The Use of Computational Intelligence for Security in Named Data Networking

by

Amin Karami

Submitted to the Computer Architecture Department (DAC)
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Computer Architecture

at the

UNIVERSITAT POLITEĆNICA DE CATALUNYA BARCELONA TECH (UPC)

January 2015

Department .....
Computer Architecture Department (DAC)
January 2015

Supervisor .....
Dr. Manel Guerrero-Zapata
UPC Campus Nord

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Escola de Doctorat

Acta de calificación de tesis doctoral

Curso académico:

Nombre y apellidos

Programa de doctorado

Unidad estructural responsable del programa

Resolución del Tribunal

Reunido el Tribunal designado a tal efecto, el doctorando / la doctoranda expone el tema de la su tesis doctoral titulada

_____.

Acabada la lectura y después de dar respuesta a las cuestiones formuladas por los miembros titulares del tribunal, éste otorga la calificación:

☐ NO APTO ☐ APROBADO ☐ NOTABLE ☐ SOBRESALIENTE

(Nombre, apellidos y firma) Presidente/a		(Nombre, apellidos y firma) Secretario/a
(Nombre, apellidos y firma) Vocal	(Nombre, apellidos y firma) Vocal	(Nombre, apellidos y firma) Vocal

_____, _____ de _____ de _____

El resultado del escrutinio de los votos emitidos por los miembros titulares del tribunal, efectuado por la Escuela de Doctorado, a instancia de la Comisión de Doctorado de la UPC, otorga la MENCIÓN CUM LAUDE:

☐ SÍ ☐ NO

(Nombre, apellidos y firma) Presidente de la Comisión Permanente de la Escuela de Doctorado	(Nombre, apellidos y firma) Secretario de la Comisión Permanente de la Escuela de Doctorado

_____, _____ de _____ de _____

The Use of Computational Intelligence for Security in Named Data Networking

by

Amin Karami

Submitted to the Computer Architecture Department (DAC)
on January 2015, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Computer Architecture

Abstract

Information-Centric Networking (ICN) has recently been considered as a promising paradigm for the next-generation Internet, shifting from the sender-driven end-to-end communication paradigm to a receiver-driven content retrieval paradigm. In ICN, content -rather than hosts, like in IP-based design- plays the central role in the communications. This change from host-centric to content-centric has several significant advantages such as network load reduction, low dissemination latency, scalability, etc. One of the main design requirements for the ICN architectures -since the beginning of their design- has been strong security.

Named Data Networking (NDN) (also referred to as Content-Centric Networking (CCN) or Data-Centric Networking (DCN)) is one of these architectures that are the focus of an ongoing research effort that aims to become the way Internet will operate in the future. Existing research into security of NDN is at an early stage and many designs are still incomplete. To make NDN a fully working system at Internet scale, there are still many missing pieces to be filled in. In this dissertation, we study the four most important security issues in NDN in order to defense against new forms of -potentially unknown- attacks, ensure privacy, achieve high availability, and block malicious network traffics belonging to attackers or at least limit their effectiveness, i.e., anomaly detection, DoS/DDoS attacks, congestion control, and cache pollution attacks. In order to protect NDN infrastructure, we need flexible, adaptable and robust defense systems which can make intelligent -and real-time- decisions to enable network entities to behave in an adaptive and intelligent manner. In this context, the characteristics of Computational Intelligence (CI) methods such as adaption, fault tolerance, high computational speed and error resilient against noisy information, make them suitable to be applied to the problem of NDN security, which can highlight promising new research directions. Hence, we suggest new hybrid CI-based methods to make NDN a more reliable and viable architecture for the future Internet.

Biography

Amin Karami was born in Tehran, Iran. He received a bachelor's degree in Computer Engineering majoring Hardware from Islamic Azad University of Qazvin, Iran in 2007 and a master's degree in Informatics from University of Skövde, Sweden in 2011. Since February 2012, he pursued his PhD degree in Computer Architecture at Universitat Politécnica de Catalunya Barcelona Tech (UPC), Spain. His research interest lies in the computational intelligence methodologies and approaches, optimization based on evolutionary computation techniques, hybrid intelligent systems, next-generation Internet, named data networking, network security, uncertainty visualization, information fusion, and big data analysis.

Acknowledgments

First and foremost, I would like to express my deepest gratitude to my advisor, Dr. Manel Guerrero-Zapata for his constant encouragement, support, patience, and providing me with an excellent atmosphere for doing research. I would like to thank my colleagues at UPC, Amir Darehshoorzadeh and Tarek AlSkaif for their valuable feedback and guidance, which significantly improved the quality of my research.

Finally, my very special thanks to my beloved parents and family without whose inspiration I may never consider pursuing a Ph.D. degree, and wonderful wife, Niloofar Farahani, without whose support I could never finish this journey.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background of Named Data Networking

Today, several research projects, both in Europe (PSIRP [3], 4WARD [4], PURSUIT [1], SAIL [2], COMET [3] and CONET [5]) and in the US (CCN [6], DONA [7] and NDN [8]) investigate new network architectures based on Information-Centric Networking (ICN) paradigm. These approaches differ with respect to their specific architectural properties, assumptions and objectives. In general, They have been proposed as a solution for a viable and vital replacement for the current IP-based Internet due to the fundamental limitations of the Internet in supporting today's content-oriented services [2, 9, 10, 11]. This change from host-centric to content-centric has several significant advantages such as network load reduction, low dissemination latency, scalability, etc. One of the main design requirements for the ICN architectures -since the beginning of their design- has been strong security [12, 13, 14]. Named Data Networking (NDN) [15] is a prominent example and ongoing research effort of ICN design. The main goal of NDN is to support the dissemination of named content rather than the current host-centric (end-to-end) delivery of content to a named host. Fig. 1-1 shows the fundamental difference between NDN and IP-based Internet. The process of requesting a content (data) by a consumer is also illustrated in Fig. 1-2. Users only value `what` they download and are not interested about `where` content is actually stored. The IP layer is the opposite and only cares about the `where`, not about the `what`. This gap becomes a source of several problems such as security.

In NDN, a consumer asks for a *Content* (*Data*) by sending an *Interest* request using name prefix

[1] http://www.fp7-pursuit.eu/
[2] http://www.sail-project.eu/
[3] http://www.comet-project.org/

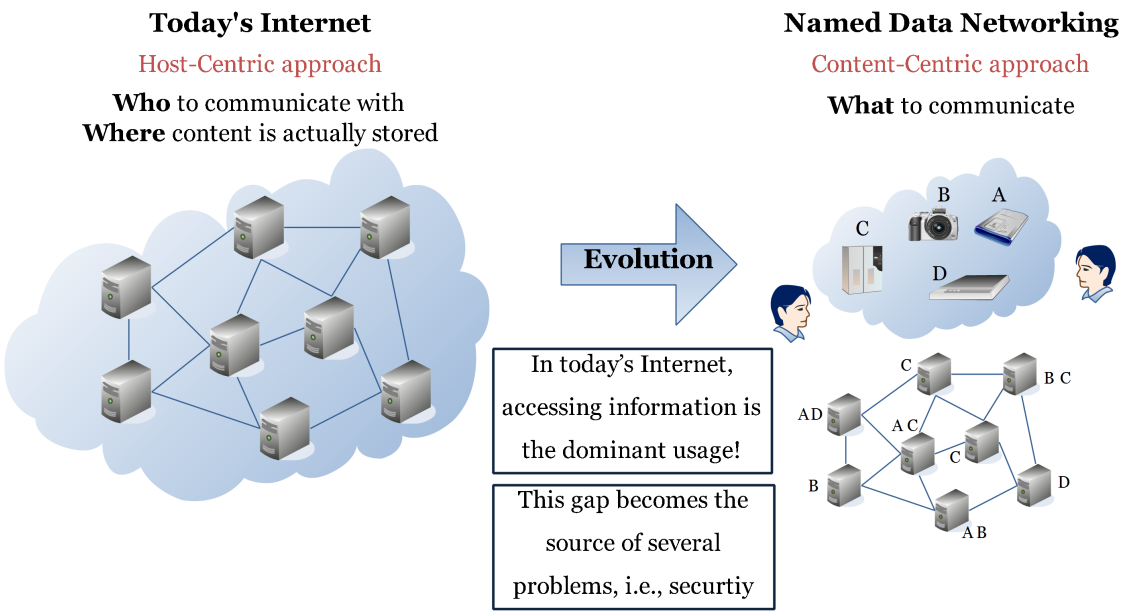
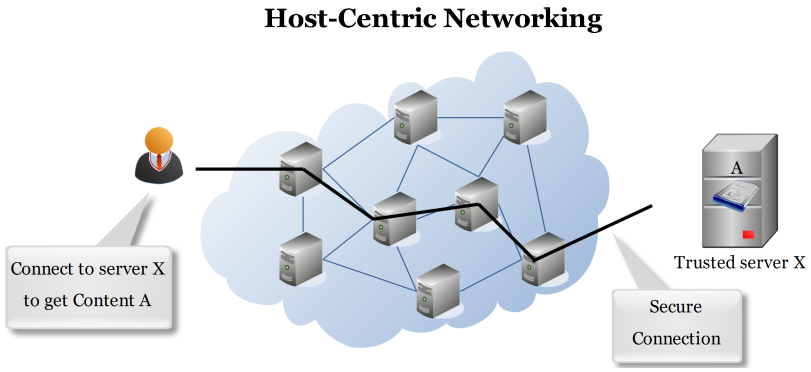
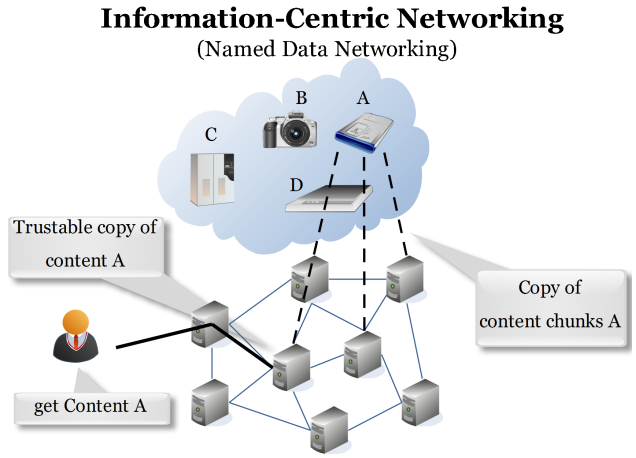


Figure 1-1: NDN (content-centric) vs. IP (host-centric)

(content identifier) instead of today's IP prefix (content location). An Interest packet is routed towards the location of the content's origin where it has been published. Any router (intermediate node) on the way checks its cache for matching copies of the requested content. The requested content is returned by any node that holds a copy of the content in its cache. On the way back, all the intermediate nodes store a copy of the content in their caches to satisfy subsequent users interested in that content (i.e., in-network caching). In this paradigm, storage for caching information is part of the basic network infrastructure so that NDN (and all ICN architectures) enables efficient and application-independent caching structure. NDN improves data availability by integrating caching within the network and ensure the integrity and provenance of content by moving from securing the communication channel to securing the content [6, 16]. All communication in NDN is performed using two distinct types of packets: `Interest` and `Data` (Fig. 1-3). Both types of packets carry a `name`, which uniquely identifies a piece of data that can be carried in one data packet [17, 18]. Data names in NDN are hierarchically structured, e.g., eight fragment of a YouTube video would be named `/youtube/videos/A45tR7Kg5/8`. In addition to the data name, each Interest packet also carries a random `nonce` generated by the consumer. A router checks both the name and nonce of each received Interest packet. If a newly arrived Interest packet carrying the same name as a previously received Interest packet from a different consumer, or a previously forwarded Interest



(a) Host-Centric Networking



(b) Information-Centric Networking

Figure 1-2: Host-Centric Networking vs. Information-Centric Networking

looped back, the router drops the Interest packet. Therefore Interest packets cannot loop. Each

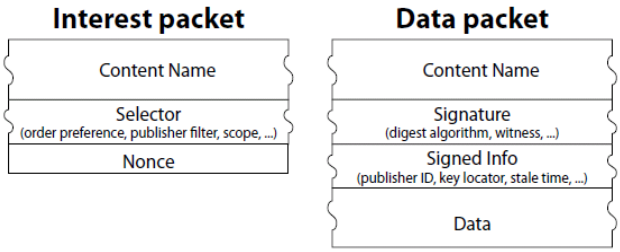


Figure 1-3: CCN packet types

NDN router maintains three major data structures [19]:

1. The Pending Interest Table (PIT) holds all not yet satisfied Interest packets that were sent upstream towards potential data sources. Each PIT entry holds one or multiple incoming physical interfaces and their corresponding Interest packets.

2. The Forwarding Information Base (FIB) forwards Interest packets to one or multiple physical network interfaces based on the forwarding strategies. The strategy module makes forwarding decisions for each Interest packet.

3. The Content Store (CS) or buffer memory temporarily buffers data packets for data retrieval efficiency.

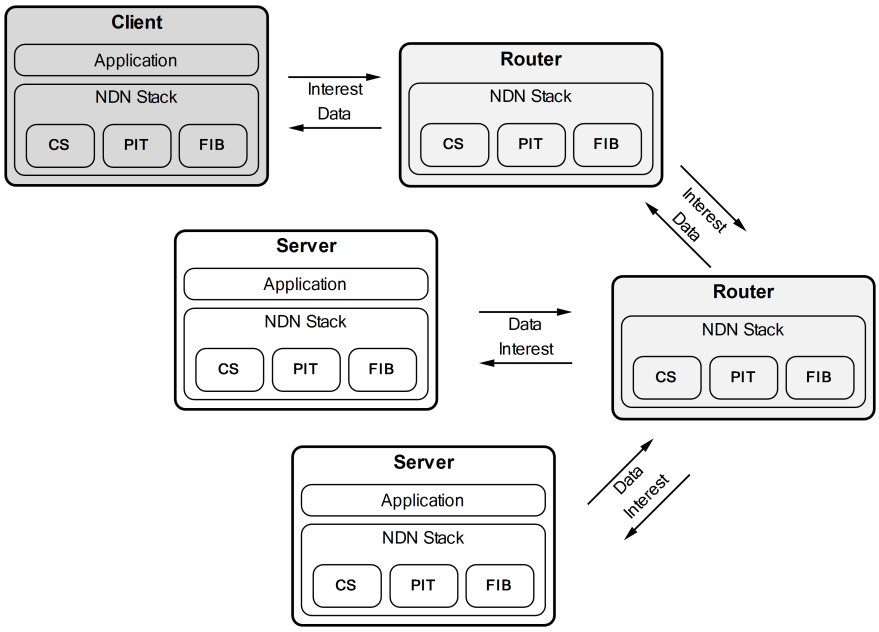


Figure 1-4: Overview of the NDN architecture

Fig. 1-4 gives an overview of the NDN architecture. According to Fig. 1-4, when a NDN router receives an Interest packet, it first checks its CS (cache). If there is no copy of the requested content, it looks up its PIT table. If the same name is already in the PIT and the arrival interface of the present Interest is already in the set of arrival interface of the corresponding PIT entry, the Interest is discarded. If a PIT entry for the same name exists, the router updates the PIT entry by adding a new arrival interface to the set. The Interest is not forwarded further. Otherwise, the router creates a new PIT entry and forwards the present Interest using its FIB. When an Interest packet is satisfied by the content's origin where it was published, on the way back, all the intermediate nodes store a copy of content in their caches to answer to probable same Interest requests from subsequent requester [12, 20].

Security in the NDN needs to be implemented differently than in current, host-centric networks. In the current Internet most security mechanisms are based on host authentication and then trusting

the data that the host delivers. In contrast, in the NDN, security mechanisms must be relied upon Information Object (IO) itself, not based on the network nodes [6, 7]. Consequently, new *information-centric security* concepts are required that base security on the information itself [21]. With this new paradigm new kinds of attacks and security challenges -from Denial of Service (DoS) to (user and cache) privacy attacks- will arise that we must provide an efficient and effective security mechanism in NDN. Exiting research into the security in NDN is at an early stage and many designs are still incomplete. To make NDN a practically working system at Internet scale, there are still many pieces to be filled in. This new Internet architecture should be resilient against existing security problems to defense against new forms of -potentially unknown- attacks, achieve high availability of content, ensure data and cache privacy, and block malicious network traffics belonging to attackers or at least limit their effectiveness. To do so, we picked four central topics which take NDN toward a more reliable and viable architecture including anomaly detection, mitigating DoS/DDoS attacks, congestion control/avoidance and mitigating cache pollution attacks. The research problems and our countermeasures are presented in the next sections.

1.2 Research Problem

The following features are the most security-related challenges which NDN should be resilient against them or at least limit their effectiveness with the current systems:

- **Anomaly Detection:** Attacks and anomalies are deliberate actions against data, contents, software or hardware that can destroy, degrade, disrupt or deny access to a computer network [22]. Hence, the contents should be resilient against both DoS and new forms of -potentially unknown- attacks or at least limit their effectiveness [23]. In order to disarm new kinds of attacks, anomalous traffics, and any deviation, not only the detection of the malevolent behavior must be achieved, but the network traffic belonging to the attackers should be also blocked [24, 25, 26]. In this dissertation, we present an intelligent method for anomaly detection in NDN which is described in detail in Chapter 3 and published on **NEUROCOMPUTING, Elsevier, [JCR-2013: 2.005 Q1] (28/121 Q1 COMPUTER SCIENCE, ARTIFICIAL INTELLIGENCE)** [12]:

  A. Karami and M. Guerrero-Zapata (2015), *A Fuzzy Anomaly Detection System based on Hybrid PSO-Kmeans in Content-Centric Networks*, Neurocomputing, 149:Part C, pp. 1253-1269.

And an upcoming book chapter in:
Advanced Research on Hybrid Intelligent Techniques and Applications. IGI Global, Hershey, Pennsylvania (USA), undergoing editing for 2015. [27]

- **Cache Pollution Attacks:** The ubiquitous in-network caching is a key NDN feature as it reduces overall latency and improves bandwidth utilization for popular content [9, 20, 28, 29, 30]. However, pervasive caching strengthens the security problem of cache pollution attacks in two generic classes: locality-disruption and false-locality [31, 32, 33, 34]. Locality-disruption attacks continuously generate requests for new unpopular files to force routers (i.e., the victims of the attack) to cache unpopular content, thus degrading cache efficiency by ruining the cache file locality. False-locality attacks repeatedly request the same set of unpopular (i.e., fake popular) files, thus degrading the hit ratio by creating a false file locality at cache. In this dissertation, a new cache replacement method in NDN is developed to detect and mitigate these two types of cache pollution attacks in a timely manner, which is described in detail in Chapter 4 and undergoing second revision on 27th Sept. 2014 in **COMNET, Elsevier, [JCR-2013: 1.282 Q2] (17/50 Q2 COMPUTER SCIENCE, HARDWARE & ARCHITECTURE)** [35]:
  A. Karami, M. Guerrero-Zapata (2014), *An ANFIS-based cache replacement method for mitigating cache pollution attacks in Named Data Networking* In COMNET, Elsevier, undergoing 2nd revision on 27th Sept. 2014.

- **DoS/DDoS Attacks:** In contrast to today's Internet, a key goal of the NDN project is "security by design" [17, 32, 36]. Unlike the current Internet (host-based) approach in which security, integrity and trust should be provided in the communication channel, CCN secures content (information) itself and puts integrity and trust as the content properties [19, 37]. However, with this new paradigm, new kinds of attacks and anomalies -from Denial of Service (DoS) to privacy attacks- will arise [38, 39]. The big question is how resilient will this new NDN architecture be against DoS/DDoS attacks [17, 23]. An adversary can take advantage of two features unique to NDN namely Content Store (CS) and Pending Interest Table (PIT) to mount DoS/DDoS attacks specific to NDN such as Interest flooding attacks and content poisoning [23, 40]. In this dissertation, we present a new intelligent hybrid algorithm for proactive detection of DoS attacks and adaptive mitigation reaction in NDN, which is described in detail in Chapter 5 and published on **NEUROCOMPUTING, Elsevier, [JCR-2013:**

2.005 Q1] (28/121 Q1 COMPUTER SCIENCE, ARTIFICIAL INTELLIGENCE) [41]:

A. Karami, M. Guerrero-Zapata (2015), *A Hybrid Multiobjective RBF-PSO Method for Mitigating DoS Attacks in Named Data Networking*, Neurocomputing, 151:Part 3, pp. 1262-1282.

And an upcoming book chapter in:

Advanced Research on Hybrid Intelligent Techniques and Applications. IGI Global, Hershey, Pennsylvania (USA), undergoing editing for 2015 [42].

- **Congestion:** Congestion takes place in NDN routers when the number of arrival data packets is higher than the queue's capacity which causes an overflow in the routers' buffer [43, 44]. When this happens a high data packet loss and increase in the end-to-end delay occur affecting negatively on the performance, stability and robustness of the network [45, 46]. This leads to under-utilization of the available resources and degradation of throughput and quality of service [47, 48]. This research work develops an Adaptive Congestion Control Protocol in NDN (ACCPndn) by learning capacities to control congestion traffics before they start impacting the network performance, which is described in Chapter 6 and submitted to **JNCA, Elsevier, [JCR-2013: 1.772 Q1] (28/121 Q1 COMPUTER SCIENCE, ARTIFICIAL INTELLIGENCE)** [49]:

  A. Karami, M. Guerrero-Zapata (2014), *ACCPndn: Adaptive Congestion Control Protocol in Named Data Networking by learning capacities* In JNCA, Elsevier, submitted on 10th May 2014.

  And later, another idea talked and presented at Xerox Co.:

  M. H. Ardestani, **A. Karami**, P. Sarolahti and J. Ott (2013), *Congestion Control in Content-Centric Networking using Neural Network*, Talk and Presentation in CCNxCon 2013, 5-6th September 2013, Parc (Xerox Co.), California, USA [50].

In order to protect NDN infrastructure, we need flexible, adaptable and robust defense systems or protocols, which can make intelligent decisions (ideally, in real-time) in detecting wide variety of threats and unknown attacks, achieving high availability of content, ensuring data and cache privacy, and blocking malicious network traffics belonging to attackers or at least limit their effectiveness. Computational Intelligent (CI) techniques seem promising to enhance NDN security measures, and have been increasingly applied in the area of information security and information assurance. CI is a well-established paradigm, where new theories and concepts with a sound bi-

ological understanding have been evolving. The most current computer systems and algorithms have many characteristics of biological computations that are difficult or impossible to do with conventional computations.
Computational Intelligence (CI) techniques and methodologies [51] known for their ability to adopt and to exhibit fault tolerance, high computational speed and error resilient against noisy information, compensate for the limitations of security challenges in NDN. Thus, the methods of Computational Intelligence (CI) provide a practical alternative for solving mathematically intractable and complex problems [52].

1.3 Contribution

As we observed, the demand for secure NDN to defense against new forms of -potentially unknown- attacks, achieve high availability, protect privacy, and block adversaries' traffics becomes an important issue that must be carefully considered. In this dissertation, we propose novel solutions for security in NDN that are based on the application of Computational Intelligence (CI) methodologies and approaches in order to face with large-scale, unknown DoS/DDoS and (user and cache) privacy attacks, and network congestion without becoming a single point of failure or initiating latency to the network flows to enable network entities to behave in an adaptive and intelligent manner in a dynamic environment. This approach can also be an innovative solution because NDN can adopt itself to unpredictable complex evolution of network environments and security problems.

The aim of this dissertation is to propose some solutions to explore how the core methods of Computational Intelligence (CI), which encompass artificial neural networks, fuzzy sets, evolutionary computation methods, swarm intelligence, soft computing, and other similar computational models can be employed in the NDN context to solve some of the security-related challenges -from DoS attacks to privacy attacks- with the current systems.

The contributions of this dissertation can be summarized as follows:

1. We proposed a novel fuzzy anomaly detection system based on the hybridization of PSO and K-means clustering algorithms. This system consists of two phases: the training phase with two simultaneous cost functions as well-separated clusters by DBI and local optimization by MSE, and the detection phase with two combination-based distance approaches as classification and outlier. Experimental results and analysis show the proposed method in the training phase is very effective in determining the optimal number of clusters, and has a very high

detection rate and a very low false positive rate at the same time. In the detection phase, the proposed method clearly outperforms other applied method in terms of AUC (area under the ROC curve), accuracy, sensitivity and specificity. In addition, the times of increment on computational time of proposed method is relative smaller than the other considered methods. Please refer to Chapter 3 for our detailed contribution.

2. We proposed a novel ANFIS-based cache replacement method to mitigate two generic cache pollution attacks namely false-locality and locality-disruption in NDN. Simulation results showed that the proposed method provides very accurate results as compared to LRU and LFU algorithms independently and in conjunction with CacheShield scheme. Experimental results and analysis show the proposed ANFIS-based cache replacement method is very effective in determining and mitigating the fake content, and has a very high detection rate of locality-disruption attacks to replace them when new content is added to a full cache in a timely manner. The extensive analysis satisfies the objectives of the experiment and ensure that the proposed ANFIS-based caching for mitigating cache pollution attacks can yield high accuracy as compared to other methods without very much computational cost. Please refer to Chapter 4 for our proposed countermeasure.

3. NDN is a newly proposed future Internet architecture which it is important to address its resilience in face of DoS/DDoS attacks. We examined the most current instances of DoS/DDoS attacks to show that an adversary with limited resources can serve service degradation for legitimate users. We then introduced our intelligent hybrid algorithm for proactive detection of DoS attacks and adaptive reaction for mitigating. In the detection phase, a combination of multiobjective evolutionary optimization and RBF neural network has been applied. This approach consists of two phases: training/optimization and prediction/classification. In the training phase, we investigate the implementation of a multiobjective approach and PSO in the design of RBF neural network in order to improve the accuracy of classification problems. We apply NSGA II to determine the Pareto solutions of RBF units' centers in terms of the well-separated centers through DBI and their local optimization through MSE. Then, the optimization and tuning of the units' widths and output weights are accomplished by using the PSO, where the each particle encodes a set of widths and weights. Moreover, the structure of this step is simple and easy to implement, yet very effective in terms of several performance criteria. In the prediction phase, we employ a simple algorithm to classify efficiency

the new input patterns with the minimum misclassification error. This hybrid algorithm was applied on four benchmarking data sets to verify the algorithm accuracy and robustness in classification problems. Subsequently, after constructing a more accurate classifier (detector), we performed a simple adaptive reaction algorithm by enforcing explicit limitations against adversaries which was very effective and efficient for shutting down the attackers with the robust recovery from network failures and accuracy more than 90% in terms of the average of Interest satisfaction ratio for legitimate users, the PIT usage, the number of received contents (throughput), and a very low false positive rate over 10 simulation runs. Please refer to Chapter 5 for our proposed countermeasure against DoS/DDoS attacks in NDN.

4. We developed an Adaptive Congestion Control Protocol in Named Data Networking (ACCPndn) by learning capacities that works in two phases. The first phase -adaptive training- forecasts the source of congestion together with the amount of congestion in NDN routers with a Timed-Lagged Feedforward Network (TLFN) optimized by hybridization of PSO and GA. The second phase -fuzzy avoidance- employs a non-linear fuzzy logic-based control system based on the outcomes of first phase, which it makes a proactive decision in each router per interface to control and/or prevent packet drop well enough in advance. Extensive simulations and experimental results show that ACCPndn sufficiently satisfies the performance metrics and outperforms two previous proposals such as NACK and HoBHIS in terms of the minimal packet drop and high-utilization (retrying alternative paths) in bottleneck links to mitigate congestion traffics. In addition, it is found to be scalable with respect to varying bandwidths, delays, packet generation, and replacement policies in cache and PIT table. Please refer to Chapter 6 for our proposed countermeasure against congestion in NDN.

5. Uncertainty is widely spread in real-world data. Uncertain data -in computer science- is typically found in the area of sensor networks where the sensors sense the environment with certain error. Traffic uncertainty refers to traffic volumes belong to more than one pattern (i.e. both normal and attack), and associated with each pattern by a set of membership levels. We investigate the implementation of fuzzy set theory through the application of Fuzzy C-means clustering algorithm in the context of SOM neural network in order to improve the accuracy of visualizing uncertain data bases. The experimental results over the uncertain traffics in Named Data Networking show that the proposed method is effective and precise in terms of the applied performance criteria. We suggest this research work to help to foster discussions

and new research ideas among our readers. We plan to improve the proposed method for various uncertain models and big uncertain network traffic data in the future. Please refer to Chapter 7 for our proposed method.

1.4 Thesis Plan

As can been seen from the above-mentioned sections, NDN has a very clear and a promising network architecture being considered as a possible replacement for the current IP-based (host-centric) Internet infrastructure in order to overcome the fundamental limitations of the current Internet. However, strong security has been one of the main design requirements for this architecture. Fig. 1-5 depicts how our work in this dissertation fits into the global picture.

Chapter 3 looks at the anomaly detection through a fuzzy anomaly detection system based on hybrid PSO-Kmeans algorithm. Experimental results in this chapter demonstrate that the proposed algorithm can achieve to the optimal number of clusters, well-separated clusters, as well as increase the high detection rate and decrease the false positive rate at the same time when compared to some other well-known clustering algorithms.

Chapter 4 then looks into the cache pollution attacks and its countermeasures. In this research work, a new cache replacement method based on Adaptive Neuro-Fuzzy Inference System (ANFIS) is presented to mitigate the cache pollution attacks in NDN. The ANFIS structure is built using the input data related to the inherent characteristics of the cached content and the output related to the content type (i.e., healthy, locality-disruption, and false-locality). The proposed method detects both false-locality and locality-disruption attacks as well as a combination of the two on different topologies with high accuracy, and mitigates them efficiently without very much computational cost as compared to the most common policies.

Chapter 5 looks into the countermeasures against DoS/DDoS attacks in NDN. We present a new intelligent hybrid algorithm for proactive detection of DoS attacks and adaptive mitigation reaction in NDN. The evaluation through simulations in this chapter shows that the proposed intelligent hybrid algorithm (proactive detection and adaptive reaction) can quickly and effectively respond and mitigate DoS attacks in adverse conditions in terms of the applied performance criteria.

Chapter 6 tackles the congestion problem. NDN is subject to congestion when the number of data packets that reach one or various routers in a certain period of time is so high than its queue gets overflowed. To address this problem many congestion control protocols have been

proposed in literature which, however, they are too high sensitive to their control parameters as well as unable to predict congestion traffic well enough in advance. This research work develops an Adaptive Congestion Control Protocol in NDN (ACCPndn) by learning capacities to control congestion traffics before they start impacting the network performance. Extensive simulations and experimental results show that ACCPndn sufficiently satisfies the applied performance metrics and outperforms some previous proposals in terms of the minimal packet drop and high-utilization (retrying alternative paths) in bottleneck links to mitigate congestion traffics.

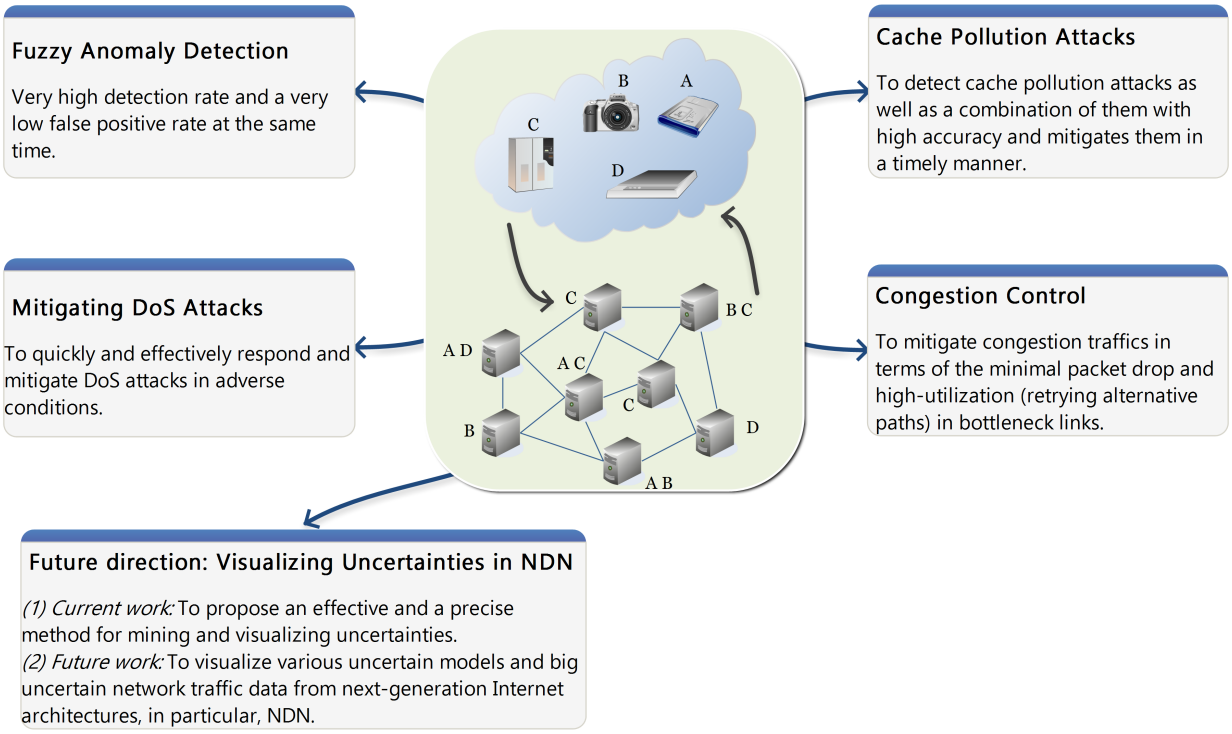


Figure 1-5: Architectural overview: how our work fits into the global picture

Finally, Chapter 7 provides a new research direction for our readers towards the visualization of uncertain network traffic data in NDN. Uncertainty is widely spread in real-world data. Uncertain data -in computer science- is typically found in the area of sensor networks where the sensors sense the environment with certain error. Mining and visualizing uncertain data is one of the new challenges that face uncertain databases. We present a new intelligent hybrid algorithm that applies fuzzy set theory into the context of the Self-Organizing Map to mine and visualize uncertain objects. The algorithm is tested in some benchmark problems and the uncertain traffics in NDN.

Experimental results indicate that the proposed algorithm is precise and effective in terms of the applied performance criteria. This is our initial idea as a future work to help to foster discussions and new research ideas among our readers by some improvements such as visualizing big uncertain network traffic data in NDN.

Obviously, there can be still other missing pieces in this picture that are not addressed in this dissertation. However, we believe that the five topics we picked are central to take NDN toward a practically working system.

A final note on the terminologies used in this dissertation: we may use NDN and CCN terms interchangeably. Interests are sometimes also referred to as requests while data packets may be called content or responses.

1.4.1 Dissertation Roadmap

The rest of the dissertation is organized as follows.
Chapter 2 presents an overview of Information-Centric Networking (ICN) architecture. Among the several ICN architectures, we picked NDN as the basis of our architecture due to its popularity and existence of open source prototype and a ns-3 based simulator as ndnSIM [53]. We propose the NDN naming and security in Section 2.2. We also describe the core methods of Computational Intelligence (CI) in Section 2.3.

We then move on to discuss papers related to the four picked specific topics we studied in this dissertation, namely anomaly detection (Chapter 3), mitigating cache pollution attacks (Chapter 4), mitigating DoS/DDoS attacks (Chapter 5), and congestion control (Chapter 6). Chapter 7 provides a new research direction into the visualization of uncertain network traffic data in NDN.

Finally, Chapter 8 summarizes the entire dissertation's contributions and pointed out a few possible directions for further research.

Chapter 2

Background

This section describes common ICN architectures and follows by naming and security issues in ICN. At last, the Computational Intelligence (CI) methodologies and approaches are described.

2.1 ICN Architectures

People exploit the Internet to get contents such as web pages, music or video files. These users only value `what` they download and are not interested about `where` content is actually stored. The IP layer is the opposite and only cares about the `where`, not about the `what`. The gap between the actual usage of the Internet and the services offered by IP becomes the source of several problems, i.e., usability, performance, security, and mobility. Therefore, new approach is necessary to directly provide users with contents at the network layer, instead of providing communication channels between hosts. This paradigm is shifting from IP to chunks of named content.

ICN is a promising network architecture being considered as a possible replacement for the current IP-based Internet infrastructure. In ICN with a top-down approach, content -rather than hosts, like in IP-based design- plays the central role in the communications. In this section, we provide a survey of the most prominent ICN architectures in the literature. Please refer to [21, 30, 54] for more complete surveys of ICN.

2.1.1 CCN/NDN

Jacobson et al. [6] proposed the Content-Centric Networking (CCN) architecture in PARC which puts named contents at the thin waist of the protocol stack. The main idea in the CCN is that, an Interest request for a content object is routed towards the location of the content's origin where it

has been published. Any router or intermediate node on the way checks its cache for matching copies of the requested content. If a cached copy of any piece of Interest request is found, it is returned to the requester along the path the request came from. On the way back, all the intermediate nodes store a copy of content in their caches to answer to probable same Interest requests from subsequent requester (see Fig. 2-1).

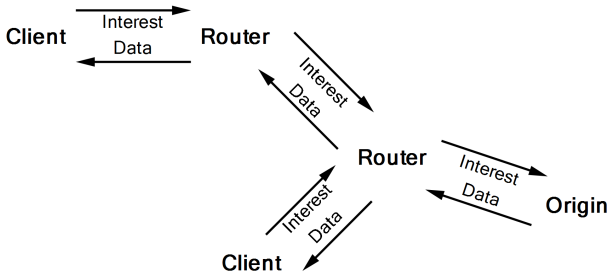


Figure 2-1: CCN/NDN architecture overview

CCN is widely recently applied in the research community and become one of the Future Internet Architecture (FIA) projects funded by National Science Foundation (NSF) under the new name of Named Data Networking (NDN) [8]. In NDN, a strategy layer mediates between the named data layer and the underlying network technologies to optimize resource usage, e.g., to select a link in a multi-homed node, while a security layer applies security functionality directly on named data. We have already employed this architecture in detail in Chapter 1 since it is the basis of our ICN architecture.

2.1.2 PSIRP/PURSUIT

Publish-Subscriber Internet Routing Paradigm (PSIRP), later continued by the Publish-Subscribe Internet Technology (PURSUIT) is a ICN project supported by European Union FP7 (The Seventh Framework Program) [55]. The PURSUIT architecture consists of three separate functions: rendezvous, topology management and forwarding. When the rendezvous function matches a subscription to a publication, it directs the topology management function to create a route between the publisher and the subscriber. This route is finally used by the forwarding function to perform the actual transfer of data (see Fig. 2-2).

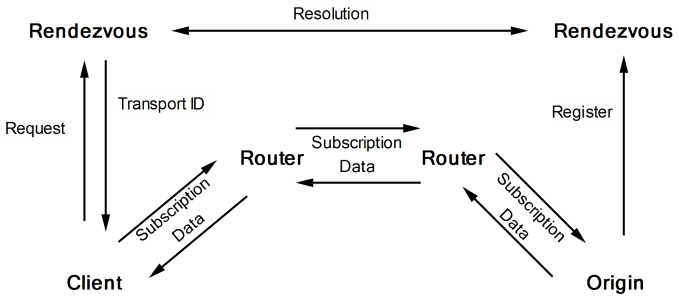


Figure 2-2: PSIRP/PURSUIT architecture overview

2.1.3 4WARD/SAIL

The Architecture and design for the future Internet (4WARD) [1] project and its continuation Scalable and Adaptive Internet Solutions (SAIL) [2], both funded by the EU Framework 7 Programme, are investigating designs for the Future Internet. In 4WARD/SAIL Information Objects (IOs) are published into the network, registered with a Name Resolution Service (NRS). The NRS also is used to register network locators that can be used to retrieve data objects that represents the published IOs. When a receiver want to retrieve an IO, the request for the IO is resolved by the NRS into a set of locators. These locators are then used to retrieve a copy of the data object from the best available source(s) (see Fig. 2-3).

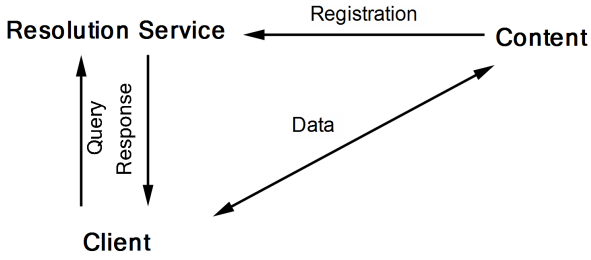


Figure 2-3: 4WARD/SAIL architecture overview

2.1.4 DONA

the Data Oriented Network Architecture (DONA) [7] from UC Berkeley is one of the first complete ICN architectures, as it radically changes naming by replacing the hierarchical URLs with flat names. In DONA, each data is given a self-certifying flat name and name resolution and data caching by Data handlers (DHs). DHs are responsible for routing clients' requests to nearby copies

[1] Available: http://www.4ward-project.eu/
[2] Available: http://www.sail-project.eu/

of the data. DONA allows a client to request a piece of data by its name, rather than the producer's address. The architecture contains two fundamental operational primitives, namely: FIND -allows a client to request a particular piece of data by its name not its location; and REGISTER -allows content providers to indicate their intent to serve a particular data object.

2.2 Naming and Security

Naming plays an important role in the ICN architectures. In today's Internet architecture, the storage locations of information are mainly named, e.g., Uniform Resource Locators (URLs) relating to a network node and file structure to name files, and Internet Protocol (IP) addresses to name the interfaces of the storage nodes. In contrast in ICN we name the information itself, i.e., naming Information Objects (IOs) via location-independent object identifier. Naming is closely related to security in several ICN architectures. In current Internet, security is mainly based on trusting the source of information via authentication and securing the data channel via encryption. In the ICN, security cannot be applied to the storage location as the network and users should benefit from any available copy of the desired content. Consequently, new information-centric security concepts are required that base security on the information itself. A popular approach followed by several ICN architectures is to integrate security aspects with the naming concept (the object IDs). The five general technical security goals are defined in the ICN context as follows:

- Confidentiality: only valid/legal entities (users or systems) can read secured information (IOs and corresponding metadata).
- Data integrity: it applies by self-certification techniques to identify accidental or intentional changes to IOs and the corresponding metadata.
- Accountability: the owner of information can be authenticated and identified by:
  - Owner authentication: Binds the information securely to a virtual entity, e.g., by a pseudonym or a public/private key pair.
  - Owner identification: Binds the information securely to a real-world entity, e.g., a person's unique identity or an institution.
- Availability: The IOs and corresponding metadata published in the network must be available and accessible for (authorized) entities. The in-network caching widely plays an important role in availability.

- Access Control: Access (i.e., read, write, execute) to IOs and/or corresponding metadata can be restricted to the authorized entities.

2.3 Computational Intelligence (CI) methods

Computational Intelligence (CI) is a fairly new research field with competing definitions [51]. How is it related to others branches of computer science such as artificial intelligence (AI), data mining, intelligent agents (systems), knowledge discovery in data (KDD), machine learning (intelligence), natural computing, parallel distributed processing, pattern recognition, probabilistic methods, graphical methods, soft computing, multivariate statistics, optimization and operation research? This is very confusing issue that means different definition with different people. For example, a definition of CI is [56]: CI is the study of the design of Intelligent Agents (IA) and IA is a system that acts intelligently based on its circumstances and its goal. It learns from experience, and it makes appropriate choices given perceptual limitations and finite computation.

Another definition is from Prof. Bezdek, the father of fuzzy pattern recognition theory which defined a computationally intelligent system as follows [57]: *A system is called* ***computationally intelligent*** *if it deals only with numerical (low-level) data, has a pattern recognition component, and does not use knowledge in the AI sense; and additionally, when it (begins to) exhibit (i) computational adaptivity; (ii) computational fault tolerance; (iii) speed approaching human-like turnaround, and (iv) error rates that approximate human performance.*

It states that a computationally intelligence system should be characterized with the capability of computational adaption, fault tolerance, high computational speed and less error-prone to noisy information sources. Computational adaption refers to the system should be capable of changing its parameters following some guidelines (e.g., optimizing criteria) and depending on the temporal changes in its input and output instances. Most of the ANN models and Evolutionary Algorithm (EA) satisfy this characteristic. Computational fault tolerance is a general characteristic of a parallel and distributed system. In a parallel or distributed system, computational resources like variables, procedures and software tools are usually replicated at the distributed units of the computing system. Hence, a damage of a few units usually does not cause malfunctioning of the entire system, parameters the same resources are available at other units. ANN and fuzzy logic have their inherent characteristics of fault tolerance, GA and belief networks too can be configured in a parallel fashion to provide users the benefit of fault tolerance. Moreover, based on the Bezdek's

definition a computationally intelligent system should deal with low level (numerical) information and should not use knowledge in the AI sense. Since fuzzy logic, ANN, GA, and belief networks satisfy these characteristics [58, 59].

CI is different from the well-known field of Artificial Intelligence (AI). Artificial Intelligence (AI) aims to emulate human intelligence in forms of machines so that enable act and think like the human beings. This intelligence is the computational part of the ability to achieve goals in the world. The AI Depot[3] in 2001 uses the following definition: "Artificial Intelligence is a branch of science which deals with helping machines find solutions to complex problems in a more human-like fashion. This usually involves borrowing characteristics from human intelligence, and applying them as algorithms in a computer friendly way."

AI (a high-level cognitive function) handles symbolic knowledge representation, while CI (a low-level cognitive function) handles numeric representation of information [60]. Although there is not yet full agreement on what computational intelligence exactly is, there is a widely accepted view on the areas belonging to CI [51]: artificial neural networks, fuzzy sets, evolutionary computation, swarm intelligence, and soft computing. In continue, we briefly review some core methods of CI that have been applied to solve current security-related challenges in NDN.

2.3.1 K-means algorithm

The K-means algorithm [61] groups the set of data points into a predefined number of the clusters in terms of a distance function. The most widely used method is Euclidean distance in which a small distance implies a strong similarity whereas a large distance implies a low similarity. The Eq. (2.1) shows the Euclidean distance calculation between two data points (x and y) with N objects in a n-dimensional space.

$$Distance(x, y) = \sqrt{\sum_{i=1}^{N}(x_i - y_i)^2} \tag{2.1}$$

The standard K-means algorithm is summarized as follows:

1 Randomly initialize the K cluster centroids.

2 Assign each object to the group with the closest centroid. Euclidean distance measures the minimum distance between data objects and each cluster centroid.

