Towards Resilient Critical Infrastructures: Application of Type-2 Fuzzy Logic in Embedded Network Security Cyber Sensor

Ondrej Linda, Milos Manic, Jim Alves-Foss
University of Idaho
Idaho Falls, ID, USA
olinda@uidaho.edu, misko@ieee.org, jimaf@uidaho.edu

Todd Vollmer
Idaho National Laboratory
Idaho Falls, ID, USA
denis.vollmer@inl.gov

Abstract— **Resiliency and cyber security of modern critical infrastructures is becoming increasingly important with the growing number of threats in the cyber environment. This paper proposes an extension to a previously developed fuzzy logic based anomaly detection network security cyber sensor via incorporating Type-2 Fuzzy Logic (T2 FL). In general, fuzzy logic provides a framework for system modeling in linguistic form capable of coping with imprecise and vague meaning of words. T2 FL is an extension of Type-1 FL which proved to be successful in modeling and minimizing the effects of various kinds of dynamic uncertainties. In this paper, T2 FL provides a basis for robust anomaly detection and cyber security state awareness. In addition, the proposed algorithm was specifically developed to comply with the constrained computational requirements of low-cost embedded network security cyber sensors. The performance of the system was evaluated on a set of network data recorded from an experimental cyber security test-bed.**

Keywords-— Anomaly Detection; Cyber Sensor; Embedded Systems; Type-2 Fuzzy Logic; Online Clustering;

I. INTRODUCTION

The need for resilient control systems is increasing with the elevated levels of cyber security threats in the modern world. The resilient control system was defined in [1] as follows: "… *one that maintains state awareness and an accepted level of operational normalcy in response to disturbances, including threats of an unexpected and malicious nature*" [1]. This paper reports on work that is part of an ongoing research effort to demonstrate that computational intelligence techniques, such as fuzzy logic [2]-[4], artificial neural networks [5], [6], support vector machines [7], genetic algorithms [8], or unsupervised clustering algorithms [9]-[10] can significantly contribute to increased resilience and state awareness of modern control systems. The attractiveness of computational intelligence comes from the ability to learn from multi-dimensional non-linear data [11].

Modern critical infrastructure control systems are typically composed of interconnected computer based stations. The systems exchange crucial information via the computer network. Improving the resiliency and state awareness of these critical components, which can be found in systems such as SCADA or nuclear power plants, is inevitably conditioned by increased cyber security [12], [13]. A compromised critical infrastructure system might have security, public safety, industrial or economical consequences [14]. Therefore, security monitoring systems specifically developed for critical infrastructures is an obvious need [1]. These systems include network traffic anomaly detection.

This paper proposes an extension to a previously developed learning algorithm for a fuzzy logic based network traffic anomaly detection system via incorporating Type-2 Fuzzy Logic (T2 FL) [4]. Type-1 Fuzzy Logic Systems (T1 FLSs) are popular in many engineering areas due to their ability to cope with linguistic uncertainty originating in the imprecise and vague meaning of words. However, dynamic uncertainties such as uncertainties about the measurements activating the system or uncertainty about the training data used to tune the FLS can lead to performance deterioration [15].

Recent advances in T2 FL theory and its emerging practical applications have shown that T2 FL offers robust system control together with the capability to cope with various sources of uncertainties and disturbances [16]-[19]. In this paper, T2 FL provides the basis for robust anomaly detection and cyber security state awareness. The previously proposed anomaly detection algorithm was specifically developed for the constrained resources of embedded network security cyber sensors [4], [20]. This learning algorithm builds a fuzzy rule base, which describes the previously seen normal network communication behavioral patterns. This fuzzy rule base is constructed directly from the stream of incoming packets using an online version of the nearest neighbor clustering algorithm. Subsequently, the set of extracted clusters is transformed into a set of fuzzy rules.

This paper extends the previous work, by extending the membership functions of each fuzzy rule into T2 Fuzzy Sets (FSs) and then utilizing the T2 fuzzy inference process to compute the final classification. The major contribution of the presented work is increased state awareness of the implemented network security cyber sensor via uncertainty modeling using the T2 FLS.

The rest of this paper is structured as follows. Section II provides a brief overview of T2 fuzzy logic. The cyber security test-bed utilized in this work is described in Section III. Section IV and V explain the network behavior modeling and the proposed T2 fuzzy rule extraction method, respectively. The results of experimental evaluation of the system are presented in Section VI, and Section VII concludes the paper.

II. Type-2 Fuzzy Logic systems

This Section provides a brief background overview of T2 Fuzzy Logic Systems (FLSs). Type-1 Fuzzy Logic Systems (T1 FLSs) have been successfully applied in many engineering areas due to their ability to cope with linguistic uncertainty originating in the imprecise and vague meaning of words. However, dynamic uncertainties such as uncertainties about the measurements or the uncertainty about the training data used to tune the FLS might lead to performance deterioration [15]. This performance deterioration can be attributed to the fact that T1 FLSs use precise T1 fuzzy membership functions, parameters of which are fixed once the design process is finalized. Type-2 Fuzzy Logic Systems (T2 FLSs), originally proposed by Zadeh [21], alleviate this issue by using T2 FSs with membership degree that are themselves fuzzy sets.

