

[6] J.Gomez and D.Dasgupta, "Evolving Fuzzy Classifiers for Intrusion Detection ", Proceedings of the IEEE 2002.

[7] J.Gayathri Monicka , Dr.N.O.Guna Srkhar, et al. "Performance Evaluation of Membership Functions on Fuzzy Logic Controlled AC Voltage Controller for Speed Control of Induction Motor Drive" , International Journal of Computer Applications (0975-8887),Volume 13-No.5, January 2011.

[8]KDD CUP 1999 DATASET: http://kdd.ics.uci.edu/databases/kddcup99/

[9] W.Wei, Z.Xiangliang, et al. "Attribute Normalization in Network Intrusion Detection", 10th International Symposium on Pervasive Systems, 2009.

[10]J.Mendel. "Fuzzy logic systems for engineering", IEEE, 83(3):345{377, March 1995.

[11]A short Fuzzy Logic Tutorial, April 8, 2010

[12] Math Works, "Fuzzy Logic Toolbox User's Guide", Jan 2008.