[3] http://ai-depot.com

3 Recalculate the cluster centroid vector, using

$$m_j = \frac{1}{n_j} \sum_{\forall data_p \in C_j} data_p \tag{2.2}$$

where, m_j denotes the centroid vector of the cluster j, n_j is the number of the data vectors in cluster j, C_j is the subset of the data vectors from cluster j, and $data_p$ denotes the pth data vector.

4 Repeat step 2 until the centroids do not change any more in the predefined number of iteration or a maximum number of iterations has been reached.

2.3.2 Genetic Algorithm (GA)

Genetic Algorithm (GA) is a search heuristic and stochastic optimization technique based on biological evolution theory and genetic principles developed by Holland on 1975. GA adopts a group of simulated encoded chromosomes and calculates the fitness function of these chromosomes. GA applies three kinds of genetic operators: selection, crossover and mutation to produce next generation. This evolution process continues until the stopping criteria are met. The selection operator chooses chromosomes from a population for later breeding (recombination or crossover). The crossover operator combines (mates) two chromosomes (parents) to produce a new chromosome (offspring). The idea behind crossover is that the new chromosomes might be better than both of the parents if it takes the best characteristics from each of the parents. The mutation operator alters one or more gene values in a chromosome from its initial state. This can result in entirely new gene values being added to the gene pool. With these new gene values, the genetic algorithm might be able to arrive at better solution than was previously possible. Mutation helps the genetic algorithm to avoid being trapped in a local optimal. GA is appropriate for large-sized and nonlinear space problems which solution is unpredictable [62]. One of the main advantages of the use of GA is that it is less likely to fall into a certain local minimum or maximum [63, 64].

2.3.3 Particle Swarm Optimization (PSO)

The PSO was firstly introduced by Kennedy and Eberhart in 1995 [65]. It was inspired by the social behavior of a bird flock or fish school. It is a population based meta-heuristic method that optimizes a problem by initializing a flock of birds randomly over the search space where each

bird is referred as a "particle" and the population of particles is called "swarm". The particles move iteratively around in the search space according to a simple mathematical formula over the particle's position and velocity to find the global best position. In the n-dimensional search space, the position and the velocity of i th particle at t th iteration of algorithm is denoted by vector $X_i(t) = (x_{i1}(t), x_{i2}(t), ..., x_{in}(t))$ and vector $V_i(t) = (v_{i1}(t), v_{i2}(t), ..., v_{in}(t))$, respectively. This solution is evaluated by a cost function for each particle at each stage of algorithm to provides a quantitative value of the solution's utility. Afterwards, a record of the best position of each particle based on the cost value is saved. The best previously visited position of the particle i at current stage is denoted by vector $P_i = (p_{i1}, p_{i2}, ..., p_{in})$ as the personal bests. During this process, the position of all the particles that gives the best cost until the current stage is also recorded as the global best position denoted by $G = (g_1, g_2, ..., g_n)$. The structure of the velocity and the position updates is depicted in Fig. 2-4. Each iteration is composed of three movements: in the first movement, particle moves slightly toward the front in the previous direction with the same speed. In the second movement, it moves slightly toward the previous itself best position. Finally, in the third movement, moves slightly toward the global position.

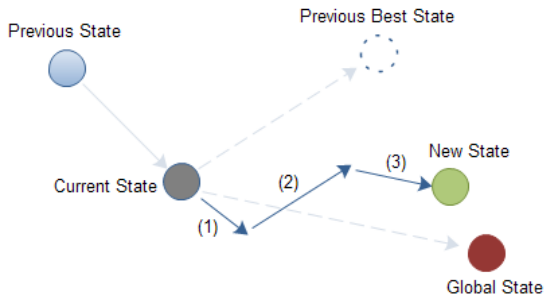


Figure 2-4: Description of velocity and position updates in PSO for a 2-dimensional parameter space

At each iteration, the velocity and the position of each particle are defined according to Eqs. (2.3) and (2.4), respectively:

$$V_i(t) = \omega * V_i(t-1) + c_1\varphi_1(P_i - X_i(t-1)) + c_2\varphi_2(G - X_i(t-1)) \tag{2.3}$$

$$X_i(t) = X_i(t-1) + V_i(t) \tag{2.4}$$

Where, ω denotes the nonzero inertia weight factor that introduces a preference for the particle to

continue moving in the same direction. Decreasing the inertia over time introduces a shift from the exploratory (global search) to the exploitative (local search) mode [66, 67]. Generally, the inertia weight ω is reduced linearly. There are several selection strategies of inertia weight ω which have been described in [68, 69]. c_1 and c_2 are positive constant (social) parameters called acceleration coefficients which control the maximum step size between successive iterations. φ_1 and φ_2 are two independently positive random number drawn form a uniform distribution between 0.0 and 1.0. According to [67], a good starting point is to set ω_{start} to 0.9, ω_{end} to 0.4, and $c_1 = c_2 = 2$.
The velocity and position of a particle might end up positioning the particle beyond the boundary $[Var_{min}, Var_{max}]$ of the search space. Therefore, the need of having a scheme which can bring such particles back into the search space. In our proposal, we apply *Set On Boundary* strategy. According to this strategy the particle is reset on the bound of the variable which it exceeds [70]. Let X_C denote a current velocity or position of a solution, then X_C is set to X_C^{new} as follows:

$$X_C \rightarrow X_C^{new} = \begin{cases} -0.1 * (Var_{max} - Var_{min}) \\ \quad if X_C < lowerbound \\ \\ 0.1 * (Var_{max} - Var_{min}) \\ \quad if X_C > upperbound \end{cases} \tag{2.5}$$

An additional strategy called velocity reflection is also applied. Velocity reflection allows those particles that move toward the outside the boundary to move back into the search space according to Eq. (2.6).

$$V_i(t+1) \rightarrow -V_i(t+1) \tag{2.6}$$

2.3.4 Fuzzy Set

In classical set theory, an element either belongs or not to a set of elements. Therefore, the membership evaluation is boolean. A more flexible approach would be fuzzy set theory, where elements belong to sets with certain degree of membership that takes its value in the interval [0 1]. This makes fuzzy set theory suitable for complex models where some things are not either entirely true nor entirely false and where the problems are somehow ambiguous or it is needed to manage subjective judgments or opinions. In our scenario, it could be used to decide things like "Is the PIT entry rate average or very high?". Therefore, fuzzy set theory can be successfully employed when

most of the decision making attributes are qualitative in nature, with the possibility of subjective assessment [71]. In fuzzy set theory, a *linguistic variable* is a variable whose values are words or sentences in natural or artificial language [72]. A fuzzy rule is defined as a conditional statement in the form:

$$IF\ x\ is\ A\quad THEN\ y\ is\ B \tag{2.7}$$

Where x and y are linguistic variables; A and B are linguistic values determined by fuzzy sets on the universe of discourse X and Y, respectively. These rules are then mathematically represented by a membership function. The membership provides a measure of the degree of presence for every element in the set [73]. A fuzzy system often consists of four main parts: fuzzification, rules, inference engine, and defuzzification [19]. In the fuzzification step, a crisp set of input data is converted to a fuzzy set using fuzzy linguistic terms and membership functions. In step 2, a list of fuzzy statements are constructed to create what is called "rule base". That rule base will be used to process the fuzzy data by a computational unit, which will output again fuzzy sets. In the defuzzification step, that fuzzy output is mapped to a crisp (non-fuzzy) output using the membership functions.

Fuzzy set in anomaly detection

Fuzzy set theory is a method of representing the vagueness and imprecision which is appropriate for anomaly detection for two major reasons [51, 74]:

1. The anomaly detection problem involves many numeric attributes in collected audit data and various derived statistical measurements. Building models directly on numeric data causes high detection errors, and

2. The security itself involves fuzziness, because the boundary between the normal and abnormal is not well defined.

Fuzzy logic also can work with other popular data mining technique as outlier detection. Since malicious behavior is naturally different from normal behavior, abnormal behavior should be considered as outliers [75, 76]. Fuzzy logic can help to construct more abstract and flexible patterns for intrusion detection and thus greatly increase the robustness and adaption ability of detection systems [51]. Hence, fuzzy approach can reduce the false positive rate with higher reliability in determining intrusive activities, due to any data (normal or attack) may be similar (closest distance) to some clusters.

2.3.5 Multilayer Perceptron (MLP) neural networks

In the last years, various neural network models have been developed for different applications including signal processing, pattern recognition, system modeling and so on [77]. The multi-layer perceptron (MLP) with back-propagation learning is the most popular and commonly used neural network structure due to its simplicity, effectiveness and excellent performance in many applications that require to learn complex patterns [78, 79]. Multi Layer perceptron (MLP) is a feed-forward neural network with one or more hidden layers between input and output layer. Feed-forward means that data flows are in the forward direction, from input to output layer. MLP can solve problems which are not linearly separable [80]. A graphical representation of a MLP is shown in Fig. 2-5. In the training phase of the MLP, the training set is presented at the input layer and the

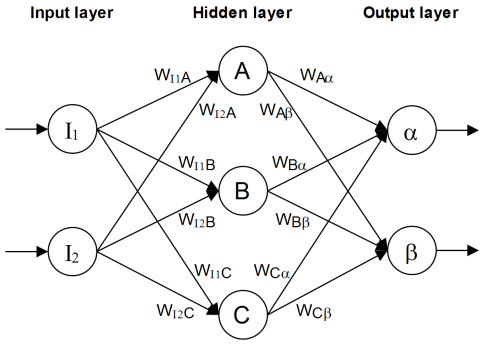


Figure 2-5: Structure of a three-layer MLP

parameters of the network (weights and biases) are dynamically adjusted using gradient-descent based delta-learning rule (back-propagation learning) to achieve the desired output [81, 82]. The training process of MLP neural network is defined as follows:

`Step 1:` *Network initialization.* The connection weights and bias of the network are initialized randomly, setting up the network learning rate η, the error threshold ε, and the maximum iterations T.

`Step 2:` *Data preprocessing.* Data samples are usually partitioned into three sets: training, validation and test. The training sets are used for training (to adjust the weights and biases) the network; the validation sets are the part that assesses or validates the predictive ability of the model during the training to minimize overfitting; the test sets are used for independent assessment of the model's predictive ability (generalization performance) after training.

`Step 3:` *Training network.* input the training sets into MLP, compute network predicted output values, and calculate the error E between output and the target value. The error function is defined

as follows:

$$E = \frac{1}{2}\sum_{k}^{m}(\hat{y}(k) - y(k))^2 \tag{2.8}$$

Where,

$$\hat{y}(k) = \phi_k(\sum_{i=1}^{m} W_{ik}h(i)) \quad k = 1, 2, ..., l. \tag{2.9}$$

Where, ϕ_k is an activation function for neuron k in hidden layer, $h(i)$ is the output value for neuron (node) i in the hidden layer, W_{ik} is weight connection between neuron i in hidden layer and neuron k in output layer, l and m are the number of neurons for output layer and the hidden layer, respectively. $h(i)$ is calculated by:

$$h(i) = \phi_i(\sum_{j=1}^{n} W_{ij}X_j + b_i) \quad i = 1, 2, ..., m \tag{2.10}$$

Where, ϕ_i is an activation function for neuron i in hidden layer, W_{ji} is the weight connection between neuron i and input j, X is input value, and b_i is the bias connection of neuron i in hidden layer.

`Step 4:` *Updating the weights and biases.* update network weights and biases according to the prediction error E, making the predictive value of the network as close to actual values through a Back-propagation algorithm.

`Step 5:` *Judgment of whether the end condition is met.* If $E <= \varepsilon$, network training is stopped and go to step 7.

`Step 6:` *Judgment of whether an overfitting has occurred.* If accuracy of the validation error has not been satisfied network training is stopped and go to step 7; otherwise, return to step 3 to continue training.

`Step 7:` *Judgment of generalization performance.* Run test data set by trained network for generalization performance measurement.

`Step 8:` *Further usage.* if the prediction error of the network is acceptable, use the network for further usage; otherwise, go to the Step 1 and train the network again until an ideal network with desire accuracy is found.

2.3.6 RBF Neural Networks

Radial Basis Function (RBF) is a kind of feed-forward neural networks, which were developed by Broomhead and Lowe in 1998 [83]. This type of neural networks use a supervised algorithm

and have been broadly employed for classification and interpolation regression [84]. As compared to other neural networks, RBF neural networks have better approximation characteristics, faster training procedures and simple network architecture. For these reasons, researchers have continued working on improving the performance of RBF learning algorithms [85, 86]. The RBF neural networks have three layers architecture including a single hidden layer of units. The first layer has n input units which connects the input space to the hidden layer. The hidden layer has m RBF units, which transforms the input units to the output layer. The output layer, consisting of l linear units. The output layer implements a weighted sum of hidden unit outputs. The input layer is non-linear while the output is linear. Due to non-linear approximation properties in RBF, this type of networks are able to model the complex mappings [87]. The real output in output layer is given by:

$$y_s(X) = \sum_{j=1}^{k} w_{js}\phi(\frac{\|P - C_j\|}{\sigma_j}) \quad for\ 1 \leq s \leq l \tag{2.11}$$

Where y_s is s-th network output, P is an input pattern, w_{js} is the weight of the link between j-th hidden neuron and s-th output neuron, C_j is the center of the j-th RBF unit in the hidden layer, and σ_j is the width of the j-th unit in the hidden layer. The ϕ denotes to an basis (activation) function. The Gaussian activation function is used in this dissertation, which is given by [88]:

$$\phi_j(r) = exp(-\frac{\|P - C_j\|^2}{2\sigma_j^2})\ \ j = 1, 2, 3, ..., p \tag{2.12}$$

Where r is the variable of radial basis function (ϕ).

2.3.7 Adaptive Neuro-Fuzzy Inference System (ANFIS)

ANFIS is a class of adaptive networks whose functionality is equivalent to a fuzzy inference system which generates a fuzzy rule base and membership functions automatically [89]. ANFIS is an integration of neural network architectures with fuzzy inference system (FIS) to map a couple of inputs-output data patterns. An ANFIS constructs a FIS (*if-then* rules) whose membership function parameters are adjusted using either backpropagation algorithm or in combination with a least squares type of method [90, 91]. An ANFIS architecture consists of a fuzzy layer, product layer, normalized layer, defuzzy layer, and summation layer. A typical architecture of ANFIS with two inputs (x and y), nine rules and one output (f) is depicted in Fig. 2-6. Among many FIS models, the 1st order Sugeno fuzzy model is the most widely applied adaptive technique with high

interpretability and computational efficiency for different problems [90, 92]. For a 1st order of

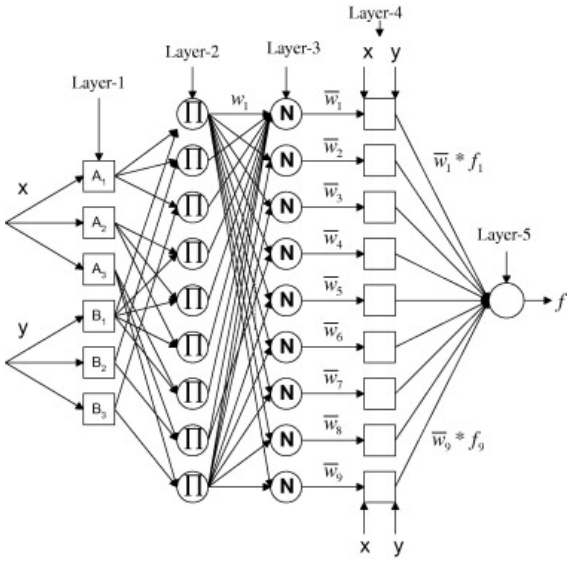


Figure 2-6: ANFIS architecture with two inputs and nine rules [1]

Sugeno fuzzy model, a typical rule set with two fuzzy *if-then* rules can be expressed as:

$$if x \text{ is } A_1 \text{ and } y \text{ is } B_1 \text{ then } f_1 = p_1 x + q_1 y + r_1 \tag{2.13}$$

$$if x \text{ is } A_2 \text{ and } y \text{ is } B_2 \text{ then } f_2 = p_2 x + q_2 y + r_2 \tag{2.14}$$

where A_i and B_i are the fuzzy sets in the antecedent and p_i, q_i, and r_i are the linear output parameters that are determined during the training process. As in Fig. 2-6, an ANFIS consists of five layers and nine *if-then* rules as follows:

Layer-1: all the square nodes in this layer are adaptive nodes. The outputs of layer 1 are the fuzzy membership grade of the inputs, which are given by:

$$O_{1,i} = \mu_{Ai}(x), \ for \ i = 1, 2, 3 \ \ O_{1,i} = \mu_{Bi-3}(y), \ for \ i = 4, 5, 6 \tag{2.15}$$

where x and y are inputs to node i, and A_i and B_i are linguistic labels for inputs. $O_{1,i}$ is the membership function of A_i and B_i. $\mu_{Ai}(x)$ and $\mu_{Bi-3}(y)$ can adopt any fuzzy membership function. For instance, if a Gaussian membership function is employed:

$$\mu_{Ai}(x) \ , \ \mu_{Bi-3}(y) = exp\left[-(\frac{x - c_i}{a_i})^2\right] \tag{2.16}$$

where c_i and a_i are the parameter set of the membership function. These parameters in this layer are referred to a premise parameters.

Layer-2: Every node in this layer is a fixed node with a circle node label $\prod$ which multiplies the incoming signals and sends the product out. The output of this layer can be represented as:

$$O_{2,i} = w_i = \mu_{Ai}(x) \times \mu_{Bi-3}(y), \quad i = 1, 2, 3, ..., 9 \tag{2.17}$$

Each node output represents the firing strength of a rule.

Layer-3: Every node in this layer is also a fixed circle node labeled N, indicating that they play a normalization role to the firing strengths from the previous layer. The outputs of this layer can be represented as:

$$O_{3,i} = \bar{w_i} = \frac{w_i}{(w_1 + w_2 + ... + w_9)}, \quad i = 1, 2, 3, ..., 9 \tag{2.18}$$

Layer-4: In this layer, the square nodes are adaptive nodes. The output of each node in this layer is the product of the normalized firing strength and a 1st order polynomial (for a 1st order Sugeno model):

$$O_{4,i} = \bar{w_i} \; . \; f_i = w_i \; . \; (p_i x + q_i y + r_i), \quad i = 1, 2, 3, ..., 9 \tag{2.19}$$

where w_i is the output of layer 3 and three parameters $\{p_i,\ q_i,\ r_i\}$ are the parameter set which will be referred to as consequent parameters.

Layer-5: The single node in this layer is a circle node labeled $\sum$ (overall output) that performs the summation of all incoming signals:

$$O_{5,i} = f = \sum_i \bar{w_i} f_i = \frac{\sum_i w_i f_i}{\sum_i w_i} \tag{2.20}$$

ANFIS has a hybrid learning rule algorithm which integrates the gradient descent and the least squares methods to train and adjust the premise and consequent parameters [93]. The hybrid learning algorithm is composed of a forward pass and a backward pass. The least squares method (forward pass) is used to optimize the consequent parameters with the premise parameters fixed to minimize the measured error in layer 4. In the backward pass, the premise parameters are updated by the gradient descent method [89, 93].

2.3.8 Non-dominated Sorting Genetic Algorithm (NSGA II)

NSGA II is one of the most widely and popular multi-objective optimization algorithms with three considerable properties including fast non-dominated sorting approach, fast crowded distance estimation procedure and simple crowded comparison operator [94]. Fig. 2-7 shows the NSGA II procedure. Generally, NSGA II can be roughly detailed as following steps [94, 95, 96]:

Step 1: Population initialization

A set of random solutions (chromosomes) with a uniform distribution based on the problem range and constraint are generated. The first generation is a $N \times D$ matrix. N and D are identified as the number of chromosomes and decision variables (genes), respectively.

Step 2: Non-dominated sort

Sorting process based on non domination criteria of the initialized population.

Step 3: Crowding distance

Chromosomes are classified to the Pareto fronts using:

$$d_{I_j} = \sum_{m=1}^{M} \frac{f_m^{I_{j+1}^m} - f_m^{I_{j-1}^m}}{f_m^{Max} - f_m^{Min}} \tag{2.21}$$

Where, d_{I_j} is crowded distance of jth solution, M is number of objectives, $f_m^{I_{j+1}^m}$ and $f_m^{I_{j-1}^m}$ are values of mth objective for $(j-1)$th and $(j+1)$th solution, f_m^{Max} is maximum value of mth objective function among solutions of the current population, f_m^{Min} is minimum value of mth objective function among solutions of the current population, I_j is the jth solution in the sorted list and $(j-1)$ and $(j+1)$ are two nearest neighboring solutions on both sides of I_j. Afterwards, the algorithm searches the nearest points (solutions) with more value of d_{I_j}. Solutions in the best-known Pareto set should be uniformly distributed and diverse over of the Pareto front in order to provide the decision maker a true picture of trade-offs. Then, Pareto fronts are ranked from the best to the worst.

Step 4: Selection

The selection of chromosomes is carried out to select appropriate chromosomes (parents) using the crowded tournament operator. The crowded tournament operator compares different solutions with two criteria, (1) a non-dominated rank and (2) a crowding distance in the population. In this process, if a solution dominates the others, it will be selected as the parent. Otherwise, the solution with the higher value of crowding distance (highest diversity) will be selected.

Step 5: Genetic algorithm operators

There are a variety of recombination (crossover) and mutation operators.

Step 6: Recombination and selection

The offspring population is combined with the current generation population and the total population is sorted based on non-domination. The new generation is filled by chromosomes from each front subsequently until the population size exceeds the current population size N.

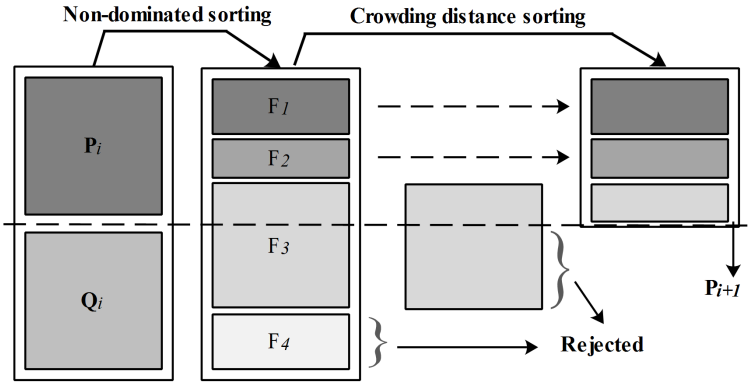


Figure 2-7: Schematic of the NSGA II procedure

2.3.9 Self-Organizing Map (SOM)

SOM (also known as Kohonen SOM) is a very popular algorithm based on competitive and unsupervised learning [97]. The SOM projects and represents higher dimensional data in a lower dimension, typically 2-D, while preserving the relationships among the input data. A diagram of the SOM is shown in Fig. 2-8. The main process of SOM is generally introduced in three phases:

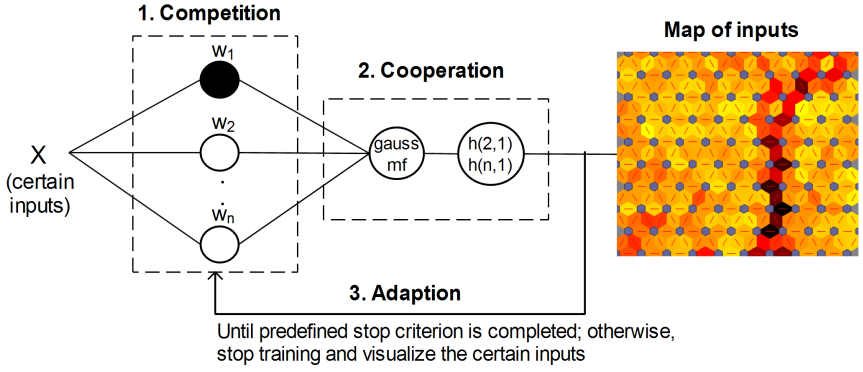


Figure 2-8: The SOM structure. The dark color of neuron in the competition phase indicates the wining neuron.

competition, cooperation and adaptation.

1. Competition: the output of the neuron in SOM computes the distance (generally Euclidean distance) between the weight vector and input vector:

$$arg\ min_i\ \|x - w_j\|,\ j = 1, 2, ..., l \tag{2.22}$$

   Where, x is the input vector, w_j is the j^{th} neuron's weight vector. The neuron j which satisfies the above condition is called the Best Matching Unit (BMU) and is the "winner" of the competition.

2. Cooperation: the winning neuron determines the spatial location of a topological neighborhood of excited neurons, thereby providing the basis for cooperation among neighboring neurons. The closest neighbors tend to get excited more than those further away while the neurons near-by the winner on the lattice get more chance to adapt. A distance function $h(n, i)$ which satisfies the above requirements can be the Gaussian function:

$$h(j, i) = exp(\frac{-{d_{j,i}}^2}{2\sigma^2}) \tag{2.23}$$

   Where, $h(j, i)$ is the topological area centered around the wining neuron i, $d_{j,i}$ is the lateral distance between winning neuron i and cooperating neuron j, and σ is the radius influence. The Gaussian function is symmetrical and decreases monotonically to zero as the distance goes to infinity.

3. Adaption: since SOM is self-adaptive, the winner and its neighbors increase their discriminant function value relative to the current input. All neurons in the neighborhood of the winner are updated in order to make sure that adjacent neurons have similar weight vectors [98]. In practice, the appropriate weight update equation is as follows:

$$w_j = w_j + \eta h(j, i) * (x - w_j) \tag{2.24}$$

   Where, η is a learning rate, i is the index of winning neuron, w_j is the weight of the neuron j and $h(j, i)$ function has formulated in Equation 2.23.

These three phases are repeated during the training, until the changes become smaller than a predefined threshold or the maximum number of iterations is met.

Chapter 3

A Fuzzy Anomaly Detection System based on Hybrid PSO-Kmeans Algorithm

Attacks and anomalies are deliberate actions against data, contents, software or hardware that can destroy, degrade, disrupt or deny access to a computer network [22]. Hence, the contents should be resilient against both DoS and new forms of (unknown) attacks or at least limit their effectiveness [23]. In order to disarm new kinds of attacks, anomalous traffics, and any deviation, not only the detection of the malevolent behavior must be achieved, but the network traffic belonging to the attackers should be also blocked [24, 25, 26]. In an attempt to tackle with the new kinds of anomalies and the threat of future unknown attacks, many researchers have been developing Intrusion Detection Systems (IDS) to help filter out known malware, exploits and vulnerabilities [22, 99]. Anomaly detection systems are becoming increasingly vital and valuable tools of any network security infrastructure in order to mitigate disruptions in normal delivery of network services due to malicious activities, Denial of Service (DOS) attacks and network intrusions [100, 101]. An IDS dynamically monitors logs and network traffics, applying detection algorithms to identify potential intrusions and anomalies within a network [102].

In recent years, data mining techniques specially unsupervised anomaly detection have been employed with much success in the area of intrusion detection [103, 104, 105]. Generally, unsupervised learning or cluster analysis algorithms have been utilized to discover natural groupings of objects and find features inherent and their deviations with similar characteristics to solve the detection

problems of the abnormal traffics and unknown forms of new attacks [106, 107]. Data clustering algorithms can be either hierarchical or partitioning [108, 109]. In this dissertation, we focus on the partitioning clustering and in particular, a popular method called K-means clustering algorithm. The K-means algorithm is one of the most efficient clustering algorithms [61, 110, 111]. This algorithm is simple, easy to implement, straightforward, suitable for large data sets, and very efficient with linear time complexity [112]. However, it suffers from two main drawbacks: (1) the random selection of centroid points and determining the number of clusters may lead to different clustering results, (2) The cost function is not convex and the K-means algorithm may contain many local optimum [113]. In the previous work [19], we employed K-means clustering in our anomaly detection system over CCN. But, the results were not appropriate due to the large number of clusters, trapping in the local optimum solution, and changing results by running the algorithm with the constant parameters in several times. However, if good initial clustering centroids can be assigned by any of other global optimal searching techniques, the K-means would work well in refining the cluster centroids to find the optimal centroids [114, 115]. To overcome these drawbacks, we present a fuzzy anomaly detection system in two phases: training and detection. In the training phase, we apply a meta-heuristic algorithm called PSO (Particle Swarm Optimization) which can find the optimal or near optimal solution by the least iterations [116, 117, 118]. We employ the combination of the ability of global search of the PSO with a novel boundary handling approach and the fast convergence of the K-means to avoid being trapped in a local optimal solution.

On the other hand, the most clustering methods usually try to minimize the Mean Square Error (MSE) between data points and their cluster centroids [119, 120]. The MSE is not suitable for determining the optimal number of clusters. Since it decreases, the number of the clusters increase. We develop our method for globally optimal placement of data points as well-separated clusters by low intra-cluster cohesion and high inter-cluster separation. But the optimal placement can increase MSE [121]. Thus, we apply MSE for local optimization, i.e., in the case of each cluster separately to decrease the error caused by corresponding data points and their cluster centroids. This simultaneous approach -application of two cost functions (well-separated clusters and local optimization)- in PSO can lead to the optimal number of clusters and well-separated clusters. When the optimal placement of clusters centroids and objects are defined, they are sent to the second phase. In the detection phase, we apply a novel fuzzy decision approach to give a fuzzy detection of normal or abnormal results in the new monitoring data that do not appear in the training data set. Because fuzzy approach can reduce the false positive rate with higher reliability in determining intrusive

activities, due to any data (normal or attack) may be similar to some clusters.

Table 3.1: Comparison of hybrid PSO + K-means approaches in clustering problems

Approach	Raw data	Parameters value	Cost function	Contribution
Junyan Chen (2012) [122]	a commercial website log file with 726 clients and 52 pages which clustered separately to 15, 25 and 35 classes	iteration: 50	$\sum_{j=1}^{m}\sum_{\forall x_n} d(x_n, z_{i,j})$, x_n is the data point, and z_{ij} refers to the j_{th} cluster centroid of the i_{th} particle, and d is the position of the particles.	an hybrid PSO for initial seeds in K-means by incorporating the multidimensional asynchronism and stochastic disturbance model to the velocity, called MSPSO-K.
Zhenkui et al. (2008) [123]	city coordinates of Hopfield-10 TSP (10 records) and Iris (150 records)	$c1 = c2 = 1.3$, w linearly reduces from 1.0 to 0.3, iteration: 10, population size: 10 (first data), 130 (second data)	$\frac{(1)max(\sum_{\forall x_i \in y_j} \frac{d(x_i,y_j)}{\lvert y_j \rvert})}{(2)min(d(y_i,y_j)),\forall i,j,i\neq j}$, (1) is the maximum value of the mean of distances within same classes, and (2) is the minimum value of distances between classes.	a combination of the core idea of K-means with PSO, which it leads to the clustering algorithm with low error rate as compared to K-means.
Cui & Potok (2005) [124]	artificial data sets: ds1 (414, 6429, 9), ds2 (313, 5804, 8), ds3 (204, 5832, 6), ds4 (878, 7454, 10) (*1st*: number of documents, *2nd*: number of terms, *3rd*: number of classes)	$c1 = c2 = 1.49$, $w = 0.72$ (in the PSO, w reduces 1% at each iteration but for hybrid it is constant), iteration: 50, population size: 50	ADVDC $= \frac{\sum_{i=1}^{N_c}[\frac{\sum_{j=1}^{P_i} d(O_i, m_{i,j})}{P_i}]}{N_c}$, $m_{i,j}$ denotes the jth document vector belongs to the cluster i, O_i is the centroid vector of ith cluster, P_i stands for the document number belongs to the cluster C_i, and N_c stand for the cluster number.	an hybrid PSO-Kmeans document clustering algorithm presents to performs fast document clustering. The cluster quality measured with ADVDC (average distance between documents and the cluster centroid) which the smaller ADVDC value results the more compact clusters.
Merwe & Engelbrecht (2003) [125]	two 2-dimensional artificial data set (n=400 with c=2 and n=600 with c=4), Iris (n=150, c=3, d=4), Wine (n=178, c=3, d=13), Breast-cancer (d=9, c=2), Automotive (n=500, d=11), n: number of data, c: number of class, d: number of attribute	$c1 = c2 = 1.49$, $w = 0.72$, iteration: 1000, population size 10	$\frac{\sum_{j=1}^{N_c}[\sum_{\forall Z_p \in C_{ij}} \frac{d(Z_p, m_j)}{\lvert C_{ij} \rvert}]}{N_c}$, $\lvert C_{ij} \rvert$ is the number of data vectors belonging to cluster C_{ij}, m_j refers to the j_{th} cluster centroid, Z_p denotes the centroid vector of cluster j, and N_c is the number of the cluster centroid vectors.	the result of the K-means algorithm utilized as one particle, while the rest of the swarm is initialized randomly. The quality is measured by the low intra-cluster (distance between data within a cluster), and high inter-cluster distance (distance between the centroids of the clusters).
Xiao et al. (2006) [126]	1st data set for training and developing normal clusters (97,278 normal samples) and the 2nd data set for evaluation (60,593 normal and 250,436 attack samples) from KDDCup 1999	w decreases linearly by $(w_1 - w_2) * \frac{Max\_iter - iter}{Max\_iter} + w_2$, limit the velocity to $sign(v_{id})v_{dmax}$ if it exceeds a positive constant value v_{dmax}	$f = \frac{1}{1+J_c}$, $J_c = \sum_{j=1}^{k}\sum_{X_i \in C_j} d(X_i, Z_j)$, $d(X_i, Z_j)$ is Euclidean distance between a data point X_i and the cluster center Z_j.	it is an anomaly intrusion detection system based on combination of PSO (for initializing K cluster centroids) and K-means (for local search ability to stable the centroids). The results show a false positive rate of 2.8% and the detection rate of 86%.
Our approach	1st data set for training (5,240 normal and 530 attack instances), 2nd and 3rd data sets for evaluation (2,110 normal and 866 attack, and 1,545 normal and 486 attack instances) from three CCN scenarios	*c1=c2=2*, w linearly decreases by $w * Wdamp$ (Inertia Weight Damping Ratio), position and velocity limit by Eqs. (2.5) and (2.6), iteration: 1000, number of particles: 25	well-separated clusters through DBI (Eq. (3.5)) and local optimization through MSE (Eq. (3.1)).	a fuzzy anomaly detection method in two phases, training and detection (section 3.3). This method leads to well-separated clusters, high detection rate, and low false positive rate at the same time as compared to some other well-known methods.

3.1 Related Work

So far, there has been no attempt to further evaluate and compare hybrid intelligent algorithms for anomaly detection in NDN/CCN. In this dissertation, we are concerned with a first attempt to investigate and compare the performance of some hybrid intelligent algorithms for anomaly detection over CCN's traffics, then introduce a novel method to outperform preexisting algorithms. Using hybrid algorithms for improving the clustering performance is not a novel idea. The novelty of our proposed method is using a swarm intelligence algorithm, specifically PSO algorithm, with K-means in order to optimize clustering results based on two simultaneous metrics: (1) well-separated clusters by low intra-cluster and high inter-cluster distances and (2) local optimization by MSE (Mean Square Error). We apply a new boundary handling approach for PSO algorithm to not only select linearly the best set of parameters but fulfill also exploration and exploitation issues. Then, we propose a fuzzy detection method by the combination of two distance-based methods as classification and outlier. We design this hybrid system over CCNs to find the optimal number of clusters with high separation from neighbor clusters and low compactness of local data points, increase detection rate, and decrease false positive rate at the same time. Table 3.1 summarizes the comparison of applied PSO with K-means in different domains and with various parameters.

3.2 Clustering Problem

Mean Square Error (MSE) is the average pairwise distance between data points and the corresponding cluster centroids. Usually distance is Euclidean distance, but other metrics are also used. Given the set of cluster centroids (c), the set of corresponding data points (x), c_x denotes the cluster centroid corresponding to the x, and N is the number of data points, MSE can be calculated as:

$$MSE = \frac{1}{N} \sum_{i=1}^{N} d(x_i, c_x)^2 \tag{3.1}$$

In order to determine the correct and the optimal number of clusters, we must choose the validation criteria. There are several methods (such as K-means) which try to minimize the MSE between data vectors and their cluster centroid to verify the clustering goodness [119, 127]. But, MSE is not enough and suitable metric for determining the number of the clusters, since it decreases as the number of cluster increases. In fact, the optimal MSE would be number of the cluster equals to data set points, and the MSE=0. Therefore, we apply Davies Bouldin Index (DBI) [128] as the

criterion since, in our experiments, we have found it quite reliable among the variety of alternative internal clustering validation metrics; with regard to pointing out the correct number of clusters. DBI takes into account both compactness (intra-cluster diversity) and separation (inter-cluster diversity) criteria that makes similar data points within the same clusters and places other data points in distinct clusters. The intra-cluster diversity of a cluster j is calculated as:

$$MSE_j = \frac{1}{N}\sum_{i=1}^{N} d(x_i, c_x)^2 \tag{3.2}$$

The inter-cluster distance of the cluster i and j is measured as the distance between their centroids c_i and c_j. According to Eq. (3.3), the closeness of the two clusters can be calculated by the sum of their MSE divided by the distance of their centroids.

$$Closeness_{i,j} = \frac{MSE_i + MSE_j}{d(c_i, c_j)} \tag{3.3}$$

Small value of $Closeness_{i,j}$ denotes that the clusters are separated and a large value denotes that the clusters are close to each other. To calculate DBI value, the highest value from Eq. (3.3) is assigned to cluster as its cluster similarity:

$$Closeness_i = max(Closeness_{i,j}), i \neq j \tag{3.4}$$

Finally, the overall DBI validity is defined according to Eq. (3.5), which the lower DBI value means better clustering result.

$$DBI = \frac{1}{M}\sum_{i=1}^{M} Closeness_i \tag{3.5}$$

3.3 Proposed Fuzzy Anomaly Detection System

This section presents the details of our proposed method. Proposed fuzzy anomaly detection system consists of two phases: training and detection. Fig. 3-1 shows the proposed fuzzy anomaly detection system steps. Each phase is also described as follows.

3.3.1 Training Phase

The training phase is based on the hybridization of PSO and K-means clustering algorithm with two simultaneous cost functions: well-separated clusters (low intra-cluster distance and high inter-

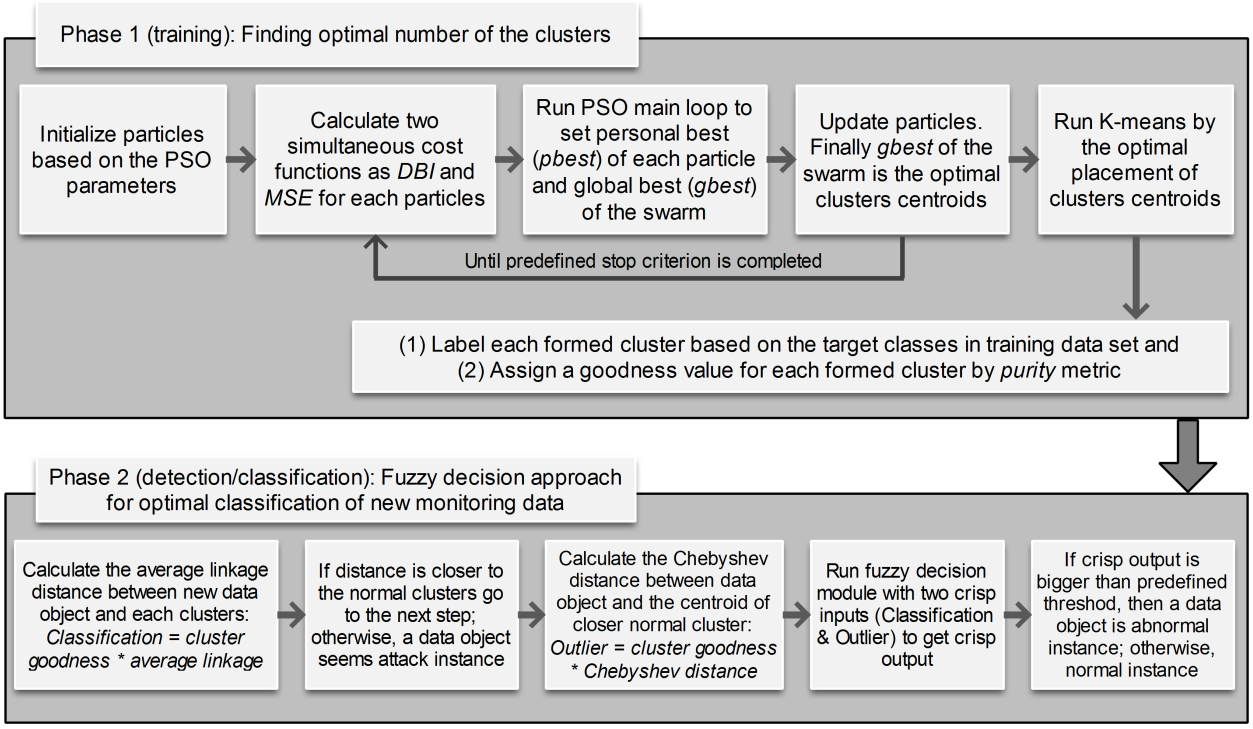


Figure 3-1: Two steps of the proposed fuzzy anomaly detection system

cluster distance) by DBI and local optimization by MSE to find the optimal number of clusters. Before training process, data samples should be normalized into [0 1], when dealing with parameters of different units and scales [129, 130]. The steps of the training phase is presented as follows:

Step 1: Define problem and PSO parameters

1.$nVar$: number of the cluster centroids, $nPop$: size of the population;

2. Define constriction coefficients parameters, $c1 = c2 = 2$ and initially $w = 1$;

3. Define inertia weight damping ratio ($Wdamp = 0.99$) to linearly decrease w;

4. Define position and velocity limits as $Var_{max} = 1$ and $Var_{min} = 0$;

5. An initial population is generated based on the $nPop$ with following parameters:

$particle.Position$ = a $m \times nVar$ matrix of random numbers generated from the continuous uniform distributions with lower (Var_{min}) and upper (Var_{max}) endpoints. m denotes size of the data set features;

$particle.Cost$ = calculate the DBI for each particle based on the generated $particle.position$;

$particle.Velocity$ = a zero matrix in $m \times nVar$ size;

$particle.Sol$ = [], (Sol is a structure of two objective functions: Cost1 (DBI) and Cost2 (MSE));

$particle.Best.Position$ = [] (keep the personal best of the position);

$particle.Best.Cost = []$ (keep the personal best of the cost);

$particle.Best.Sol = []$ (keep the personal best of the Sol);

6. $Globalbest = []$ (keep the global best of swarm);

7. Repeat the following loop until the target or maximum iteration is completed:

8. Select $Particle(i), i = 1, 2, ..., nPop$ and run the following PSO algorithm for $Particle(i)$:

8.1. Update velocity by Eq. (2.3);

8.2. Apply velocity limits by Eq. (2.5);

8.3. Update position by Eq. (2.4);

8.4. Velocity mirror effect by Eq. (2.6);

8.5. Apply position limits by Eq. (2.5);

8.6. Evaluation of two objective functions, DBI by Eq. (3.5) and MSE by Eq. (3.1);

8.7. Update personal best:

$if\ (particle(i).Cost == particle(i).Best.Cost)\ AND$

$(particle(i).Sol.MSE < particle(i).Best.Sol.MSE)$

$particle(i).Best.Position = particle(i).Position;$

$particle(i).Best.Sol = particle(i).Sol;$

$else\ if\ (particle(i).Cost < particle(i).Best.Cost)$

$particle(i).Best.Position = particle(i).Position;$

$particle(i).Best.Cost = particle(i).Cost;$

$particle(i).Best.Sol = particle(i).Sol;$

end

$end;$

8.8. Update global best:

$if((particle(i).Best.Cost == GlobalBest.Cost)\ AND$

$(particle(i).Best.Sol.MSE < GlobalBest.Sol.MSE))$

$OR\ (particle(i).Best.Cost < GlobalBest.Cost)$

$GlobalBest = particle(i).Best;$

$end;$

9. if $i > nPop$ go to the step 10; otherwise, set $i = i + 1$ and go to the step 8;

10. Update w by $w = w * Wdamp$;

11. If the maximum iteration or predefined target is not reached, set $i = 1$ and go to the step 7;

Otherwise, run K-means clustering algorithm by the obtained positions of cluster centroids from

PSO algorithm.

After the main procedure of training phase, each formed cluster is labeled based on the target (original) classes in training data set. It is highly probable that the clusters containing normal data (correct classification) will have a number of abnormal data (incorrect classification) and vice versa. Therefore, we assigned a goodness value in range of [0 1] for each formed cluster by purity metric. The purity metric determines the frequency of the most common category/class into each cluster:

$$Purity = \frac{1}{n}\sum_{q=1}^{k}\max_{1\leq j\leq l} n_q^j \tag{3.6}$$

Where, n is the total number of samples; l is the number of categories, n_q^j is the number of samples in cluster q that belongs to the original class $j(1 \leq j \leq l)$. A large purity (close to 1) is desired for a good clustering. If the all samples (data) in a cluster have the same class, the purity value set to 1 as a pure cluster. This purity metric (goodness value) is used in the detection phase.

3.3.2 Detection Phase

The defined optimal placement of cluster centroids and data objects from training phase are sent to the second phase for outlier and anomaly detection when new monitoring data enter. In the detection phase, a fuzzy decision approach applied to detect attacks and anomalies. We deploy a combination of two distance-based methods, i.e., classification and outlier:

1 **Classification**: The distances between a data object and each clusters are calculated using the *goodness value of the cluster* $\times$ *average linkage*. Average linkage approach considers small variances, because it considers all members in the cluster rather than just a single point. However, it tends to be less influenced by extreme values than other distance methods [131]. A data object is classified as normal if it is closer to the one of the normal clusters than to the anomalous ones, and vice versa. This distance-based classification allows detecting known kind of abnormal or normal traffics with similar characteristics as in the training data set.

2 **Outlier**: An outlier (noise) is a data object that differs considerably from most other objects, which can be considered as an anomaly. For outlier detection, only the distance to the normal clusters (obtained from classification phase) is calculated by *goodness value of the closer normal cluster* $\times$ *Chebyshev distance*. In the Chebyshev distance (Eq. (3.7)), distance between two vectors is the greatest of their differences along any coordinate dimension. It

allows to detect better new anomalies that do not appear in the training data set. Because it takes into account the maximum value distance approach between any coordinate dimension that would lead to become more strict against data objects measurement.

$$D_{chebyshev}(p, c) = max(|p_i - c_i|) \tag{3.7}$$

Where, p is the data object and c is the centroids of the normal cluster with standard coordinates p_i and c_i.