The Mamdani type of T2 FLS considered in this paper maintains a fuzzy rule base populated with fuzzy linguistic rules in an implicative form. Consider rule R_k that is described as follows [16]:

$$\text{Rule } R_k\text{: } \mathbf{IF}\ x_1 \text{ is } \widetilde{A}_1^k\ \mathbf{AND} \ldots \mathbf{AND}\ x_n \text{ is } \widetilde{A}_n^k\ \mathbf{THEN}\ y_k \text{ is } \widetilde{B}^k \qquad (1)$$

Here, symbols $\widetilde{A}_i^k$ and $\widetilde{B}^k$ denote the i^{th} input T2 FS and the output T2 FS of the k^{th} rule, respectively, where n is the dimensionality of the input vector $\vec{x}$ and y_k is the associated output variable. A general T2 FS $\widetilde{A}$ can be described by its membership function $\mu_{\widetilde{A}}(x,u)$, where $x \in X$ and $u \in J_x$ [15]:

$$\widetilde{A} = \int_{x\in X}\int_{u\in J_x} \mu_{\widetilde{A}}(x,u)/(x,u) \quad J_x \subseteq [0,1] \qquad (2)$$

In Eqn. (2), variable x and u are the primary and the secondary variables and J_x denotes the interval support of the secondary membership function. The operator $\int\int$ denotes the union over all possible values of x and u, and $\mu_{\widetilde{A}}(x,u) \in [0,1]$. By restricting all secondary membership grades of T2 FS $\widetilde{A}$ to 1, an Interval T2 (IT2) FS is created:

$$\widetilde{A} = \int_{x\in X}\int_{u\in J_x} 1/(x,u) \qquad J_x \subseteq [0,1] \qquad (3)$$

The IT2 FLS is used in this work because of its computational inexpensiveness and ease of implementation [22]. In addition, the number of design parameters of IT2 FLSs is substantially smaller when compared to the full-blown general T2 FLSs. By instantiating the variable x into a specific value x', the vertical slice of the IT2 FS can be obtained as:

$$\mu_{\widetilde{A}}(x = x', u) = \mu_{\widetilde{A}}(x') = \int_{u\in J_{x'}} 1/u \quad J_{x'} \subseteq [0,1] \qquad (4)$$

The domain of the primary memberships J_x defines the Footprint-Of-Uncertainty (FOU) of FS $\widetilde{A}$:

$$FOU(\widetilde{A}) = \bigcup_{x\in X} J_x \qquad (5)$$

The FOU of an IT2 FS (schematically depicted in Fig. 3) can be completely described by its upper and lower membership functions:

$$FOU(\widetilde{A}) = \bigcup_{\forall x\in X} (\underline{\mu}_{\widetilde{A}}(x), \overline{\mu}_{\widetilde{A}}(x)) \qquad (6)$$

This constitutes a substantial simplification when compared to the general T2 fuzzy sets. Here, only two T1 fuzzy membership functions (the upper and the lower fuzzy set) are used to describe the IT2 FS $\widetilde{A}$. This simplification is then transferred through the inference mechanism of the IT2 FLS, taking advantage of the modified interval T2 fuzzy join and meet operations [22]. The interval join and meet operations work exclusively with the FOU of the IT2 fuzzy sets, thus removing much of the computational burden associated with processing of general T2 fuzzy sets.

In order to obtain a crisp output value, the resulting IT2 output fuzzy set $\widetilde{B}$ is first type-reduced and then defuzzified. The interval centroid can be described by its boundary points y_l and y_r. The final crisp defuzzified value y can be computed as the mean of the centroid interval:

$$y = \frac{(y_l + y_r)}{2} \qquad (7)$$

III. Experimental Cyber Security Test-Bed

This Section describes the hardware platform used for the implemented cyber sensor and the experimental network data acquisition test-bed.

A. Embedded Network Security Cyber Sensor

The Tofino embedded network security device, depicted in Fig. 1, is manufactured by Byres Security Inc. [23]. Originally, the device was developed for pre-emptive threat detection, termination and reporting, specifically tailored for the needs of SCADA and industrial control systems. Its major advantages are primarily its low-cost and ease of deployment in real world systems. In the presented work, the Tofino cyber sensor was used as an embedded development platform for implementation of the proposed anomaly based detection learning algorithm.

The Tofino platform consists of an Arcom Vulcan single board computer. The main processor is an Intel IXP425 XScale processor running at 533MHz with 64MB of DRAM and 32MB of flash memory. Two Ethernet ports are provided along with two USB ports. The operating system is based on the OpenWRT distribution of Linux.