The proposed fuzzy detection method consists of two inputs (classification and outlier), one output, and four main parts: fuzzification, rules, inference engine, and defuzzification. In fuzzification step, a crisp set of input data are converted to a fuzzy set using fuzzy linguistic terms and membership functions. In step 2, we construct rule base. Afterwards, an inference is made and combined based on a set of rules. In the defuzzification step, the results of fuzzy output are mapped to a crisp (non-fuzzy) output using the membership functions. Finally, if the crisp output is bigger than a predefined threshold, an object is considered as an abnormal instance; otherwise, an object is a normal instance. This fuzzy approach can improve our performance criteria (high detection rate and low false positive rate at the same time) as compared to a non-fuzzy approach.

3.4 Experimental Results and Discussion

3.4.1 Performance Measurement

We compared and evaluated the training phase of our proposed method with standalone PSO and K-means algorithms as well as preexisting methods from the literature as [122], [123], [124], [125], and [126] which used different parameters and cost functions. We also employed both MSE and DBI criteria on all evaluations. In order to evaluate the performance of each method, we use the Detection Rate (DR), False Positive Rate (FPR) and F-measure criteria. The detection rate is the number of intrusions detected by the system from Eq. (3.8), the false positive rate is the number of normal traffics that was incorrectly classified as intrusion from Eq. (3.9) and F-measure is the weighted harmonic mean of precision (positive predictive value) and recall (detection rate) from Eq. (3.11).

$$DR\ (Recall) = \frac{TruePositive}{TruePositive + FalseNegative} \tag{3.8}$$

$$FPR = \frac{FalsePositive}{FalsePositive + TrueNegative} \tag{3.9}$$

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive} \tag{3.10}$$

$$F - measure = 2 \times \frac{Precision \ \times \ Recall}{Precision + Recall} \tag{3.11}$$

True negative and true positive correspond to a correct operating of the system when traffics are successfully predicted as normal and attacks, respectively. False positive refers to normal traffics when are predicted as attack, and false negative is attack traffic when incorrectly predicted as normal traffic.

Table 3.2: The five applied benchmark data sets

Data set	No. of features	No. of classes	No. of patterns
Iris	4	3	150
Glass	9	6	214
Wine	13	3	178
Ionosphere	34	2	351
Zoo	17	7	101

3.4.2 Benchmarking the proposed method

To assess the robustness and accuracy of our proposed method, we applied the five classic benchmark problems from the UCI machine learning repository [132]. Table 3.2 shows the main characteristics of these data sets. Our proposed method and the other methods mentioned in section 3.4.1 have been employed to these problems. All experiments were run 20 times, and the average classification error (Ave.) and its standard deviation (S.D.) were computed. In the experiments, 70% of data set is used as training data set in the training phase and the rest is considered as testing data set in the detection phase in order to validate the functionality of the proposed method. We assume that the normal clusters denote the correct classification and abnormal (attack) clusters denote the incorrect classification. For instance, given a data object `d` in a test data set belongings to class `A`. If it gets assigned to class `B` by the proposed classification method in the second phase, class `B` is an incorrect class/category for data object `d`. Hereby, the formed cluster belongings to class `B` is assumed to be an abnormal cluster for the data object `d`. In contrast, if data object `d` is closer to a cluster labeled class `A` (we called it normal cluster), the outlier distance should be calculated. Then, according to the detection/classification phase of the proposed method, both classification and outlier values are sent to the fuzzy module. If the crisp output is smaller than the predefined threshold, data object `d` seems normal instance (correct classification); otherwise, it seems anomalous instance (incorrect classification). The results have been summarized in Table

3.3. It can be seen in the table that our proposed fuzzy method tends to obtain a more accurate classification rate (Ave.) and lower standard deviation (S.D.) as compared to other methods.

Table 3.3: Classification error (%) for our proposed method and applied methods

Method	Type	Criteria	Data set				
			Iris	Glass	Wine	Ionosphere	Zoo
K-means	Training	Ave.	6.86	19.54	18.2	11.64	10.83
		S.D.	2.34	3.61	3.66	3.28	2.73
	Test	Ave.	5.53	17.59	18.26	11.12	9.42
		S.D.	2.32	3.12	3.76	3.1	2.6
PSO (MSE)	Training	Ave.	5.42	17.41	17.8	10.66	10.35
		S.D.	2.14	3.08	3.01	2.86	2.73
	Test	Ave.	4.84	16.41	16.81	9.59	9.64
		S.D.	2.24	3.3	2.98	2.78	2.2
PSO (DBI, MSE)	Training	Ave.	4.9	16.85	17.46	10.75	9.98
		S.D.	1.73	3.01	2.56	3.14	2.48
	Test	Ave.	4.59	16.08	16.41	9.17	8.64
		S.D.	1.62	2.85	2.41	2.72	2.6
PSO-Kmeans (MSE)	Training	Ave.	5.1	16.89	17.54	11.94	11.4
		S.D.	1.23	3.08	3.56	2.91	2.55
	Test	Ave.	4.77	16.81	17.48	9.96	9.35
		S.D.	1.26	3.1	3.26	2.78	2.6
Chen [122]	Training	Ave.	4.87	16.32	15.24	11.16	8.58
		S.D.	1.28	3.32	3.4	2.48	2.4
	Test	Ave.	5.4	16.07	15.08	9.92	8.06
		S.D.	1.4	3.63	2.92	2.39	2.02
Zhenkui [123]	Training	Ave.	5.92	16.54	16.34	10.42	10.03
		S.D.	1.35	3.47	3.4	3.36	3.3
	Test	Ave.	5.76	16.43	15.6	9.88	10.05
		S.D.	1.5	3.51	3.04	2.68	2.75
Cui [124]	Training	Ave.	5.84	18.72	16.98	12.24	11.52
		S.D.	1.34	3.78	3.3	3.79	3.17
	Test	Ave.	5.48	17.18	15.82	11.86	9.56
		S.D.	1.32	3.61	2.98	3.61	3.25
Merwe [125]	Training	Ave.	6.01	18.59	17.65	10.45	9.31
		S.D.	1.97	4.54	4.76	4.87	5.01
	Test	Ave.	5.98	17.64	16.16	11.06	9.11
		S.D.	1.75	4.85	5.02	4.85	3.97
Xiao [126]	Training	Ave.	4.91	16.29	15.62	11.18	9.49
		S.D.	1.23	3.33	3.9	2.98	2.35
	Test	Ave.	4.52	16.18	15.14	10.22	8.09
		S.D.	1.38	3.11	3.01	2.84	2.23
Our Method PSO-Kmeans (DBI, MSE)	Training	Ave.	4.01	14.44	14.88	10.04	7.98
		S.D.	1.03	2.29	2.16	2.31	2.11
	Test	Ave.	3.58	13.14	13.04	9.03	7.47
		S.D.	0.98	2.12	2.01	2.26	1.88

3.4.3 Feature Construction

We employed simple features that can be extracted by inspecting the headers of the network packets. These intrinsic features are the duration of the connection, source host, destination host, source interface, and destination interface [133]. We also used the following features in each 2 seconds

time interval:

1 *Total number of packets* sent from and to the given interface in the considered time interval,

2 *Total number of bytes* sent from and to the given interface in the considered time interval,

3 *Number of different source-destination pairs* matching the given content name that being observed in the considered time interval.

The motivation of the first two features is that the number of packets and bytes allow to detect anomalies in traffic volume, and the third feature allows to detect network and interface scans as well as distributed attacks, which both result in increased number of source-destination pairs [134].

Table 3.4: CCNx Traffic Generation

Type of traffic	Applied tools
Normal (5240 records)	(1) *ccnsendchunks* with *ccncatchunks2* (2) *ccnputfile* with *ccngetfile* (3) *ccnchat*
Attack (530 records)	(1) *ccndsmoketest* for (distributed) *Interest flooding* attack (2) make abnormal traffics to saturate channels by sending very small contents (decreasing buffer size) from owner of origin, called *Abnormal Source Behavior* (3) do not forward contents deliberately to requester(s), called *Abnormal Unreachable Content Behavior*

3.4.4 Training Phase

Since there is no reference data for content-centric networks as well as real Internet traffic, we used the CCNx software of PARC (www.ccnx.org) to run a scenario for generating of CCN traffics in a local testbed. This local testbed includes 13 Linux (Ubuntu) machines, three of them acting as servers (content origins) and the other ones as clients. Then, we ran wireshark tool to capture CCNx packets. We performed the following experiments with the main tools in CCNx: ccnsendchunks (to upload objects/files into the CCN repository), ccncatchunks2 (to receive desired contents and to write them to stdout), ccnputfile (to publish a local file in the CCNx repository), ccngetfile (to retrieve published content and writes it to the local file), ccndsmoketest (to send the large number of Interests -Interest flooding attacks- toward a host/network), and ccnchat (to run a chat channel). We conducted three attack instances for both training and detection phases including Interest flooding attacks, flooding a victim router by sending too many small contents from owner of origin content (we called it *Abnormal Source Behavior*) and making content unreachable for requesters

(we called it *Abnormal Unreachable Content Behavior*). We also carried out an anomaly instance in the detection phase as serving fake response (we called it *Abnormal Forwarder Capacity Behavior*) which does not appear in the training data set. The structure of the generated traffics are shown in Table 3.4 for training and Tables 3.5 and 3.6 for testing data sets. For the PSO algorithm,

Table 3.5: First scenario of CCNx traffic

Type of traffic	Applied tools
Normal (2110 records)	(1) *HttpProxy* application to run a HTTP proxy that converts HTTP Gets to CCN data. (2) *ccnputfile* with *ccngetfile* (3) *ccnchat*
Attack (866 records)	(1) *ccndsmoketest* for *Interest flooding* attack (2) *Abnormal Source Behavior* (3) make capacity limitation in count of content objects by forwarder/router to discard cached content objects deliberately as *Abnormal Forwarder Capacity Behavior*

Table 3.6: Second scenario of CCNx traffic

Type of traffic	Applied tools
Normal (1545 records)	(1) *ccnsendchunks* with *ccncatchunks2* (2) *ccnputfile* with *ccngetfile* (3) *HttpProxy* application
Attack (492 records)	(1) *Abnormal Source Behavior* (2) *Abnormal Unreachable Content Behavior* (3) *Abnormal Forwarder Capacity Behavior*

we used swarm size of 25 particles, the number of iterations set to 1000, and other parameters set according to subsection 3.3.1. The proposed hybrid method was implemented by the MATLAB software on an Intel Pentium 2.13 GHz CPU, 4 GB RAM running Windows 7 Ultimate.

3.4.5 Detection Phase

We use MATLAB fuzzy logic toolbox for fuzzy rule based intrusion detection. The detection phase is structured with the following components:

1 Two fuzzy set of input variables: Classification and Outlier;
classification membership: *Very Close, Close, Average, Far, Very Far*; outlier membership: *Close, Average, Far.*

2 A fuzzy set of output variable: Alarm; alarm membership: *Normal, Less Prone, High Prone, Abnormal.*

3 Fuzzy membership functions: see section 3.4.7.

4 Fuzzy rules: 15 rules (Tables 3.7 and 3.8).

Table 3.7: Rules Matrix

Outlier	Classification (Cls.)				
	Very close	Close	Average	Far	Very far
Close	Normal	Normal	Normal	Low prone	Low prone
Average	Low prone	Low prone	High prone	High prone	High prone
Far	High prone	High prone	Abnormal	Abnormal	Abnormal

Table 3.8: Some fuzzy rules in proposed fuzzy system

IF *Cls.=Average* and *Outlier=Close* **THEN** *Alarm=Normal*
IF *Cls.=Close* and *Outlier=Average* **THEN** *Alarm=LowProne*
IF *Cls.=High* and *Outlier=Average* **THEN** *Alarm=HighProne*
IF *Cls.=Very far* and *Outlier=Far* **THEN** *Alarm=Abnormal*

5 Inference: Mamdani fuzzy inference by fuzzy set operations as *max* and *min* for *OR* and *AND*, respectively.

6 Defuzzifier: Center of Gravity algorithm:

$$Center\ of\ Gravity = \frac{\int_{min}^{max} u\ \mu(u)\ d(u)}{\int_{min}^{max} \mu(u)\ d(u)} \tag{3.12}$$

Where, u denotes the output variable, μ is the membership function after accumulation, and *min* and *max* are lower and upper limit for defuzzification, respectively.

A sample solution area (fuzzy inference) of proposed fuzzy detection phase is given in Fig. 3-2.

3.4.6 Results of Training Phase

In this section, the performance of proposed method and preexisting methods from the literature are compared. Since null clusters might appear in the results, these clusters are removed and we count the correct number of K. The experiments on each method were repeated 10 times independently with several K values.

The trends of minimum and maximum ratio of the DR and the FPR at the same time for applied methods are shown in Fig. 3-3. Detailed results are also given in Table 3.9. The proposed method outperforms other preexisting methods in terms of the DR, the FPR and the F-measure at the same time. The PSO (DBI and MSE) could satisfy DR by 99% when initial K is between 300 and 500. However, it could not satisfy a suitable FPR. By the hybridization of K-means

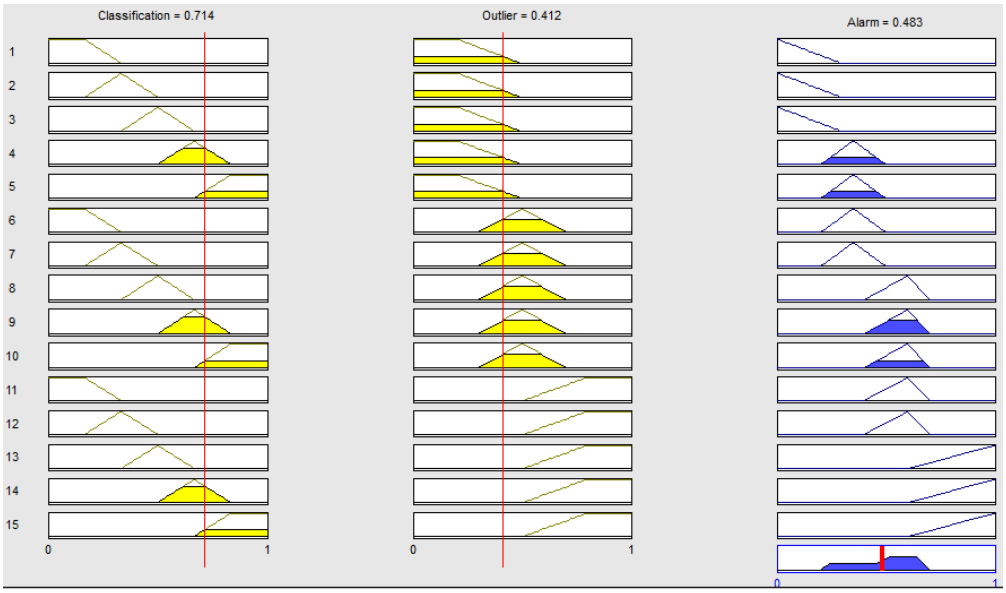


Figure 3-2: The sample solution area (fuzzy inference) of proposed fuzzy detection system

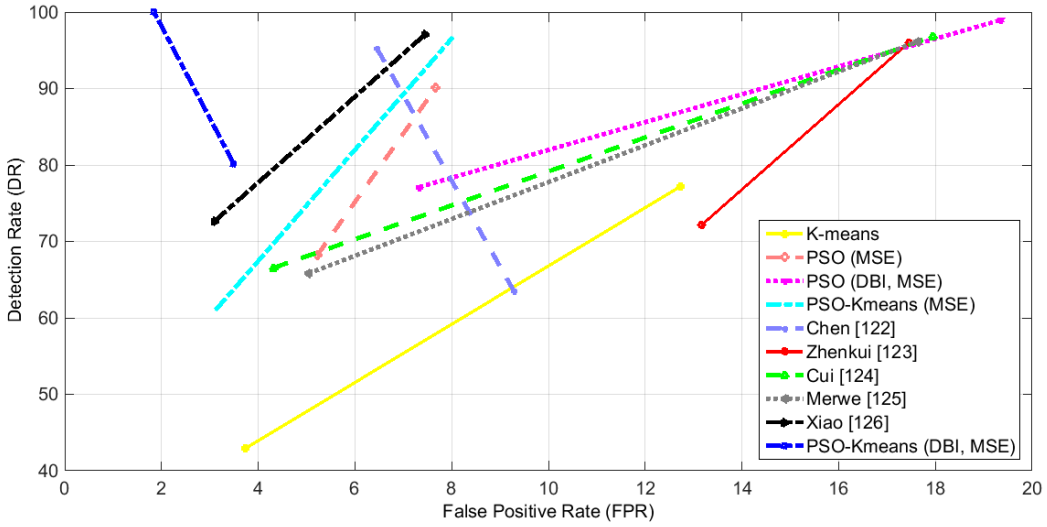


Figure 3-3: The trends of minimum and maximum combination of DR and FPR at the same time

algorithm and PSO (DBI and MSE), we could gain suitable results by very low FPR and very high DR at the same time. In contrast, none of other methods meet very high DR and very low FPR at the same time. According to the Table 3.9, by increasing of initial parameter K, results are more efficient with the optimal number of clusters, high detection rate, low false positive rate and greater F-measure at the same time. The results clearly show that our proposed method offers the best optimized solution in comparison with the other methods when K=400 by DR=100%,

Table 3.9: Comparison of our proposed method with some other methods

K	Criteria	Kmeans	PSO (MSE)	PSO (DBI, MSE)	PSO-Kmeans (MSE)	Chen [122]	Zhenkui [123]	Cui [124]	Merwe [125]	Xiao [126]	Our Method PSO-Kmeans (DBI, MSE)
50	Correct K	12	10	10	14	15	15	17	18	18	10
	DR (%)	56.18	68.18	77.11	69.12	71.12	73.92	76.55	74.65	73.12	80.22
	FPR (%)	9.19	5.22	7.33	13.157	12.05	12.15	9.43	9.96	8.12	3.489
	F-measure (%)	67.81	78.62	83.58	75.77	77.6	79.4	82.7	83.11	80.65	87.32
75	Correct K	15	12	10	15	15	14	18	18	16	14
	DR (%)	47.24	68.18	77.11	61.05	63.5	72.14	66.55	65.78	74.12	80.22
	FPR (%)	9.338	4.28	3.704	3.122	9.287	13.165	4.32	5.03	9.12	3.489
	F-measure (%)	60.31	79.05	85.28	74.36	76.15	77.81	77.87	76.84	80.85	87.32
100	Correct K	15	15	14	15	16	17	20	19	16	14
	DR (%)	47.24	68.18	77.11	67.145	64.5	72.14	76.55	75.83	76.12	80.22
	FPR (%)	8.558	6.431	7.839	7.819	7.182	12.314	8.12	9.12	12.8	3.489
	F-measure (%)	60.61	78.06	83.35	76.72	75.11	78.17	82.85	80.68	80.52	87.32
125	Correct K	17	10	15	18	18	15	21	21	17	11
	DR (%)	56.18	68.18	77.11	65.123	66.5	72.14	66.55	67.89	77.12	80.22
	FPR (%)	4.738	4.102	3.505	3.935	8.134	9.637	2.98	3.78	10.023	3.489
	F-measure (%)	69.81	79.31	85.37	77.02	76.12	79.33	78.5	78.95	82.38	87.32
150	Correct K	11	14	16	13	14	14	15	16	17	16
	DR (%)	42.93	68.18	77.11	71.147	71.119	72.14	76.55	77.93	77.14	80.22
	FPR (%)	3.738	1.345	1.345	2.101	5.98	12.508	8.88	7.64	12.209	1.314
	F-measure (%)	58.53	80.43	86.41	82.12	80.28	78.09	82.51	85.89	81.43	88.38
175	Correct K	22	22	20	21	25	31	30	32	17	20
	DR (%)	71.903	68.18	77.11	70.548	72.119	83.34	78.95	76.89	77.06	80.22
	FPR (%)	4.489	3.13	3.002	2.44	3.98	15.98	14.14	13.54	3.096	2.738
	F-measure (%)	81.51	79.58	85.61	81.55	81.88	83.55	81.71	82.48	85.53	87.68
200	Correct K	16	18	18	19	22	21	20	19	20	18
	DR (%)	64.24	71.11	77.11	74.343	72.119	73.34	72.95	74.35	81.66	80.22
	FPR (%)	2.738	3.002	1.376	2.739	9.98	12.436	12.15	13.14	14.096	1.314
	F-measure (%)	76.81	81.67	88.71	83.95	79.16	78.9	78.76	80.32	83.37	88.38
250	Correct K	16	17	15	19	19	16	21	18	18	15
	DR (%)	64.24	70.34	77.11	71.01	82.119	72.245	75.95	79.45	72.66	80.22
	FPR (%)	2.738	2.013	3.91	4.11	15.95	5.86	1.16	3.12	3.101	2.738
	F-measure (%)	76.81	81.61	85.18	81.08	82.85	81.1	85.74	86.83	82.66	87.7
300	Correct K	21	20	14	20	21	18	18	19	17	14
	DR (%)	74.82	88.27	99	88.132	81.44	89.911	90.106	88.34	94.109	100
	FPR (%)	10.314	9.12	17.352	9.19	11.209	17.33	24.51	26.93	16.91	9.117
	F-measure (%)	80.74	89.36	91.44	89.28	84.5	86.59	83.86	81.14	88.71	95.64
350	Correct K	21	25	26	22	23	23	26	25	20	26
	DR (%)	77.22	90.122	99	88.668	88.44	95.22	92.20	93.67	97.109	100
	FPR (%)	12.38	16.981	9.676	10.254	9.209	9.164	12.12	11.39	7.454	6.809
	F-measure (%)	81.4	86.96	94.84	89.1	89.45	93.13	90.19	90.83	94.91	96.71
400	Correct K	16	21	25	19	21	28	28	26	27	27
	DR (%)	77.22	90.122	99	92.55	95.29	94.005	96.20	94.23	97.077	100
	FPR (%)	12.38	16.981	17.998	10.45	6.45	15.45	13.12	14.67	14.968	1.847
	F-measure (%)	81.37	86.95	90.89	91.21	94.64	89.82	91.92	88.45	91.76	98.99
500	Correct K	23	24	21	22	25	31	33	33	27	21
	DR (%)	77.22	90.122	99	96.68	94.29	96.005	96.78	96.15	96.807	100
	FPR (%)	12.738	7.672	19.368	8.018	16.45	17.45	17.94	17.63	12.216	12.379
	F-measure (%)	81.25	91.09	90.59	94.43	89.42	89.88	90.01	90.03	92.54	94.17

FPR=1.847%, F-measure=98.99% and the correct number of K=27. We show the fluctuation of variations of two cost functions during the training phase in Figs. 3-4 and 3-5. The results clearly show that by changing of clustering values based on DBI, MSE changes in a irregular manner through the different iterations. For instance, in the last iteration, the minimum MSE is 8.391, but

the lowest MSE is in iteration 915 by 8.3458. When DBI is decreasing to find optimal clustering results in the iterations between 100 and 800, there are many fluctuations for MSE value. We

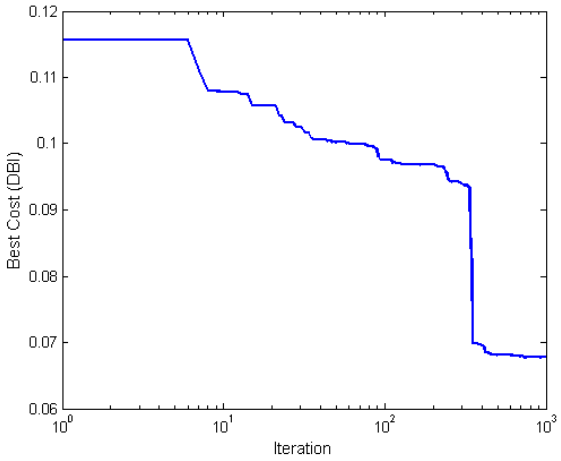


Figure 3-4: 1st cost function (DBI) in 1000 iterations

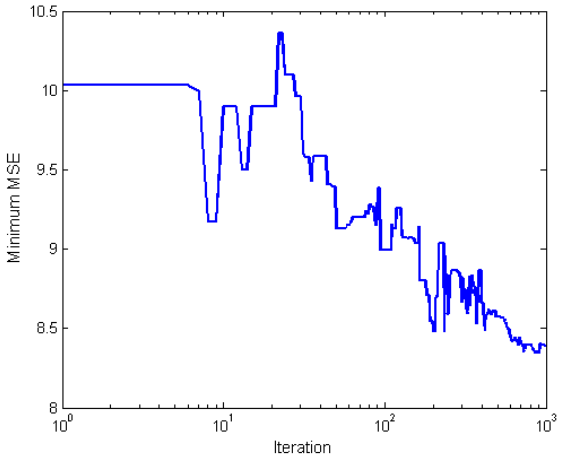


Figure 3-5: 2nd cost function (MSE) in 1000 iterations

also show the trend of changes of DBI and MSE values during the training phase when the DR was 100% and K is between 300 and 500 (Figs. 3-6 and 3-7). According to Fig. 3-6, the best and the worst procedure of reducing the DBI value are for K=300 and 400, respectively. In contrast, the best and the worst procedure of reducing the MSE value are for K=500 and 300 as shown in Fig. 3-7. The best DBI value for K=300 led to the worst value in MSE. Moreover, the highest changes for minimizing the two applied cost functions during the training phase are for K=400 and 500. These results verify that the MSE parameter cannot be singly used as a good performance

criterion for finding the optimal placement of clusters centroids and data objects. We send the optimal outcomes from our proposed method (DR = 100%, FPR = 1.847%, F-measure = 89.99% and K = 27) and the best combination of the DR, the FPR and the F-measure from other methods to the second phase for fuzzy anomaly detection.

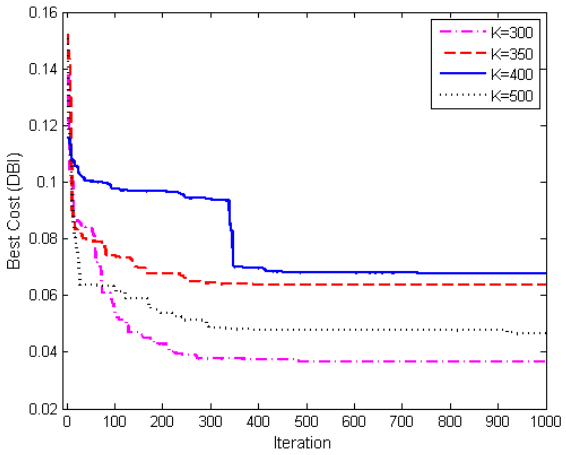


Figure 3-6: The best cost (DBI) of four clustering results

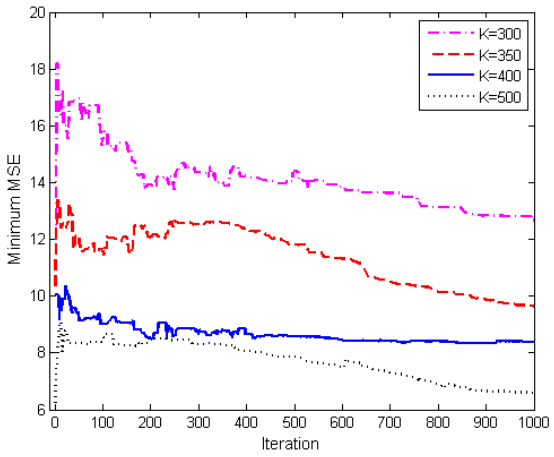
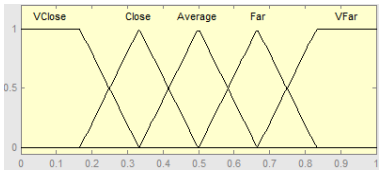


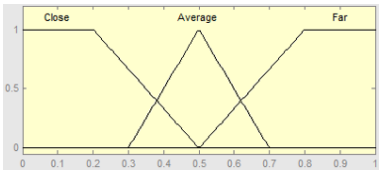
Figure 3-7: The MSE value of four clustering results

3.4.7 Results of Detection Phase

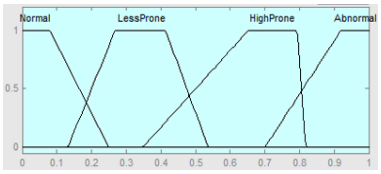
In order to obtain results on how the proposed fuzzy anomaly detection system can perform in real scenarios, we applied it to packet traces recorded at two scenarios with 17 Linux machines (10 clients, 4 servers, and 3 routers). These traces are from CCNx data repository of the University of



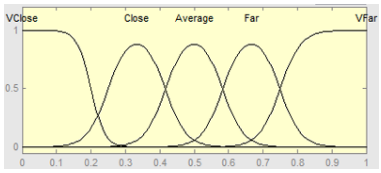
(a) Classification (trapmf)



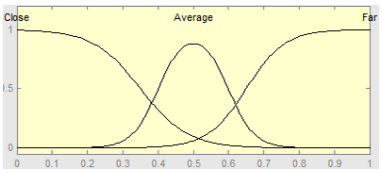
(b) Outlier (trapmf)



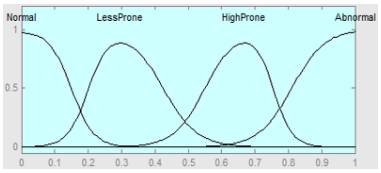
(c) Alarm (trapmf)



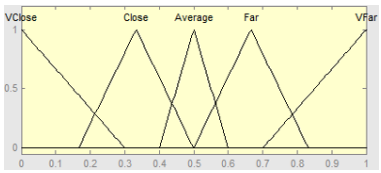
(d) Classification (dsigmf)



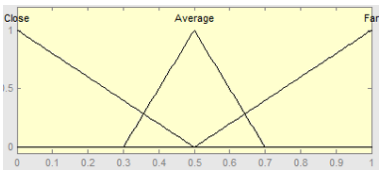
(e) Outlier (dsigmf)



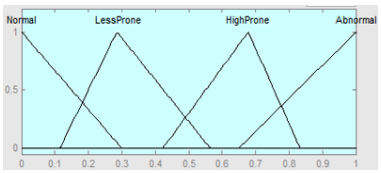
(f) Alarm (dsigmf)



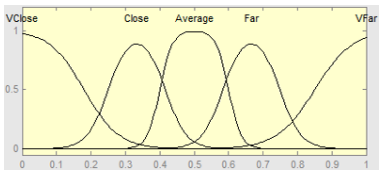
(g) Classification (trimf)



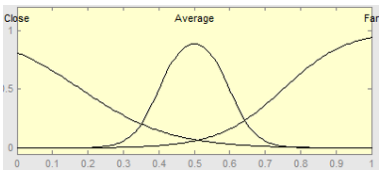
(h) Outlier (trimf)



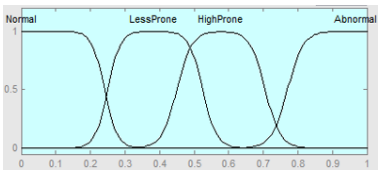
(i) Alarm (trimf)



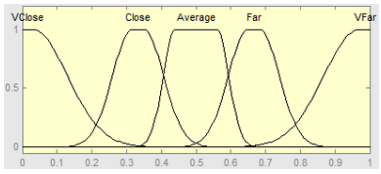
(j) Classification (psigmf)



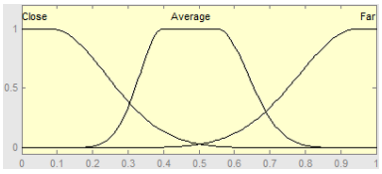
(k) Outlier (psigmf)



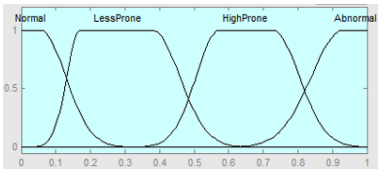
(l) Alarm (psigmf)



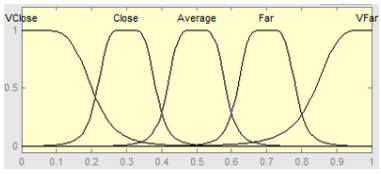
(m) Classification (gauss2mf)



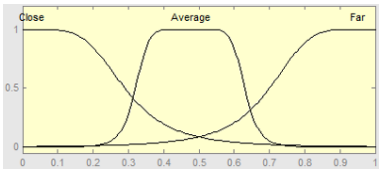
(n) Outlier (gauss2mf)



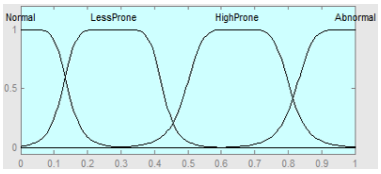
(o) Alarm (gauss2mf)



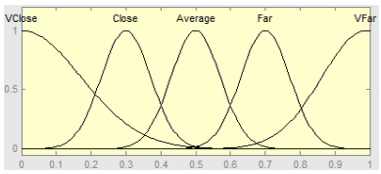
(p) Classification (gbellmf)



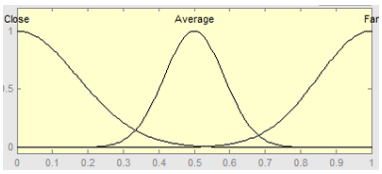
(q) Outlier (gbellmf)



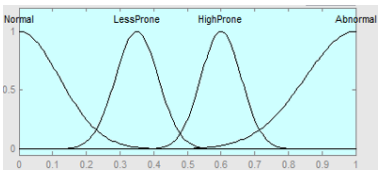
(r) Alarm (gbellmf)



(s) Classification (gaussmf)



(t) Outlier (gaussmf)



(u) Alarm (gaussmf)

Figure 3-8: Seven applied membership functions in detection phase (two inputs and one output)

Politecnica Catalunya (UPC) which are shown in Tables 3.5 and 3.6. Each trace file contains about 20 minutes of monitored traffic. According to Tables 3.5 and 3.6, there is a new type of normal traffic (*HttpProxy*) and a new type of anomaly traffic (*Abnormal Forwarder Capacity Behavior*) which have not appeared in the training data set. We also define a threshold as $d_{threshold}$=0.5. Each new monitored CCN packet is sent as input to the fuzzy detection phase in order to detect attacks and anomalies. According to the proposed fuzzy anomaly detection system (section 3.3.2), we calculate the classification distance to find the nearest cluster. If the distance is closer to one of the normal clusters, we calculate the outlier. If the outlier outcome is bigger than a predefined threshold, the packet is treated as an anomaly. In contrast, if the classification distance is closer to one of the attack clusters, it gets treated as an attack packet.

Based on the different fuzzy membership functions, the fuzzy detection method produces different results. To find the most ideal system, we apply seven membership functions for each applied methods including trapmf (Trapezoidal-shaped), dsigmf (Difference between two sigmoidal functions), trimf (Triangular-shaped), psigmf (Product of two sigmoidal), gauss2mf (Gaussian combination), gbellmf (Generalized bell-shaped), and gaussmf (Gaussian curve). Fig. 3-8 illustrates the applied membership functions. We integrated each method by optimal results gained from the training phase (Table 3.9) with our proposed fuzzy detection method in the second phase. Afterwards, we compare the performance of each method based on the RMSE, minimum and maximum error between target output and predicted output. The comparison results between methods in two applied data sets (Tables 3.5 and 3.6) are summarized in Table 3.10. We found out that the RMSE between target and predicted output is absolutely different. We marked the three best results for each membership function. The most appropriate results based on the RMSE, minimum and maximum error include our proposed method (PSO-Kmeans (DBI, MSE)), PSO (DBI, MSE), methods [122] and [126], respectively. By the integration of DBI (well-separated cost) and MSE (local optimization cost), PSO could considerably improve the results in detection phase. As shown, our proposed method is very well suited for most of the membership functions based on the less RMSE, minimum and maximum error values. Performance of trapmf and gauss2mf MF in our proposed method are better than other MF and applied methods. For anomaly detection performance measurement, we continue our experiment by applying well-performing and preexisting methods from Table 3.10 on the aforementioned data sets. The performance of fuzzy detection approach is also compared with the non-fuzzy approach. In order to validate the CCNx traffic classification performance of our fuzzy detector, we use the Receiver Operating Characteristic (ROC) curve anal-

Table 3.10: Comparison of membership functions for fuzzy anomaly detection purposes

Methods	Data set	Criteria	trapmf	dsigmf	trimf	psigmf	gauss2mf	gbellmf	gaussmf
K-means	Table 3.5	RMSE	0.1037	0.2397	0.2713	0.2268	0.3039	0.1943	0.1949
		Min error	-0.253	-0.2992	-0.2216	-0.9755	-1.5892	-1.058	-0.4240
		Max error	0.8934	0.8635	0.995	0.9334	1.0023	0.9683	0.9875
	Table 3.6	RMSE	0.0663	0.1596	0.1211	0.1238	0.2202	0.2703	0.2009
		Min error	-0.8817	-0.2088	-0.432	-0.3606	-1.1779	-0.4503	-0.6663
		Max error	0.9994	0.935	0.6265	0.3563	0.9966	0.9880	1.008
PSO (MSE)	Table 3.5	RMSE	0.0759	**0.092**	0.103	0.1162	0.0953	0.1046	**0.0865**
		Min error	-0.2625	**-0.7765**	-0.3011	-0.727	-0.6035	-0.6303	**-0.6541**
		Max error	0.4862	**0.7926**	0.4364	0.9918	0.3987	0.8653	**0.8337**
	Table 3.6	RMSE	0.1295	0.1592	0.2129	0.2604	0.1665	0.1785	0.1728
		Min error	-0.9444	-0.2385	-0.2004	-1.0268	-0.3541	-1.2154	-0.4445
		Max error	0.8943	0.8267	1.0337	0.9501	0.8683	0.4131	0.9616
PSO (DBI, MSE)	Table 3.5	RMSE	0.6525	0.1817	0.1541	0.1432	0.2024	**0.0723**	0.1587
		Min error	-0.7457	-0.276	-0.6548	-0.4627	-0.2643	**-0.584**	-0.9233
		Max error	0.937	0.8859	0.4301	0.9398	0.9489	**0.5771**	0.8303
	Table 3.6	RMSE	**0.0524**	0.0892	0.1251	0.1225	**0.0669**	0.25	0.0925
		Min error	**-0.5833**	-0.5382	-0.7248	-0.2865	**-0.6324**	-0.9254	-0.4052
		Max error	**0.6592**	0.8299	0.9487	0.7618	**0.5671**	0.9618	0.8465
PSO-Kmeans (MSE)	Table 3.5	RMSE	0.1096	0.1381	0.2582	0.2608	0.3255	0.1737	0.1931
		Min error	-0.5418	-0.2839	-0.3055	-1.173	-1.0158	-0.1510	0.8461
		Max error	0.854	0.9903	1.1168	0.915	0.9992	0.9425	0.8425
	Table 3.6	RMSE	0.2002	0.1597	**0.0979**	0.1466	0.2331	0.2647	0.168
		Min error	-1.1255	-1.1157	**-0.5717**	-1.3077	-1.005	-0.198	-0.4575
		Max error	0.9525	0.6084	**0.5921**	0.1089	0.998	1.043	0.9459
Chen [122]	Table 3.5	RMSE	0.0927	0.1093	**0.0722**	**0.1135**	0.0935	**0.0763**	**0.0581**
		Min error	-0.3177	-0.5658	**-0.3461**	**-0.5126**	-0.092	**-0.6623**	**-0.6063**
		Max error	0.4688	0.8788	**0.8808**	**0.8765**	1.003	**0.5867**	**0.6139**
	Table 3.6	RMSE	0.1156	0.3435	0.1826	0.2317	0.2817	0.23	0.2393
		Min error	-0.5278	-0.6531	-0.8078	-0.982	-0.9648	-0.1718	-0.565
		Max error	0.8821	0.9217	0.7279	0.825	1.0119	0.9886	1.032
Zhenkui [123]	Table 3.5	RMSE	0.1507	0.2584	0.1868	0.2916	0.2523	0.1115	0.2968
		Min error	-0.4221	-0.6492	-0.8722	-0.3394	-1.074	-0.4625	-1.038
		Max error	0.9439	0.7947	0.78	0.836	1.008	0.3892	0.9654
	Table 3.6	RMSE	0.1919	0.2442	0.0971	0.1749	0.1374	0.1288	0.1163
		Min error	-0.2277	-0.6492	-0.3084	-0.5541	-0.6253	-0.7965	-0.3109
		Max error	1.0243	0.8691	0.8129	0.8973	0.9699	0.9148	0.8623
Cui [124]	Table 3.5	RMSE	0.0917	0.1971	0.2805	0.2059	0.2891	0.1737	0.1568
		Min error	-0.5883	-0.494	-0.9252	-0.7737	-0.8936	-0.9185	-0.6149
		Max error	0.7866	0.9858	0.9913	1.4086	1.479	1.007	0.6044
	Table 3.6	RMSE	0.1749	0.13	0.2525	0.1282	0.2481	0.209	0.1788
		Min error	-0.5433	-0.5966	-0.6027	-0.3625	-0.9461	-1.139	-0.902
		Max error	0.9719	0.4311	0.7168	1.0516	1.085	1.005	0.391
Merwe [125]	Table 3.5	RMSE	0.0921	0.201	0.2612	0.2112	0.2761	0.1872	0.1691
		Min error	-0.593	-0.5143	-0.8982	-0.8754	-0.9012	-0.9218	-0.6241
		Max error	0.7957	0.9936	0.9984	1.4148	1.502	1.019	0.6502
	Table 3.6	RMSE	0.1791	0.1256	0.2485	0.1432	0.2516	0.215	0.1889
		Min error	-0.5553	-0.6041	-0.6081	-0.3702	-0.9333	-1.114	-0.924
		Max error	0.9784	0.4394	0.7221	1.0464	1.094	1.055	0.403
Xiao [126]	Table 3.5	RMSE	0.1442	**0.0948**	0.1206	**0.0811**	0.0961	**0.0848**	0.1106
		Min error	-0.3528	**-0.5687**	-0.6512	**-0.5823**	-0.209	**-0.5186**	-0.3415
		Max error	1.0159	**0.872**	0.556	**0.7106**	0.8354	**0.8223**	0.8651
	Table 3.6	RMSE	0.2885	0.1871	0.2245	0.2043	0.1849	0.1968	0.3799
		Min error	-1.391	-1.005	-0.8121	-1.1521	-0.803	-0.2025	-1.3634
		Max error	1.0382	0.805	1.0565	0.4807	0.9676	0.9299	0.8228
Our Method	Table 3.5	RMSE	**0.0617**	0.2525	0.1191	**0.0653**	**0.0664**	0.1176	0.3219
		Min error	**-0.4157**	-1.0143	-1.0819	**-0.5434**	**-0.581**	-0.3657	-1.0182
		Max error	**0.6002**	0.9994	0.6676	**0.5124**	**0.4562**	0.8798	1.003
	Table 3.6	RMSE	**0.0531**	**0.0738**	**0.0691**	0.2165	**0.0657**	0.1491	**0.0519**
		Min error	**-0.5215**	**-0.5281**	**-0.671**	-0.5261	**-0.5759**	-0.7349	**-0.5331**
		Max error	**0.5208**	**0.5365**	**0.488**	0.8954	**0.6468**	0.8061	**0.5982**

Table 3.11: The 2 × 2 contingency table (confusion matrix)

True label	Predicted label	
	Negative	Positive
Negative	a	b
Positive	c	d

ysis, Area Under the Curve (AUC), accuracy, specificity and sensitivity (recall). The ROC curve provides a way to visually represent how the trade-off between false positive and detection rate varies for different values of the detection threshold [135]. The AUC summarizes the classification performance of the classifier in the range [0 1] in which the higher the AUC, the easier to distinguish attacks from normal traffic [136]. The other applied performance measures can be summarized as a 2 × 2 table (confusion matrix in Table 3.11):

1. Accuracy: $(a + d)/(a + b + c + d)$
2. Specificity (true negative rate): $a/(a + b)$
3. Sensitivity (recall): $d/(c + d)$

Figs. 3-9 and 3-10 present the fuzzy and non-fuzzy ROC curves of our proposed method and the other applied methods for 1st scenario. Figs. 3-11 and 3-12 present the ROC curve for both fuzzy and non-fuzzy approaches in 2nd scenario. As it can be seen in these figures, the detection rate and the false positive rate of our proposed method (PSO-Kmeans (DBI, MSE)) are better than in the other methods. This implies a higher number of the correct detection and a lower number of the false positives. Table 3.12 shows the results of fuzzy and non-fuzzy (crisp) anomaly detection for two applied testing data sets. As shown in this table, our proposed method classifies data objects better than the other approaches based on AUC, accuracy, sensitivity and specificity. In addition, the non-fuzzy anomaly detection approach is often not sufficient in detecting many types of attacks as compared to a fuzzy detection method.

3.4.8 Computational Order

The computational order of standard PSO algorithm is $O(I \cdot S \cdot Cost)$, where I is the required generation number, S is the population size, and $Cost$ is the cost function. The computational complexity of evaluating the cost function depends on the particular cost function under consideration. The applied cost functions in preexisting methods ([122, 123, 124, 125, 126]) are $O(N \cdot K)$, where N is the number of data samples and K is the number of clusters. The computational order

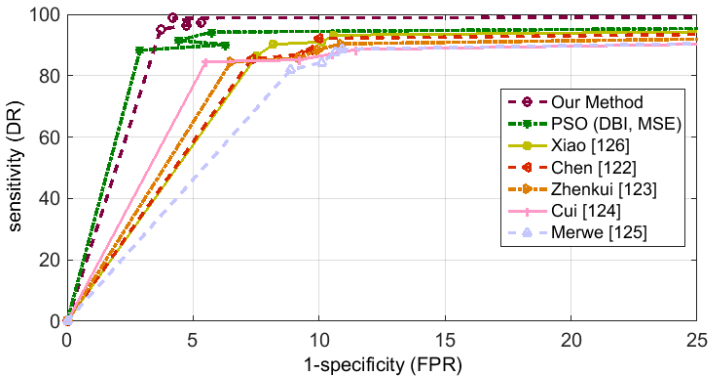


Figure 3-9: ROC curves corresponding to the proposed method and other applied methods for 1st scenario (fuzzy approach)

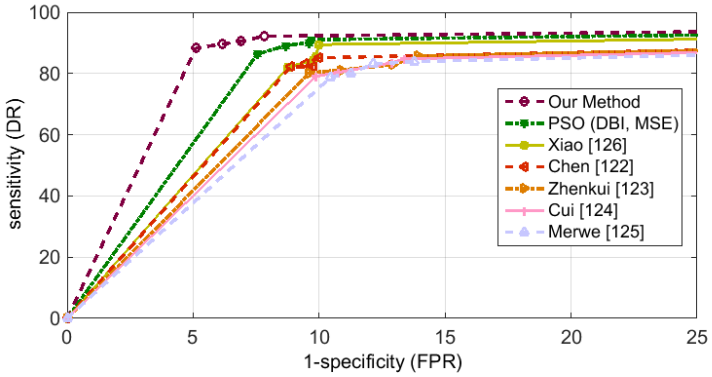


Figure 3-10: ROC curves corresponding to the proposed method and other applied methods for 1st scenario (non-fuzzy approach)

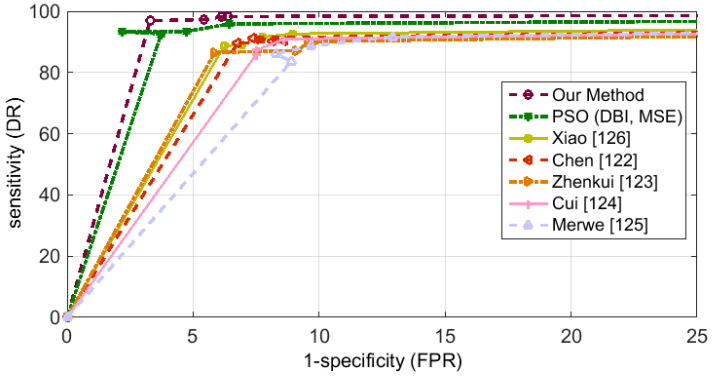


Figure 3-11: ROC curves corresponding to the proposed method and other applied methods for 2nd scenario (fuzzy approach)

of K-means algorithm is $O(T \cdot N \cdot K)$, where T is the number of iterations. The computational order of proposed training method and preexisting methods from the literature are shown in Table 3.13.

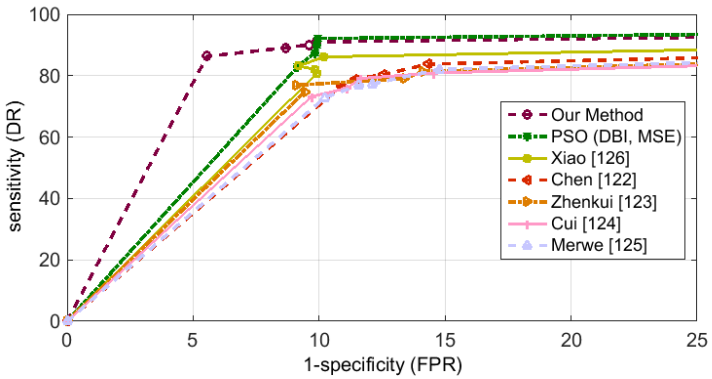


Figure 3-12: ROC curves corresponding to the proposed method and other applied methods for 2nd scenario (non-fuzzy approach)

Table 3.12: Fuzzy (non-fuzzy) anomaly detection for two applied testing data sets

Method	AUC	Accuracy		Sensitivity (recall)		Specificity	
		mean	S.D.	mean	S.D.	mean	S.D.
Data set 1: **Table 3.5**							
Our Method	97.44	94.48	0.97	96.88	1.54	95.52	0.79
	(93.26)	(89.07)	(2.35)	(90.15)	(1.41)	(93.51)	(1.02)
PSO (DBI, MSE)	95.36	91.38	1.34	91.02	2.45	94.18	1.51
	(91.41)	(87.3)	(2.48)	(89.15)	(2.03)	(91.12)	(1.29)
Xiao [126]	92.39	89.61	2.73	89.2	2.83	91.4	1.28
	(89.87)	(81.74)	(3.9)	(82.76)	(2.97)	(88.4)	(1.37)
Chen [122]	91.92	88.18	2.84	88.21	2.89	90.98	1.94
	(87.37)	(81.64)	(3.71)	(83.33)	(3.7)	(87.73)	(1.17)
Zhenkui [123]	91.37	89.29	2.98	87.07	3.04	90.11	2.08
	(87.14)	(81.18)	(3.88)	(82.13)	(3.57)	(87.61)	(2.21)
Cui [124]	90.87	88.63	3.02	87.1	3.12	90.01	2.18
	(86.78)	(80.51)	(3.76)	(82.21)	(3.85)	(87.15)	(2.4)
Merwe [125]	89.4	87.74	3.01	86.63	3.31	89.41	2.15
	(86.12)	(80.2)	(3.58)	(81.68)	(3.72)	(87.05)	(2.31)
Data set 2: **Table 3.6**							
Our Method	97.41	94.45	0.99	97.65	0.67	96.7	0.99
	(92.29)	(88.14)	(2.84)	(89.15)	(2.03)	(91.57)	(1.36)
PSO (DBI, MSE)	95.91	92.01	1.01	93.81	1.43	94.93	1.8
	(90.98)	(86.8)	(2.68)	(88.18)	(3.96)	(90.3)	(1.39)
Xiao [126]	92.92	89.84	2.83	88.49	2.19	91.58	1.74
	(88.64)	(81.06)	(3.49)	(82.3)	(3.19)	(86.32)	(1.83)
Chen [122]	92.18	89.14	2.78	87.3	0.75	90.43	1.14
	(86.67)	(80.19)	(3.9)	(81.82)	(3.09)	(85.55)	(1.98)
Zhenkui [123]	91.71	87.11	2.74	87.21	0.8	90.1	1.22
	(86.11)	(80.1)	(3.99)	(81.9)	(3.18)	(85.33)	(2.05)
Cui [124]	91.47	86.98	2.86	87.17	0.91	90.02	1.34
	(85.61)	(80.06)	(3.92)	(81.76)	(3.41)	(85.3)	(2.03)
Merwe [125]	90.08	85.49	3.03	86.66	1.03	89.43	1.53
	(85.86)	(80.01)	(3.99)	(80.54)	(3.68)	(85.11)	(2.61)

Time Complexity

We compare the computational time of algorithms on the training data set. Table 3.14 shows the computational time and the times of increment on computational time of the six methods.

Table 3.13: The computational order of the six methods

Methods	Cost function	Algorithm
Our Method	$O(MSE) + O(DBI) = O(N \cdot K) + O(K^2)$	$O(PSO) + O(K-means)$
Xiao [126]	$O(MSE) = O(N \cdot K)$	$O(PSO) \times O(K-means)$
Chen [122]	$O(MSE) = O(N \cdot K)$	$O(PSO) \times O(K-means)$
Zhenkui [123]	$O(MSE) = O(N \cdot K)$	$O(PSO) \times O(K-means)$
Cui [124]	$O(MSE) = O(N \cdot K)$	$O(PSO) + O(K-means)$
Merwe [125]	$O(MSE) = O(N \cdot K)$	$O(PSO) + O(K-means)$

Table 3.14: The computational time of the six methods

Methods	Computational time (sec)	Increment time (sec)
Our Method	791.412	92.381
Xiao [126]	1348.297	478.146
Chen [122]	1203.459	401.678
Zhenkui [123]	1301.763	424.829
Cui [124]	711.359	207.412
Merwe [125]	723.286	289.764

Table 3.14 demonstrates that the proposed method (PSO+Kmeans (DBI, MSE)) seems to be less time consuming than the other methods except methods [124] and [125] due to the application of a single cost function. But the proposed method can find the better solution with less times of increment on computational time than the other five methods due to its fast convergence speed. The results show that the proposed method with the new strategy of cost function -application of two simultaneous cost functions- can yield high accuracy as compared to other methods without very much computational cost.

3.4.9 Discussion

In this dissertation, a fuzzy anomaly detection system has been proposed for content-centric networks. This system applies a new hybrid approach with PSO and K-means in two phases: training and detection. In the training phase, we propose an hybridization of Particle Swarm Optimization (PSO) and K-means algorithm with two simultaneous cost functions as well-separated clusters by DBI and local optimization by MSE. The algorithm utilizes the iteratively global search ability of PSO to find optimal or near optimal cluster centroids and local search ability of K-means to avoid being trapped in a local optimal solution. A new boundary handling approach is also utilized in the PSO to not only select linearly the best set of parameters but fulfill also exploration and exploitation issues. When the optimal placement of clusters centroids and objects are defined, they are sent to the second phase. In the detection phase, we employ a fuzzy approach by the combination of two distance-based methods as classification and outlier to detect anomalies in new monitoring

data.

Convergence of the proposed fuzzy anomaly detection system is studied for finding the global and optimal results and measuring the suitable performance over different CCN traffic flows (Table 3.9 from training phase and Tables 3.10 and 3.12 from detection phase). Experimental results show that the applied CCN traffic flows could be used well for both training and detection phase as well as preexisting methods from the literature.

Convergence of the proposed method is also studied for finding global classification of different benchmarking data sets as Iris, Glass, Wine, Ionosphere and Zoo. Experimental results (Table 3.3) show the accuracy and the robustness of our proposed method based on the average of correct classification and lower standard deviation as compared to other methods.

The feasibility and efficiency of proposed system in training phase compared to nine different approaches. Table 3.9 depicts the final results using K-means, PSO (MSE), PSO (DBI, MSE), PSO-Kmeans (MSE), methods [122], [123], [124], [125], [126], and our proposed method as PSO-Kmeans (DBI, MSE). The proposed training phase outperforms other methods based on the optimal results as DR = 100%, FPR = 1.847% and F-measure = 98.99 %. In the training phase, future work is needed in the application of multi-objective optimization techniques. Moreover, detection phase results are very capable for anomaly detection purposes. The various membership functions are employed to demonstrate the effectiveness of our proposed method among applied well-performing methods in Table 3.10. In the most cases, the proposed anomaly detection method performed better than other methods based on the RMSE, minimum and maximum error between target and predicted output at the same time. Specifically, optimal results gained by trapmf and gauss2mf MF. In the detection phase, future work is needed in the application of non-linear membership functions.

Our proposed method and the other methods use different parameter settings and were repeated 10 times independently to find the global results in the training phase; therefore, the effect of tuning parameters on performance of the methods are studied.

We continue our anomaly detection performance measurements by applying well-performing and preexisting methods (from Table 3.10) and our proposed method over two applied data sets (Tables 3.5 and 3.6). As shown in Figs. 3-9-3-12 and Table 3.12, the proposed fuzzy and non-fuzzy anomaly detection phase can outperform other methods. In addition, the times of increment on computational time of proposed method is relative smaller than the other considered methods (Table 3.14).

3.5 Conclusion

In this dissertation, we proposed a novel fuzzy anomaly detection system based on the hybridization of PSO and K-means clustering algorithms over Content-Centric Networks (CCNs). This system consists of two phases: the training phase with two simultaneous cost functions as well-separated clusters by DBI and local optimization by MSE, and the detection phase with two combination-based distance approaches as classification and outlier. Experimental results and analysis show the proposed method in the training phase is very effective in determining the optimal number of clusters, and has a very high detection rate and a very low false positive rate at the same time. In the detection phase, the proposed method clearly outperforms other applied method in terms of AUC (area under the ROC curve), accuracy, sensitivity and specificity. In addition, the times of increment on computational time of proposed method is relative smaller than the other considered methods.

We are currently working on several improvements of the presented approach with the application of computational intelligence methodologies (such as multi-objective optimization techniques) to propose a robust method to improve the accuracy of detection rate and reduce false positive rate over different CCNs traffics.

Chapter 4

An ANFIS-based Cache Replacement Method for Mitigating Cache Pollution Attacks

The ubiquitous in-network caching is a key NDN feature as it reduces overall latency and improves bandwidth utilization for popular content [9, 28, 29, 30]. However, pervasive caching strengthens the security problem of cache pollution attacks in two generic classes: locality-disruption and false-locality [31, 32, 33, 34]. Locality-disruption attacks continuously generate requests for new unpopular files to force routers (i.e., the victims of the attack) to cache unpopular content, thus degrading cache efficiency by ruining the cache file locality. False-locality attacks repeatedly request the same set of unpopular (i.e., fake popular) files, thus degrading the hit ratio by creating a false file locality at cache.

Cache replacement algorithms play an important role in the analysis of cache pollution attacks [2, 137, 138]. Cache replacement refers to the process that a cache capacity becomes full and old content must be removed to make a space for new content. However, the most replacement algorithms and policies are susceptible to a subclass of pollution attacks [137, 138]. These algorithms and policies consider just one criterion and ignore other criteria that may influence on the caching efficiency and suffer from cache pollution attacks [139, 140, 141]. In this dissertation, a new cache replacement method in NDN is developed to detect and mitigate these two types of cache pollution attacks. The proposed method is based on the relationship between inherent characteristics of the cached content and the content type (i.e., attack or non-attack). Many researchers have proposed

meaningful relationship between a series of nonlinear input-output data patterns using Adaptive Neuro-Fuzzy Inference System (ANFIS) [92, 142, 143, 144]. ANFIS is a beneficial method to handle linguistic concepts and find nonlinear relationships between inputs and outputs, which is a combination of the strength of Artificial Neural Network (ANN) and fuzzy systems [145, 146]. In ANFIS, neural networks extract automatically fuzzy rules from numerical data through the learning process, and the membership functions are adaptively adjusted. The whole proposed ANFIS-based cache replacement method contains three steps: the input-output data patterns are extracted from the NDN scenarios at first. The input features are the inherent characteristics and statistical data of the cached content, and the output is the numerical value which refer to the type of the content, i.e., locality-disruption, false-locality or healthy. After that, the accuracy of constructed ANFIS is verified under different cache pollution circumstances. And finally, the constructed model is established in a simulation environment to be integrated with NDN topologies as a novel cache replacement method to mitigate cache pollution attacks in a timely manner.

The main objective of the proposed method is to enable the caching efficiency through a novel nonlinear cache replacement method in the presence of the cache pollution attacks and satisfy some applied performance metrics. The evaluation through simulations shows that the proposed nonlinear cache replacement method based on ANFIS provides benefits in cache robustness and mitigating cache pollution attacks with high accuracy in a timely manner. We then illustrate that the proposed method provides a suitable compromise between overhead and applied performance metrics as compared to some common existing countermeasures.

4.1 Related Work

As a new Internet architecture proposal, there is very limited work recently regarding to mitigation of cache pollution attacks in NDN. Park et al. [33] propose a detection approach against locality disruption attacks using randomness checks of a matrix in CCN. They apply a filtering approach and a statistical sequential analysis (i.e., cumulative sum (CUSUM) algorithm) to detect low-rate attacks. Since the analysis is based on a very simple CCN scenario, the results cannot be extended to a larger CCN topology. Conti et al. [2] introduce a lightweight detection technique for detecting locality-disruption attacks. However, authors do not apply any reaction method for mitigating attacks. Xie et al. [32] introduce a technique, called CacheShield with the goal of improving NDN cache robustness against locality disruption attacks. In CacheShield, when a router receives a

content object, the CS evaluates a shielding function based on a logistic function that determines whether the content object should be cached. The CacheShield must run continuously even when no attack is in progress and store a large amount of statistics at each router that may reduce the space available to cache content. Paper [2] shows that CacheShield is ineffective against some pollution attacks and introduces new attacks specific to CacheShield. Ghali et al. [34] propose a ranking algorithm for cached content that allows routers to probabilistically distinguish good and bad content. This ranking is based on statistics collected from consumers' actions following delivery of content objects. Authors evaluate the performance of their ranking algorithm with inactive adversaries. They also assume that any fake content has a valid version till the proposed algorithm detects fake versions. The ranking algorithm must store several versions of the same content to detect a valid version, and therefore consume routers' storage and computing resources such as FIB for returning back the different possible versions of a same content.

4.2 An ANFIS-based cache replacement method for mitigating cache pollution attacks

In this section, we design and construct the material of proposed ANFIS-based cache replacement method for mitigating cache pollution attacks in NDN. Afterwards, we use simulation to evaluate the effectiveness of the proposed method in two considered NDN topologies. The proposed ANFIS-based cache replacement architecture is depicted in Fig. 4-1. The detail of the method is proposed as follows.

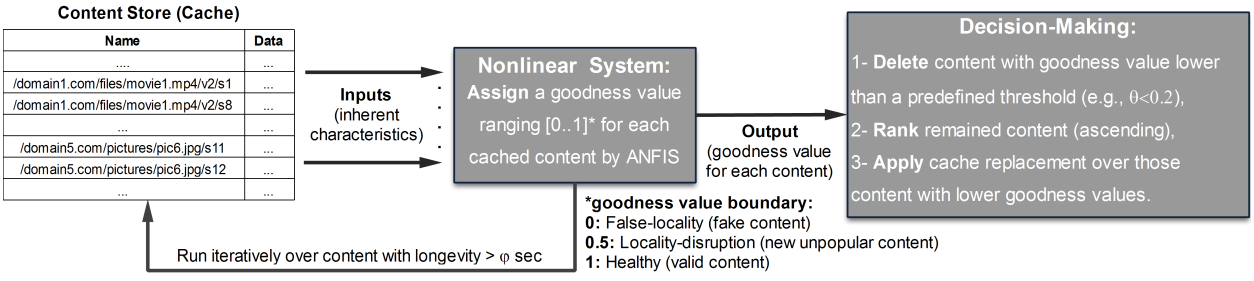


Figure 4-1: Schematic of the proposed ANFIS-based cache replacement method in NDN

4.2.1 Data preparation

The detection of cache pollution attacks is hard because all requested content are uncorrupted. The traditional detection methods usually observe and learn the legitimate users' traffic patterns and detect attacks and anomalies when such patterns change. To address this challenge, we analyze the inherent characteristics of cache pollution attacks and design a nonlinear approximation function through ANFIS to detect locality-disruption and false-locality attacks separately. In order to formulate the problem and construct an ideal method regarding the relationship between inputs (i.e., inherent characteristics of cached content) and output (i.e., content type) data, we define a set of parameters that govern the proposed ANFIS-based cache replacement method. We extract the input parameters based on published research articles such as [34, 147, 148], our observation during the design, and experts' opinion. The considered input parameters are defined as follows:

1. The cached content's longevity (*Longevity*). This corresponds to the time that content has remained in the cache between the time of content being cached and the current time.

2. The cached content's frequency access (*Frequency*). This corresponds to the estimation of content's access frequency. An Exponentially Weighted Moving Average (EWMA) method is employed as a filter to obtain a recent estimate of the access frequency rate. It can also identify the possible aberrant behavior of content's access frequency. EWMA applies weighting factors which decrease exponentially. The weighting for each older data decreases exponentially, giving much more importance to recent observations while still not discarding older observations entirely [149]. The degree of weighting decrease is expressed as a constant smoothing factor β, a number between 0 and 1. EWMA formula is defined as:

$$\overline{\mu}_n = \beta\overline{\mu}_{n-1} + (1-\beta)x_n \tag{4.1}$$

   where $\overline{\mu}_n$ is the exponentially weighted moving average of the past measurements and x_n is the number of content's access frequency in the n-th time interval. We apply the six recent time intervals (i.e., each 0.25 second) to calculate EWMA efficiently.

3. The Standard Deviation of content's access frequency in recent six time intervals (*Std.*). This parameter allows to distinguish the type of content request distribution. The Std. of an uniform distribution is (close to) zero, while other types of distribution such as normal and skewed are not (close to) zero.

4. The last access to the content (*Last Retrieval*). It corresponds to the time interval between the last time of content being used and the current time.

5. The percentage of cache hit (*Hit Ratio*). It corresponds to the content cache hit vs. the total content cache hit in the recent time interval.

6. The variance of an entire population of repeated requests for a same content from the local interfaces (*Interface turnout*). The variance allows to detect distributed cache pollution attacks, when all local interfaces return the same content continuously. If all local interfaces return the similar rate of content, the variance is close to zero.

The output of each data pattern is a goodness value which determines the type of content ranging [0..1]. The boundary of assigned goodness value is defined as: 0 (false-locality or fake content), 0.5 (locality-disruption or new unpopular content), and 1 (healthy). We apply EWMA criterion in the last three time intervals to calculate the average goodness value for each content over a period of time.

According to Fig. 4-1, the applied ANFIS model is automatically executed after every time interval (we set it to 1 second) over cached content with longevity more than a threshold (we set 0.25 second) in order to rank all content based on the goodness value. The initial goodness value for new incoming content with the longevity less than the threshold is set to one (healthy content). After running ANFIS, each content gets a goodness value from the healthy (good) to fake (bad). Those content with the goodness value less than a predefined threshold (we set $\theta < 0.2$) are removed from the cache due to their fake type. This allows to possible valid (healthy) content be replaced with the fake version. Then, the remaining content is sorted in ascending order based on the goodness value for cache replacement when a new content enters and the cache has not enough space for storing. Thus, the proposed method can efficiently and accurately mitigate the false-locality (by removing the fake content) and the locality-disruption (by removing those content with the lower goodness value for cache replacement) attacks in a timely manner.

4.2.2 Materials of ANFIS

During the training process, ANFIS tries to minimize the training error between the target output (i.e., the type of cached content) and the actual output of the ANFIS. The input-output data samples are collected based on the section 4.2.1. In particular, we set cache size to infinite to not to apply any cache replacement algorithm during the training process (see section 4.3).

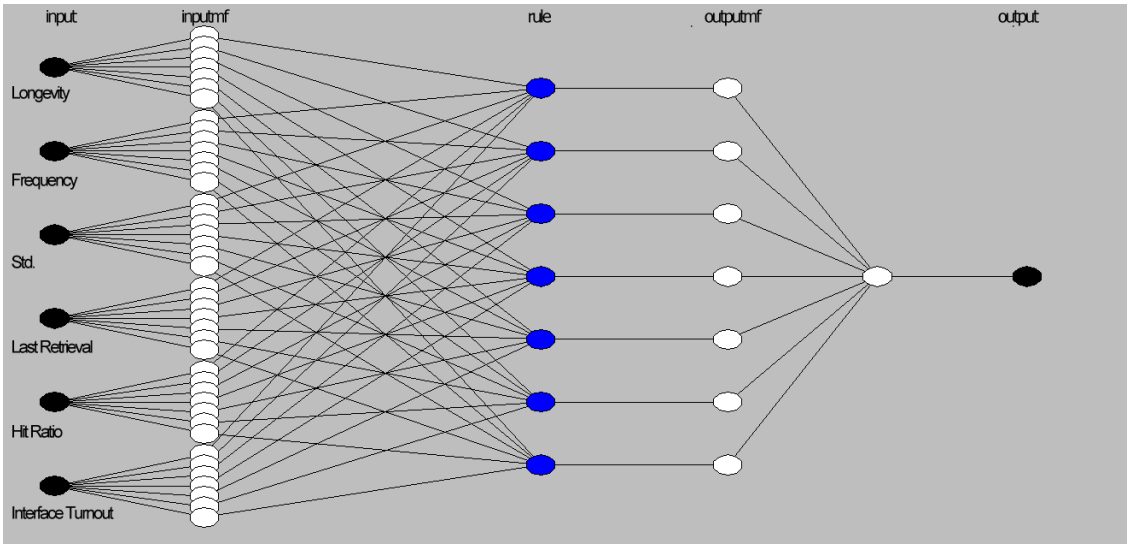


Figure 4-2: The structure of the proposed ANFIS

In this work, ANFIS is established in MATLAB environment. The ANFIS has 6 inputs and one output. Before the training process, data samples should be normalized into [0..1], when dealing with parameters of different units and scales [129, 130]. All variables of ANFIS have "gaussmf" membership function. The gaussmf is a kind of smooth membership functions, so the resulting model has a high accuracy [150]. The "and", "or" and "defuzzification" methods in ANFIS are selected as "product", "max" and "center of gravity", respectively. Fig. 4-2 shows the structure of the constructed ANFIS model as well as the number of fuzzy if-then rules. Before training process, the ANFIS structure is initialized by the fuzzy c-mean method through the input-output data, and its parameters are optimized by least squares and gradient descent algorithms.

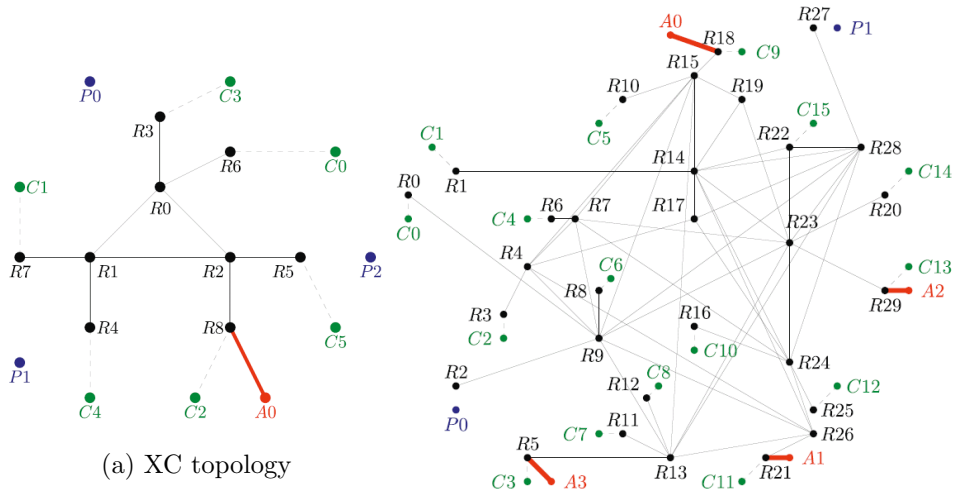
4.3 Experimental setup

This section describes the considered network topologies, simulation environment, followed by the modeling of cache pollution attack strategies.

4.3.1 Simulation environment

We evaluate cache pollution attacks and countermeasures discussed in this section via simulations. We rely on open-source ndnSIM [53] package, a module for ns-3 developed at UCLA as part of the NDN project. The ANFIS-based cache replacement method was firstly implemented with MATLAB on an Intel Pentium 4 3.0 GHz CPU, 4 GB RAM running Windows 7 Ultimate. Then,

it was compiled as a C++ shared library using the MATLAB compiler in order to integrated it with the ndnSIM environment.



(a) XC topology

(b) DFN topology

Figure 4-3: Considered network topologies [2]

The Experiments are performed over two topologies, as illustrated in Fig. 4-3: Xie-complex (XC) and the German Research Network (DFN). The XC and DFN topologies have been identified in previous works as meaningful topologies for simulation [2, 151]. There are several commonly used symbols to identify the type of nodes in NDN networks (such as Fig. 4-3), including `Cx`, `Px`, `Rx`, and `Ax` to represent `x`-th consumer, producer, router and adversary nodes, respectively [2, 40]. In our configurations, we set nodes' PIT size to [500..800] entries randomly. The Interest expiration time was also set to the default timeout of 4000 ms. We set the link delay and queue length parameters to fixed values for every node. In particular, we set delay and queue length to 10 ms and 500, respectively. The requests of regular consumers (we call them *honest consumers*) follow a three types of pattern: Zipf-like, exponentially and batch (i.e., generating a specified number of Interests at specified points of simulation) distributions [53]. We also configure the pattern frequency of Interest packets ranging [100..800], where each honest consumer changes five times the frequency randomly. We apply randomly two different replacement policies in PIT table (i.e., perform different actions when limit on number of PIT entries is reached) including LRU and persistent policies. The nodes' cache capacity was randomly set to [100..400] content. We set the low and medium size for cache capacity to evaluate the accuracy and robustness of the proposed ANFIS-based method sufficiently.

The simulation runs over two and a half hours. The collected input-output data pattern during the

simulation is divided into three blocks as training (70%) to fit (train) the ANFIS model, the first testing (15%) and the second testing (remaining last 15%) to confirm the ANFIS accuracy after training. During the simulation, honest consumers request content based on the above-mentioned configurations. False-locality (see section 4.3.2) and locality-disruption (see section 4.3.3) attacks are issued by adversaries between 0s-30s and 50s-80s, respectively. Finally, the last thirty seconds between 100s-130s, adversaries launch both attacks at the same time to measure the effect of the proposed ANFIS-based cache replacement method in the simultaneous presence of the both attacks.

4.3.2 False-locality

We consider two types of content poisoning implementation: proactive and active attacks. Firstly, we consider a proactive content poisoning attack whereby adversaries anticipated a set of Interest packets for a set of valid content. Adversaries inject fake content into router caches. Assuming a consumer sends an Interest packet which is received by an intermediate router (R_i) and an entry is added to its PIT table. When a router or producer satisfies the Interest packet with a fake content and returns back to the R_i, all the intermediate nodes in the way back as well as R_i are polluted with a fake content. A range of honest consumers are not satisfied if an Interest returns a fake content. After receiving a fake content, they always send the same Interest packet until receive valid content. A range of adversaries behave in the opposite manner. They always ask for bogus content. The adversaries request content according to the uniform distribution. Secondly, we consider an active content poisoning attack whereby adversaries ask some fake content during the simulation run.

We use simulation to measure how many honest consumers can retrieve healthy (valid) content and how fast they can do so when the router caches are poisoned. For proactive scenarios, all routers are pre-populated with different rate of fake content objects, 50%, 80%, and 95% of all the content when the simulation runs. In active scenarios, adversaries request a series of fake content in which intermediate routers are populated with fake content ranging 30%, 50%, and 70% of all the content. We demonstrate that the proposed scheme outperforms the most common policies as Least Frequently Used (LFU) and Least Recently Used (LRU) algorithms in terms of applied performance criteria.

4.3.3 Locality-disruption

We assume that the adversaries can predict Interest packets from a set of honest consumers to issue Interests for attack purposes. The adversaries can issue Interest packets with 5%, 50%, and 90% of the total number of Interest packets issued by honest consumers according to the uniform distribution. This allows to explore the effects of low, moderate and high attacks, and whether the proposed ANFIS-based countermeasure is able to identify and mitigate them.
To summarize multiple statistics in the absence and presence of attacks, we define the metric similar to Deng et al. [138] as the key measure of the effectiveness of the attack as:

$$Hit\ damage\ ratio = 1 - \frac{HR(non-attack) - HR(attack)}{HR(non-attack)} \tag{4.2}$$

Where, $HR(non-attack)$ and $HR(attack)$ denote the hit ratio of honest consumers in the absence/presence of an attack, respectively. When the *Hit damage ratio* is (close to) zero, the attack is completely ineffective, while it is (close to) one, the caching feature is completely under attack. We then demonstrate that the proposed scheme outperforms the most common policies as LFU, LRU independently and in conjunction with CacheShield [32] in terms of applied performance criteria. Xie et al. in [32] introduce CacheShield, a method to shield NDN routers from locality disruption attacks.

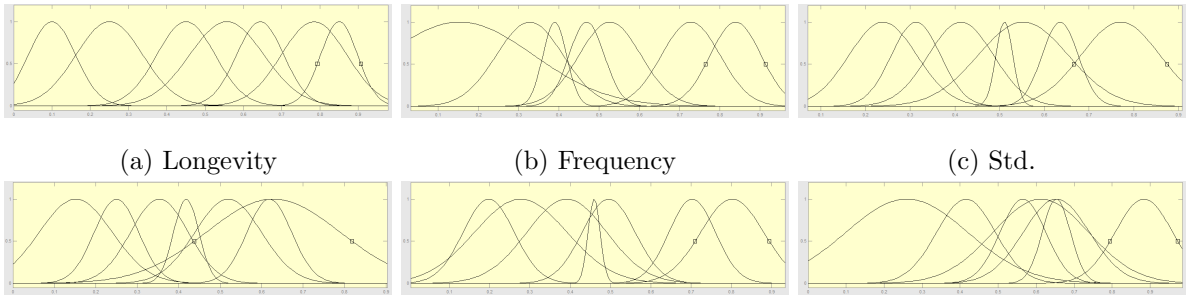
4.4 Experimental results

In this section, we demonstrate through simulations that the proposed ANFIS-based cache replacement method satisfies in a much better way the applied performance criteria as compared to the preexisting methods. Our countermeasure is tested over the two considered topologies in Fig. 4-3. Each router implements the proposed ANFIS-based technique discussed in Section 4.2.

4.4.1 Results of ANFIS design

The training data used for constructing ANFIS model is the obtained statistical data (see section 4.3) from DFN topology. The constructed ANFIS model is used as a cache replacement method over both XC and DFN topologies in order to test its performance and robustness against cache pollution attacks.

Based on the hybrid training process in ANFIS through the number of constructed cluster



(a) Longevity (b) Frequency (c) Std.

(d) Last retrieval (e) Hit ratio (f) Interface Turnout

Figure 4-4: Final membership functions of the input data

centers by fuzzy c-mean clustering, there are seven fuzzy rules. The number of training epochs is 500 and the error tolerance was set to the default value, which is zero. The initial step-size and the increase and decrease rates were set to 0.01, 0.8, and 1.2, respectively. These configuration settings in our application are set up to cover a wide range of learning tasks, which lead to optimization of the training process. Fig. 4-4 illustrates the final membership functions of input data. To show the efficiency of the model, different performance metrics are applied including Mean Square Error (MSE), Mean Absolute Error (MAE), Root Mean Square Error (RMSE), Standard Deviation of the error (Std.), and Quantile-Quantile plot (Q-Q plot) followed by Pearson and Kendall tau_b correlation coefficient divided to training and testing data sets. Numerical results are shown in Figs. 4-5-4-8. The plots demonstrate the correspondence between the real values (content type) and corresponding output values predicted by the ANFIS model, indicating that the ANFIS model we have developed is accurate.

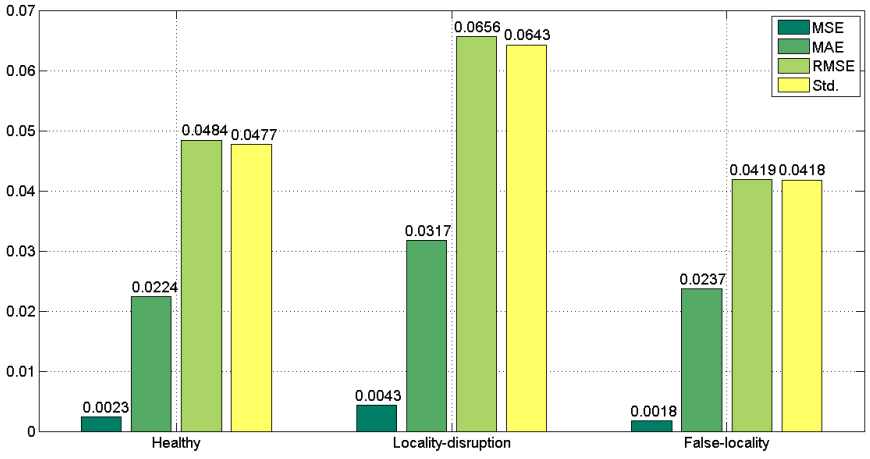


Figure 4-5: The statistical results on training data set

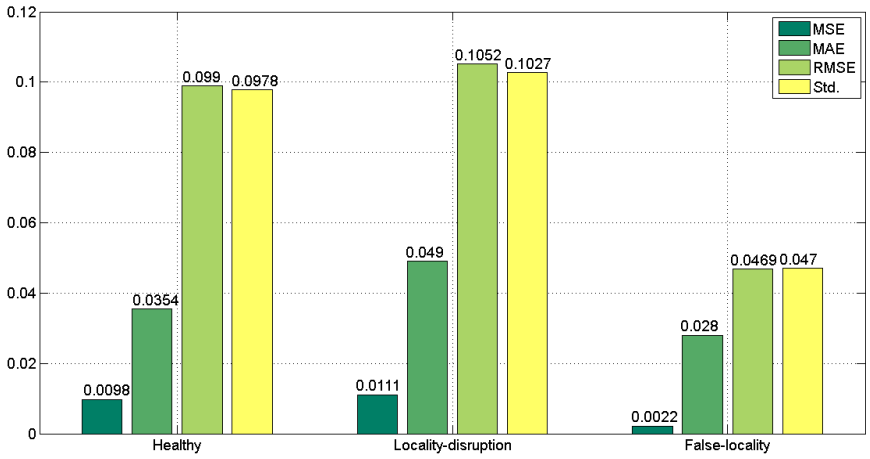


Figure 4-6: The statistical results on 1st testing data set

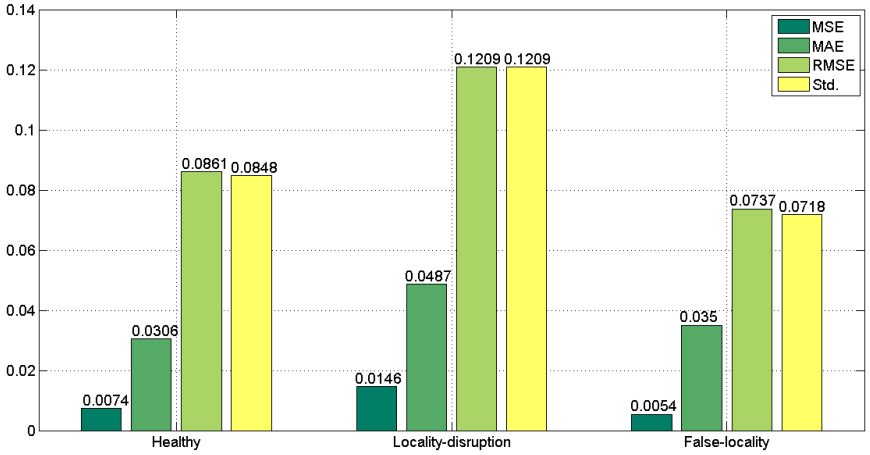
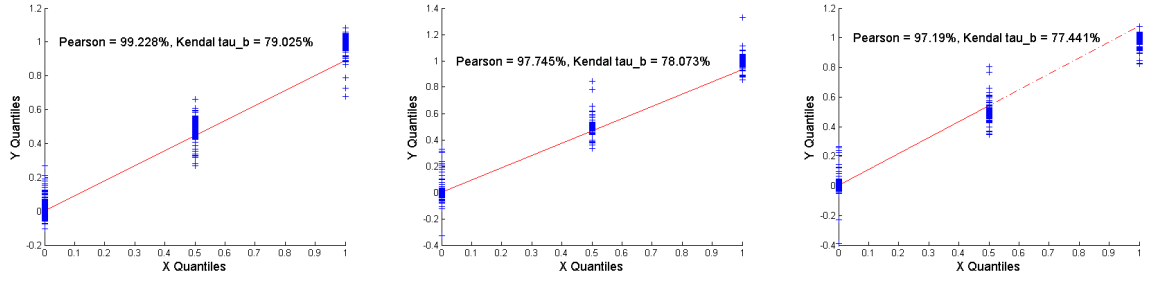


Figure 4-7: The statistical results on 2nd testing data set



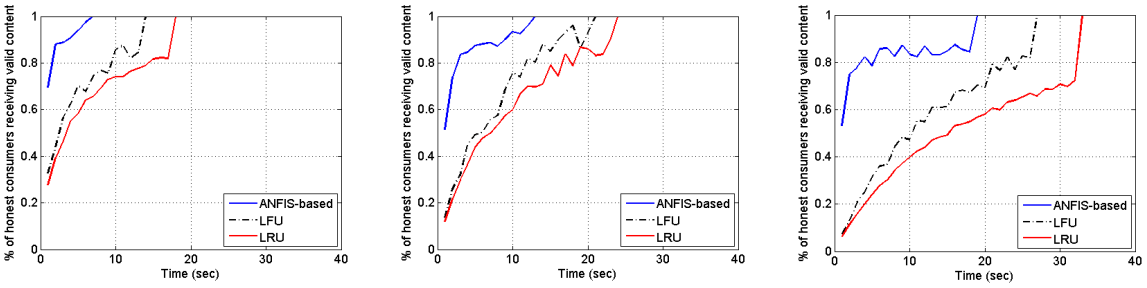
(a) Numerical results on training set (b) Numerical results on 1st testing set (c) Numerical results on 2nd testing set

Figure 4-8: Q-Q plot and statistical results

4.4.2 Mitigating false-locality

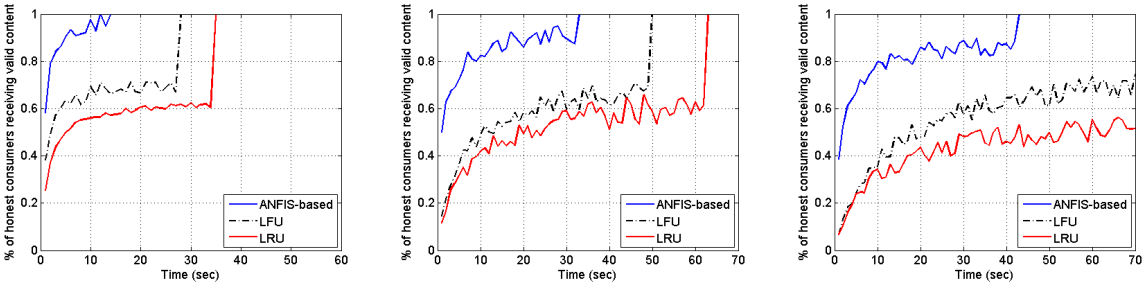
We first evaluate the effectiveness of proposed ANFIS-based cache replacement method in a simple network topology using the XC network. Fig. 4-9 illustrates the average behavior of three methods with different pre-populated fake content rate within 10 runs. The proposed ANFIS-based cache replacement method is more accurate and outperforms other methods in terms of the faster full convergence.

After verifying the correct behavior of ANFIS-based cache replacement method in the XC topology, we consider a more complex network topology using DFN network in Fig. 4-10.



(a) 50% pre-populated (b) 80% pre-populated (c) 95% pre-populated

Figure 4-9: Results of different pre-populated fake content in XC topology (mean of 10 runs)



(a) 50% pre-populated (b) 80% pre-populated (c) 95% pre-populated

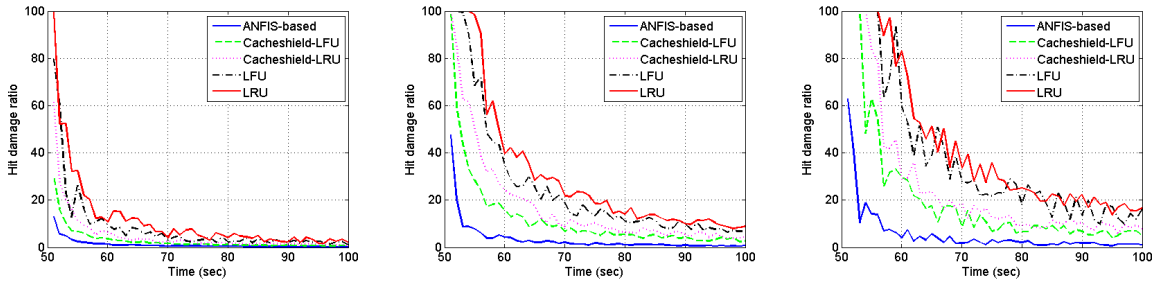
Figure 4-10: Results of different pre-populated fake content in DFN topology (mean of 10 runs)

Fig. 4-10 shows the average of experimental results by ANFIS-based, LRU, and LFU cache replacement methods within 10 runs. As shown in this figure, there is a considerable benefits of the proposed countermeasure implemented by ANFIS model in faster full convergence of the honest consumers. With increased rate of pre-populated fake content, the LRU and LFU methods perform an insignificant behavior in removing the fake content from the caches. Whereas, the proposed ANFIS-based method performs more accurate and efficient in removing the fake content

from the caches and satisfies all the honest consumers in a timely manner.

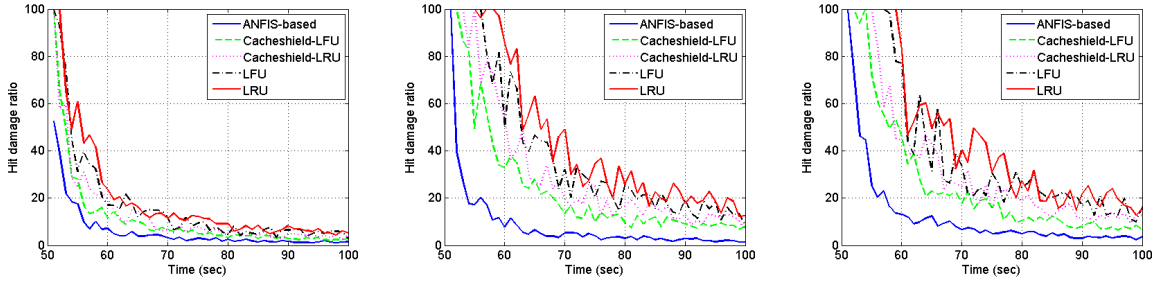
4.4.3 Mitigating locality-disruption

Simulation results in Figs. 4-11 and 4-12 show that our cache replacement technique can quickly detect the content placed with the goal of performing locality-disruption attacks and replace them when a new content is added to a full cache. These figures show that routers using ANFIS-based cache replacement method successfully outperforms four applied cache replacement algorithms in a timely manner. The most stunning result is the extreme vulnerability of the LRU and the LFU to pollution attacks.