The focus of the implementation is at a very low level with an envisioned deployment just before some critical equipment,

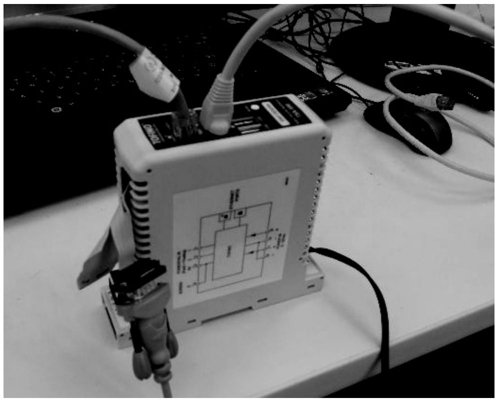


Fig. 1 Photo of the TOFINO network security cyber sensor plugged-in into the test system.

such as a Programmable Logic Controller (PLC). With the increasingly common usage of network based control systems and the current deployment of smart grid systems hundreds, thousands and possibly millions of cyber sensor devices will be deployed. This makes the cost of an implemented hardware solution a relevant concern. The selected Tofino hardware platform constitutes such a low-cost solution.

B. *Control System Experimental Test-Bed*

The experimental hardware test-bed system that was used for network data acquisition represents several aspects of an operational control system, such as operational control structure, control system network and hardware control of actual physical processes. A schematic view of the test-bed is depicted in Fig. 2. RSView32, a Rockwell Software HMI product, provides an integrated component based interface for monitoring of the system behavior. A Moxa EDS-505A Ethernet switch provides network connectivity for the controller. All network traffic to and from the controller is transported via this switch. A Linux laptop with the *tcpdump* software application was attached to the system allowing for network traffic capturing and monitoring. Finally, a second Linux based laptop representing the intruder-compromised machine was attached to a third port.

The control system itself consists of an Allen-Bradley MicroLogix 1100 PLC [24]. Attached to the PLC are 6 lighted buttons, 7 lights, 2 potentio-meters, 2 temperature sensors and a small electric fan constituting both digital and analog input/output points. All of the items are capable of being controlled individually from the PLC or directly by pressing a button.

TABLE I
SELECTED WINDOW-BASED FEATURES

Num. of IP addresses	Num. packets with 0 win. size
Avg. interval between packets	Num. packets with 0 data length
Num. of protocols	Average window size
Num. of flag codes	Average data length

IV. ONLINE BEHAVIOR MODELING FOR ANOMALY IDS

This Section presents the learning algorithm for the fuzzy logic based anomaly detection using an embedded network security cyber sensor.

A. *Feature Extraction from Packet Stream*

In a previous work of the authors, an Artificial Neural Network (ANN) based intrusion detection system was developed [6]. The ANN was trained on a sub-set of available network traffic features extracted by a window-based technique applied directly to the stream of packets. This feature extraction technique is also utilized in the presented work. The inherent time series nature of the packet stream data is described by a vector capturing the statistical behavior of the network traffic. This windowing technique extracts the statistical features from a limited set of consecutive packets.

As described in [6], a window of specified length is shifted over the stream of network packets. At each position of the window a feature vector is computed. As next arriving packet is pushed into the window, the last packet is removed from the end. The effects of different window sizes on the classification performance of the algorithm were studied in the previous work of the authors [4].

Table I summarizes the list of extracted statistical features from the packet window. This set of features was empirically selected based on analysis of the recorded network traffic and the motivation to most accurately capture the time series nature of the packet stream. For further details and evaluation of the feature extraction refer to [6].

B. *Rule Extraction via Online Clustering*

The proposed rule extraction algorithm takes into account the constrained computational resources of the available embedded network security cyber sensor. Other learning approaches, such as IDS-NNM algorithm [6], pursue an offline learning approach once all training data have been acquired. However, such a learning process is typically

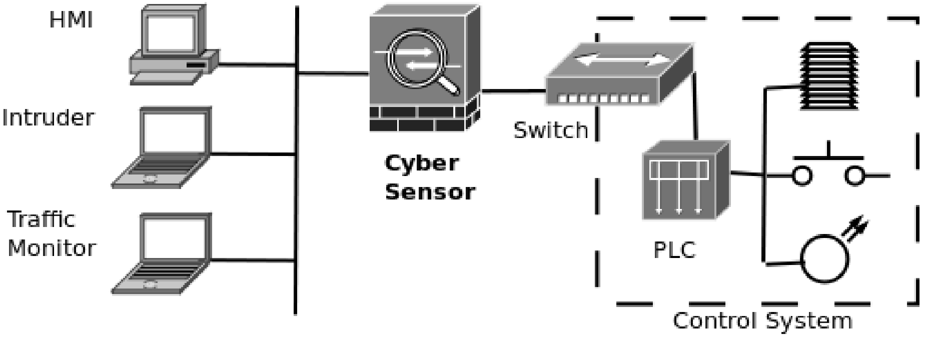


Fig. 2 Schematic diagram of the network test bed with the security cyber sensor.

computationally unfeasible for embedded devices, given the usual network traffic density of these devices [20].