(a) 5% locality-disruption (b) 50% locality-disruption (c) 90% locality-disruption

Figure 4-11: Results of Hit damage ratio for locality-disruption attack in XC topology (mean of 10 runs)

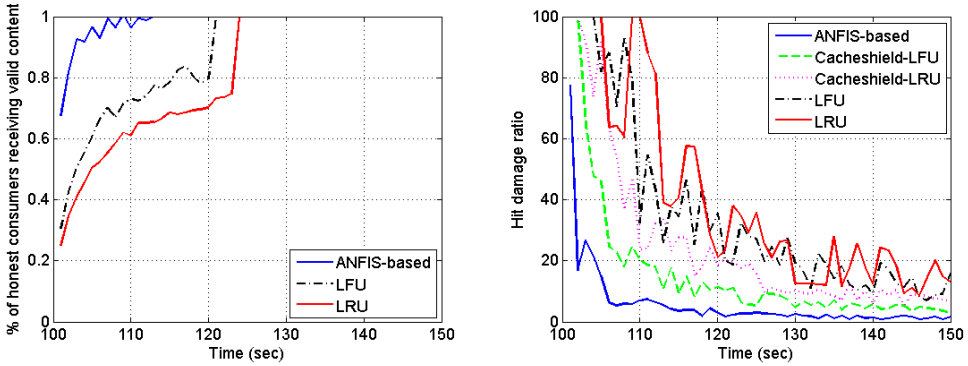


(a) 5% locality-disruption (b) 50% locality-disruption (c) 90% locality-disruption

Figure 4-12: Results of Hit damage ratio for locality-disruption attack in DFN topology (mean of 10 runs)

The experimental results in Figs. 4-11a-4-11c and 4-12a-4-12c indicate that the ANFIS-based cache replacement technique is more resilient than the preexisting methods against locality-disruption attacks. Despite the fact that the hit damage ratio is still quite high by ANFIS-based technique in the early times of the simulation, the application of the ANFIS-based technique is quite effective

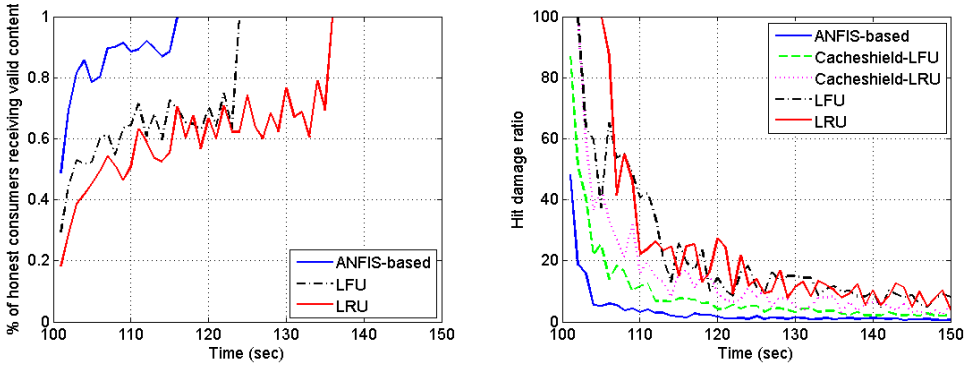
and more reliable against low, middle, and high rate pollution attacks.



(a) Receiving valid content by 30% false-locality

(b) The hit damage ratio by 70% locality-disruption

Figure 4-13: The results for 30% false-locality and 70% locality-disruption in XC topology (mean of 10 runs)



(a) Receiving valid content by 50% false-locality

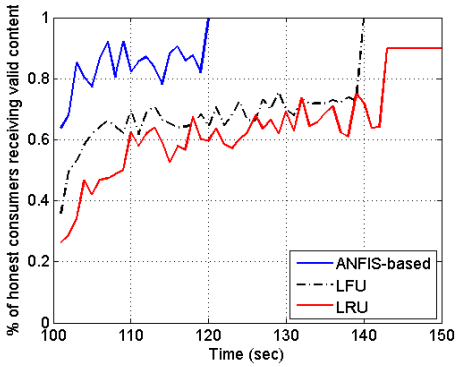
(b) The hit damage ratio by 50% locality-disruption

Figure 4-14: The results for 50% false-locality and 50% locality-disruption in XC topology (mean of 10 runs)

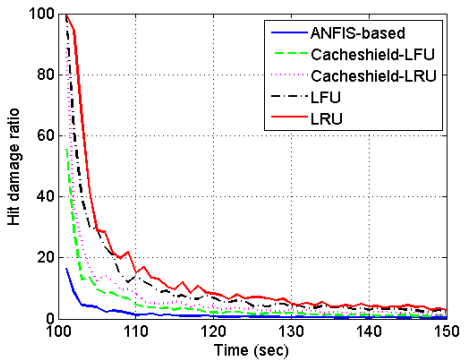
4.4.4 Mitigating combination of both attacks at the same time

Adversaries can launch both false-locality and locality-disruption attacks at the same time. For instance, the same set of attackers can launch false-locality attacks by pre-populating 50% of the total honest consumers' Interest requests, and at the same time they start locality-disruption attacks to interfere the content locality by requesting the rest 50% of the honest consumers' Interest requests in the caches.

According to the proposed ANFIS-based cache replacement method discussed in section 4.2, the

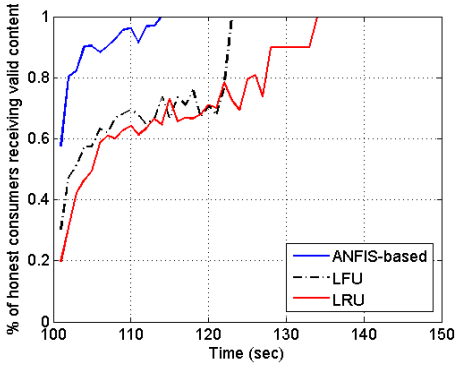


(a) Receiving valid content by 70% false-locality

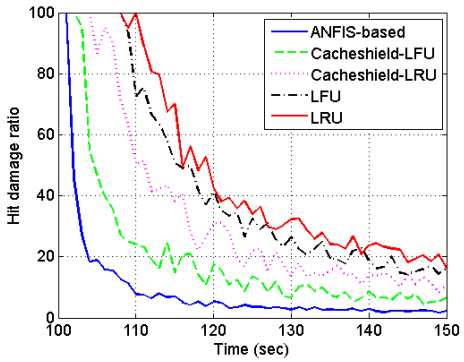


(b) The hit damage ratio by 30% locality-disruption

Figure 4-15: The results for 70% false-locality and 30% locality-disruption in XC topology (mean of 10 runs)



(a) Receiving valid content by 30% false-locality

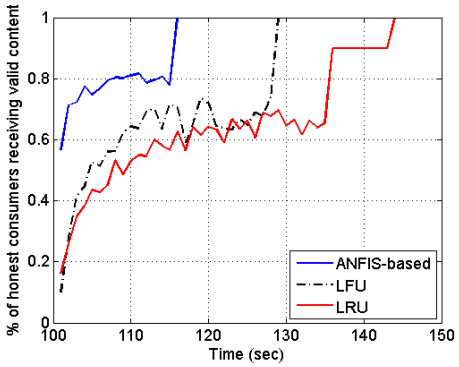


(b) The hit damage ratio by 70% locality-disruption

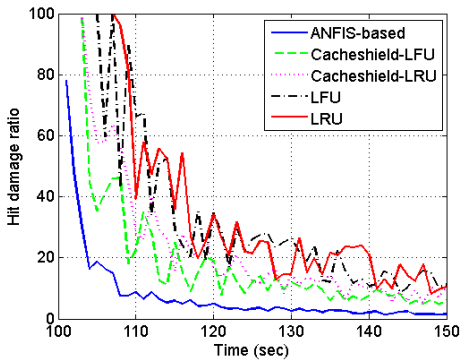
Figure 4-16: The results for 30% false-locality and 70% locality-disruption in DFN topology (mean of 10 runs)

existence of locality-disruption attacks will not affect the detection of false-locality attacks and vise versa. First, the proposed method tries to detect false-locality attacks by assigning goodness value close to zero, and once detected, they are removed from the caches. Then those content with the goodness value close to 0.5, detected as locality-disruption attacks, would be replaced separately when a new content enters and cache space is full.

We vary the behavior of attackers for executing false-locality and locality-disruption attacks between 100 and 130 seconds of the simulation run. Figs. 4-13-4-15 and 4-16-4-18 are shown the results of mitigating both cache pollution attacks at the same time with different strategies in XC and DFN topologies, respectively. Experimental results demonstrate that our proposed method is more resilient and more accurate than preexisting methods to the mixture of attacks. The most stunning

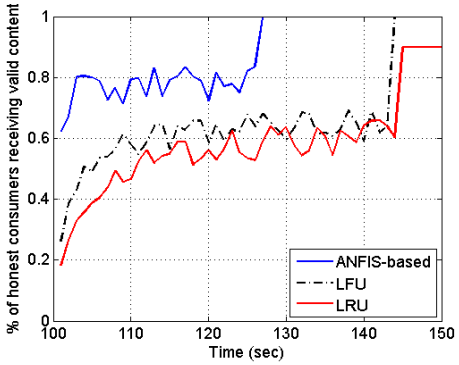


(a) Receiving valid content by 50% false-locality

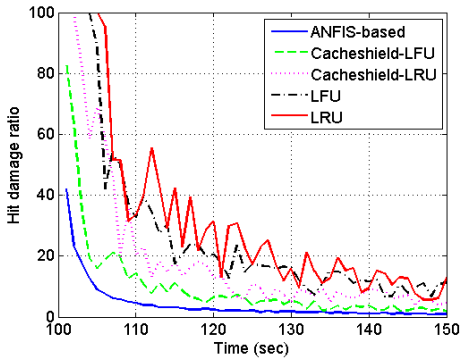


(b) The hit damage ratio by 50% locality-disruption

Figure 4-17: The results for 50% false-locality and 50% locality-disruption in DFN topology (mean of 10 runs)



(a) Receiving valid content by 70% false-locality



(b) The hit damage ratio by 30% locality-disruption

Figure 4-18: The results for 70% false-locality and 30% locality-disruption in DFN topology (mean of 10 runs)

result is the extreme vulnerability of the LRU and LFU algorithms to the active false-locality attacks as compared to the proactive false-locality attacks. Thus, the proposed ANFIS-based cache replacement mechanism in the considered simulation environments offers visibly promising performance in presence of cache pollution attacks.

4.4.5 The overhead cost

In this section, we assess the overhead cost of our proposed method and preexisting schemes in presence of adversaries. In particular, we are interested in determining the overhead of the average number of arrival data packets for legitimate users in routers and the operation overhead of the methods.

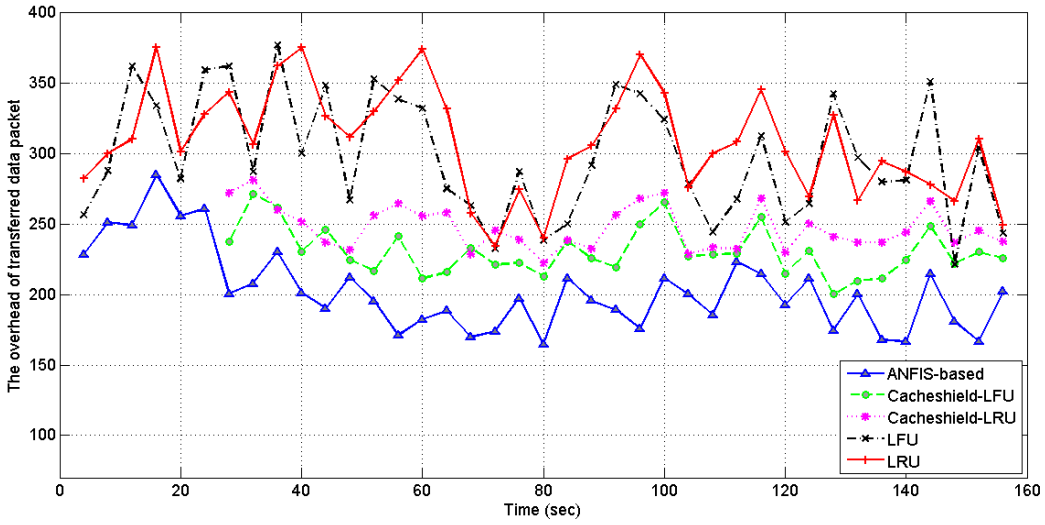


Figure 4-19: The average of arrival data packets in XC topology

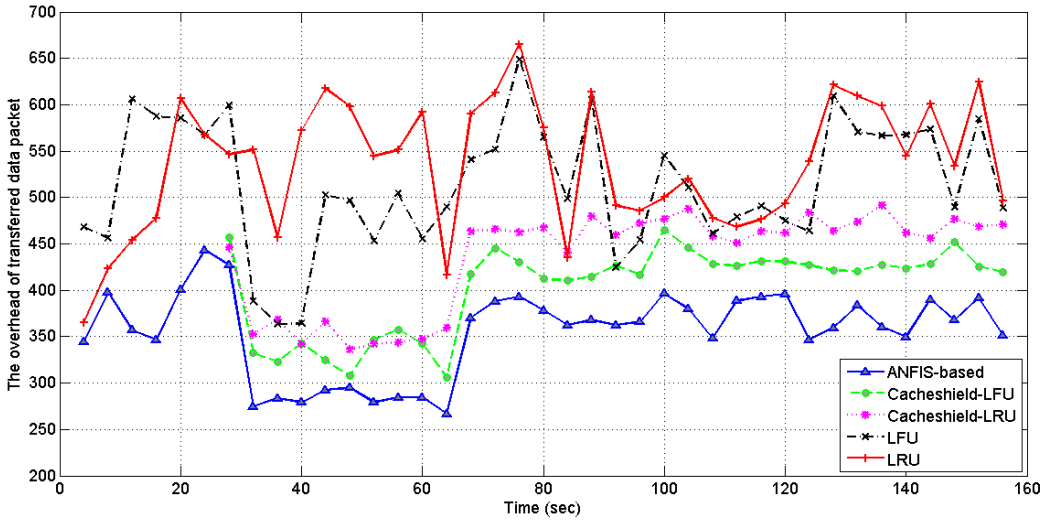


Figure 4-20: The average of arrival data packets in DFN topology

1. The overhead of the average of arrival data packets: It guarantees that this amount of data packet was actually transferred over the channel during the cache pollution attacks. Figs. 4-19 and 4-20 show the average of overhead of transmitted data packets in routers in the XC and DFN networks, respectively. We can observe that the our proposed method outperforms other methods based on the lower overhead of data transmission. Our results confirm that the most data packets were able to cache to the closest edge routers (i.e., close routers to the legitimate consumers) by mitigating effectively both attacks. Our results also show that the overhead of transmitting data

packets by LRU and LFU algorithms are greater than our proposed method and the CacheShield, making the attack more effective.

2. The operation overhead: This is the amount of processing time to execute the caching algorithms within operating system. Table 4.1 shows that the proposed method seems to be less time consuming than the other methods except LRU and LFU algorithms when attacks do not run simultaneously. The results in Table 4.1 indicate that the proposed approach can improve the performance as compared to LRU and LFU algorithms in terms of the operation overhead up to 6.93% and 7.15 %, and 5.15% and 5.78% in XC and DFN topologies respectively, when both cache pollution attacks are simultaneously implemented. According to the obtained results, by increasing rate of attacks, the overhead of our proposed method is considerably decreased as compared to LRU and LFU. The results from Table 4.1 also confirm that the our proposed method outperforms sufficiently the CacheShield-LRU and CacheShield-LFU methods in terms of the operation overhead up to 14.56% and 16.79%, and 21.67% and 23.14% in XC and DFN, respectively.

Table 4.1: Comparing operation overhead achieved by the proposed scheme over other methods (mean of 10 runs)

Time (sec)	Attack	Percent of worsening (↓) and improving (↑) (%)			
		LRU	LFU	CacheShield-LRU	CacheShield-LFU
XC topology:					
0-50 (false-locality attacks)	50%	↓ 8.83	↓ 7.98	-	-
	80%	↓ 7.11	↓ 4.36	-	-
	95%	↓ 5.32	↓ 2.81	-	-
50-100 (locality-disruption attacks)	5%	↓ 8.62	↓ 7.37	↑ 14.41	↑ 14.51
	50%	↓ 8.47	↓ 6.29	↑ 13.11	↑ 14.24
	90%	↓ 6.83	↓ 3.98	↑ 14.28	↑ 15.76
100-150 (combination of both attacks)	30-70%	↑ 2.74	↑ 3.31	↑ 13.18	↑ 16.07
	50-50%	↑ 3.59	↑ 4.64	↑ 14.56	↑ 15.34
	70-30%	↑ 6.93	↑ 7.15	↑ 14.01	↑ 16.79
DFN topology:					
0-50 (false-locality attacks)	50%	↓ 9.52	↓ 7.45	-	-
	80%	↓ 8.17	↓ 6.43	-	-
	95%	↓ 8.03	↓ 5.14	-	-
50-100 (locality-disruption attacks)	5%	↓ 9.21	↓ 9.33	↑ 19.73	↑ 18.33
	50%	↓ 9.01	↓ 7.75	↑ 20.11	↑ 19.03
	90%	↓ 6.83	↓ 7.24	↑ 18.91	↑ 20.76
100-150 (combination of both attacks)	30-70%	↑ 2.42	↑ 3.66	↑ 20.34	↑ 23.02
	50-50%	↑ 3.84	↑ 4.03	↑ 20.13	↑ 22.38
	70-30%	↑ 5.15	↑ 5.78	↑ 21.67	↑ 23.14

To evaluate the effectiveness and efficiency of the proposed method, we illustrate that the proposed ANFIS-based method provides a suitable compromise between overhead (i.e., the overhead of the arrival data packets in Figs. 4-19 and 4-20, and the operation overhead of the algorithms in Table 4.1) and applied performance metrics including the percentage of legitimate consumers

receiving valid content (Figs. 4-9-4-10 and 4-13a-4-18a) and the hit damage ratio (Figs. 4-11-4-12 and 4-13b-4-18b) as compared to common existing countermeasures. Therefore, the extensive analysis satisfies the objectives of the experiment in terms of the applied performance metric and ensure that the proposed ANFIS-based caching for mitigating cache pollution attacks in NDN can yield high accuracy as compared to other methods without very much computational cost.

4.5 Conclusion

In this dissertation, we proposed a novel ANFIS-based cache replacement method to mitigate two generic cache pollution attacks namely false-locality and locality-disruption in NDN. Simulation results showed that the proposed method provides very accurate results as compared to LRU and LFU algorithms independently and in conjunction with CacheShield scheme. Experimental results and analysis show the proposed ANFIS-based cache replacement method is very effective in determining and mitigating the fake content, and has a very high detection rate of locality-disruption attacks to replace them when new content is added to a full cache in a timely manner. The extensive analysis satisfies the objectives of the experiment and ensure that the proposed ANFIS-based caching for mitigating cache pollution attacks can yield high accuracy as compared to other methods without very much computational cost.

Chapter 5

A Hybrid Multiobjective RBF-PSO Method for Mitigating DoS Attacks

In contrast to today's Internet, a key goal of the NDN project is "security by design" [17, 32, 36]. Unlike the current Internet (host-based) approach in which security, integrity and trust should be provided in the communication channel, CCN secures content (information) itself and puts integrity and trust as the content properties [19, 37]. However, with this new paradigm, new kinds of attacks and anomalies -from Denial of Service (DoS) to privacy attacks- will arise [38, 39]. The big question is how resilient will this new NDN architecture be against DoS/DDoS attacks [17, 23]. An adversary can take advantage of two features unique to NDN namely Content Store (CS) and Pending Interest Table (PIT) to mount DoS/DDoS attacks specific to NDN such as Interest flooding attacks and content poisoning [23, 40].

The first goal of any protection scheme against DoS attack is the early detection (ideally long before the destructive traffic build-up) of its existence [40, 152]. In order to disarm DoS/DDoS attacks and any deviation, not only the detection of the malevolent behavior must be achieved, but the network traffic belonging to the attackers should be also blocked [24, 25, 104]. Thus, a predictor (detector) should take an appropriate action to thwart the attacks and should be able to adjust itself to the changing dynamics of the anomalies/attacks [23, 153]. In an attempt to tackle with the new kinds of DoS attacks and the threat of future unknown attacks and anomalies, many researchers have been developing intelligent learning techniques as a significant part of the current research on DoS attacks detection [19, 154]. The most popular approach for DOS/DDoS attacks prediction is using Artificial Neural Networks (ANNs) classification [91, 155, 156]. ANNs have become one of

the most vital and valuable tools in solving many complex practical problems [107, 157], among which the Radial basis function (RBF) neural networks have been successfully applied for solving dynamic system problems, because they can predict the behavior directly from input/output data [158, 159]. RBF networks have many remarkable characteristics, such as simple network structure, strong learning capacity, better approximation capacities and fast learning speed. The difficulty of applying the RBF networks is in network training which should select and estimate properly the input parameters including centers and widths of the basis functions and the neuron connection weights [87, 157, 160]. In order to find the most appropriate parameters, an optimization algorithm can be used [161, 162]. An optimization algorithm will attempt to find an optimal choice that satisfies defined constraints and make an optimization criterion (performance or cost index) maximize or minimize [161]. Hence, to improve the prediction accuracy and robustness of the RBF network, network parameters (centers, widths and weights) should be simultaneously tuned [157]. Some of the existing algorithms to achieve that are given in [157, 160, 163, 164, 165]. Almost all algorithms compute the optimal estimation of the basis function centers by mean of error minimization, i.e., accuracy based on Mean-Square Error (MSE) [160, 165, 166, 167]. However, MSE is not suitable for determining the optimal position of basis function centers. Since the MSE decreases, the number of centers increases [12]. To accomplish this task, we develop our proactive detection algorithm for globally well-separating units' centers and their local optimization by MSE (decreasing the error caused by corresponding data points and their centers, separately). But the optimal placement and well-separated centers can increase MSE [121]. It is generally accepted that well-separated (external separation of) centers and their local optimization (internal homogeneity) are conflicting objectives [12, 168]. This trade-off is a well-known problem as the Multiobjective Optimization Problem (MOP) [150, 169, 170, 171]. This dissertation applies NSGA II (Non-dominated Sorting Genetic Algorithm) proposed by Deb et al. (2002) to solve this problem, as it has recently been frequently applied to various scenarios [172, 173, 174, 175]. On the other hand, for (near) optimal estimation and adjustment of two others RBF parameters (units' widths and output weights), we implement Particle Swarm Optimization (PSO) that favors global and local search of its interacting particles which has proved to be effective in finding the optimum in a search space [117, 118, 176]. When the DoS attacks by the proposed intelligent predictor are identified, the second phase (i.e., adaptive mitigation reaction) is triggered by enforcing explicit limitations against adversaries. The contribution of this work is summarized in three objectives. The first objective of this work is to develop an algorithm to resolve the hybrid learning problem of a RBF network using multiobjective

optimization and particle swarm optimization to obtain a simple and more accurate RBF network-based classifier (predictor). The second objective is utilization of this optimized RBF network-based predictor in proactive detection of the DoS/DDoS attacks in NDN. The third objective is introducing a new algorithm to enable NDN routers to perform quickly and effectively adaptive reaction (recovery) from network problems, in order to keep track of legitimate data delivery performance and effectively shutting down malicious users' traffic.

There are three main advantages of the proposed prediction (classification) method; first, the proposed method can be applied to classification of any real-world problem; second, it gives better results in terms of the low misclassification, accuracy and robustness for some benchmark problems. And third, it provides a promising performance in prediction of DoS attacks in NDN. Moreover, the evaluation through simulations shows that the proposed intelligent hybrid algorithm (proactive detection and adaptive reaction) can quickly and effectively respond and mitigate DoS attacks in adverse conditions in terms of the applied performance criteria.

5.1 DoS attacks in NDN

The new variations of DoS attacks might be quite effective against NDN. An adversary can take advantage of two features unique in NDN routers as CS and PIT to mount DoS/DDoS attacks into NDN. There are two major categories of DoS attacks in NDN infrastructure [23, 31]:

1. *Interest Flooding Attack (IFA)*: It is partly due to the lack of authentication of Interest packets (source). Anyone can generate Interests packets and any middle router (node) only knows that a particular Interest packet entered on a specific interface.

2. *Content/Cache Poisoning*: The adversary tries to make routers forward and cache corrupted or fake data packets in order to prevent consumers from retrieving the original (legitimate) content.

5.1.1 Interest Flooding Attack

In this type of attack, the adversary (controlling a set of possibly geographically distributed zombies) generates a large number of Interest packets aiming to (1) overwhelm PIT table in routers in order to prevent legitimate users to satisfied their Interest packets and (2) swamp the target content producers [23]. There are three types of Interest flooding attacks, based on the type of content requested:

1. *existing or static*: it is quiet limited since in-network content caching provides a built-in countermeasure. If several zombies from different paths generate large number of Interest packets for an existing content which settles in all intervening routes' caches, these Interest packets for the same content can not propagate to the producer(s) since they are satisfied by cached copies.

2. *dynamically-generated*: There is no benefits via caching copies. Since requested content is dynamic, all Interest packets are routed to content producer(s), thus consuming bandwidth and router PIT table. Also, content producer might waste considerable computational resources due to the signing the content (per-packet operation) which is itself expensive.

3. *non-existent (unsatisfied Interests)*: Such Interest packets cannot be collapsed by routers, and are routed toward the content producer(s). This type of Interest packets take up space in router PIT table until they expire. A large number of non-existent Interest packets in PIT table lead to legitimate Interest packets being dropped in the network.

5.2 Related Work

As a new Internet architecture proposal, there is very limited work recently regarding to mitigation of DoS/DDoS attacks in Named Data Networking. Gasti et al. [23] performed initial analysis of NDN's resilience to DoS attacks. This work identifies two new types of attacks specific to NDN (Interest flooding and content/cache poisoning) and discusses effects and potential countermeasures. However, the paper does not analyze DoS attacks and their countermeasures. Afanasyev et al. [17] presented three mitigation algorithms (token bucket with per interface fairness, satisfaction-based Interest acceptance and satisfaction-based pushback) that allow routers to exploit their state information to thwart Interest flooding attacks. Among these three mitigation algorithms, satisfaction-based pushback mechanism could effectively shut down malicious users while preventing legitimate users from service degradation. This work uses a simple and static attackers model (sending junk Interests as fast as possible), and it does not consider intermediate router's cache and always forwards all the way to the producer. Compagno et al. [40] introduced a framework for local and distributed Interest flooding attack mitigation, in particular, rapid generation of large numbers of Interest for non-existent contents that saturate the victim router's PIT. Authors simulated a simple attackers model, and their countermeasure has been able to use around 80-90% of the available bandwidth in the most cases during the attacks. Dai et al. [177] proposed Interest

traceback as a counter measure against NDN DDoS attacks, which traces back to the originator of the attacking Interest packets. In this paper, when PIT exceeds its threshold, Interest traceback is triggered. This method responds to the attack by generating spoofed Data packets to satisfy the long-unsatisfied Interest packets in the PIT by tracing back to the Interest originators. This method is not proactive, makes overhead in the network by increasing of made spoofed contents. It leads to middle routers cache bogus contents. This paper also assumes that the long-unsatisfied Interests in the PIT is adversary and others unsatisfied Interest are normal usages. Another shortcoming of this method is that the router drops the incoming packet rate of the interface which has too many long-unsatisfied Interest packets. As a result of this independent decision, the probability of legitimate Interests being forwarded decreases rapidly as the number of hops between the content requester and producer. Choi et al. [38] provided an overview of threats of Interest flooding attacks for non-existent contents on NDN. Authors simulated and explained the effect of Interest flooding DoS attacks by a simple scenario over the quality of services for legitimate Interest packets from normal users due to PIT full. However, they do not analyze DoS attacks and their countermeasures.

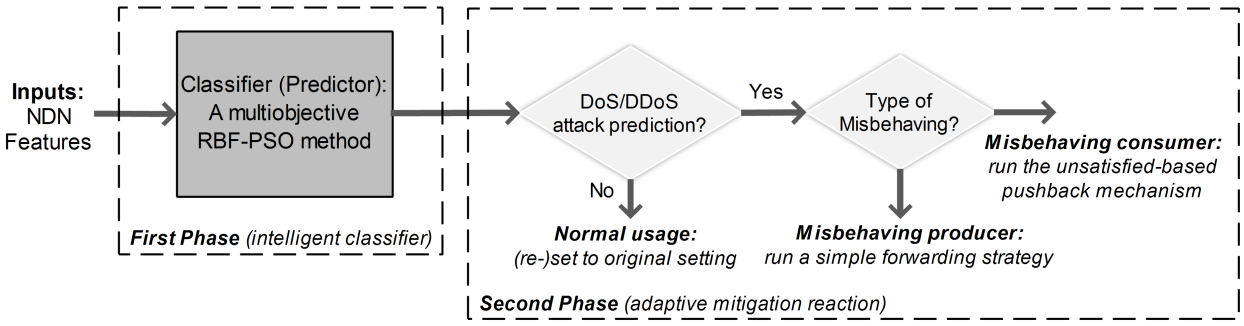


Figure 5-1: The overview of the proposed DoS mitigation method in NDN

5.3 The proposed hybrid intelligent method

In this section, we introduce our method, a two-phase framework for mitigating DoS attacks in NDN. The first phase being proactive detection (see section 5.3.1) and the second one adaptive reaction (see section 5.6.2). The proposed predictor in the first phase is a global framework so that we can use the predictor in other networks. In this dissertation, we apply the proposed predictor successfully on some benchmark problems and NDN and leave further investigations in other networks to future work. A diagram of the two phases of the proposed method is shown in Fig. 5-1.

5.3.1 The proposed intelligent classifier (predictor)

This section presents the details of proposed intelligent algorithm for classification problems. Our approach composes of two main phases. It is depicted in Fig. 5-2. Each phase is given in the next subsections.

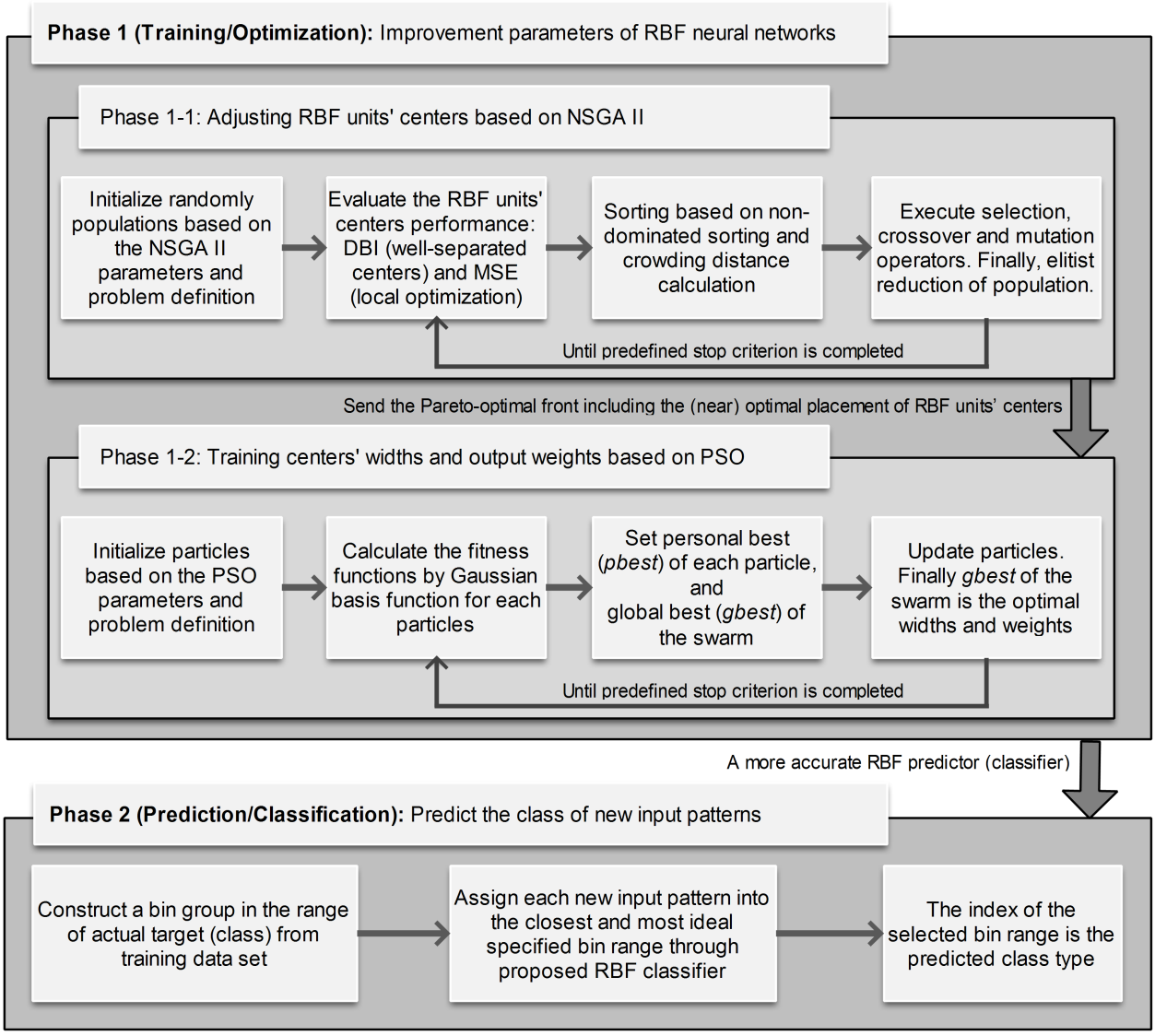


Figure 5-2: Proposed intelligent algorithm for more accurate classification

Phase 1: Improvement of RBF parameters

In the first phase -training (optimization)- we introduce a new hybrid optimization approach for designing RBF neural networks which can be implemented for real-world problems. Firstly, a new

multiobjective optimization algorithm as NSGA II for adjusting centers of the RBF units is introduced. This algorithm obtains various non-dominated sets that provide an appropriate balance between two conflicting objectives: well-separated and local optimization of RBF centers. Secondly, PSO algorithm has been applied to simultaneously tune widths of the RBF units and output weights through well-placed centers. The algorithm is presented below:

A. First part (ADJUSTING RBF UNITS' CENTERS BASED ON NSGA II):

1. Problem definition:

   1-1- population size (N), maximum iteration ($Iter_{Max}$), crossover percentage ($pCrossover$), number of parents (offspring) after crossover operator ($nCrossover = 2\times round(pCrossover\times \frac{N}{2})$), mutation percentage ($pMutation$), number of mutants after mutation ($nMutation = round(pMutation \times N)$), mutation rate ($mu$), mutation step size ($sigma = 0.1$).

2. Initialize population:

   2-1- Generate the initial populations (individuals) P, including $P_1, P_2, ..., P_N$.

   2-2- Calculate the two conflicting cost functions as DBI and MSE (presented in section 5.3.1) for each population.

   2-3- Rank all populations according to their non-dominance.

   2-4- Calculate the crowding distances for all populations to keep the population diversity (Eq. 2.21).

   2-5- Sort the non-dominated solutions in descending crowding distance and rank values.

3. NSGA II main loop:

   3-1- Execute the evolution process including crossover and mutation operators:

   a. Execute crossover operator, $PopCrossover$ (we adopt the two-point crossover).

   b. Execute mutation operator, $PopMutation$ (A Gaussian distributed random number with mean zero and variance 1 is used [178, 179]).

   c. Merge populations:

   $P = [P \;\; PopCrossover \;\; PopMutation]$.

   3-2- Run steps 2-3 (rank), 2-4 (crowding distance) and 2-5 (sort) over the merged P.

   3-3- Truncate/Select the generated population P to the range of population size: $P = P(1 : N)$.

   3-4- Run steps 2-3 (rank), 2-4 (crowding distance) and 2-5 (sort) over the truncated P.

3-5- Store Pareto-optimal front (non-dominated set) in the archive as $PF1$.

3-6- Repeat Step 3 until termination condition ($Iter_{Max}$) is reached.

3-7- Keep the final $PF1$ including the (near) optimal placement of RBF units' centers.

B. Second part (CALCULATING WIDTHS OF THE RBF UNITS AND OUTPUT WEIGHTS BASED ON PSO ALGORITHM):

1. Problem definition:

   1-1- population size (N), maximum iteration ($Iter_{Max}$) and number of RBF Kernel obtained from $PF1$ in phase A ($nKernel$).

   1-2- Upper and lower bound of width (σ) and weight (w) variables.

   1-3- Adjust the PSO parameters: inertia weight (ω) which is linearly decrease by Eq. 5.1, acceleration coefficients ($c1 = c2 = 2$), and two random numbers ($r1$ and $r2$) which distributed uniformly in [0 1].

   $$\omega = \omega_{max} - t \cdot \frac{(\omega_{max} - \omega_{min})}{T} \tag{5.1}$$

   Where $\omega_{max}, \omega_{min}, T$ and t denote the maximum inertia weight, the minimum inertia weight, the total and the current number of iterations, respectively.

2. Initialize population for each particle:

   2-1- Generate the initial populations (particle positions) including:

   $Particle(i)$.Position.σ and $Particle(i)$.Position.w. $i = 1, 2, ..., N$.

   $particle(i)$.Position.σ = Continuous uniform random numbers between $\sigma.Lower$ and $\sigma.Upper$ in size of $nKernel$.

   $particle(i)$.Position.w = Continuous uniform random numbers between $w.Lower$ and $w.Upper$ in size of $nKernel$.

   2-2- Initialize velocity vectors in a feasible space for each particle:

   $particle(i)$.Velocity.σ = a $nKernal$ size zero matrix.

   $particle(i)$.Velocity.w = a $nKernal$ size zero matrix.

   2-3- Evaluate each particle by Gaussian basis function in each RBF units (Eq. 2.11). Calculate Gaussian basis function with two tuned parameters (σ -centers' widths- and w -output weights- from PSO) and optimal placement of RBF units' centers from archive $PF1$.

   2-4- Initially, personal best ($lbest$) is the current calculated cost.

3. Set the global best ($gbest$) to a particle with the lowest cost.

4. PSO main loop:

   4-1- Update velocity for each particle by Eq. 2.3.

   4-2- Control the lower (V_{min}) and upper (V_{max}) bounds of velocity:

   $V_{min} \leq V_i t \leq V_{max}$. Where, i (particle id)=1, 2, ..., N and t (iteration number)=1, 2, ..., $Iter_{Max}$.

   4-3- Update position by Eq. 2.4.

   4-4- If the current velocity and position are outside of the boundaries, they take the upper bound or lower bound. They are multiplied by -1 so that they search in the opposite direction (mirroring to feasible search space).

   4-5- Update personal best (*lbest*): if the current particle cost is better than the previous (recorded in *lbest*) particle cost, then set the current particle cost as the personal best.

   4-6- Update global best (*gbest*): if the current personal best is better than the global best, then set the current personal best as the global best in the swarm.

5. Repeat Step 4 until termination condition (Max_{Iter}) is reached. Otherwise, *gbest* is the optimized RBF units' widths and output weights.

Phase 2: classification of new input patterns

In the second phase -prediction (classification)- we classify (predict) the class type of new input patterns, which we do not know about their target classes in prior. The classification is calculated by defining bins. Data samples should be normalized into [0 1], when dealing with parameters of different units and scales [130]. Since data set is normalized in range of [0 1], bin values should be defined in this range. The number of bin ranges are equal to the number of target classes in training phase. Then, we can determine which data object falls into a specified bin range. For instance, if the number of target class in a particular data set is five classes, then the range of bin values can be organized in the range of [0 0.25 0.5 0.75 1]. Hereafter, constructed RBF neural network from first phase is executed over the input patterns. The RBF output is always a decimal number between [0 1]. This output assigns to the closest and most ideal index of specified bin range, e.g., if output=0.65, then the input pattern falls into fourth bin. It means that the predicted class is four. The pseudo-code of classification computation is given below:

1- Define some input parameters:

LowEdge = lower bound of target class.

$UpEdge$ = upper bound of target class.

$NumBins$= number of target classes in training data set.

$BinEdges$= Generate linearly spaced vectors between $LowEdge$ and $UpEdge$ in the size of $NumBins$, where the bin range is equal to the number of target class.

2- Assign input patterns into the closest index of specified bin range. The index of bin range is the predicted classes of input patterns.

Objective functions in NSGA II

Two objective functions are used to evaluate the RBF network units' centers performance. The two objective functions for minimization problems are:

1. Local optimization based on Mean Square Error (MSE):

Given the set of centers (c), the set of corresponding data objects (x), c_x denotes the center corresponding to the x, and N is the number of data points, MSE can be calculated as:

$$MSE = \frac{1}{N}\sum_{i=1}^{N} d(x_i, c_x)^2 \tag{5.2}$$

2. Well-separated (well-placed) RBF units' centers based on Davies-Boulding Index (DBI).

Based on our experiments [12], we have found it quite reliable. DBI [128] takes into account both compactness and separation criteria that makes similar data points within the same centers and places other data points in distinct centers. The compactness of a group of data objects with corresponding center is calculated based on the MSE. The separation is measured by the distance between centers c_i and c_j. In general, the DBI is given by:

$$\frac{1}{NC}\sum_{i} max_{j,j\neq i}\frac{[\frac{1}{n_i}\sum_{x\epsilon C_i} d(x, c_i) + \frac{1}{n_j}\sum_{x\epsilon C_j} d(x, c_j)]}{d(c_i, c_j)} \tag{5.3}$$

Where, NC is the number of centers, x is the corresponding data objects, n_i is the number of data objects belonging to the center c_i.

5.4 Benchmarking the proposed intelligent classifier (predictor)

For assurance of robustness and accuracy of our proposed intelligent hybrid classifier (predictor), we applied the four classic benchmark problems from the UCI machine learning repository [132]. Table 5.1 shows the main characteristics of these data sets. In the experiments, 70% of data set

Table 5.1: The four applied benchmark data sets

Data set	No. of features	No. of classes	No. of patterns
Wine	13	3	178
Iris	4	3	150
Ionosphere	34	2	351
Zoo	17	7	101

is used as training data set and the rest is considered as testing data set in order to validate the functionality of the proposed method. We evaluated different performance criteria including Mean Square Error (MSE), Standard Deviation (Std.), Standard Error of Mean (SEM), Confidence Interval (CI) by 95% and the number of incorrect classification (Err.). Firstly, we adjust RBF units' centers based on MSE as a frequently used cost function (minimization objective) in the literature. We employ four optimization algorithms which are widely used and adopted successfully in different applications including PSO [180, 181, 182, 183], Genetic Algorithm (GA) [184, 185, 186], Imperialist Competitive Algorithm (ICA) [187, 188, 189] and Differential Evolution (DE) [190, 191, 192]. The experiments on each algorithm were repeated 20 times independently to find the optimal considered performance criteria. Tables 5.2-5.5 show the comparison of (best) results over applied benchmarking problems. As seen in these Tables, PSO performs better results in estimation of RBF units' centers as compared to others based on the applied performance measures. The second optimal results have also performed by GA. However, we have evaluated all results as the (near) optimal adjustment of units' centers for adjusting two others RBF network parameters.

Table 5.2: adjusting RBF units' centers in Wine

n	Pop.	Iter.	MSE	Std.	SEM	CI (95%)
PSO:						
20	20	1500	0.19224	0.1235	0.0101	[0.182 0.671]
40	30	2000	0.16474	0.1207	0.0104	[0.176 0.649]
70	35	2500	0.14989	0.1013	0.0088	[0.165 0.572]
GA:						
20	20	1500	0.19423	0.1242	0.0104	[0.188 0.68]
40	30	2000	0.16532	0.1222	0.0105	[0.196 0.671]
70	35	2500	0.3729	0.1072	0.0093	[0.171 0.583]
ICA:						
20	20	1500	0.41448	0.1421	0.0123	[0.349 0.907]
40	30	2000	0.3396	0.1235	0.0107	[0.327 0.812]
70	35	2500	0.30012	0.124	0.0107	[0.291 0.777]
DE:						
20	20	1500	0.38732	0.1484	0.0128	[0.314 0.895]
40	30	2000	0.41173	0.1555	0.0134	[0.318 0.928]
70	35	2500	0.41586	0.1442	0.0125	[0.346 0.911]

Secondly, for adjusting the RBF units' widths and output weights, we integrate the optimal placement of centers from four applied optimization algorithm (from Tables 5.2-5.5) with PSO. The obtained results are shown in Tables 5.6-5.9. The classification error (Err.) is calculated based on

Table 5.3: adjusting RBF units' centers in Iris

n	Pop.	Iter.	MSE	Std.	SEM	CI (95%)
PSO:						
25	35	2000	0.00957	0.0309	0.0036	[0.009 0.169]
35	50	2500	0.00661	0.0346	0.0033	[0.006 0.142]
40	70	3000	0.00541	0.0362	0.0032	[0.007 0.135]
GA:						
25	35	2000	0.01078	0.0394	0.0037	[0.019 0.173]
35	50	2500	0.00975	0.0439	0.0041	[0.0072 0.175]
40	70	3000	0.00598	0.0365	0.0033	[-0.001 0.138]
ICA:						
25	35	2000	0.02359	0.0666	0.0063	[0.011 0.269]
35	50	2500	0.01497	0.0438	0.0041	[-0.209 0.239]
40	70	3000	0.01332	0.0368	0.0035	[0.037 0.182]
DE:						
25	35	2000	0.02396	0.0595	0.0056	[0.026 0.26]
35	50	2500	0.02376	0.049	0.0046	[0.05 0.242]
40	70	3000	0.02352	0.0584	0.0055	[0.131 0.153]

Table 5.4: adjusting RBF units' centers in Ionosphere

n	Pop.	Iter.	MSE	Std.	SEM	CI (95%)
PSO:						
40	60	3000	0.90357	0.4709	0.0297	[-0.104 1.763]
50	80	4000	0.81119	0.457	0.0282	[-0.079 1.673]
60	90	4000	0.74164	0.4496	0.0284	[-0.085 1.631]
GA:						
40	60	3000	1.043	0.4953	0.0299	[-0.111 1.836]
50	80	4000	0.9489	0.4615	0.0285	[-0.086 1.763]
60	90	4000	0.9394	0.4501	0.0278	[-0.093 1.741]
ICA:						
40	60	3000	2.113	0.5575	0.0344	[0.25 2.436]
50	80	4000	1.9462	0.4792	0.0295	[0.37 2.25]
60	90	4000	1.8535	0.4743	0.0292	[0.347 2.206]
DE:						
40	60	3000	2.6211	0.5671	0.035	[0.405 2.629]
50	80	4000	2.6249	0.5878	0.0362	[0.358 2.663]
60	90	4000	2.5915	0.5493	0.0339	[0.437 2.59]

Table 5.5: adjusting RBF units' centers in Zoo

n	Pop.	Iter.	MSE	Std.	SEM	CI (95%)
PSO:						
40	50	2000	0.75405	0.23	0.0264	[-0.288 1.289]
50	70	2500	0.67409	0.2622	0.0301	[0.198 1.318]
60	90	3000	0.68884	0.2563	0.0274	[0.249 1.253]
GA:						
40	50	2000	0.75469	0.2793	0.032	[0.296 1.371]
50	70	2500	0.68008	0.3057	0.0351	[0.201 1.366]
60	90	3000	0.69329	0.2654	0.0281	[0.315 1.277]
ICA:						
40	50	2000	1.1539	0.303	0.0348	[0.315 1.377]
50	70	2500	0.9867	0.3112	0.0357	[0.334 1.554]
60	90	3000	1.0088	0.2826	0.0324	[0.41 1.518]
DE:						
40	50	2000	1.96	0.3213	0.0369	[0.733 1.933]
50	70	2500	1.9406	0.2829	0.0325	[0.81 1.919]
60	90	3000	1.8115	0.2736	0.0314	[0.782 1.855]

our proposed algorithm in the second phase.

As seen in these tables, PSO is almost able to achieve better results than the other methods in

Table 5.6: Classification of Wine data set based on RBF-PSO optimization algorithm

n	Pop.	Iter.	Training data set					Test data set				
			MSE	Std.	CI (95%)	SEM	Err.	MSE	Std.	CI (95%)	SEM	Err.
Units' centers by PSO:												
20	25	2000	0.00838	0.0912	[-0.157 0.158]	0.00692	**2**	0.01078	0.109	[-0.208 0.2]	0.01567	**2**
40	30	2500	0.00586	0.08024	[-0.158 0.157]	0.00676	**0**	0.01389	0.11475	[-0.25 0.199]	0.01814	**3**
70	40	3000	0.00519	0.07145	[-0.135 0.146]	0.00617	**1**	0.01316	0.11656	[-0.22 0.237]	0.0174	**3**
Units' centers by GA:												
20	25	2000	0.0084	0.093	[-0.164 0.166]	0.00725	**1**	0.01082	0.10907	[-0.216 0.192]	0.01568	**3**
40	30	2500	0.00598	0.08227	[-0.171 0.173]	0.00624	**1**	0.01479	0.11874	[-0.265 0.201]	0.0179	**4**
70	40	3000	0.00525	0.07254	[-0.137 0.148]	0.00626	**2**	0.01501	0.12183	[-0.261 0.216]	0.01836	**3**
Units' centers by ICA:												
20	25	2000	0.00917	0.09615	[-0.188 0.189]	0.0083	**1**	0.01688	0.13139	[-0.262 0.254]	0.0198	**4**
40	30	2500	0.00716	0.08496	[-0.166 0.167]	0.00734	**1**	0.01483	0.11884	[-0.234 0.216]	0.01742	**3**
70	40	3000	0.00677	0.08255	[-0.159 0.165]	0.00713	**1**	0.01576	0.12683	[-0.038 0.024]	0.01912	**3**
Units' centers by DE:												
20	25	2000	0.01159	0.10808	[-0.212 0.212]	0.00933	**2**	0.02135	0.14608	[-0.309 0.264]	0.02202	**3**
40	30	2500	0.00906	0.09555	[-0.187 0.188]	0.00825	**1**	0.01401	0.11895	[-0.252 0.211]	0.01778	**3**
70	40	3000	0.00648	0.08082	[-0.159 0.158]	0.00698	**2**	0.01327	0.11926	[-0.256 0.211]	0.01787	**3**

Table 5.7: Classification of Iris data set based on RBF-PSO optimization algorithm

n	Pop.	Iter.	Training data set					Test data set				
			MSE	Std.	CI (95%)	SEM	Err.	MSE	Std.	CI (95%)	SEM	Err.
Units' centers by PSO:												
25	35	2000	0.007	0.07407	[-0.175 0.125]	0.0079	**2**	0.01347	0.10954	[0.211 0.219]	0.01844	**3**
35	50	2500	0.00435	0.06626	[-0.13 0.13]	0.00623	**2**	0.01419	0.09469	[-0.189 0.182]	0.01692	**2**
40	70	3000	0.00429	0.07007	[-0.132 0.131]	0.00682	**3**	0.05785	0.08827	[-0.189 0.157]	0.01551	**2**
Units' centers by GA:												
25	35	2000	0.00781	0.08877	[-0.174 0.174]	0.00835	**2**	0.01406	0.11579	[-0.223 0.231]	0.01903	**5**
35	50	2500	0.00454	0.06768	[-0.132 0.133]	0.00636	**1**	0.01416	0.10704	[-0.206 0.213]	0.01759	**3**
40	70	3000	0.00432	0.07391	[-0.145 0.145]	0.00544	**2**	0.05557	0.0734	[-0.162 0.126]	0.01206	**2**
Units' centers by ICA:												
25	35	2000	0.0079	0.08717	[-0.153 0.149]	0.00725	**2**	0.01517	0.12285	[-0.238 0.243]	0.01897	**3**
35	50	2500	0.00543	0.07405	[-0.146 0.145]	0.00696	**3**	0.01542	0.12586	[-0.244 0.249]	0.02069	**2**
40	70	3000	0.00455	0.07008	[-0.137 0.137]	0.00563	**2**	0.05092	0.10541	[-0.217 0.196]	0.01732	**3**
Units' centers by DE:												
25	35	2000	0.00782	0.08188	[-0.149 0.149]	0.00721	**2**	0.01378	0.11638	[-0.15 0.15]	0.01913	**3**
35	50	2500	0.00493	0.07666	[-0.123 0.124]	0.00592	**2**	0.01822	0.09973	[-0.202 0.189]	0.01608	**2**
40	70	3000	0.00625	0.07941	[-0.153 0.158]	0.00747	**3**	0.05956	0.09912	[-0.192 0.197]	0.01629	**3**

Table 5.8: Classification of Ionosphere data set based on RBF-PSO optimization algorithm

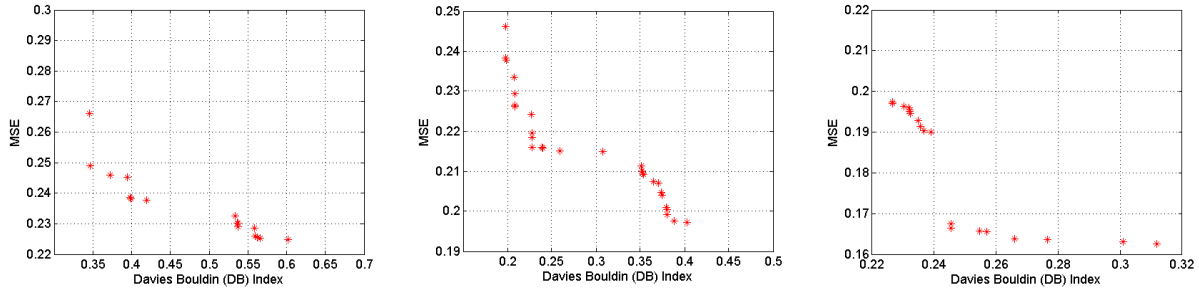
n	Pop.	Iter.	Training data set					Test data set				
			MSE	Std.	CI (95%)	SEM	Err.	MSE	Std.	CI (95%)	SEM	Err.
Units' centers by PSO:												
30	60	2000	0.05403	0.22663	[-0.444 0.444]	0.01395	**17**	0.05838	0.2387	[-0.423 0.513]	0.02544	**5**
40	80	2500	0.05172	0.22785	[-0.448 0.445]	0.01405	**16**	0.05553	0.23233	[-0.409 0.502]	0.024767	**3**
50	90	3000	0.0466	0.20488	[-0.401 0.403]	0.01244	**14**	0.04855	0.21235	[-0.373 0.459]	0.02289	**4**
Units' centers by GA:												
30	60	2000	0.05464	0.23407	[-0.451 0.467]	0.01443	**19**	0.05436	0.22923	[-0.395 0.504]	0.02443	**5**
40	80	2500	0.06114	0.24773	[-0.485 0.487]	0.01527	**20**	0.07284	0.27006	[-0.502 0.557]	0.02878	**7**
50	90	3000	0.05673	0.23859	[-0.473 0.462]	0.01471	**17**	0.05943	0.24339	[-0.448 0.507]	0.02594	**4**
Units' centers by ICA:												
30	60	2000	0.07042	0.2658	[-0.528 0.514]	0.01639	**19**	0.06913	0.26367	[-0.497 0.537]	0.0281	**7**
40	80	2500	0.06699	0.25932	[-0.508 0.509]	0.01599	**20**	0.06238	0.24981	[-0.427 0.552]	0.02662	**5**
50	90	3000	0.06389	0.25318	[-0.491 0.502]	0.01561	**21**	0.06058	0.24449	[-0.441 0.518]	0.026	**5**
Units' centers by DE:												
30	60	2000	0.05847	0.24228	[-0.474 0.476]	0.01492	**18**	0.06325	0.24815	[-0.438 0.535]	0.02645	**4**
40	80	2500	0.05798	0.24125	[-0.474 0.472]	0.01487	**15**	0.05386	0.22932	[-0.406 0.493]	0.02444	**3**
50	90	3000	0.06175	0.24898	[-0.489 0.487]	0.01535	**16**	0.06379	0.2508	[-0.452 0.532]	0.02673	**6**

terms of the classification error and other applied metrics. Experimental results demonstrate that even though the ICA and the DE with not so proper results in obtaining RBF units' centers could successfully provide low classification error. Unlike the suitable number of correct classification by ICA and DE, they do not usually perform well in terms of MSE, Std., CI (95%) and SEM as compared to PSO and GA. Since the number of correct classification is the major criterion in the classification problems, it can be concluded that the MSE (as minimization objective) is not a suitable performance metric for finding the (near) optimal placement of units' centers. To

Table 5.9: Classification of Zoo data set based on RBF-PSO optimization algorithm

n	Pop.	Iter.	Training data set					Test data set				
			MSE	Std.	CI (95%)	SEM	Err.	MSE	Std.	CI (95%)	SEM	Err.
Units' centers by PSO:												
30	50	2000	0.00156	0.03974	[-0.077 0.079]	0.00455	**3**	0.00394	0.0552	[-0.113 0.104]	0.01104	**3**
40	70	2500	0.00093	0.03024	[-0.059 0.059]	0.00366	**1**	0.00471	0.07	[-0.14 0.135]	0.01401	**5**
50	90	3000	0.00095	0.03114	[-0.061 0.061]	0.00335	**3**	0.00607	0.07197	[-0.157 0.131]	0.01439	**4**
Units' centers by GA:												
30	50	2000	0.00222	0.47477	[-0.093 0.094]	0.00544	**4**	0.00904	0.08952	[-0.212 0.139]	0.0179	**7**
40	70	2500	0.00141	0.03783	[-0.074 0.074]	0.00433	**5**	0.00595	0.07695	[-0.167 0.134]	0.01539	**4**
50	90	3000	0.00115	0.03422	[-0.067 0.067]	0.00392	**4**	0.00628	0.06858	[-0.147 0.122]	0.01383	**4**
Units' centers by ICA:												
30	50	2000	0.00211	0.04628	[-0.089 0.093]	0.0053	**5**	0.00706	0.08286	[-0.155 0.17]	0.01657	**4**
40	70	2500	0.0011	0.0334	[-0.065 0.066]	0.00383	**2**	0.00487	0.06209	[-0.155 0.17]	0.01241	**6**
50	90	3000	0.00102	0.03228	[-0.063 0.064]	0.0037	**2**	0.00826	0.08935	[-0.2 0.151]	0.01787	**4**
Units' centers by DE:												
30	50	2000	0.00159	0.04024	[-0.079 0.078]	0.00461	**3**	0.00514	0.07046	[-0.158 0.119]	0.01409	**5**
40	70	2500	0.00136	0.03712	[-0.073 0.073]	0.00425	**4**	0.00848	0.08664	[-0.206 0.134]	0.01733	**6**
50	90	3000	00113	0.03385	[-0.066 0.066]	0.00388	**2**	0.00635	0.074	[-0.155 0.135]	0.0148	**4**

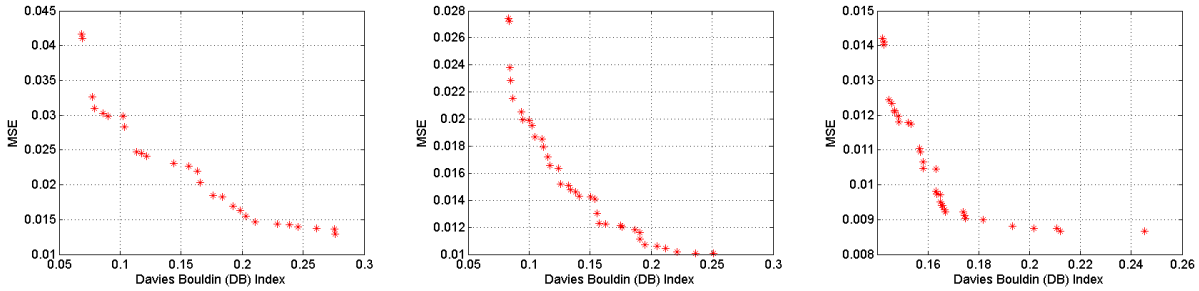
confirm convincingly this claim, we present a multiobjective approach to find the (near) optimal placement of centers. According to the first part of the proposed method (see Fig. 5-2), NSGA II was applied over benchmarking problems by two conflicting objectives (DBI and MSE) in order to find the well-separated centers and their local optimization, respectively. The experiment on proposed algorithm was repeated 5 times independently to find the optimal performance metrics. Figs. 5-3-5-6 are depicted the optimal Pareto front solutions of (near) well-placed of RBF units' centers through DBI (x-axis) and MSE (y-axis). We are going to show that for constructing final RBF neural networks, MSE is not solely the ideal accurate criterion.



(a) Multi objective (n=20) (b) Multi objective (n=40) (c) Multi objective (n=70)

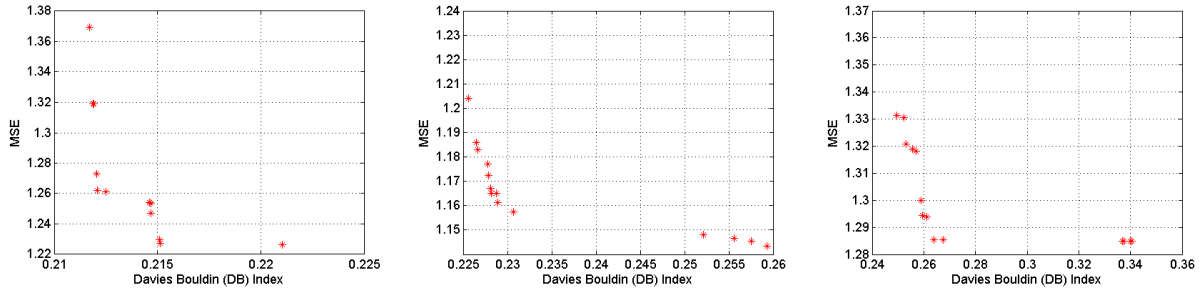
Figure 5-3: Optimal Pareto fronts of Wine data set

Afterward, we integrate the optimal placement of units' centers (obtained by our two-objective approach in Figs. 5-3-5-6) with the PSO (see second step of the first phase in Fig. 5-2) in order to optimize and tune units' widths and output weights. We run the PSO algorithm with all the optimal Pareto front solutions of units' centers. The first five optimal results are demonstrated based on the minimum classification error in both training and testing data sets in Tables 5.10-5.13. As seen in these tables, the first five optimal Pareto solutions outperform significantly the other methods by single-objective approach in Tables 5.6-5.9 based on the MSE, Std. and the num-



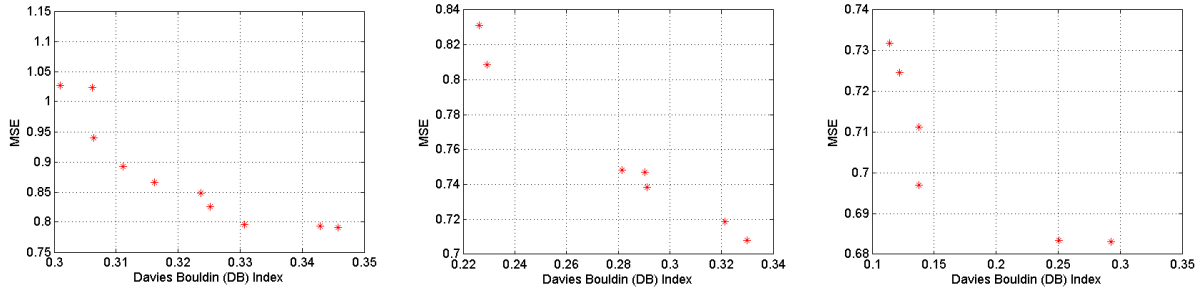
(a) Multi objective (n=20) (b) Multi objective (n=50) (c) Multi objective (n=70)

Figure 5-4: Optimal Pareto fronts of Iris data set



(a) Multi objective (n=60) (b) Multi objective (n=80) (c) Multi objective (n=90)

Figure 5-5: Optimal Pareto fronts of Ionosphere data set



(a) Multi objective (n=50) (b) Multi objective (n=70) (c) Multi objective (n=90)

Figure 5-6: Optimal Pareto fronts of Zoo data set

ber of misclassification error. Other applied performance metrics outperforms the single-objective approach over 90% of results.

The results show that the proposed method can provide several well-placed RBF units' centers as compared to the traditional (single-objective) approaches through MSE criterion. To sum up, MSE is not an unique criterion to evaluate the performance of the units' centers in RBF networks. A new hybrid optimization approach for well-separated centers (such as by DBI) and their local

Table 5.10: Classification of Wine data set based on proposed method

n	Pop.	Iter.	NSGA II		Training data set					Test data set				
			MSE	DBI	MSE	Std.	CI (95%)	SEM	Err.	MSE	Std.	CI (95%)	SEM	Err.
			0.2252	0.566	0.0063	0.0801	[-0.157 0.158]	0.0069	**0**	0.0104	0.1033	[-0.222 0.191]	0.017	**1**
			0.2249	0.6022	0.007	0.0811	[-0.164 0.166]	0.0072	**0**	0.0105	0.1015	[-0.227 0.2]	0.0183	**1**
20	25	2500	0.2303	0.5362	0.0073	0.0813	[-0.169 0.17]	0.0074	**0**	0.0105	0.1043	[-0.232 0.201]	0.0187	**1**
			0.2376	0.4196	0.0076	0.0819	[-0.162 0.171]	0.0075	**1**	0.0106	0.1087	[-0.23 0.21]	0.018	**1**
			0.2452	0.3941	0.0081	0.0835	[-0.167 0.174]	0.0078	**1**	0.0103	0.1076	[-0.217 0.202]	0.0162	**2**
			0.2004	0.3799	0.0049	0.0702	[-0.138 0.138]	0.006	**0**	0.0107	0.1032	[-0.221 0.184]	0.0155	**1**
			0.1973	0.4032	0.0049	0.0703	[-0.137 0.139]	0.006	**0**	0.0131	0.1135	[-0.246 0.2]	0.0163	**1**
40	30	2500	0.1993	0.3803	0.0051	0.0765	[-0.149 0.151]	0.0066	**0**	0.0138	0.1143	[-0.247 0.201]	0.0166	**1**
			0.2095	0.3532	0.0052	0.0795	[-0.155 0.157]	0.0068	**0**	0.0131	0.1143	[-0.251 0.198]	0.0166	**1**
			0.2074	0.3646	0.0055	0.0798	[-0.156 0.157]	0.007	**0**	0.0138	0.1131	[-0.259 0.185]	0.0162	**2**
			0.1638	0.2661	0.0039	0.0632	[-0.125 0.123]	0.0054	**0**	0.0126	0.1106	[-0.243 0.191]	0.0166	**1**
			0.1657	0.2547	0.0044	0.0667	[-0.131 0.131]	0.0057	**0**	0.0126	0.11	[-0.244 0.188]	0.0165	**1**
70	40	3500	0.1636	0.2767	0.0044	0.0669	[-0.128 0.135]	0.0057	**0**	0.0129	0.1138	[-0.241 0.205]	0.0171	**1**
			0.1630	0.3011	0.0045	0.0678	[-0.133 0.132]	0.0058	**0**	0.0127	0.1138	[-0.233 0.213]	0.0171	**2**
			0.1674	0.2454	0.0051	0.0696	[-0.142 0.131]	0.0061	**0**	0.0131	0.1156	[-0.243 0.21]	0.0174	**2**

Table 5.11: Classification of Iris data set based on proposed method

n	Pop.	Iter.	NSGA II		Training data set					Test data set				
			MSE	DBI	MSE	Std.	CI (95%)	SEM	Err.	MSE	Std.	CI (95%)	SEM	Err.
			0.0136	0.2759	0.0046	0.0685	[-0.134 0.134]	0.0064	**1**	0.0118	0.109	[-0.198 0.23]	0.0179	**2**
			0.0129	0.0403	0.0052	0.0729	[-0.142 0.144]	0.0068	**2**	0.0128	0.1027	[-0.168 0.235]	0.0217	**2**
25	35	2500	0.0183	0.184	0.0055	0.0745	[-0.145 0.147]	0.007	**1**	0.0126	0.1002	[-0.173 0.22]	0.021	**2**
			0.0142	0.2386	0.0058	0.0748	[-0.15 0.151]	0.0072	**1**	0.0099	0.1012	[-0.197 0.2]	0.0166	**2**
			0.0169	0.1924	0.0063	0.0748	[-0.156 0.157]	0.0075	**2**	0.0124	0.1091	[-0.191 0.237]	0.0214	**2**
			0.0111	0.1911	0.0031	0.0565	[-0.111 0.111]	0.0053	**1**	0.0139	0.0929	[-0.153 0.191]	0.02	**1**
			0.0121	0.1752	0.0034	0.0589	[-0.115 0.116]	0.0055	**1**	0.0139	0.0913	[-0.146 0.212]	0.02	**1**
35	50	3000	0.0106	0.2048	0.0034	0.059	[-0.115 0.116]	0.0055	**1**	0.0136	0.0918	[-0.162 0.191]	0.0206	**1**
			0.0116	0.1910	0.0037	0.0616	[-0.121 0.121]	0.005	**2**	0.0129	0.0924	[-0.164 0.198]	0.02	**1**
			0.0105	0.2122	0.0038	0.062	[-0.123 0.121]	0.0058	**1**	0.0124	0.0906	[-0.16 0.196]	0.0207	**1**
			0.0095	0.1650	0.0041	0.0663	[-0.129 0.131]	0.0062	**1**	0.0399	0.052	[-0.058 0.046]	0.026	**1**
			0.0088	0.1932	0.0041	0.0679	[-0.133 0.134]	0.0063	**1**	0.039	0.0566	[-0.053 0.169]	0.0256	**1**
40	70	3500	0.0091	0.1744	0.0042	0.0681	[-0.133 0.135]	0.0064	**1**	0.0347	0.0598	[-0.074 0.161]	0.024	**1**
			0.0087	0.2108	0.0042	0.0682	[-0.132 0.135]	0.0064	**1**	0.0323	0.0528	[-0.056 0.152]	0.0233	**1**
			0.0097	0.1636	0.0042	0.069	[-0.134 0.136]	0.0064	**2**	0.0346	0.0483	[-0.045 0.144]	0.0242	**1**

Table 5.12: Classification of Ionosphere data set based on proposed method

n	Pop.	Iter.	NSGA II		Training data set					Test data set				
			MSE	DBI	MSE	Std.	CI (95%)	SEM	Err.	MSE	Std.	CI (95%)	SEM	Err.
			1.2472	0.2147	0.0528	0.2246	[-0.442 0.439]	0.0134	**12**	0.0512	0.2064	[-0.371 0.438]	0.0213	**2**
			1.2266	0.2210	0.0532	0.2254	[-0.442 0.442]	0.0135	**11**	0.0538	0.2091	[-0.372 0.448]	0.0216	**2**
30	60	2000	1.2539	0.2147	0.0533	0.2257	[-0.443 0.442]	0.0135	**12**	0.0539	0.2022	[-0.366 0.472]	0.02095	**2**
			1.2297	0.2151	0.0533	0.2217	[-0.431 0.438]	0.0139	**12**	0.053	0.2113	[-0.389 0.439]	0.0207	**2**
			1.262	0.2121	0.0533	0.2218	[-0.437 0.433]	0.0139	**11**	0.0536	0.211	[-0.377 0.451]	0.0207	**2**
			1.1434	0.2593	0.043	0.2079	[-0.403 0.413]	0.0128	**9**	0.0375	0.1902	[-0.331 0.415]	0.0202	**2**
			1.1651	0.2282	0.0437	0.2094	[-0.409 0.412]	0.0129	**10**	0.042	0.2011	[-0.349 0.44]	0.0214	**1**
40	80	2500	1.1481	0.2521	0.0439	0.2099	[-0.41 0.413]	0.0129	**9**	0.0437	0.2048	[-0.354 0.45]	0.0218	**1**
			1.1574	0.5508	0.0445	0.2113	[-0.407 0.422]	0.013	**11**	0.0446	0.2067	[-0.357 0.454]	0.022	**2**
			1.1465	0.2556	0.0453	0.213	[-0.404 0.431]	0.0131	**11**	0.043	0.2045	[-0.36 0.442]	0.021	**2**
			1.3209	0.2531	0.0373	0.2039	[-0.395 0.405]	0.0147	**11**	0.044	0.2104	[-0.384 0.442]	0.0245	**2**
			1.2998	0.2591	0.0399	0.2024	[-0.399 0.305]	0.0138	**11**	0.0425	0.211	[-0.38 0.447]	0.0231	**1**
50	90	3000	1.3313	0.2496	0.0412	0.2026	[-0.401 0.39]	0.0139	**11**	0.0427	0.2088	[-0.399 0.44]	0.0243	**2**
			1.2945	0.2596	0.0435	0.2031	[-0.403 0.394]	0.0143	**10**	0.0426	0.2117	[-0.381 0.449]	0.0336	**1**
			1.2939	0.2612	0.0452	0.2034	[-0.396 0.401]	0.0144	**10**	0.0478	0.211	[-0.395 0.433]	0.0256	**2**

Table 5.13: Classification of Zoo data set based on proposed method

n	Pop.	Iter.	NSGA II		Training data set					Test data set				
			MSE	DBI	MSE	Std.	CI (95%)	SEM	Err.	MSE	Std.	CI (95%)	SEM	Err.
			0.7917	0.3458	0.0005	0.0227	[-0.044 0.045]	0.0026	**0**	0.0038	0.0522	[-0.132 0.073]	0.012	**1**
			0.7954	0.3307	0.0005	0.024	[-0.047 0.047]	0.0027	**1**	0.0035	0.0512	[-0.126 0.075]	0.0112	**1**
50	40	3000	0.826	0.3252	0.0005	0.0244	[-0.048 0.049]	0.0028	**1**	0.0033	0.0522	[-0.113 0.092]	0.0102	**1**
			0.793	0.3429	0.0006	0.026	[-0.051 0.051]	0.0029	**0**	0.003	0.0521	[-0.11 0.094]	0.0118	**1**
			0.8924	0.3111	0.0008	0.0298	[-0.059 0.058]	0.0031	**1**	0.0031	0.0527	[-0.092 0.115]	0.01	**2**
			0.7185	0.3212	0.0004	0.0218	[-0.042 0.044]	0.0025	**0**	0.0045	0.0602	[-0.104 0.132]	0.0116	**1**
			0.747	0.2903	0.0007	0.0278	[-0.054 0.055]	0.0031	**0**	0.0041	0.0598	[-0.121 0.114]	0.0109	**2**
70	50	3000	0.7383	0.2911	0.0008	0.0289	[-0.057 0.057]	0.0033	**1**	0.003	0.0563	[-0.112 0.108]	0.0112	**2**
			0.7081	0.239	0.0008	0.0291	[-0.056 0.058]	0.0033	**0**	0.0043	0.0605	[-0.123 0.115]	0.0115	**1**
			0.7482	0.2816	0.0008	0.0294	[-0.058 0.058]	0.0033	**1**	0.0029	0.055	[-0.108 0.107]	0.011	**2**
			0.6831	0.2927	0.0004	0.0223	[-0.044 0.044]	0.0025	**0**	0.0055	0.0621	[-0.142 0.102]	0.0161	**2**
			0.6834	0.2508	0.0004	0.021	[-0.041 0.041]	0.0024	**0**	0.0059	0.0601	[-0.135 0.101]	0.0112	**2**
90	60	3000	0.6969	0.1378	0.0006	0.0258	[-0.05 0.051]	0.0029	**0**	0.006	0.0665	[-0.142 0.119]	0.0112	**2**
			0.7113	0.1317	0.0007	0.0282	[-0.055 0.056]	0.0032	**1**	0.0058	0.0665	[-0.146 0.115]	0.0113	**2**
			0.7246	0.1221	0.0002	0.0171	[-0.033 0.034]	0.0019	**0**	0.0026	0.0516	[-0.109 0.094]	0.0103	**1**

optimization (such as by MSE) in estimation of RBF units' centers would fit considerably the performance requirements.

5.5 Evaluation environment

We use simulations to quantify effects of DoS attacks and their countermeasures. In this work, we used the open-source ndnSIM [53] package, which implements NDN protocol stack for NS-3 network simulator (`http://www.nsnam.org/`), to run simulations for evaluating the performance of considered mitigation method. ndnSIM simulation environment reproduces the basic structures of a NDN node (i.e., CS, PIT, FIB, strategy layer, and so on). The proposed detection method (first phase) was implemented by the MATLAB software on the Intel Pentium 2.13 GHz CPU, 4 GB RAM running Windows 7 Ultimate. This algorithm deployed to C++ project integrating as a C++ shared library using the MATLAB compiler. Then, this C++ program was integrated with ndnSIM environment to be able to adjust in the simulation environment. The proposed adaptive reaction was also implemented with C++ in ndnSIM environment. We demonstrate through simulations that our countermeasure satisfies considerably applied performance metrics as compared to two recently applied DoS attack mitigation methods namely satisfaction-based pushback and satisfaction-based Interest acceptance [17]. We perform 10 times simulation runs to calculate the average performance metrics. Our experiments are performed over two topologies

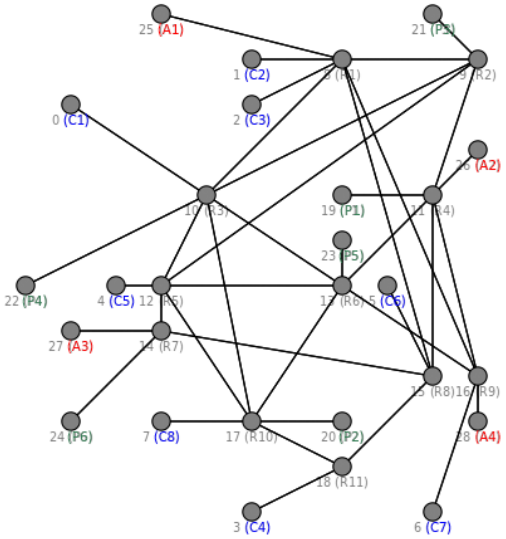


Figure 5-7: DFN-like topology

shown in Figs. 5-7 and 5-8. Fig. 5-7 corresponds to DFN-like (Deutsche Forschungsnetz as the

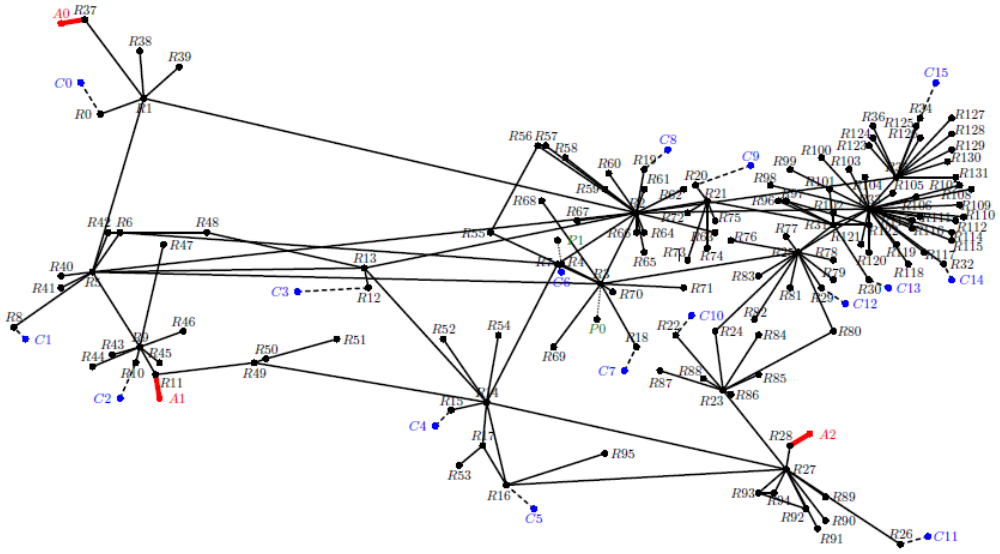


Figure 5-8: AT&T topology

German Research Network) [193], and Fig. 5-8 corresponds to the AT&T network [194]. We use the symbols Cx, Px, Rx, and Ax to represent x-th consumer, producer, router and adversary nodes, respectively. In spite of various arguments and experiments, there is no typically and properly justification for NDN parameters and they have specified based on authors' experiences and designs [10]. The experimental setup (i.e., attack and non-attack traffics modeling) is performed over two applied topologies as follows. For attack effectiveness, we examine the performance of the network's data packet delivery and satisfied Interest rate under the different scenarios (see DoS attacks issues in section 5.1):

1. Interest flooding attack (dynamically-generated Interest packets) for the existent Data.

2. Interest flooding (dynamically-generated Interest packets) for the non-existent Data. It can be in the form of brute-force attack (very high distribution of Interest) or normal distribution of Interest.

3. Hijacking, in which a producer silently drops all incoming Interest traffic in downstream interfaces.

4. Content poisoning (bogus data packets), in which a producer deliberately signs data packets with a wrong key. We assume that the routers firstly check the signature filed of data packet, then cache and route the packet toward its destination if the signature is valid. Hence, the bogus data packets cannot be cached in the intermediate routers.

In our configurations, we set nodes' PIT size to 120 KB, while the Interest expiration time was set to the default timeout of 4 sec. We set the link delay and queue length parameters to fixed values for every node in the simulated topologies. In particular, we set delay and queue length to 10 ms and 400 for both considered topologies, respectively. The PIT entries replacement policy was adopted to the least-recently-used (the oldest entry with minimum number of incoming faces will be removed if PIT size reached its limit) as a widely used strategy. The nodes' cache capacity was set to 1000 contents and cache replacement policy was set to least-recently-used method. The other system settings of investigated network topologies are summarized in Table 5.14. As shown in this table, we ran various traffic patterns in which each configuration changes in every 10 simulation runs in order to perform different network characteristics. We first analyze the topologies without any adversarial

Table 5.14: Network parameters considered

Node	Distribution	Pattern	Frequency	Run time (min.)	Producer	Goal
DFN-like topology (Fig. 5-7)						
C1	randomize	uniform	[100 500]	0-40	P1	normal
C2	randomize	exponential	[100 500]	2-40	P2	normal
C3	Zipf-Mandelbort (α = [0.5 0.9])	exponential	[100 500]	3-40	P3	normal
C4	randomize	uniform	[100 500]	4-40	P6	normal
C5	Zipf-Mandelbort (α = [0.5 0.9])	exponential	[100 500]	3-40	P2, P3	normal
C6	randomize	uniform	[100 500]	5-40	P3	normal
C7	randomize	uniform	[100 500]	7-16, 22-31	P6, P4	sign data with the wrong key
C8	randomize	exponential	[100 500]	8-18, 25-40	P1	normal
A1	randomize	uniform	[1500 3000]	7-16	P1	Interest flooding for existence data
A2	Zipf-Mandelbort (α = [0.5 0.9])	uniform	[1500 3000]	22-31	no producer	Interest flooding for non-existence data
A3	randomize	uniform	[400 800]	7-16	P5 (hijacker)	hijacking incoming Interest packets
A4	randomize	exponential	[1500 3000]	22-31	P6	Interest flooding for existence data
AT&T topology (Fig. 5-8)						
C0, C7	randomize	uniform	[200 600]	0-50	P0, P1	normal
C1, C8	randomize	exponential	[200 600]	2-50	P0	normal
C2, C9	randomize	exponential	[200 600]	3-50	P1	normal
C3, C10	randomize	uniform	[200 600]	4-50	P1	normal
C4, C11	Zipf-Mandelbort (α = [0.5 0.9])	exponential	[200 600]	5-50	P0, P1	normal
C5, C12, C13	randomize	uniform	[200 600]	6-50	P0, P1	normal
C6, C14, C15	randomize	uniform	[200 600]	8-50	P1	normal
A0	randomize	uniform	[1000 3000]	7-25	P1	Interest flooding for existence data
A0	randomize	exponential	[1000 3000]	30-45	P1	Interest flooding for existence data
A1	Zipf-Mandelbort (α = [0.5 0.9])	exponential	[500 1000]	7-25	P0	sign data with the wrong key
A1	Zipf-Mandelbort (α = [0.5 0.9])	uniform	[1000 3000]	30-45	no producer	Interest flooding for non-existence data
A2	randomize	exponential	[1000 3000]	7-25	no producer	Interest flooding for non-existence data
A2	randomize	uniform	[1000 3000]	30-45	P1	Interest flooding for existence data

traffic, then with adversarial traffic, finally consideration of the proposed mitigation method over the illegitimate traffics. Our assumption is that, the behavior of legitimate (honest) consumers is unchanged in duration of the simulation, and the adversary is not allowed to control routers. To study the performance of our proposed countermeasure algorithm under range of conditions, we varied the percentage of attackers and their run times in the considered topologies in Table 5.14.

5.6 The proposed countermeasure: proactive detection and adaptive reaction

In this section, we introduce our method, a two phases -detection and reaction- framework for mitigating DoS attacks in NDN.

5.6.1 Detection Phase

This step adopts our proposed intelligent classifier from section 5.3.1. We choose the DFN-like topology (Fig. 5-7) in the training phase with the recommended parameter settings in Table 5.14. We then apply this trained network for the detection purposes in both DFN-like and AT&T topologies.

NDN routers can easily keep track of unsatisfied (expired) Interests and use this information for DoS attack countermeasures such as, pending Interests per outgoing and incoming interfaces, and pending Interests per namespace. The proper combining/choosing of statistic parameters in NDN routers for maximum effectiveness against attacks and anomalies, minimum disordering of legitimate traffics, and distinguishing between 'good' and 'bad' Interest packets are research challenges [17, 23]. Hence, we employed simple intrinsic features from the network which is shown in Table 5.15 (i.e., the input features in the RBF neural network).

Table 5.15: Feature construction

Feature	**Description**
InInterests	a number of arrival Interest in an interface
InData	a number of arrival data in an interface
InSatisfiedInterests	a number of satisfied Interests where interface was part of the incoming set
InTimedOutInterests	a number of timed out Interests where interface was part of the incoming set
OutInterests	a number of sent Interest from an interface
OutData	a number of sent data from an interface
OutSatisfiedInterests	a number of satisfied Interests where interface was part of the outgoing set
OutTimedOutInterests	a number of timed out Interests where interface was part of the outgoing set
DropInterests	a number of dropped Interest in an interface
DropData	a number of dropped data in an interface
SatisfiedInterests	a total number of satisfied Interests
TimedOutInterests	a total number of timed out Interests

In the training process, all the features beginning with 'In' are suitable for prediction of the misbehaving consumers and the features by 'Out' are suitable for prediction of the misbehaving producers. Taking into account only a specific or a group (e.g., 'In' or 'Out') of features may cause the detection algorithm to report a wrong prediction. For example, if there are two PIT

entries that share the same prefix and one Data packet arrives, there will be two entries of In/Out satisfied Interest but only one In/Out Data, since both Interests can be satisfied with the same Data. Hence, if a number of In/Out Data be more than the In/Out satisfied Interest for a given interface or vice versa, it would not be a misbehaving. Another instance is that, Interest packets from a consumer are possible to arrive to several routers and perhaps several producers that can satisfy the Interests. Corresponding data packet will send back from producer(s). A router in the middle way, receives the first packet from any producer and will forward it to the consumer and remove the PIT entry. When the second Data object arrives to the router, it will be discarded by the routers as unsolicited. Hence, it is more likely that a rate of In/Out Data or DropData be more than In/Out Interest rate and vice versa in a corresponding interface. Obviously, it is not an attack or anomaly behavior. Also, in a given interface, the rate of the InInterest may be less that the SatisfiedInterest rate which in due to the portion of the satisfaction rate comes from the previous time interval. On the other hand, the rate of the OutData may be more than the InInterest rate, which is for routing the cached data for satisfying incoming Interest packets. To sum up, different parameters mentioned by our detection module act as weights and counterweights for misbehaving consumer and producer detection purposes.

For constructing a predictor module based on the RBF neural network, at first the centers, widths and weights are computed and adjusted using training set 75% of data set, and then the remaining part of the data set as the test set, is used to validate the trained network functionality. We trained and evaluated the network with various number of RBF units, where the three optimal results are summarized in Table 5.16. The optimal Pareto front solutions by NSGA II are also depicted in Fig. 5-9. We computed the MSE, Std., CI (95%), SEM and classification error for both training and testing parts. The histogram analysis of the classification error distribution and the regression analysis of the misclassification are shown in Figs. 5-10 and 5-11, respectively. As seen in these Figures and Table 5.16, third parameter settings could provide the better results as compared to the two others in terms of the applied performance metrics. Hence, these (near) optimal parameter settings are used to construct our RBF classifier (predictor).

Table 5.16: Classification of NDN data set based on proposed method

n	Pop.	Iter.	NSGA II		Training data set					Test data set				
			MSE	DBI	MSE	Std.	CI (95%)	SEM	Err.	MSE	Std.	CI (95%)	SEM	Err.
			0.0314	0.0979	0.0099	0.0998	[-0.192 0.197]	0.0055	2	0.0235	0.1541	[-0.301 0.3]	0.0147	3
80	40	2500	0.0643	0.0979	0.0099	0.0998	[-0.186 0.206]	0.0055	2	0.0248	0.1584	[-0.32 0.29]	0.0151	3
			0.0315	0.0914	0.0095	0.095	[-0.187 0.186]	0.0054	1	0.0231	0.1527	[-0.299 0.3]	0.0142	1

As we expected (based on our proof in section 5.4), This phase constructs an optimized and

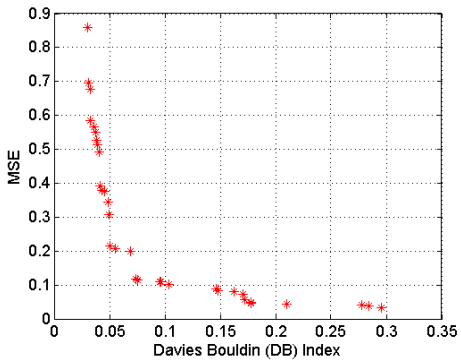
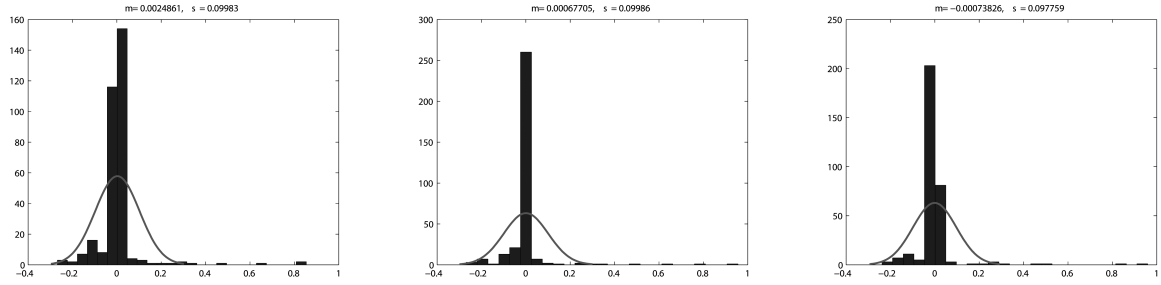
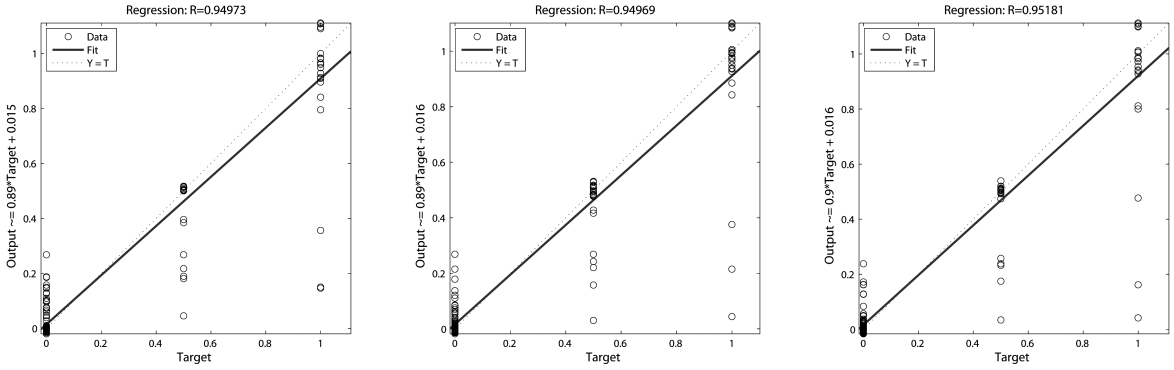


Figure 5-9: Optimal Pareto fronts of DFN-like training phase



(a) 1st histogram (b) 2nd histogram (c) 3rd histogram

Figure 5-10: The histogram analysis of the classification error distribution in DFN-like topology



(a) 1st Regression (94.97%) (b) 2nd Regression (94.96%) (c) 3rd Regression (95.18%)

Figure 5-11: Regression of the classification error between target and predicted output in DFN

more accurate RBF classifier (predictor) for our DoS attack mitigation purposes in NDN. According to the traffic flows type in the training data set (see Table 5.14), this predictor learned three types of traffic patterns including normal, malicious behavior from consumers and producers. This predictor module runs on routers, in order to continuously monitor per-interface required statistical

information. This module is executed at fixed time intervals -typically every 0.5 sec - to provide a proactive detection behavior. Finally, based on three types of prediction (normal, misbehaving consumer and misbehaving producer), we should respond an appropriate action as detailed in the next subsection.

5.6.2 Reaction Phase

Once a DoS attack from interface j of router i has been identified with the proposed proactive detector (see section 5.6.1), our reaction mechanism enables and enforces explicit limitation based on the prediction type (adversary consumer or adversary producer) for each interface. The proposed intelligent proactive detector reports misbehaving in the early stages of beginning DoS attacks. Our adaptive reaction criterion for misbehaving consumer directly depends on the local interface's Interest unsatisfied ratio and for misbehaving producer directly depends on the forwarding strategy. The original settings and Interest rate are restored once the detector module reports the normal traffic in the next time interval.

reaction regarding to misbehaving consumer

When the proposed intelligent detector module in router detects adversarial traffics from a set of interfaces, it sends an alert message on each of them. An alert message is an unsolicited content packet which belongs to a reserved namespace ("`/pushbackmessage/alert/`") in our implementation. There are two reasons for using content packet rather than Interest packet for carrying pushback message [20]:

1. during an attack, the PIT of next hop connected to the offending interface may be full, and therefore the alert message may be discarded, and

2. content packets are signed, while Interests are not. This allows routers to receive the content packets as a legitimate packet for processing.