In the previous work of the authors, a new low-cost online rule extraction technique was proposed [4]. Each rule is extracted using an online version of the adapted Nearest Neighbor Clustering (NNC) algorithm. The algorithm maintains additional information about the spread of data points associated with each cluster throughout the clustering process. Each cluster P_i of encountered normal network behavior is described by its center of gravity $\vec{c}_i$, weight w_i and a matrix of boundary parameters M_i. Hence:

$$P_i = \{\vec{c}_i, w_i, M_i\},\ \vec{c}_i = \{c_i^1, \ldots, c_i^n\},\ M_i = \begin{vmatrix} c_{i,1}^U & \cdots & c_{i,n}^U \\ c_{i,1}^L & \cdots & c_{i,n}^L \end{vmatrix} \qquad (8)$$

Here, i is the index of the particular cluster, c_i^j is the attribute value in the j^{th} dimension, $c_{i,j}^U$ and $c_{i,j}^L$ are the upper and lower bounds of the encountered values of the j^{th} attribute for data points assigned to cluster P_i and n denotes the dimensionality of the input. The algorithm maintains a set of clusters Ω. Initially, the algorithm starts with a single cluster P_1 positioned at the first supplied training data point $\vec{x}_1$. This initial data point becomes available once the shifting window is first filled with incoming network packets.

Upon acquiring a new data point $\vec{x}_i$ from the shifting window buffer, the set of clusters Ω is updated according to the NNC algorithm. First, the Euclidean distance to all available clusters with respect to the new input feature vector $\vec{x}_i$ is calculated. The nearest cluster P_a is identified. If the computed nearest distance is greater than the established maximum cluster radius parameter, a new cluster is created. Otherwise the nearest cluster P_a is updated according to:

$$\vec{c}_a = \frac{w_a \vec{c}_a + \vec{x}_i}{w_a + 1},\ w_a = w_a + 1 \qquad (9)$$

$$c_{i,j}^U = \max(x_i^j, c_{i,j}^U),\ c_{i,j}^L = \min(x_i^j, c_{i,j}^L) \quad j = 1 \ldots n \qquad (10)$$

Hence, the modified NNC algorithm also keeps track of the lower and upper bounds of the encountered input values in each dimension for every cluster, as opposed to the original NNC algorithm that only stored the cluster positions.

It is important to note here, that the cluster positions are dynamically changing as new input vectors are processed by the algorithm. Storing the min-max bounds of the input values might not provide accurate information about the real spread of input patterns for a particular cluster. This can be seen as new source of uncertainty, which is a tradeoff for the low memory requirements and the favored online learning capability. The proposed IT2 fuzzy rule extraction copes with this source of uncertainty by introducing IT2 fuzzy membership functions.

V. IT2 Fuzzy Rule Extraction and Inferencing for Anomaly Detection

Once the rule extraction phase of the learning process is finalized (e.g. user decision, time limit, limit on the number of packets, etc.), the learning algorithm maintains a final set of clusters Ω that describes the normal network communication behavioral patterns observed in the provided training data. In the next phase of the algorithm, each cluster is converted into an IT2 fuzzy logic rule. Each fuzzy rule then describes the similarity of observed network behavior with the normal network traffic.

An n-dimensional cluster P_i is transformed into its associated IT2 fuzzy rule R_i as follows. Rule R_i is composed of n antecedent IT2 FSs $\tilde{A}_i^j$, $j = 1..n$. Each fuzzy set $\tilde{A}_i^j$, located in the j^{th} dimension of the input space, is modeled using a non-symmetrical Gaussian fuzzy membership function with distinct left and right spreads. In the previous work [4], there were three parameters of the T1 fuzzy membership function, namely mean $m_{i,j}$ and the left and the right spreads $\delta_{i,j}^U$, $\delta_{i,j}^L$. These parameters have been previously extracted using the crisp fuzziness parameter α. In the new IT2 FLS based anomaly detection algorithm, the interval fuzziness parameter is provided as $[\underline{\alpha}, \overline{\alpha}]$. The wider the interval of the introduced fuzziness parameter, the more uncertainty will be modeled in the constructed IT2 FSs. The optimal values of the fuzziness parameter are application dependent and must be appropriately tuned. The interval fuzziness parameter allows for construction of IT2 fuzzy membership functions based on the computed cluster P_i in the following manner:

$$m_{i,j} = c_{i,j} \qquad (11)$$

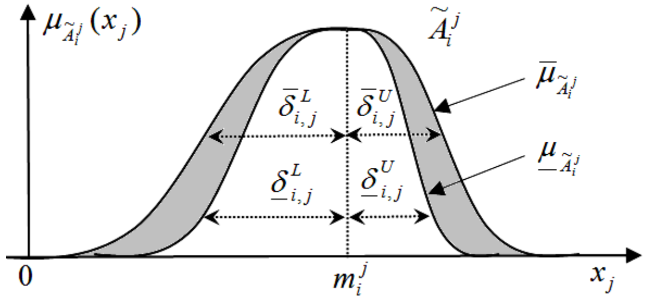
$$[\underline{\delta}_{i,j}^U, \overline{\delta}_{i,j}^U] = \left[\underline{\alpha}(c_{i,j}^U - c_{i,j}), \overline{\alpha}(c_{i,j}^U - c_{i,j})\right] \qquad (12)$$

$$[\underline{\delta}_{i,j}^L, \overline{\delta}_{i,j}^L] = \left[\underline{\alpha}(c_{i,j} - c_{i,j}^L), \overline{\alpha}(c_{i,j} - c_{i,j}^L)\right] \qquad (13)$$

The above mentioned parameters are used to construct the non-symmetric IT2 Gaussian fuzzy membership functions with uncertain spread as depicted in Fig. 3.