The payload of an alert message contains the timestamp corresponding to the generation time of the alert message, the new reduced (unsatisfied) rate and the wait time of reduction period. The formal definition of our unsatisfied-based pushback mechanism presents in Fig. 5-12. Assuming in a time interval in router `C` the predictor reports a misbehaving traffic from a consumer (neighbor node).

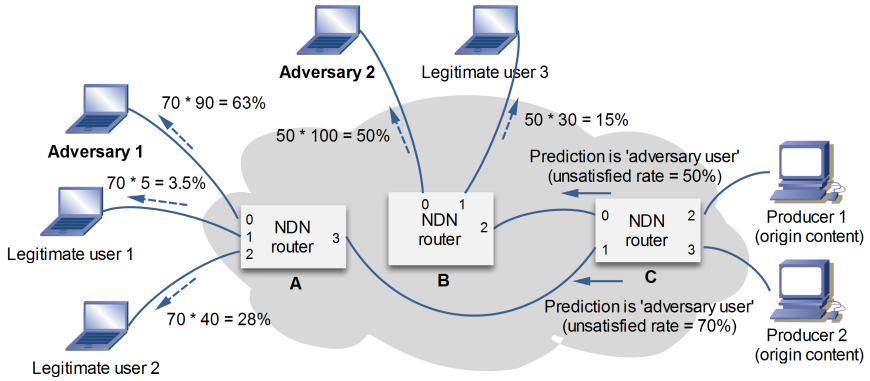


Figure 5-12: unsatisfied-based pushback example

Also, an unsatisfied rate is 50% for `eth0` and 70% for `eth1`. Our proposed reaction mechanism is as follows:

1. Router `C` will send a pushback alert message to the neighbors from `eth0` and `eth1`.

2. Routers `A` and `B`, after receiving alert message from `C` will readjust their local interfaces limit to `'announced reduced rate'` × `'local unsatisfied rate'` in each local interface. If the new limit in the corresponding interface exceeds the predefined threshold ϕ, the corresponding interface gets new reduction of Interest rate in downstream. For instance, we assume $\phi = 5\%$ so that router `B` decreases the Interest rate of `eth0` to 50% and `eth1` to 15%. Router `A` decreases the Interest rate in its three interfaces to 63%, 0 (the new limit rate (=3.5%) is under predefined threshold (=5%) and will not be changed) and 28% in `eth0`, `eth1` and `eth2`, respectively. This threshold allows bandwidth usage be consumed for legitimate traffics in the nearest next time interval and intensifies Interest rate reduction for adversaries in each next time intervals.

3. Our wait time strategy for the reduction period in neighbor nodes is an ascending penalty. If in a time interval t in interface j the misbehaving traffic be reported, a counter sets to 1 sec. If in the next time interval $t+1$ the misbehaving again be reported, a counter sets to 2 sec. Our ascending penalty method is in $2^{counter}$. Initially, $counter = 0$ and increase linearly in each time interval. The counter is set to the initial value when there is no misbehaving prediction in the next time interval. This ascending penalty intensifies the penalty for adversaries and alleviates the bandwidth usage for legitimate (honest) users.

4. Any neighbor node may obey (ignore) the announced limit rate and send Interest packets

without any restriction from the upstream interface. Our algorithm after twice refusing the alert message will band the incoming Interest packets from the corresponding interface for a long time period.

Input: `AlertMsg`, `timestamp` of alert generation, `reduced rate` and `wait time` from interface j in router i (r_i^j)
Result: (1) adaptive pushback reaction and (2) pushback alert message generation

```
1  counter_j = 0 // initial counter for generating wait time in interface j
2  φ = 5% // reduction threshold of Interest rate
   /* section:  adaptive pushback reaction                                   */
3  if AlertMsg is Pushback alert message then
4      if Verify(AlertMsg.signature) and IsFresh(AlertMsg.timestamp) then
           /* Pushback reaction                                              */
5          foreach local interface j do
6              new rate = unsatisfied rate of j × announced reduced rate ;
7              if φ < new rate then
                   /* intensify the penalty                                  */
8                  Decrease(interface j, new rate, announced wait time);
9              else
                   /* reset to original setting                              */
10                 Increase(interface j, original rate) ;
11 else
12     Drop(AlertMsg) ;
13     return ;
   /* section:  Pushback alert message generation                            */
14 if (predictor module reports the adversary consumer (neighbor) in local interface j) then
15     if (time from last sent AlertMsg to local interface r_i^j < current local time) then
           /* Pushback alert message generation                              */
16         new time interval = 2^counter_j ;
17         AlertMsg = (current timestamp of alert generation, current unsatisfied rate in local interface j,
           new time interval);
18         Send(AlertMsg to r_i^j);
19         counter_j = counter_j + 1;
20 else
       /* reset to original setting                                          */
21     counter_j = 0 ;
22     Increase(interface j, original rate) ;
```

Algorithm 1: Unsatisfied-based pushback algorithm

At the next iteration of the unsatisfied-based pushback mechanism, legitimate user(s) will be able to gradually improve their satisfaction rate and sending Interest packets on both router `A` and `B`. After applying the alert message in router `A`, the Interest rate of the adversary will be decreased to around 63% in the next iteration. It allows bandwidth usage be consumed for 2nd legitimate user, that it will considerably led to the increasing of legitimate Interests rate. If the adversary continues its misbehaving in the next times, the ascending wait time strategy will increase the penalty rate

of the illegal Interest packets. Hence, `eth1` and `eth2` interfaces in router `A` will get through and return Data, eventually resulting in a full allowance in the link between the routers `A` and `C`.
The Pseudo-code of the unsatisfied-based pushback mechanism is shown in Algorithm 1. In this algorithm, the `Decrease` function decreases the Interest rate from corresponding interface with announced parameters. After normal traffic prediction, the `Increase` function sets the default Interest rate on the corresponding interface in the next time interval. The `IsFresh` function checks the freshness of the alert message when there is no previously alert message.

reaction regarding to misbehaving producer

If the predictor module predicts a misbehaving producer from an interface j, we build an adaptive and simple forwarding strategy. The main goal is to retrieve data via the best performance path(s), and to quickly recover packet delivery problem by the other (possible) legitimate producers. When a predictor module in a router i reports a misbehaving producer in an interface j, the interface status changes to the *unavailable* (can not bring data back) and will be deactivated for a predefined time interval. This type of forwarding strategy can increase the data retrieving chance for awaiting Interest packets in the PIT table by changing the forwarding path. We apply the wait time strategy from the misbehaving consumer section (see section 5.6.2). After normal prediction in the next time intervals, the interface status changes to *available* (can bring data back). It means, it is ready for forwarding Interest packets via this interface. It is expected that in the next time intervals, when there is no any legitimate producer to satisfy the corresponding Interest packets in an interface j, the predictor module reports misbehaving consumer (neighbor) from upstream interface j, where Interest packets are susceptible to be illegal traffics. Then, the rate of incoming Interest packets should gradually decrease in upstream interface j based on our ascending penalty mechanism in previous subsection.

5.7 Experimental results and evaluation

In this section we report the experimental evaluation of countermeasures presented in Section 5.6. Our countermeasures are tested over two considered topologies in Figs. 5-7 and 5-8. Each router implements the proposed detection technique discussed in Section 5.6.1 and adaptive reaction technique discussed in 5.6.2.

We report the results based on the five conditions: baseline, attack (no countermeasure), our

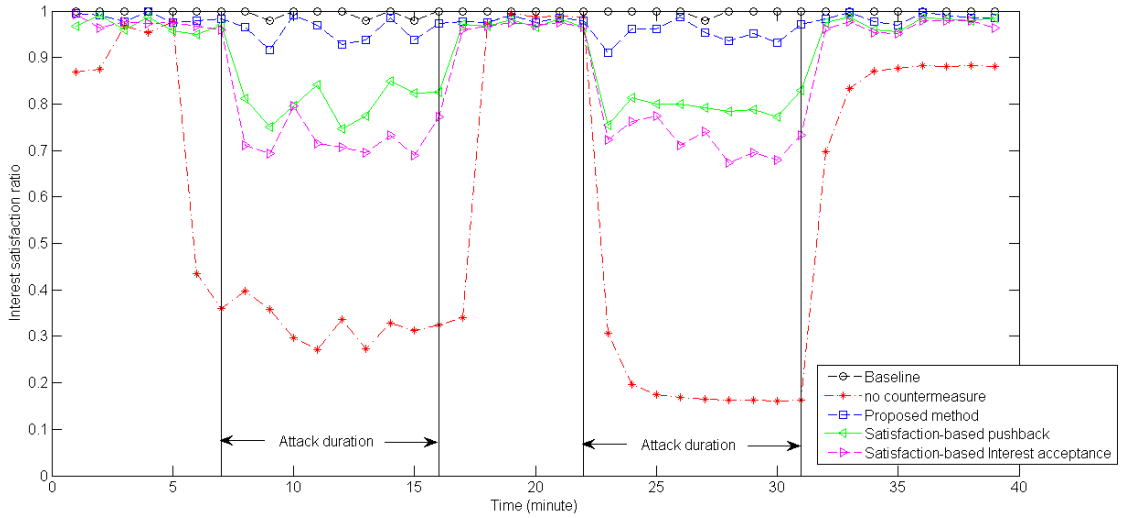


Figure 5-13: Interest satisfaction ratio for legitimate users in DFN

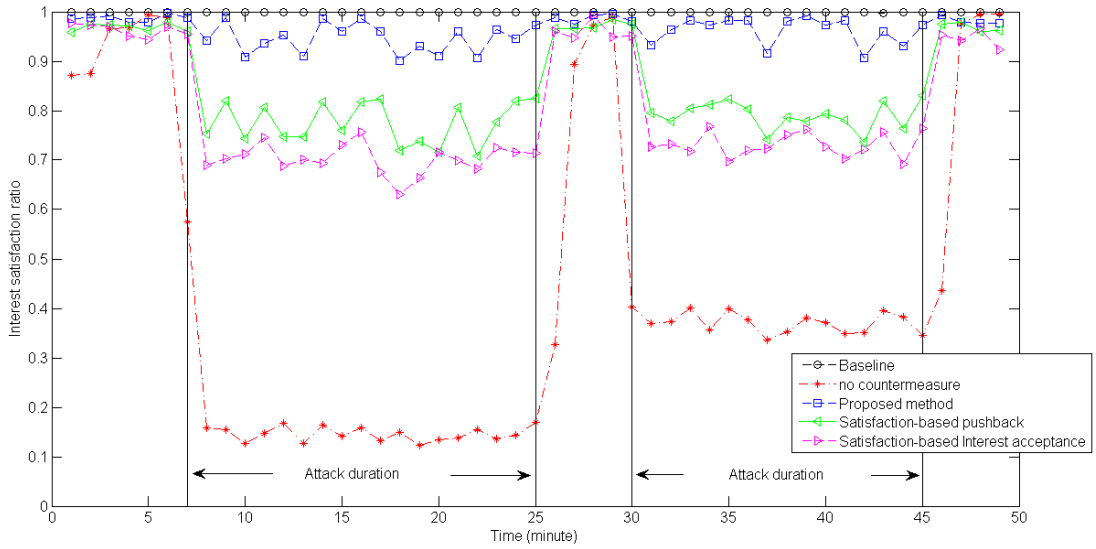


Figure 5-14: Interest satisfaction ratio for legitimate users in AT&T

proposed method, and two DoS mitigation methods applied in this work including satisfaction-based pushback and satisfaction-based Interest acceptance from [17]. Figs. 5-13 and 5-14 show the average Interest satisfaction ratio for legitimate users within 10 runs in DFN-like and AT&T topologies, respectively. Our results show that the proposed intelligent hybrid algorithm (proactive detection and adaptive reaction) is very effective for shutting down the adversary traffics and preventing legitimate users from service degradation by the accuracy more than 90% during the

attack.

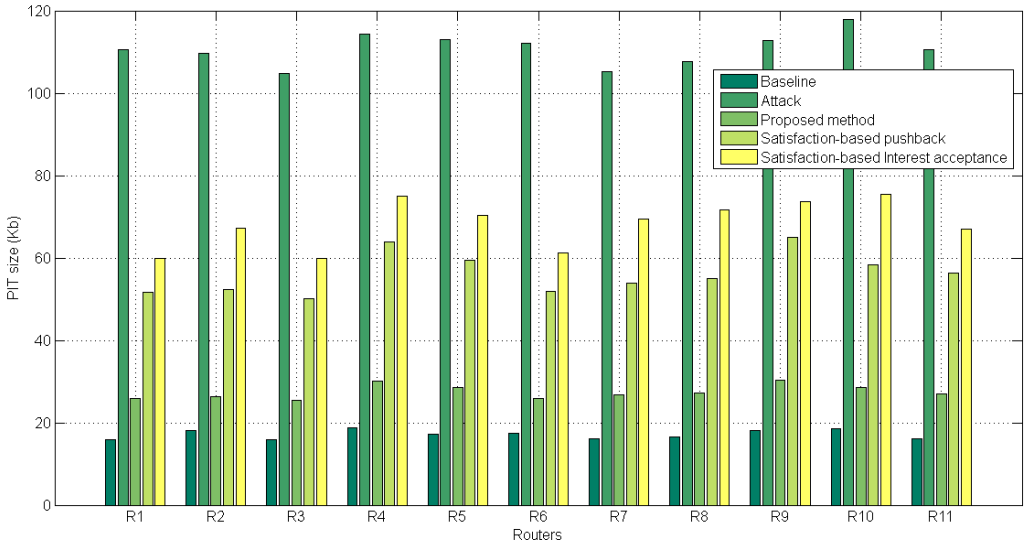


Figure 5-15: PIT usage with countermeasures in DFN

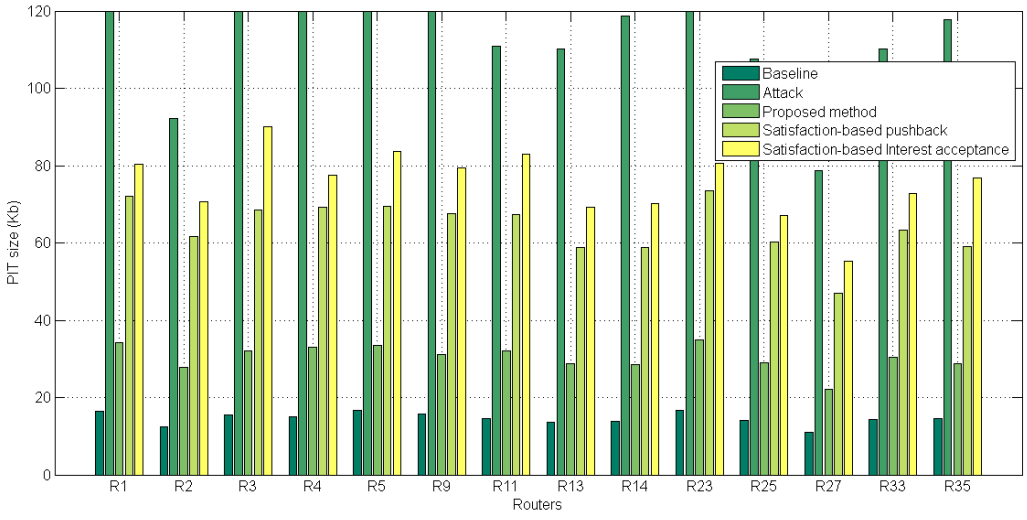


Figure 5-16: PIT usage with countermeasures in AT&T

Figs. 5-15 and 5-16 demonstrate the average PIT usage within 10 runs in five considered conditions in DFN-like and AT&T topologies, respectively. Our results show that there is a significant benefit of the proposed countermeasure in reduction of PIT usage in presence of an adversary. In Figs. 5-17 and 5-18 we show the average number of content received (throughput) in DFN-like and

AT&T topologies, respectively. The results show that the proposed countermeasure is effective and efficient in presence of adversary. For clarity, we just report measurements for those routers that are affected by the attacks for AT&T topology. The most routers in both considered topologies exhibit an interesting behavior. The proposed mechanism in both steps (detection by an intelligent hybrid method and reaction by enforcing explicit limitations against adversaries) offers visibly promising performance in presence of adversary.

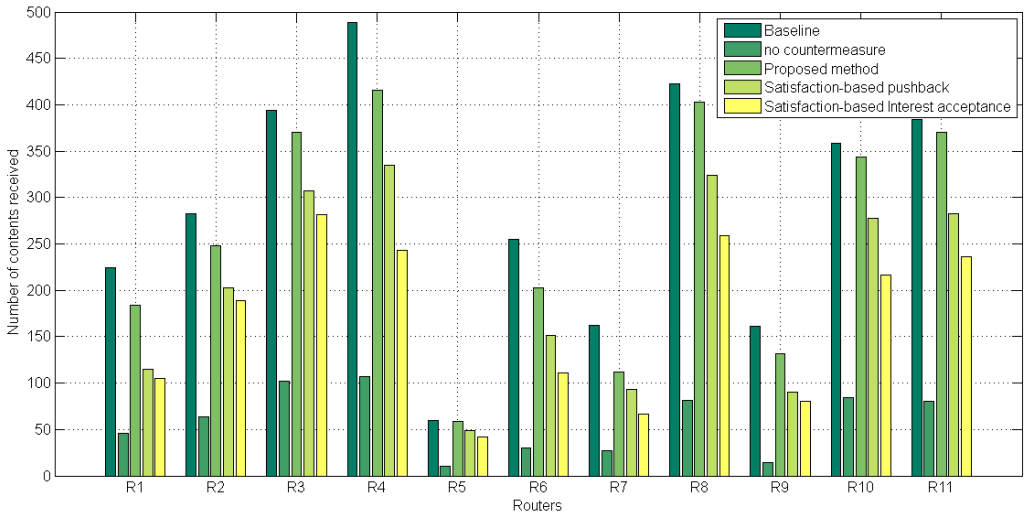


Figure 5-17: Effects of countermeasures in DFN (Throughput)

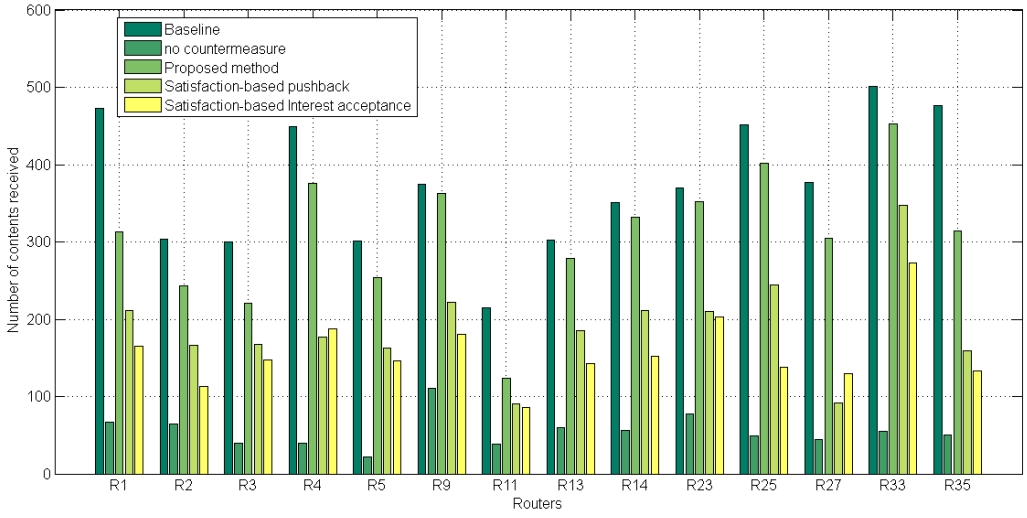


Figure 5-18: Effects of countermeasures in AT&T (Throughput)

5.7.1 Two facts of DoS/DDoS mitigation in NDN

Experimental results and analysis show that the two conditions can cause a DoS mitigation method degrades service to legitimate consumers:

1. Producers can misbehave by dropping incoming Interest packets or signing data packets with the wrong keys as they are unwilling to forward data packets to legitimate consumers. We conducted this experiment in AT&T topology by producer P0 between 7-25 seconds of simulation run (see Table 5.14), in which consumers C0, C1, C4, C5, C7, C8, C11, C12, and C13 request their desirable data packets from that producer. When our proposed predictor in an interface reports a misbehaving producer, the corresponding interface status changes to unavailable and will be deactivated. Consequently, the data retrieving chance for awaiting Interest packets increases by changing the forwarding path towards the producer P1 except C1 and C8, because other consumers can be satisfied with more than one producer. It is an expected behavior from a predictor until there is either no misbehaved producer or an extra well-behaved producer. As a result of this condition, it seems reasonable to decrease the rate of Interest packets for legitimate consumers.

2. A DoS mitigation technique should be able to detect malevolent behaviors and any deviation ideally long before the destructive traffics build-up and block the network traffics belonging to the attackers without denying services to legitimate consumers in a timely manner. If a mitigation technique cannot detect DoS attacks in a timely manner, the generated overload by DoS attacks prevents the resource from responding to legitimate traffic, or slows its response so significantly a (high) percentage of the legitimate Interest packets are completely disrupted. In this way, DoS mitigation techniques often create false positives (false alarms) by dropping legitimate Interest packets or enforcing limitations incorrectly against legitimate consumers. False positive refers to normal traffics when are incorrectly decreased by enforcing explicit limitations from our proposed unsatisfied-based pushback mechanism and other considered countermeasures during DoS/DDoS attacks (see section 5.6.2). Table 5.17 demonstrates the average rate of false positives obtained by our method and other applied countermeasures within 10 runs. This table shows that the proposed mitigation method is characterized by an extremely low false positive rate as compared to other countermeasures which is important when dealing with DoS/DDoS attacks. It can be concluded that the proposed intelligent hybrid predictor is able to detect DoS/DDoS attacks in a timely manner to prevent service

degradation for legitimate users.

Table 5.17: Comparison of false positive rate (mean of 10 runs)

Topology	No countermeasure	Satisfaction-based Interest acceptance	Satisfaction-based pushback	Our method
DFN	59.78%	21.05%	14.47%	6.44%
AT&T	66.43%	24.29%	19.13%	9.26%

A future work is needed in the classification of legitimate users' traffics as either good (non-malicious), bad (malicious) or low and high prone to attack traffics (non-malicious, but with the same properties as malicious traffics).

5.7.2 Discussion

A new intelligent hybrid algorithm (**proactive detection and adaptive reaction**) for mitigating DoS attacks in Named Data Networking has been proposed. The first part (**detection**) of this new algorithm (Fig. 5-2) consists of two phases: training/optimization and prediction/classification. In the training phase, an hybrid optimization algorithm has been developed to resolve the hybrid learning problem of RBF neural networks using multiobjective evolutionary algorithm and PSO. The first step of this phase adjusts RBF units' centers based on NSGA II through two conflicting objectives: well-separated centers (by Davies Boulding Index (DBI)) and local optimization of centers (by Mean-Squared Error (MSE)). Second step of this phase trains units' widths and output weights using PSO. This step tunes and adjusts widths and weights simultaneously by the well-separated centers from the previous step. In the prediction phase, a simple and an effective prediction algorithm has been designed to classify the new input patterns in their actual classes. This part of our hybrid algorithm has been successfully applied to define a more accurate RBF classifier over the NDN traffic flows as well as distinguish intelligently DoS attack traffics.
Convergence of the proposed RBF classifier (predictor) is studied for finding global and optimal classification of different benchmarking data sets as Wine, Iris, Ionosphere and Zoo. We applied the single-objective approach in Tables 5.2-5.5 (training units' centers) and Tables 5.6-5.9 (training units' widths, output weights and calculating the misclassification error), and our conflicting two-objective approach in Figs. 5-3-5-6 (Pareto front of the units' centers solutions) and Tables 5.10-5.13 (training units' widths, output weights and calculating the misclassification error). Experimental results confirm the accuracy and the robustness of the proposed approach based on the several performance metrics: MSE, Standard Deviation (Std.), Standard Error of Mean (SEM), Confidence Interval (CI 95%) and the number of incorrect classification.

The feasibility and efficiency of the proposed RBF classifier (predictor) method was compared to four well-known and frequently used optimization algorithms. Tables 5.6-5.9 demonstrate the final results, using PSO, Genetic Algorithm (GA), Imperialist Competitive Algorithm (ICA) and Differential Evolution (DE). The proposed algorithm in this dissertation outperforms all applied methods based on the (near) optimal results in the number of correct classification, MSE and Std. criteria. It can be concluded that the proposed intelligent hybrid algorithm is able to construct more accurate and well-tuned RBF classifier for (near) optimal classification of input patterns.
Although, the proposed method and other methods use different parameter settings. Our method was repeated 5 times and others were repeated 20 times independently to find the global results in the training/optimization phase; therefore, the effect of tuning parameters on performance of the methods are studied. We repeated the proposed training phase less than other methods to show that our two-objective approach is able to tune and adjust RBF parameters faster and more accurate than other methods.
The proposed intelligent classifier was successfully adopted in the detection phase of our countermeasure (see section 5.6.1). After constructing the intelligent hybrid classifier (predictor) module, an **adaptive reaction** mechanism by enforcing explicit limitations against adversaries was proposed to mitigate potential DoS/DDoS attacks in NDN (see section 5.6.2). Finally, convergence, feasibility and efficiency of the proposed algorithm (proactive detection and adaptive reaction) is studied for finding the optimal placement of RBF units' centers, units' widths and output weights and measuring the suitable performance over two network topologies including DFN-like (Fig. 5-7) and AT&T (Fig. 5-8). The results were promising as compared to two recently proposed DoS mitigation methods from [17] based on the average of Interest satisfaction ratio for legitimate users, the PIT usage, the number of received contents (throughput), and a very low false positive rate over 10 simulation runs.

5.8 Conclusion

NDN is a newly proposed future Internet architecture which it is important to address its resilience in face of DoS/DDoS attacks. We examined the most current instances of DoS/DDoS attacks to show that an adversary with limited resources can serve service degradation for legitimate users. We then introduced our intelligent hybrid algorithm for proactive detection of DoS attacks and adaptive reaction for mitigating. In the detection phase, a combination of multiobjective evolu-

tionary optimization and RBF neural network has been applied. This approach consists of two phases: training/optimization and prediction/classification. In the training phase, we investigate the implementation of a multiobjective approach and PSO in the design of RBF neural network in order to improve the accuracy of classification problems. We apply NSGA II to determine the Pareto solutions of RBF units' centers in terms of the well-separated centers through DBI and their local optimization through MSE. Then, the optimization and tuning of the units' widths and output weights are accomplished by using the PSO, where the each particle encodes a set of widths and weights. Moreover, the structure of this step is simple and easy to implement, yet very effective in terms of several performance criteria. In the prediction phase, we employ a simple algorithm to classify efficiency the new input patterns with the minimum misclassification error. This hybrid algorithm was applied on four benchmarking data sets to verify the algorithm accuracy and robustness in classification problems.

Subsequently, after constructing a more accurate classifier (detector), we performed a simple adaptive reaction algorithm by enforcing explicit limitations against adversaries which was very effective and efficient for shutting down the attackers with the robust recovery from network failures and accuracy more than 90% in terms of the average of Interest satisfaction ratio for legitimate users, the PIT usage, the number of received contents (throughput), and a very low false positive rate over 10 simulation runs.

Chapter 6

ACCPndn: Adaptive Congestion Control Protocol in NDN by learning capacities

Congestion takes place in NDN routers when the number of arrival data packets is higher than the queue's capacity which causes an overflow in the routers' buffer [43, 44, 50]. When this happens a high data packet loss and increase in the end-to-end delay occur affecting negatively on the performance, stability and robustness of the network [45, 46]. This leads to under-utilization of the available resources and degradation of throughput and quality of service [47, 48].

This difficulty has recently motivated researchers to explore ways of congestion control in NDN. Some of the relevant contributions are [43, 44, 195, 196, 197, 198, 199]. The main weak points of the proposed methods are: a too high sensitivity to their control parameters as well as the inability to predict congestion traffic well enough in advance. This will often bring unfair bandwidth sharing, network instability, packet loss, additional delay and so on [200, 201]. The first goal of any method against congestion can be the early detection (ideally long before the problematic traffic builds up) of its existence. If the congestion problem can be recognized in advance, changing network parameters can possibly prevent such costly network breakdowns. Network traffic prediction plays an important role in guaranteeing quality of service in computer networks [202]. The prediction of network traffic parameters is feasible due to a strong correlation between chronologically ordered values [200]. Their predictability is mainly determined by their statistical characteristics including self-similarity, multi-scalarity, long-range dependence (LRD) and a highly non-linear nature [203]. Prediction

algorithms can be embedded into network communications to improve the global performance of the network by anomaly detection, proactive congestion detection (or avoidance), and allow a better quality of service by a balanced utilization of the resources [202, 204, 205]. Contributions from the areas of operational research, statistics and computer science have lead to forecasting methods. In particular, the field of Time Series Forecasting (TSF) deals with the prediction of a chronologically ordered values [206, 207]. The goal of TSF is to model a complex system as a black-box in order to predict the systems behavior based on the historical data [200, 208].

In this dissertation, we develop ACCPndn (Adaptive Congestion Control Protocol in Named Data Networking) which is a new congestion control protocol with learning capacities. The ACCPndn focuses on two phases for congestion control before building up in NDN. **The first phase -adaptive training-** learns from the past breakdowns to how to detect the problem beforehand. This phase allows to identify the source of the congestion together with the amount of congestion. This phase uses Timed-Lagged Feedforward Neural Network (TLFN) approach. The TLFN adopts a multilayer perceptron ANN (Artificial Neural Network) and time series forecasting (TSF) [209, 210]. The major advantages of neural networks in time series forecasting are their flexible nonlinear modeling capability that there is no need to specify a particular model form and high data error tolerance [211, 212]. A Back-Propagation is a most popular NN algorithm (BPNN) to determine and adjust network parameters, weights and biases. Despite the advantages of BPNN, it has some drawbacks that the most important one being their poor trainability. It might fall to local minima and cause overfitting and failure of the network training [213, 214]. There is a recent trend to train BPNN with bio-inspired optimization algorithms for different applications [215, 216, 217]. In this work, in order to improve the performance of BPNN, a new combined algorithm namely Particle Swarm Optimization (PSO) and Genetic Algorithm (GA) is presented to optimize the weights and the biases of network, and to prevent trapping in local minima. The results show that our proposed combination of PSO/GA with TLFN (TLFN + PSO-GA) performs better than the GA/PSO, PSO and GA in terms of the applied performance criteria.

When the source(s) and the amount of congestion are identified, they are sent to the second phase for congestion control before building up. **The second phase -fuzzy avoidance-** performs a fuzzy decision-making approach to proactively respond to network congestion rather than simply wait for a congested queue to overflow and the tail drop all subsequently arriving data packets. The application of fuzzy decision-making techniques to the problem of congestion control is suitable due to the difficulties in obtaining a precise mathematical (or a formal analytical) model, while

some intuitive understanding of congestion control is available [218, 219]. Its use allows to regulate effectively the incoming Interest packets in each routers' interface.

The main objective of the proposed protocol is to enable a stable equilibrium and satisfy some basic requirements which are characterized by the utilization of multiple paths and few packet drops. The second objective is to present a scalable and fast convergence properties with respect to varying delays, bandwidths, traffic patterns, and number of users at different times utilizing the network. The evaluation through simulations shows that ACCPndn can quickly and effectively respond against congestion problems in a timely manner and performs successfully even in the large networks as compared to two recent congestion control mechanisms namely NACK [199] and HoBHIS [43] in terms of the applied performance metrics.

6.1 Related Work

The congestion difficulty has recently motivated researchers to explore ways of congestion control in NDN. Some of the relevant contributions are [43, 44, 195, 196, 197, 198, 199]. The main weak points of the proposed methods are: a too high sensitivity to their control parameters as well as the inability to predict congestion traffic well enough in advance. This will often bring unfair bandwidth sharing, network instability, packet loss and additional delay [200, 201]. Among all the proposals, HobHIS (Hop-by-hop Interest Shaping mechanism) [43] and NACK [199] are recently applied in NDN scenarios, which they slow down Interest packets on the hop after congestion rely on backpressure to alleviate congestion.

HobHIS is a rate-based congestion control mechanism for CCN. This method computes the available capacity of each CCN router in a distributed way in order to shape their conversations Interest rate and therefore, adjust dynamically their Data rate and transmission buffer occupancy. NACK mechanism also works as follows. When a NDN node can neither satisfy nor forward an Interest (e.g., the upstream node has no available interface to forward the Interest or its downstream link to forward the Data packet is congested), it sends an Interest NACK back to the downstream node. When a neighbor node receives NACK message, it sets a limit on Interest packets over its downstream interface. The problem of Interest NACK is that it consumes the PIT entries on the way, and if a neighbor is under congestion, this packet is discarded.

6.2 Time series analysis

A time series is a sequence of data points or time ordered observations $(y_1, y_2, ..., y_t)$ in which each period recorded at a specific time t. A time series forecasting is the use of a model to predict future values based on previously observed values [208, 220]. For time series feature extraction, a trace (set of events) should be converted into *time series* with the regular time intervals. This will be used as an input for the purpose of prediction. The time series would be described by the following formula [221]:

$$x(t+\tau) = f(x(t), x(t-\tau), ..., x(t-n\tau)) \tag{6.1}$$

Where, f is a Time Series Forecasting (TSF) method, τ is specified time delay and n is some integer values. The TSF methods have found applications in very wide area including finance, business, computer science, medicine, physics, chemistry and many interdisciplinary fields. Most time series modeling methods provide only a reasonable, but limited accuracy and suffer from the assumptions of stationarity and linearity [222]. To improve TSF with nonlinear characteristics, several researchers have successfully employed artificial neural networks [223, 224].

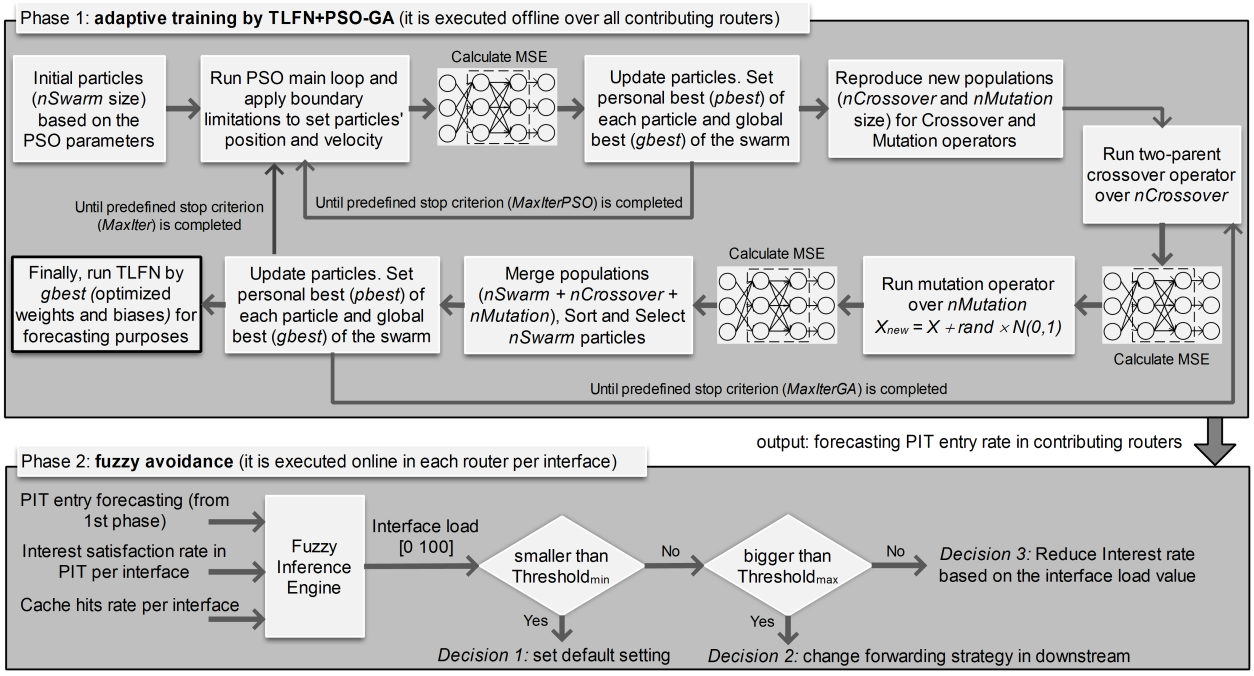


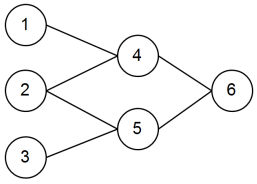
Figure 6-1: Two steps of the ACCPndn

6.3 The proposed method: ACCPndn

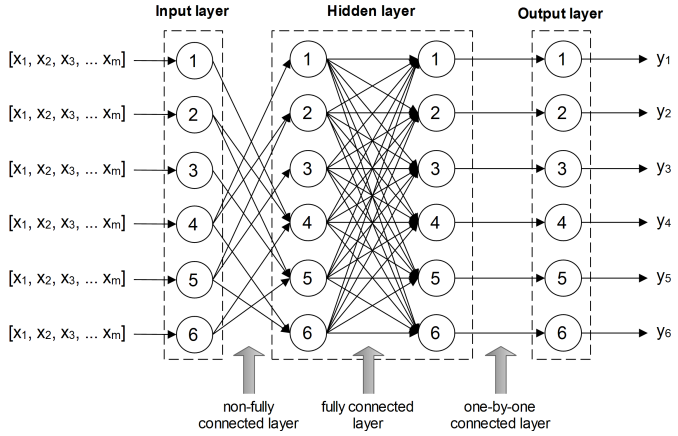
In this section, we introduce our proposal, ACCPndn: a two phase framework for congestion control. The first phase being adaptive training and the second one fuzzy avoidance. This proposal aims to avoid congestion before it builds up. A diagram of the ACCPndn is shown in Fig. 6-1.

6.3.1 Phase 1: Adaptive Training

For TLFN modeling, we try to forecast the rate at which entries are added to the PIT table in NDN routers (In the rest of the work we use the term *PIT entries rate* interchangeably). Since all exchange in NDN is Interest/Data (one Interest packet will be replied back with one data packet), the rate of new PIT entries (the expected amount of returned data) could be a good indicator of a future congestion in the router's buffer. With the prediction of new PIT entries rate in the next time interval, the arrival rate of data packets at that are susceptible to create congestion can be forecast in a timely manner. In this phase, routers learn what are the kind of many low and high frequent traffic patterns which cause an overflow in the routers' buffers and create congestion.



(a) example of data communication network



(b) neural network representing the network connectivity

Figure 6-2: Reflection of the connectivity of data communication network in a neural network considered

We adopt the nodes connectivity of the NDN routers for defining the number of neural network layers, the connectivity of layers and the number of neurons in TLFN. Fig. 6-2 shows the logic of the proposed neural network connectivity. The neural network used consists of $m \times n$ input nodes, two hidden layers and one output layer containing n node. The m denotes the number of features in the input layer and n denotes the number of the contributing routers. The input

features correspond to the PIT entries rates for a set of consecutive time intervals.

For the connectivity between input layer and first hidden layer, the neural network reflects the connectivity of the data network by only allowing links between neurons representing adjacent nodes. For instance in Fig. 6-2a, n would be six because only six nodes are actually contributing traffic. Hence, the connection between node 1 in the input layer and node 4 in the first hidden layer derived from their connectivity in the data communication network. We only allow connectivity between nodes neighbors from input neurons (nodes) to represented neurons (nodes) in the hidden layer. On the other hand, according to the definition of the NDN data communication, when a node's cache cannot satisfy Interest packets, the node forwards Interest packets toward the origin's content through intermediate nodes. Thus, there are data communications by routing Interest and returning back data packets through intermediate routers. To address this issue, we provide an extra layer in the hidden layer by a fully-connection communication. The output of the neural network is a representation of PIT entries rate forecasting in each considered routers which are suspected causing the problem in the next time interval. For instance, an output of [50 0 0 30 90 0] would mean the first, fourth and fifth routers will be faced with the new PIT entries with the probability of 50%, 30% and 90% in the next time interval, respectively.

The constructed TLFN is trained offline in order to create a pattern by learning the PIT entries rate in contributing routers in the next time interval. Afterward, we create a control agent containing this trained neural network to being placed in the simulation environment. A higher level view of our architecture is a network with a control agent existing somewhere on a node in the network. This controller should easily gather required input information (historical PIT entries rate) from contributing routers in a predefined time interval. When the controller predicts the rate of PIT entries in contributing routers, it sends the prediction rate to the corresponding routers. Then, each router per interface performs a fuzzy decision making to control or prevent the probable packet drop in a timely manner (see section 6.3.2).

For TLFN modeling, we propose a new technique, an hybrid of particle swarm optimization and genetic algorithm during the TLFN training. The TLFN + PSO-GA integrates PSO and GA to tune (optimize) weights and biases of TLFN. The computational procedures for the proposed TLFN + PSO-GA are as follows:

`Step 1:` Normalize data samples into [0 1]:

$$X = \frac{X_i - min(X_i)}{max(X_i) - min(X_i)} \tag{6.2}$$

Where $min(X_i)$ and $max(X_i)$ are the minimum and maximum value of data samples and X_i denotes the real value of each vector.

`Step 2:` Define some initial parameters: $c_1 = c_2 = 2$, $\omega_{min} = 0.3$, $\omega_{max} = 0.9$, $SwarmSize = 25$ (number of the particles), $MaxIter = 500$ (maximum number of the main iteration), $MaxIter_{PSO} = 4$ (maximum number of the PSO iteration), $MaxIter_{GA} = 2$ (maximum number of the GA iteration), $Var_{min} = 0$ (lower bound of variables -particles' position-) and $Var_{max} = 1$ (upper bound of variables).

`Step 3:` Randomly initialize a group of particles in size of $SwarmSize$. Each particle includes position (the weights and the biases of TLFN) and velocity.

`Step 4:` Calculate the particles' fitness value. The performance (fitness) function is Mean Square Error (MSE) between the actual target and output of the neural network. Afterwards, update the *pbest* (personal best of each particle) and the *gbest* (global best of the swarm).

`Step 5:` Repeat the following loop until the target or maximum sub-iteration of PSO ($MaxIter_{PSO}$) is reached:

`Step 6:` Apply PSO main loop:

1. Update velocity by Eq. 2.3.

2. Apply velocity limits: If the velocity of a particle exceeds the minimum or maximum allowed speed limit, it should bring such particle back into the search space:

$$\begin{aligned} Velocity &= max(Velocity, Vel_{Min}) \\ Velocity &= min(Velocity, Vel_{Max}) \end{aligned} \tag{6.3}$$

   Where $Vel_{Min} = 0.1 \times (Var_{Max} - Var_{Min})$ and $Vel_{Min} = -Vel_{Max}$ are the minimum and maximum values of the particles' velocity.

3. Update position by Eq. 2.4.

4. Apply velocity reflection: it allows those particles' position that move toward the outside the boundary $[Var_{Min}\ Var_{Max}]$ to move back into the search space multiplying particles' velocity by -1.

5. Apply position limits: If the position of the particle exceeds the boundary of search space,

this such particle should bring back into the feasible search space:

$$\begin{aligned} Position &= max(Position, Var_{Min}) \\ Position &= min(Position, Var_{Max}) \end{aligned} \tag{6.4}$$

6. Evaluate fitness function.

7. Update the personal best (*pbest*) and the global best (*gbest*).

`Step 7:` If $MaxIter_{PSO}$ is not reached to its predefined value go to the `step 6`; otherwise, if all particles updated and $MaxIter_{PSO}$ is reached to its predefined value go to the next step.

`Step 8:` Apply real-coded GA operators: reproduction, crossover, mutation, selection:

1. *Reproduction:* reproduce a number of individuals (chromosomes) for crossover and mutation:

$$nCrossover = \left\lfloor (pCrossover \times \frac{nSwarm}{2}) \times 2 \right\rfloor \tag{6.5}$$

Where, $pCrossover$(=0.7) is crossover percentage, $nCrossover$ is the number of parents (Offsprings).

$$nMutation = \lfloor pMutation \times nSwarm \rfloor \tag{6.6}$$

Where, $pMutation$(=0.2) is mutation percentage, $nMutation$ is the number of mutants.

2. *Crossover:* apply two-point crossover over two random selected particles for the number of $nCrossover$ particles (individuals). It creates new population set as $pop_{Crossover}$. Calculate fitness function for $pop_{Crossover}$.

3. *Mutation:* apply mutation over random selected particle for the number of $nMutation$ particles:

$$X_{new} = X \times rand \times N(0, 1) \tag{6.7}$$

It creates new population set as $pop_{Mutation}$. Calculate fitness function for $pop_{Mutation}$.

4. *Selection:* merge populations ($[nSwarm\ pop_{Crossover}\ pop_{Mutation}]$), sort merged populations based on their fitness values, and select the first $nSwarm$ particles.

`Step 9:` Update *pbest* and *gbest*.

`Step 10:` If the sub-iteration of GA algorithm ($MaxIter_{GA}$) is not reached to its predefined value

go to the `step 8`; otherwise, go to the next step.

`Step 11:` Update ω by Eq. 5.1.

`Step 12:` If the maximum iteration ($MaxIter$) or predefined target is not reached, go to the `step 5`; otherwise, the *gbest* includes the optimized parameters (weights and biases) of TLFN + PSO-GA and the network can be used for forecasting.