Using the minimum t-norm, the interval firing strength of fuzzy rule R_i is then computed as:

$$\underline{\mu}_{R_i}(\vec{x}) = \min_{j=1..n}\{\underline{\mu}_{\tilde{A}_i^j}(x_j)\} \qquad (14)$$



$\mu_{\tilde{A}_i^j}(x_j)$
$\tilde{A}_i^j$
$\overline{\mu}_{\tilde{A}_i^j}$
$\underline{\mu}_{\tilde{A}_i^j}$
$\overline{\delta}_{i,j}^L$
$\overline{\delta}_{i,j}^U$
$\underline{\delta}_{i,j}^L$
$\underline{\delta}_{i,j}^U$
0
m_i^j
x_j

Fig. 3 Illustration of the non-symmetric IT2 input Gaussian fuzzy set $\tilde{A}_i^j$.

$$\bar{\mu}_{R_i}(\vec{x}) = \min_{j=1..n}\{\bar{\mu}_{\tilde{A}_i^j}(x_j)\} \qquad (15)$$

In this specific application, the output of the fuzzy rule is a singleton fuzzy set assigning the input pattern to the normal behavior class. Hence, in this special case the fired interval output of a particular fuzzy rule is actually its own interval firing strength $[\underline{\mu}_{R_i}(\vec{x}), \bar{\mu}_{R_i}(\vec{x})]$. The final interval output centroid is obtained by applying to maximum t-conorm to the interval output of all available rules:

$$\underline{y}(\vec{x}) = \max_{i=1..C} \underline{\mu}_{R_i}(\vec{x}) \qquad (16)$$

$$\bar{y}(\vec{x}) = \max_{i=1..C} \bar{\mu}_{R_i}(\vec{x}) \qquad (17)$$

Here, C denotes the number of extracted fuzzy rules. The crisp value of the output y can be computed by defuzzifying the output interval centroid $[\underline{y}(\vec{x}), \bar{y}(\vec{x})]$ as:

$$y = \frac{(\underline{y}(\vec{x}) + \bar{y}(\vec{x}))}{2} \qquad (18)$$

The output value y together with the output interval centroid $[\underline{y}(\vec{x}), \bar{y}(\vec{x})]$ provides the basis for making decisions about the membership of input pattern $\vec{x}$ to the class of normal behavior. The following hard-partitioning scheme was proposed for the implemented IT2 FLS based anomaly detection algorithm:

If $\underline{y}(\vec{x}) >$ threshold **Then** Anomaly behavior.

Else If $\bar{y}(\vec{x}) <$ threshold **Then** Normal behavior.

Else If $\underline{y}(\vec{x}) <$ threshold$< \bar{y}(\vec{x})$ **Then** Uncertain behavior.

The detection of uncertain behavior signals that the anomaly detection system is dealing with uncertain input data and that the output decision would be subject to increased uncertainty. The final crisp Anomaly/Normal decision can then be made using the classical hard-partitioning scheme based on the crisp defuzzified value y or by performing additional in-depth analysis of the network traffic. As such, the cyber security state awareness of the protected infrastructure is increased, which is a necessary factor for promoting its resiliency.

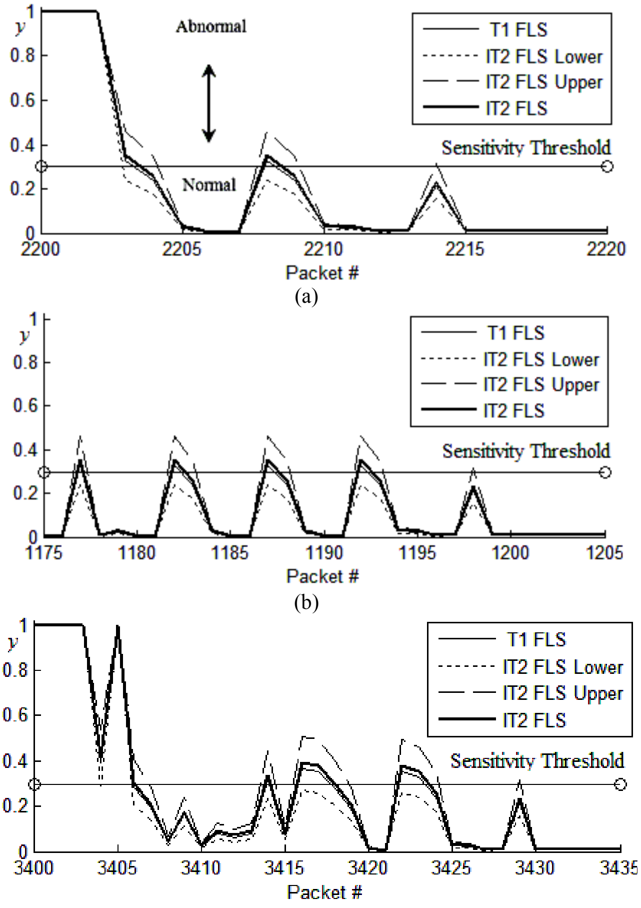
VI. Experimental Results

This section first describes the acquired experimental datasets. Next, the improved uncertainty handling is demonstrated. Finally, the classification performance is evaluated on the previously acquired testing datasets.

A. Experimental Datasets

The *Nmap* [25] and *Nessus* [26] software utilities were used to create anomalous network traffic behavior in an attempt to emulate the probes of a cyber attacker. *Nmap* is a network scanning tool commonly used to identify hosts, scan ports, operating systems and to determine applications that are listening on open ports. *Nessus* is a network scanning tool that provides auditing capabilities, vulnerability assessments and profiling information. The simulated intrusion attempts include: ARP pings, SYN stealth scans, port scanning, open port identification and others. Cyber attacks ranged from long attacks composed of many packets to very short intrusion sequences.

Two datasets of experimental data have been recorded. Because it is assumed that only normal network data are available during the training phase, the first recorded training set is composed of 6 datasets with only normal network behavior. Overall, 60,661 packets of normal network traffic were acquired including specialized normal behavior such as system initialization and system component reconnection. The normality of the recorded data was ensured by maintaining an isolated closed network and thus preventing any presence of intrusive attempts. This dataset was used only during the training phase of the algorithms. The second set is a testing set composed of 10 datasets, which include simulated abnormal behavior along with normal behavior. Overall 583,637 packets have been recorded. This datasets was not used during the training phase.



(a)

(b)

(c)

Fig. 4 Uncertainty handling with the IT2 FL based anomaly detection system for near-similar intrusion attempts (a), unusual normal behavior (b), and their combined case (c).

TABLE II
CLASSIFICATION PERFORMANCE OF THE FUZZY LOGIC BASED ANOMALY DETECTION ALGORITHM ON DIFFERENT DATASETS

Datasets	Number of Packets	Classification Rate	False Positives
Data 1	16,860	99.226 %	0.857%
Data 2	11,794	99.276 %	0.840 %
Data 3	21,904	99.327 %	0.727 %
Data 4	18,225	99.321 %	0.809 %
Data 5	34,586	99.385 %	1.372 %
Data 6	113,705	98.277 %	1.772 %
Data 7	113,557	98.339 %	1.804 %
Data 8	65,018	98.438 %	1.606 %
Data 9	69,959	98.521 %	1.519 %
Data 10	118,029	98.259 %	1.791 %
Sum / Average	583,637	98.837 %	1.310 %

B. Improved Uncertainty Handling

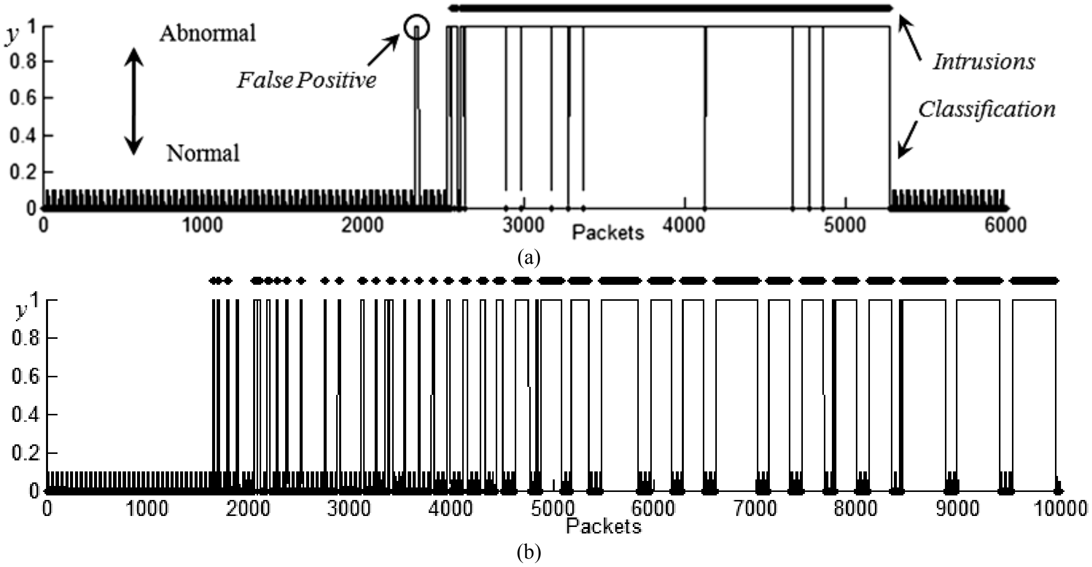
The improved uncertainty handling of the proposed IT2 FLS based anomaly detection algorithm is demonstrated in this Section. As shown in the previous work of the authors [4], [6], common intrusion attempts can be differentiated from the normal behavior using neural networks or T1 FLS. However, in specific cases, such as slightly deviating intrusion attempts or very unusual normal behavior, high amounts of uncertainty might be associated with the output decision. In these cases, the IT2 FLS offers increased state awareness as the uncertainty in the problem domain is modeled by the IT2 FSs and propagated to the final interval classification through the inference mechanism.

As an illustration, three uncertain cases are depicted in Fig. 4. In Fig. 4(a) a classification of slightly deviating intrusion attempts is depicted. It can be seen that for the selected sensitivity threshold the absence of the interval decision might lead to misleading conclusions. On the other hand, the proposed IT2 FLS based algorithm is capable of determining the increased uncertainty of the anomaly identification and thus increase the cyber security state awareness. Fig. 4(b) shows a scenario where unusual normal behavior is experienced due to disconnecting several components of the experimental test bed. Again, the presence of the uncertainty indicator provided by the IT2 FLS can lead to more informed decision making. Finally, a combined case is depicted in Fig. 4(c) where the intrusion attempts were simulated during unusual normal behavior leading to increased uncertainty of the anomaly indicator.

C. Classification Performance

The IT2 FLS based anomaly detection algorithms was applied to the 10 testing datasets with 583,637 packets in total. The algorithm was trained on the 6 training datasets composed of 60,661 normal behavior packets. Using the maximum cluster radius of 0.01 132 fuzzy rules were extracted.

Three performance measures are considered in this work: classification rate, false negative rate and false positive rate. The classification rate is the ration of all correctly classified instances. False positive rate is the rate of false alarms when there was no intrusion attempt. False negative rate is the rate of missing alarms when an intrusion occurred. The classification performance is summarized in Table II. Here, the classification rate and the false positive rates are reported for each dataset and the average values are calculated. The



(a)

(b)

Fig. 5 Anomaly detection performance of the proposed algorithm on segments of packets from datasets 2 (a), and 3 (b). Thin line represents system decision, thick line denotes the known anomalous behavior.

proposed IT2 FLS based anomaly detection system achieved 0% false negative rate in all testing scenarios. This is considered important as false negatives would mean that intrusion attempts were able to break into the system undetected. It can be observed that the algorithm maintained 1.3% average false positive rate, which can be considered a low false positive rate in the area of anomaly detection.

Fig. 5 visually demonstrates the classification of two datasets. The thin line denotes the prediction of the anomaly detection system and the thick line above the system response marks the known occurrence of the anomalous behavior as denoted in Fig 5(a). It can be seen that the proposed anomaly detection system responded well to both long and short intrusion attempts. An example of false positive case is also highlighted in Fig. 5(a).

Comparative analysis between the T1 and the IT2 FLS based approaches and determining specific scenarios when the IT2 FLS based approach provides improved classification performance is currently subject of an ongoing research effort. Up to this date, the major contribution of the presented work is the increased state awareness of the cyber sensor via uncertainty modeling using the IT2 FLS.

VII. Conclusion

This paper presented a novel IT2 FLS based anomaly detection algorithm for embedded network security cyber sensors. The anomaly detection algorithm was specifically designed to allow for both fast learning and fast classification on the constrained computational resources of the embedded device. The algorithm extracts IT2 fuzzy rules using an adapted version of the online nearest neighbor clustering algorithm directly to the stream of packets.

The proposed algorithm was tested on an experimental test-bed mimicking the environment of a critical infrastructure control system with emulated probes of a cyber attacker. The final performance evaluation was performed on a set of 10 test datasets with 583,637 packets with a wide range of anomalous network behavior. The experimental analysis yielded 98.837% correct classification rate with 0.0% false negative rate and 1.31% false positive rates. It was demonstrated the IT2 FLS is capable of improved cyber security state awareness due to improved uncertainty handling.

Acknowledgment

Work supported by the U.S. Department of Energy under DOE Idaho Operations Office Contract DE-AC07-05ID14517, performed as part of the Instrumentation, Control, and Intelligent Systems Distinctive Signature (ICIS) of Idaho National Laboratory.

References

[1] C. G. Rieger, D. I. Gertman, M. A. McQueen, “Resilient Control Systems: Next Generation Design Research,” in *Proc. 2nd IEEE Conf. on Human System Interactions*, pp. 632-636, May 2009.

[2] F. Gonzalez, D. Dasgupta, J. Gomez, M. Kaniganti, “An Evolutionary Approach to Generate Fuzzy Anomaly Signatures,” in *Proc. the IEEE Information Assurance Workshop*, pp. 251-259, June 2003.

[3] J. Gomez, D. Dasgupta, F. Gonzalez, “Detecting Cyber Attacks with Fuzzy Data Mining Techniques,” in *Proc. of the Workshop on Data Mining for Counter Terrorism and Security, 3rd SIAM Conference on Data Mining*, May, 2003.

[4] O. Linda, T. Vollmer, J. Wright, M. Manic, “Fuzzy Logic Based Anomaly Detection for Embedded Network Security Cyber Sensor,” in *Proc. IEEE Symposium Series on Computational Intelligence*, April, 2011.

[5] Z. Zhang, J. Li, C. Manikopulos, J. Jorgenson, J. Ucles, “HIDE: a Hierarchical Network Intrusion Detection System Using Statistical Preprocessing and Neural Network Classification,” in *Proc. IEEE Workshop on Information Assurance and Security*, pp. 85-90, 2001.

[6] O. Linda, T. Vollmer, M. Manic,“Neural Network Based Intrusion Detection System for Critical Infrastructures*,”* in *Proc. Int. Joint INNS-IEEE Conf. on Neural Networks*, pp. 1827-1834, June 14-19, 2009.

[7] W. Hu, Y. Liao, V. R. Vemuri, “Robust Anomaly Detection Using Support Vector Machines,” in *Proc. International Conference on Machine Learning*, 2003.

[8] G. Stein, B. Chen, A. S. Wu, K. A. Hua, “Decision Tree Classifier For Network Intrusion Detection With GA-based Feature Selection,” in *Proc. of the 43rd ACM Southeast Conference*, pp. 136-141, March 2005.

[9] S. Zhong, T. Khoshgoftaar, N. Seliya, “Clustering-based network intrusion detection,” in *Intl. Journal of Reliability, Quality and Safety*, Vol. 14, No. 2, pp. 169-187, 2007.

[10] Q. Wang, V. Mehalooikonomou, “A Clustering Agorithm for Intrusion Detection,” in *SPIE Conference on Data Mining, Intrusion Detection, Information Assurance, and Data Networks Security*, pp. 1083-1086, 2005.

[11] I. H. Witten, E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*, Morgan Kaufmann Publishers, 2005.

[12] D. Yang, A. Usynin, J. W. Hines, “Anomaly-Based Intrusion Detection for SCADA Systems,” in *Proc. of 5th Intl. Topical Meeting on Nuclear Plant Instrumentation, Control and Human Machine Interface Technologies (NPIC&HMIT 05)*, pp. , Nov. 2006.

[13] H. S. Kim, J. M. Lee, T. Park, W. H. Kwon, “Design of networks for distributed digital control systems in nuclear power plants,” *Intl. Topical Meeting on Nuclear Plant Instrumentation, Controls, and Human-Machine Interface Technologies (NPIC&HMIT 2000)*, Nov. 2000.

[14] Dana A. Shea, “Critical Infrastructure: Control Systems and the Terrorist Threat*,” Report for Congress RL31534*, February, 2003.

[15] H. A. Hagras, “A Hierarchical Type-2 Fuzzy Logic Control Architecture for Autonomous Mobile Robots,” in *IEEE Trans. Fuzzy Systems*, vol. 12, no. 4, pp. 524-539, 2004.

[16] J. M. Mendel, *Uncertain Rule-Based Fuzzy Logic Systems: Introduction and New Directions*, Upper Saddle River, NJ: Prentice Hall PTR, 2001.

[17] M. Beglarbegian, W. Melek, J. M. Mendel, “On the robustness of Type-1 and Type-2 fuzzy logic systems in modeling,” in *Information Sciences*, vol. 181, issue: 7, pp. 1325-1347, April 2011.

[18] O. Linda, M. Manic, “Interval Type-2 Fuzzy Voter Design for Fault Tolerant Systems,” in *Information Sciences*, vol. 181, issue: 14-15, pp. 2933-2950, July 2011.

[19] O. Linda, M. Manic, “Evaluating Uncertainty Resiliency of Type-2 Fuzzy Logic Controllers for Parallel Delta Robot,” in Proc. *4th IEEE Conf. on Human System Interactions*, May, 2011.

[20] R. Sommer, V. Paxson, N. Weaver, “An architecture for exploiting multi-core processor to parallelize network intrusion prevention,” *Concurrency Computation: Practice and Experience*, 21, pp. 1255-1279, 2009.

[21] L. A. Zadeh, “The Concept of a Linguistic Variable and its Approximate Reasoning - II,” in *Information Sciences*, No. 8, pp. 301-357, 1975.

[22] J. M. Mendel, R. John, F. Liu, “Interval Type-2 Fuzzy Logic Systems Made Simple,” in *IEEE Trans. on Fuzzy Systems*, vol. 14, no. 6, pp. 808-821, 2006.

[23] Tofino webpage [URL], Available: http://www.tofinosecurity.com, from March 2011.

[24] Allan Bradley PLC 5 Controller webpage, Available: http://www.ab.com/programmablecontrol/plc/, from March 2011.

[25] Nmap webpage [URL], Available: http://nmap.org, from March 2011.

[26] Nessus webpage [URL], Available: http://tenable.com/products/nessus, from March 2011.