6.3.2 Phase 2: Fuzzy Avoidance

As we explained earlier, a controller based on the trained TLFN + PSO-GA is placed in the network to gather required information for PIT entries rate forecasting in the contributing routers. In this phase, a Fuzzy Inference System (FIS) is applied to prevent probable packet drop in susceptible routers to congestion problem before building up. We deploy a combination of three criteria where each interface in contributing routers gathers them in each time interval:

1. PIT entries rate forecasting in each router through the first phase (training module).

2. Interest satisfaction rate in PIT per interface.

   We take into consideration the unique feature of NDN, i.e., one Interest packet will only return back one data packet in reverse path of the Interest packet. In a NDN router, if the number (rate) of incoming Interest packets in PIT be varied widely with the number (rate) of incoming data packets for Interest satisfaction, there might be some abnormal Interests or congestion. If the number of incoming Interest packets be more than the PIT size, PIT will apply its replacement policy for new incoming Interest packets. If the PIT removes old PIT entries to accommodate new Interest packets, returned data packets for removed PIT entries become *unsolicited*. It might be led to congestion due to the crowding of unsolicited data packets. On the other hand, current unsatisfied Interest packets in PIT table may also reach their timeout (lifetime expiration) and become *dangling state* [199]. Such dangling state can potentially block other Interest packets. When a router is congested, it can potentially lead to dangling state for unsatisfied Interest packets. We maintain the Relative Strength Index (RSI) for every interface of a router to reflect the Interest satisfaction ratio in PIT:

$$RSI = \frac{\hat{I_n}}{\hat{I_n} + \hat{D_n}} \tag{6.8}$$

   Where $\hat{I_n}$ and $\hat{D_n}$ are the average number of the placed Interests in the PIT table and the

incoming data packets of an interface at the nth time, respectively. We apply the standard Exponentially Weighted Moving Average (EWMA) with α coefficient (a lower α counts widely earlier observations) [225] to calculate the placed Interest packets in PIT and the incoming data packets periodically, e.g., once a second:

$$\begin{aligned} \hat{I_n} &= \alpha.I_n + (1-\alpha)\ \hat{I_{n-1}} \\ \hat{D_n} &= \alpha.D_n + (1-\alpha)\ \hat{D_{n-1}} \end{aligned} \tag{6.9}$$

Where I_n and D_n are the total number of incoming Interest in PIT and incoming data packets of an interface in the nth period. Generally, the reasonable RSI of every interface should be around 50%.

3. Cache hits rate per interface.

   If an interface satisfies the most arrival Interest packets by cache, it should be significantly considered in the decision making. Otherwise, if an interface of a suspected router to congestion just fills up PIT table, it should be negatively considered in decision making. We apply Exponential Moving Average (EMA) to calculate the new average of the cache hits ratio in the recent nth time interval. It applies weighting factors which decrease exponentially (the weighting for each older datum decreases exponentially):

$$Cache\ hits = \frac{C_1 + (1-\alpha)C_2 + ... + (1-\alpha)^{n-1}C_n}{1 + (1-\alpha) + ... + (1-\alpha)^{n-1}} \tag{6.10}$$

   Where, C_1 denotes to the number of current cache hit and C_n is the number of the cache hits in the recent nth time.

These three criteria themselves involve fuzziness because of bringing vague, imprecise or uncertain information along in problem solving. For instance, the exact value 0.7 (or 70%) cannot show that it is very high percentage or partially high percentage of occurrence an event (e.g., one of the three applied criteria). With the uncertainty modeling, fuzziness subjective, incomplete and imprecise data can be described. Thus, a fuzzy control system can construct a control system in terms of many-values logic dealing with reasoning, i.e., approximate rather than fixed or exact.

The output of proposed FIS is the amount of interface load in a router. Interface load means the portion of the corresponding interface in filling PIT table entries up in that router. The structure

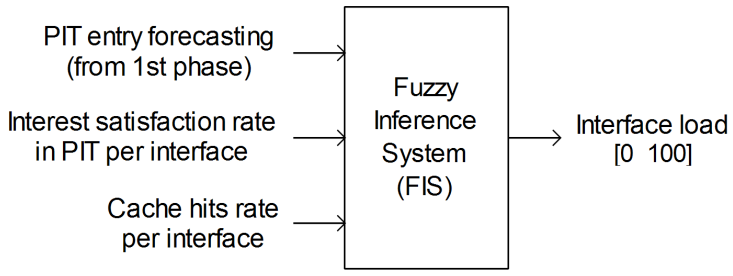


Figure 6-3: Proposed FIS for fuzzy congestion control

of the proposed fuzzy inference system is depicted in Fig. 6-3. When the controller forecasts PIT entries rate and sends them to the corresponding routers, the proposed fuzzy control system is called in each contributing router to apply three types of control per interface including (1) readjust Interest packets rate, (2) effect on forwarding strategy in the downstream and (3) set default configuration. We set two thresholds ($threshold_{min}$ and $threshold_{max}$) to make decision regarding to the crisp output of the proposed FIS. We set $threshold_{min}$ and $threshold_{max}$ to 20% and 80%, respectively. The procedure of the fuzzy decision making approach is depicted in Fig. 6-4.

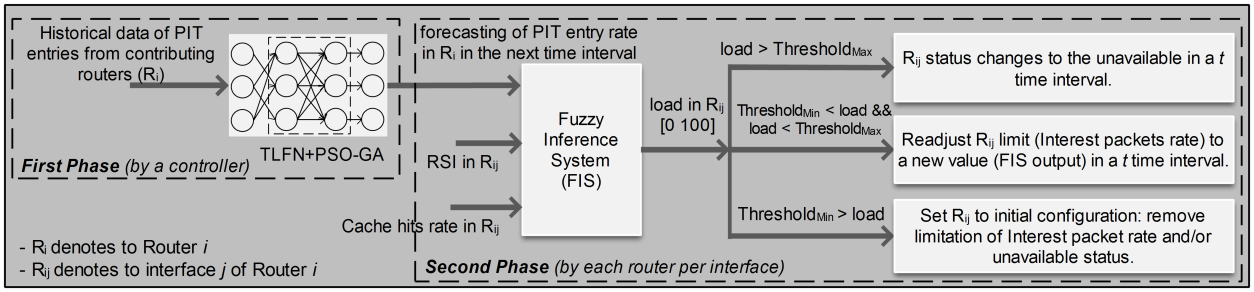


Figure 6-4: ACCPndn: the first phase by a controller and the second phase by routers per interface

According to Fig. 6-4, if the output of FIS from R_{ij} is bigger than $threshold_{max}$, there is more likely that interface j in router i will face to a malignant congestion (very high packet drop in router's buffer) and the best decision can be changing the interface j status to the unavailable (cannot bring data back) and will be deactivated for a predefined time interval t in order to the downstream (neighbor router) does not send Interest packets. It allows to downstream to forward its Interest packets via other available links. If the output of the FIS is between [$threshold_{min}$ $threshold_{max}$] in an interface j, there is more likely by controlling the rate of Interest packets in the downstream in a predefined time interval t, the probability of packet drop reduces considerably. Finally, if the output of FIS in an interface j in router i is lower than $threshold_{min}$, there is more

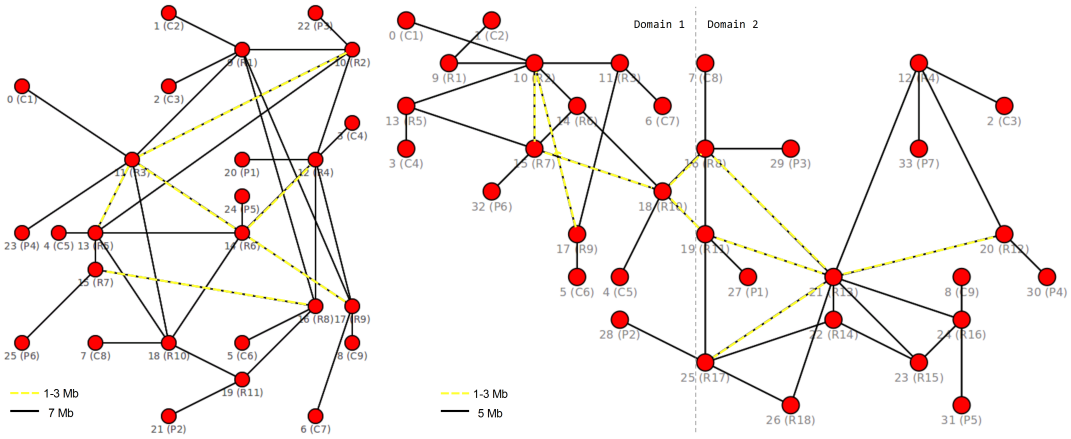
likely that there is no congestion (packet drop) in the next time interval and the configuration of an interface j can be set to its default values: set original Interest packet rate and/or make available (can bring data back) the downstream interface j.

We apply an ascending penalty for definition of time interval t during two restrictions (Interest packets rate and interface unavailability). If in an interface j in a time T, a restriction is needed, counter sets to 1 sec. If in the next time interval $T+1$ a same restriction again be reported, counter sets to 2 sec. Our ascending penalty method is in $2^{counter}$. Initially, counter sets to 0 and increase linearly in each time interval. The counter will set to the initial value when the output of FIS be lower than $threshold_{min}$ in the next time interval. This ascending penalty intensifies the restriction to avoid packet drop in the long-term.

6.4 Experimental Setup

We use simulations to quantify effect of congestion and its countermeasure. In this work, we used the open-source ndnSIM [53] package, which implements NDN protocol stack for NS-3 network simulator (`http://www.nsnam.org/`), to run simulations for evaluating the performance of proposed method. ndnSIM simulation environment reproduces the basic structures of a NDN node (i.e., CS, PIT, FIB, strategy layer, and so on). The proposed adaptive training method (first phase) was implemented by the MATLAB software on an Intel Pentium 2.13 GHz CPU, 4 GB RAM running Windows 7 Ultimate. This algorithm deployed to C++ project integrating as a C++ shared library using the MATLAB compiler. Then, this C++ program was integrated with ndnSIM environment to be able to adjust in the simulation environment. The proposed fuzzy avoidance phase was also implemented with C++ in ndnSIM environment. We choose two metrics to quantify the effectiveness of our countermeasure. First criterion is the average of utilization of multiple paths (retry alternative paths) to mitigate congestion in bottleneck links. The indicator for evaluating the utilization of bottleneck links and alternative links is the rate of *InData*. InData denotes a number of arrival data in an interface. InData guarantees that this amount of data packet was actually transferred over the channel during the congestion. Second criterion is the average of packet drop rate. If the proposed ACCPndn considerably decreases or totally removes packet drops, it can be concluded our proposed method is highly effective at mitigating/removing packet drops.

Our experiments are performed over two topologies shown in Fig. 6-5. Fig. 6-5a corresponds to DFN-like (Deutsche Forschungsnetz as the German Research Network) [193], and Fig. 6-5b



(a) DFN-like topology

(b) Switch-like topology

Figure 6-5: Considered network topologies

corresponds to the Switch-like (Swiss Education and Research Network) [194]. We use the symbols `Cx`, `Px`, and `Rx` to represent `x`-th consumer, producer and router, respectively. In spite of various arguments and experiments, there is no typically and properly justification for NDN parameters and they have specified based on authors' experiences and designs [10]. Therefore, the applied control parameters of the ACCPndn are iteratively learned under various network environments to make a real data communication in considered topologies. For scalability reasons, it is important for a congestion control protocol to be able to maintain their properties as network characteristics change. We thus set nodes' PIT size to a range of [700 1000] randomly, while the Interest expiration time was set to the default timeout of 4000 ms. We set the link delay and queue length parameters to different values for every node in the simulated topologies. In particular, we set delay and queue length to the range of 1ms-3ms and 200-500 for both considered topologies, respectively. A various PIT entries replacement policies (i.e., perform different actions when limit on number of PIT entries is reached) were adopted randomly over the nodes in both considered topologies including persistent (new entries will be rejected if PIT size reached its limit), random (when PIT reaches its limit, random entry will be removed from PIT) and least-recently-used (the oldest entry with minimum number of incoming faces will be removed). Moreover, The nodes' cache capacity was set to 1000 contents, while the caching replacement policies were set to randomly over the nodes including least-recently-used, FIFO (first-input first-output) and random policies. We ran various traffic patterns within the randomize and Zip-Mandelbort (α is in range of [0.4 0.9]) distribution. For both distribution methods, we applied uniform and exponential patterns of distribution. The

expected frequency of Interest packets generation sets to a range of 100-1000 packets per second. Each consumer changes five times the frequency randomly during simulation run. We assign some bottleneck links with yellow dash lines in both considered network topologies in Fig. 6-5. We set bandwidth in the range 1 Mb/s to 3 Mb/s randomly.

Table 6.1: Interest-Data communications

DFN-like		Switch-like	
Consumer	Producer	Consumer	Producer
C1	P1	C1	P1
C2	P2	C2	P2
C3	P5	C3	P3
C4	P4	C4	P1, P4
C5	P2, P3	C5	P3
C6	P3	C6	P2, P6
C7	P4, P6	C7	P5
C8	P1, P5	C8	P5, P6
C9	P4, P6	C9	P7

Table 6.1 is also shown the Interest-Data communications. Finally, we investigate the transient behavior of the utilization and the packet drop rate at the bottleneck links and alternative paths for congestion control and/or avoidance during simulation run.

6.5 Experimental Results

In this section we report the experimental evaluation of ACCPndn presented in Section 6.3.

6.5.1 Phase 1: adaptive training

Training phase consists of a collection of time ordered observations of PIT entries in contributing routers from two considered NDN network topologies in Fig. 6-5. Depending on the time scale, there are four main forecasting types including real-time, short-term, middle-term and long-term [226]. The real-time forecasting is the most appropriate type of PIT entries forecasting where samples not exceed a few seconds and requires an on-line forecasting and reaction in a timely manner.

The choice of the input time intervals has a crucial effect in the PIT entries forecasting performance. A small number of time intervals will provide insufficient information, while a high number of intervals will increase the probability of irrelevant input features [208]. Several configurations based on our observations of PIT entries fluctuation in considered network topologies were set. Five different sliding windows were adopted based on the predefined time interval (we set 1 sec):

- **DFN-like:**

  1. {1 2 3 6 7 8 11 12 13 24 25 26 38 39 40}
  2. {1 2 3 6 7 8 11 12 13 24 25 26}
  3. {1 2 3 4 5 8 9 10 11 12}
  4. {1 2 3 7 8 9 12 13 14 26 27}
  5. {1 2 3 6 7 8 10 11 12 23 24 25}

- **Switch-like:**

  1. {1 2 3 4 8 9 10 11}
  2. {1 2 3 4 5 10 11 12 13 14}
  3. {1 2 3 6 7 8 10 11 12}
  4. {1 2 3 10 11 12 20 21 22}
  5. {1 2 3 7 8 9 16 17 18}

Due to the application of different configuration settings on considered network topologies, we run the experiments 20 times independently to evaluate the proposed training method (see section 6.3.1) in terms of applied performance metrics. The performance of the forecasting model in training phase is evaluated by the Mean Square Error (MSE) and Symmetric Mean Absolute Percent Error (SMAPE):

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(T_i - O_i)^2 \tag{6.11}$$

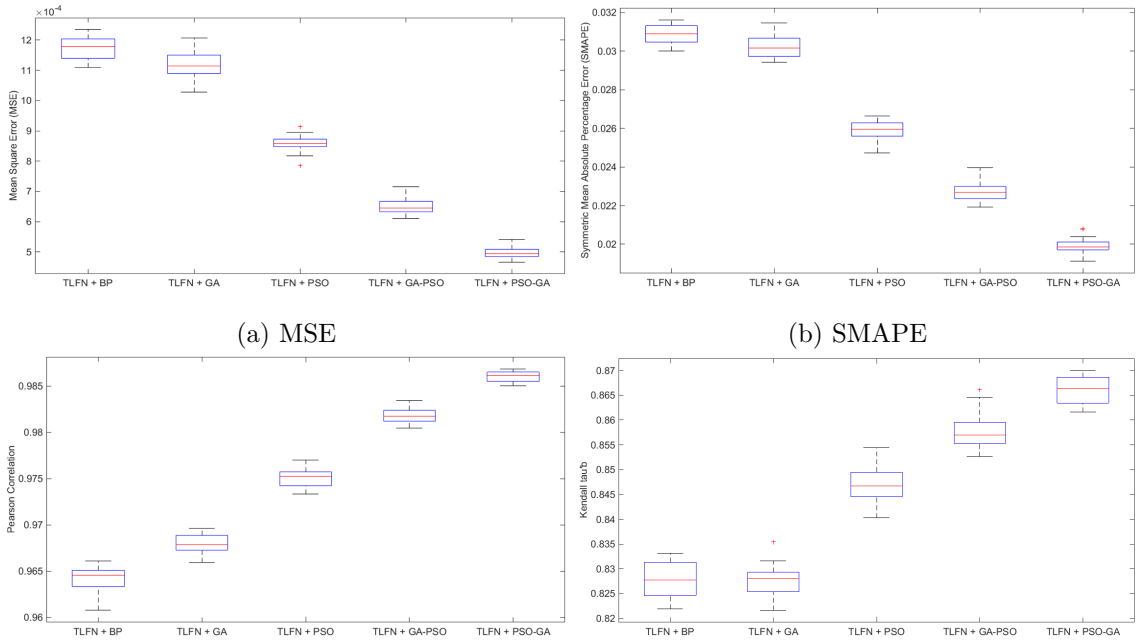
$$SMAPE = \frac{\sum_{i=1}^{n}|O_i - T_i|}{\sum_{i=1}^{n}(O_i + T_i)} \tag{6.12}$$

where T is the actual value and O is the forecast value. The MSE quantifies the difference between values implied by an estimator and the true values of the quantity being estimated. The SMAPE is an alternative to Mean Absolute Percent Error (MAPE) when there are zero or near-zero demand for items. It is a measure of accuracy of a method for constructing fitted time series values in statistics. In case of the 20 simulation runs, forecasting methods likely yield different results. Therefore, the forecasting results are investigated by statistical tests if these differences are significant [129, 130]. We have used Pearson correlation coefficient and Kendall's tau'b with 99% of confidence level implemented by MATLAB software. Moreover, the time series data from considered NDN topologies were divided into three contiguous blocks as training (70% of the series) to fit (train) the

forecasting models, validation (next 15% of the series) to evaluate the forecasting accuracy during the training and test (remaining 15% of series) to confirm the forecasting accuracy after training. In order to confirm the quality and performance of the TLFN + PSO-GA from ACCPndn, four algorithms for updating network parameters (weights and biases) are integrated with TLFN, i.e., BP learning (TLFN + BP), Genetic Algorithm (TLFN + GA), Particle Swarm Optimization (TLFN + PSO), and TLFN + GA-PSO.

DFN-like topology

Fig. 6-6 shows the optimal accuracy derived from the Table 6.2. The box plot of TLFN + PSO-GA in Figs. 6-6a and 6-6b is comparatively short as compared to other methods. This suggests that overall MSE and SMAPE values are relatively small and have a high level of agreement within 20 runs.



(a) MSE

(b) SMAPE

(c) Pearson correlation

(d) Kendall tau'b

Figure 6-6: The forecasting results in DFN (1st sliding window)

TLFN + PSO-GA provides better results than TLFN + GA-PSO which it confirms that GA performs a good local search for better particles movement in the swarm to minimize significantly both applied cost functions. Indeed, TLFN + PSO and TLFN + GA obtain quite good forecasting errors as compared to TLFN + BP by MSE and SMAPE in the almost all 20 runs, respectively. As expected, the hybridization of the optimization algorithms reveal a better performance as compared

Table 6.2: Accuracy measurement in DFN-like topology (mean of 20 runs)

Sliding window	Blocks	Criteria	TLFN + BP	TLFN + GA	TLFN + PSO	TLFN + GA-PSO	TLFN + PSO-GA
1	train	MSE	0.001133	0.0010422	0.00077854	0.00057117	0.0004177
		Std.	0.032429	0.032183	0.026545	0.021826	0.019204
		SMAPE	0.032611	0.031494	0.027822	0.023684	0.020903
	validation	MSE	0.0014095	0.001341	0.00096241	0.00072668	0.00058018
		Std.	0.036732	0.036689	0.029929	0.024589	0.022442
		SMAPE	0.031568	0.031744	0.024147	0.02386	0.020229
	test	MSE	0.0016052	0.0016073	0.00099027	0.00080267	0.00061721
		Std.	0.039529	0.039888	0.029454	0.025269	0.023385
		SMAPE	0.033571	0.031391	0.026276	0.023686	0.019503
2	train	MSE	0.0011237	0.0010276	0.00088078	0.00059297	0.00043985
		Std.	0.032953	0.031852	0.028764	0.022689	0.020326
		SMAPE	0.032451	0.031818	0.02971	0.024655	0.021336
	validation	MSE	0.0014157	0.0012751	0.00095915	0.00075775	0.00061771
		Std.	0.03706	0.035498	0.030155	0.024475	0.024081
		SMAPE	0.034949	0.034798	0.027445	0.025248	0.023069
	test	MSE	0.0015441	0.0015185	0.0011129	0.00090368	0.00068794
		Std.	0.038555	0.038259	0.032808	0.028117	0.025138
		SMAPE	0.03431	0.033378	0.028012	0.023322	0.022331
3	train	MSE	0.0011498	0.00092411	0.00092566	0.00067046	0.00047723
		Std.	0.033333	0.029759	0.030318	0.024637	0.021558
		SMAPE	0.033361	0.029302	0.029659	0.025782	0.02164
	validation	MSE	0.0013421	0.00093305	0.0010567	0.00076505	0.00045326
		Std.	0.036546	0.02955	0.032554	0.026471	0.020936
		SMAPE	0.034284	0.030544	0.031482	0.025328	0.02064
	test	MSE	0.0017045	0.001354	0.0014934	0.00095665	0.00084002
		Std.	0.041123	0.036713	0.038663	0.030416	0.028713
		SMAPE	0.032024	0.032466	0.032112	0.023657	0.023157
4	train	MSE	0.0010178	0.0010685	0.00071116	0.00057548	0.00042858
		Std.	0.03085	0.032591	0.024529	0.021784	0.018987
		SMAPE	0.031173	0.033236	0.026122	0.023645	0.020206
	validation	MSE	0.00123	0.0014125	0.0010411	0.00077095	0.00047968
		Std.	0.033612	0.037608	0.030615	0.025082	0.020201
		SMAPE	0.029963	0.032922	0.025833	0.023803	0.019677
	test	MSE	0.0015572	0.0016926	0.0011888	0.00085998	0.00070696
		Std.	0.037736	0.041193	0.03181	0.025839	0.024333
		SMAPE	0.031421	0.034397	0.025582	0.024964	0.021255
5	train	MSE	0.0012795	0.00090687	0.00093529	0.00067955	0.00051242
		Std.	0.035636	0.029365	0.030485	0.024934	0.022296
		SMAPE	0.034494	0.029723	0.030536	0.025032	0.022044
	validation	MSE	0.0013881	0.0010467	0.0012313	0.00079344	0.00059672
		Std.	0.037219	0.031785	0.035001	0.026413	0.023688
		SMAPE	0.034018	0.030513	0.031386	0.025968	0.021899
	test	MSE	0.0017149	0.0012372	0.0017008	0.00095329	0.0006225
		Std.	0.040976	0.034652	0.04131	0.029686	0.024188
		SMAPE	0.033203	0.031446	0.035687	0.026168	0.021309

to standalone TLFN trained by BP.

The short lower and upper whisker in the Figs. 6-6c and 6-6d mean that the results of several runs are not varied. As shown, the correlation between different applied algorithms with 99% of confidence level is strong and positive which is statistically significant. The significant statistical correlation between 20 runs is the proposed forecasting method in ACCPndn about 98.5% as compared to other algorithms ranging 96%-98.3%. Moreover, the values for concordance coefficient from Kendall's tau'b results are close to +1, which means that there is a large agreement between the forecasting results. The concordance coefficient of the proposed forecasting method in ACCPndn

is better than other algorithms ranging 86%-88%. Table 6.2 shows the accuracy analysis of all considered methods based on the three contiguous data blocks. This table confirms the accuracy and effectiveness of the proposed training method in ACCPndn as compared to other applied methods.

Table 6.3: Accuracy measurement in 1st domain of Switch-like (mean of 20 runs)

Sliding window	Blocks	Criteria	TLFN + BP	TLFN + GA	TLFN + PSO	TLFN + GA-PSO	TLFN + PSO-GA
1	train	MSE	0.0027635	0.0014026	0.0012365	0.00099692	0.00028537
		Std.	0.049521	0.03613	0.034767	0.029797	0.016897
		SMAPE	0.031476	0.022688	0.019648	0.018956	0.011175
	validation	MSE	0.0026484	0.0016247	0.0012282	0.00091903	0.00028195
		Std.	0.048312	0.039191	0.034395	0.028507	0.016824
		SMAPE	0.031969	0.023566	0.020324	0.018404	0.010395
	test	MSE	0.0025021	0.0014297	0.0011872	0.00093219	0.00029754
		Std.	0.047201	0.036734	0.034185	0.029122	0.017286
		SMAPE	0.029759	0.023319	0.019317	0.018784	0.011254
2	train	MSE	0.0031308	0.0016835	0.0015935	0.0011783	0.00064175
		Std.	0.051214	0.03928	0.037889	0.032353	0.024311
		SMAPE	0.032532	0.02476	0.022795	0.019996	0.015406
	validation	MSE	0.0029838	0.0018509	0.0022157	0.0013488	0.00071595
		Std.	0.050279	0.040629	0.043639	0.034572	0.026029
		SMAPE	0.031354	0.024493	0.025082	0.021603	0.015029
	test	MSE	0.0033835	0.0021883	0.0013979	0.0011571	0.00074926
		Std.	0.053024	0.044826	0.035782	0.030614	0.026482
		SMAPE	0.034141	0.027284	0.02147	0.021821	0.01703
3	train	MSE	0.0022935	0.0012346	0.0007924	0.0018308	0.00041769
		Std.	0.045562	0.034322	0.028163	0.037931	0.020247
		SMAPE	0.028708	0.021044	0.016288	0.026208	0.012339
	validation	MSE	0.0023131	0.0011145	0.00083097	0.0018396	0.00038474
		Std.	0.045365	0.033147	0.028872	0.039097	0.019515
		SMAPE	0.029805	0.019819	0.017029	0.023286	0.011485
	test	MSE	0.002415	0.0012499	0.00065984	0.0019827	0.00038495
		Std.	0.045935	0.033962	0.025743	0.037461	0.019302
		SMAPE	0.031351	0.023851	0.016036	0.025895	0.013148
4	train	MSE	0.0035232	0.0018208	0.0016862	0.0011025	0.00076384
		Std.	0.054189	0.040489	0.038967	0.030843	0.026023
		SMAPE	0.034915	0.025932	0.024281	0.01998	0.01652
	validation	MSE	0.0037375	0.0017994	0.001795	0.0010079	0.00074195
		Std.	0.056213	0.040786	0.039344	0.030075	0.026289
		SMAPE	0.034785	0.025561	0.025697	0.019636	0.016758
	test	MSE	0.0033995	0.0016328	0.0018524	0.0010152	0.00078523
		Std.	0.052888	0.038536	0.038898	0.030189	0.025749
		SMAPE	0.033843	0.024882	0.026077	0.019854	0.018404
5	train	MSE	0.0029474	0.0016809	0.0012904	0.0013499	0.00063743
		Std.	0.050302	0.039142	0.034625	0.034303	0.024057
		SMAPE	0.032465	0.024169	0.021003	0.021954	0.015227
	validation	MSE	0.0027138	0.0018163	0.0013819	0.0016331	0.00073568
		Std.	0.050049	0.041239	0.035878	0.036638	0.026195
		SMAPE	0.033539	0.025638	0.02258	0.02246	0.016764
	test	MSE	0.0033697	0.0022373	0.0013286	0.0010361	0.00069929
		Std.	0.053659	0.043984	0.035236	0.030421	0.025063
		SMAPE	0.033257	0.025781	0.021625	0.019432	0.015651

Switch-like topology

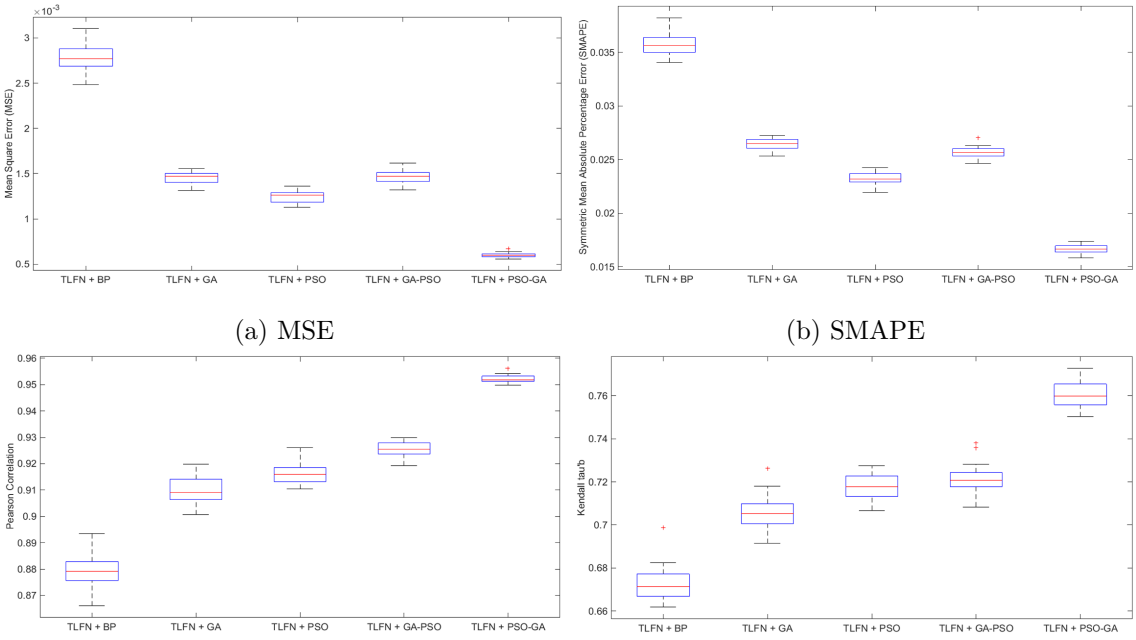
To assess the robustness and accuracy of our proposed adaptive learning method in ACCPndn, we apply a large network such as Switch-like (Fig. 6-5b) which consists of two different domains.

This large network can be formed and decomposed to two smaller domains. We have divided the Switch-like topology to two different domains where the learning control agent would be somewhere within each domain defined in the network. This decomposition of domains is depicted in Fig. 6-5b by vertical dot points. First domain consists of eight routers (R1, R2, R3, R5, R6, R7, R9 and R10) and the rest (ten routers) appears in the second domain. Tables 6.3 and 6.4 show the accuracy

Table 6.4: Accuracy measurement in 2nd domain of Switch-like (mean of 20 runs)

Sliding window	Blocks	Criteria	TLFN + BP	TLFN + GA	TLFN + PSO	TLFN + GA-PSO	TLFN + PSO-GA
1	train	MSE	0.00078039	0.00044251	0.00021409	0.00017433	0.00011026
		Std.	0.025422	0.020284	0.014198	0.01305	0.0096268
		SMAPE	0.018849	0.017016	0.010987	0.011487	0.0076496
	validation	MSE	0.00072864	0.00047302	0.00022576	0.00017944	8.9705e-05
		Std.	0.024926	0.020619	0.014449	0.013109	0.0088663
		SMAPE	0.014813	0.012406	0.0087261	0.0083463	0.005614
	test	MSE	0.00069999	0.00050847	0.00021194	0.00017898	0.00011247
		Std.	0.024613	0.021609	0.014142	0.012926	0.009727
		SMAPE	0.011392	0.010343	0.0073818	0.0061609	0.0044295
2	train	MSE	0.00076631	0.00044568	0.00022041	0.00017571	0.00010996
		Std.	0.025261	0.020386	0.01436	0.012979	0.0096048
		SMAPE	0.019583	0.018088	0.012323	0.01154	0.0076941
	validation	MSE	0.00079727	0.00053439	0.00019751	0.00014825	0.00011068
		Std.	0.026169	0.022109	0.013594	0.011986	0.0095327
		SMAPE	0.014879	0.011658	0.0085069	0.0075746	0.0058163
	test	MSE	0.00063682	0.00057829	0.00016224	0.00014329	0.00011101
		Std.	0.023495	0.023281	0.012444	0.011503	0.0097995
		SMAPE	0.010801	0.011493	0.0067112	0.006306	0.0044249
3	train	MSE	0.00072844	0.00047774	0.00023336	0.00017478	0.00011944
		Std.	0.024666	0.021061	0.014714	0.013013	0.010026
		SMAPE	0.019106	0.017303	0.012139	0.010906	0.0084846
	validation	MSE	0.00078986	0.00048046	0.00022541	0.00019941	8.9746e-05
		Std.	0.026048	0.02059	0.01467	0.013837	0.0087935
		SMAPE	0.014024	0.013093	0.0085223	0.0083947	0.0054335
	test	MSE	0.00062868	0.00037375	0.00023874	0.0001957	0.00011615
		Std.	0.023194	0.018667	0.014835	0.013796	0.0098775
		SMAPE	0.011816	0.009308	0.0065265	0.0066622	0.0046408
4	train	MSE	0.00074778	0.00045258	0.00023706	0.00016688	0.00011633
		Std.	0.025178	0.020635	0.014863	0.012552	0.0098328
		SMAPE	0.018742	0.017888	0.011695	0.011892	0.0082162
	validation	MSE	0.00068194	0.00044909	0.00024731	0.00019363	0.00011298
		Std.	0.024043	0.020479	0.015402	0.013691	0.0097654
		SMAPE	0.013292	0.011891	0.0089466	0.0089644	0.0055512
	test	MSE	0.00084249	0.00046449	0.00019938	0.00016508	9.4057e-05
		Std.	0.026658	0.020962	0.013698	0.012447	0.0089862
		SMAPE	0.013349	0.0102	0.0061604	0.006017	0.0042135
5	train	MSE	0.00074874	0.00046469	0.00023834	0.00018502	0.00011359
		Std.	0.024946	0.020733	0.014892	0.013297	0.009748
		SMAPE	0.019537	0.018489	0.01269	0.011497	0.0078074
	validation	MSE	0.00082051	0.00046043	0.00023944	0.00019519	0.00011848
		Std.	0.026099	0.020665	0.015184	0.013656	0.0101
		SMAPE	0.01426	0.010704	0.0089525	0.008051	0.0057206
	test	MSE	0.00064585	0.00049763	0.00020805	0.00016224	0.00011711
		Std.	0.023395	0.021497	0.013989	0.012436	0.010086
		SMAPE	0.010787	0.01028	0.0068957	0.0059625	0.0043928

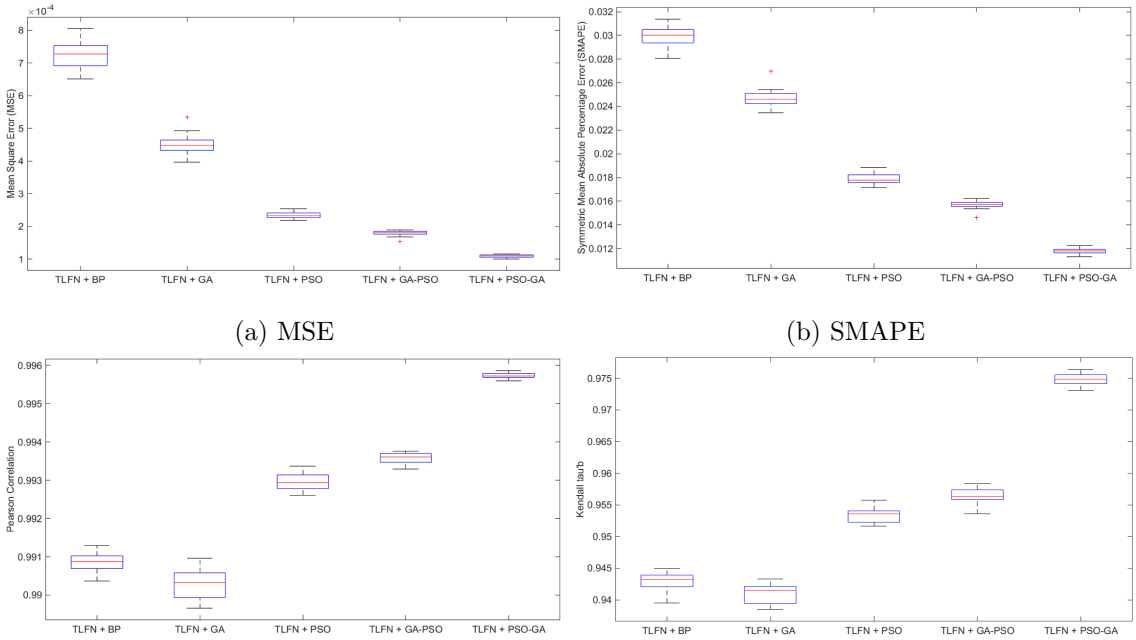
analysis of all considered methods based on the three contiguous data blocks. These table confirm the accuracy and effectiveness of the proposed training method in ACCPndn as compared to other applied methods.



(a) MSE (b) SMAPE

(c) Pearson correlation (d) Kendall tau'b

Figure 6-7: The forecasting results in 1st domain of Switch-like (1st sliding window)



(a) MSE (b) SMAPE

(c) Pearson correlation (d) Kendall tau'b

Figure 6-8: The forecasting results in 2nd domain of Switch-like (2nd sliding window)

The optimal forecasting performance of the first and the second domain is depicted in Figs. 6-7 and 6-8, respectively. The box plots clearly illustrate that the proposed TLFN + PSO-GA is able

to provide roughly appropriate performance in terms of the MSE and SMAPE.
The extensive analysis in Figs. 6-7c-6-7d and Figs. 6-8c-6-8d demonstrate that the correlation and the concordance coefficient between results by the proposed training algorithm is more significant than other applied hybridization during 20 runs. Similar to the results of DFN-like topology (see section 6.5.1), TLFN by PSO-GA training satisfies performance criteria more appropriate than by GA-PSO in Switch-like topology. Moreover, as we expected, the application of the optimization algorithms can perform a better performance as compared to standalone training by BP.

6.5.2 Phase 2: fuzzy avoidance

In this study, MATLAB fuzzy logic toolbox is used for fuzzy rule based decision-making regarding to congestion control. The second phase is structured with following components:

1. Three fuzzy set of input variables: (1) RSI rate in R_{ij}, (2) Predicted PIT entries rate in R_i and (3) Cache hits rate in R_{ij}; membership functions: *Low, Medium, High.*

2. A fuzzy set of output variable: Interface load; membership functions: *Negligible, Small load, Moderate load, Overloaded.*

3. Fuzzy membership functions: Since the *sigmoid* membership function [227, 228] is inherently open to the right or to the left; thus, it is appropriate for representing concepts such as "Low", "High" or "Negligible", "Overloaded". The *guass2mf* is also employed for middle linguistic values ("Medium", "Small load", "Moderate load"). The *gauss2mf* is a kind of smooth membership functions, so the resulting model has a high accuracy. It also covers the universe sufficiently which leads to the completeness of a fuzzy system [150]. The membership functions of input and output variables are shown in Figs. 6-9-6-12.

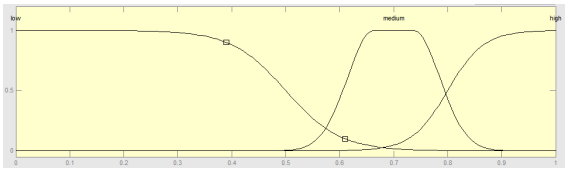


Figure 6-9: RSI membership functions (input)

4. Fuzzy rules: 27 rules. The nonlinear control-decision surface is shaped by the constructed rule base and the linguistic values of the inputs and output variables in Fig. 6-13. According to Fig. 6-13, the cache hit ratio plays an important role for decision making next to the

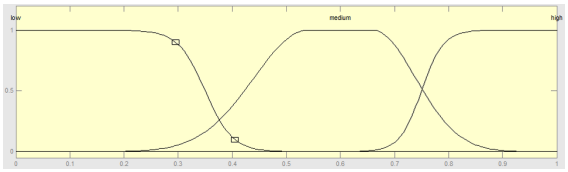


Figure 6-10: PIT entries membership functions (input)

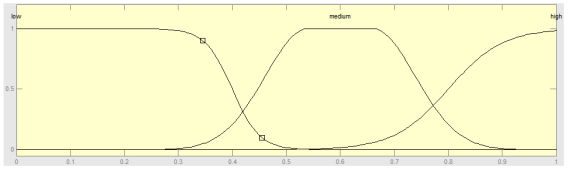


Figure 6-11: Cache hits membership functions (input)

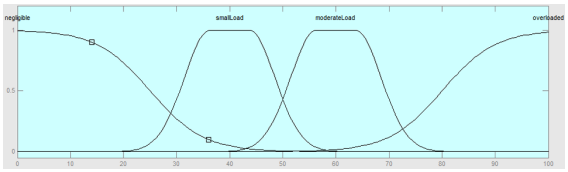
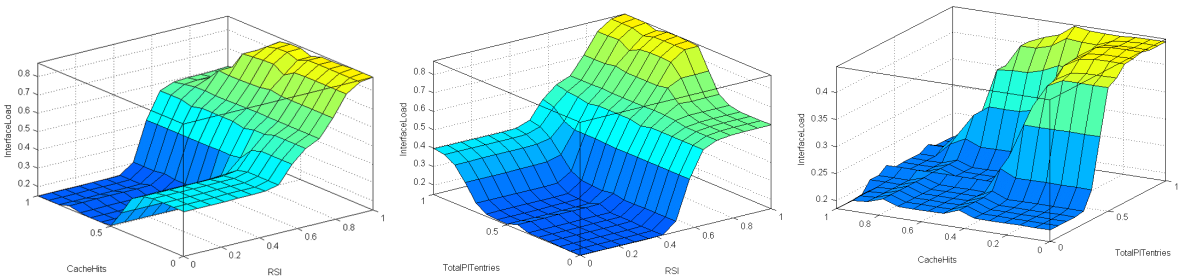


Figure 6-12: Interface load membership functions (output)



(a) RSI vs. Cache hits (b) RSI vs. Total PIT entries (c) Total PIT entries vs. Cache hits

Figure 6-13: The surface of the proposed fuzzy control system

RSI and PIT entries forecasting, while, the high cache hit ratio might bring the not highly interface load and the low cache hit ratio might bring the highly interface load. Moreover, the RSI criterion plays a main role as far as the increasing the RSI will lead to high interface load. A sample of constructed rule base is as follows:

❶ if `CacheHits` is *low* and `PITentry` is *high* and `RSI` is *high* ➔ `InterfaceLoad` is *Overloaded*

❷ if `CacheHits` is *medium* and `PITentry` is *high* and `RSI` is *medium* ➔ `InterfaceLoad` is *ModerateLoad*

❸ if `CacheHits` is *high* and `PITentry` is *high* and `RSI` is *medium* ➔ `InterfaceLoad` is *Smal-*

lLoad

❹ if `CacheHits` is *high* and `PITentry` is *medium* and `RSI` is *low* ➔ `InterfaceLoad` is *Negligible*

5. Inference: Mamdani fuzzy inference by fuzzy set operations as max and min for OR and AND, respectively.

6. Defuzzifier: Center of Gravity algorithm:

$$Center\ of\ Gravity = \frac{\int_{min}^{max} u\ \mu(u)\ d(u)}{\int_{min}^{max} \mu(u)\ d(u)} \tag{6.13}$$

Where, u denotes the output variable, μ is the membership function after accumulation, *min* and *max* are lower and upper limit for defuzzification, respectively. A sample solution area (fuzzy inference) of proposed fuzzy detection phase is given in Fig. 6-14.

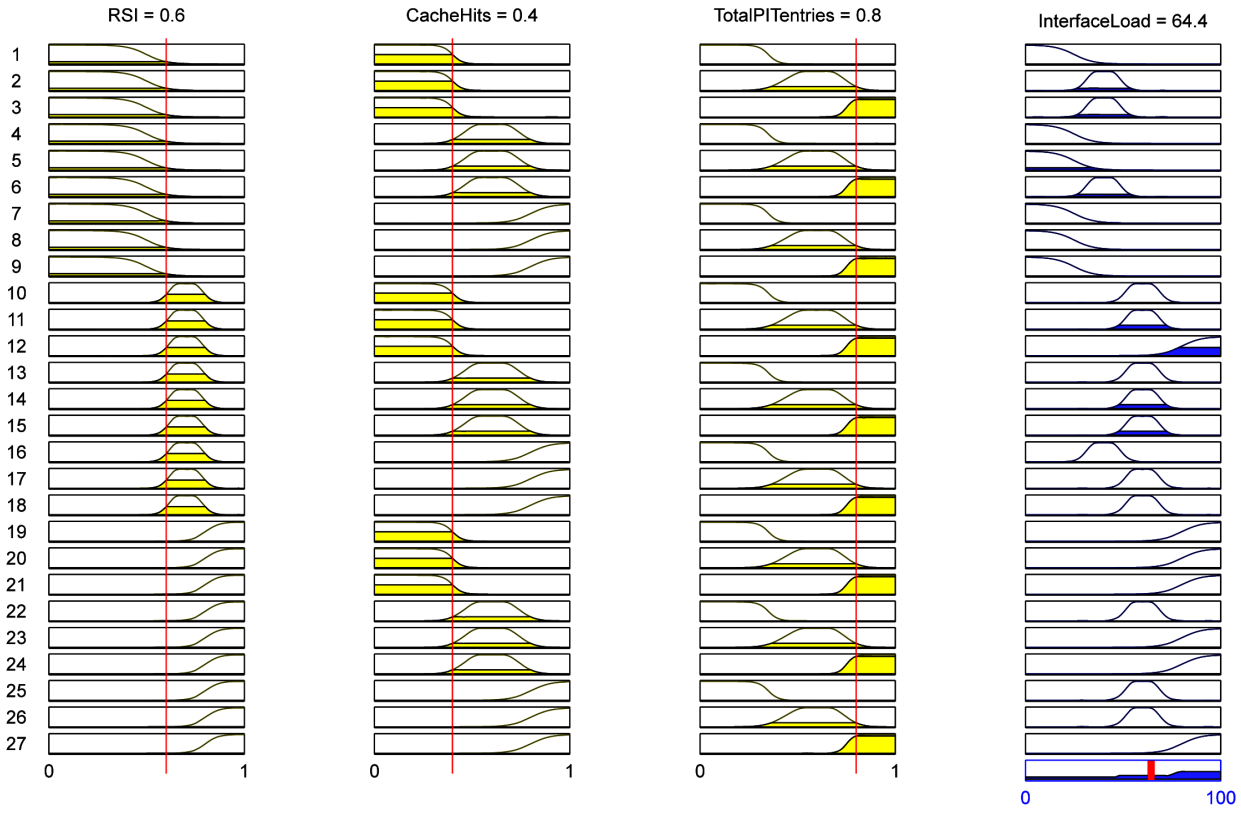
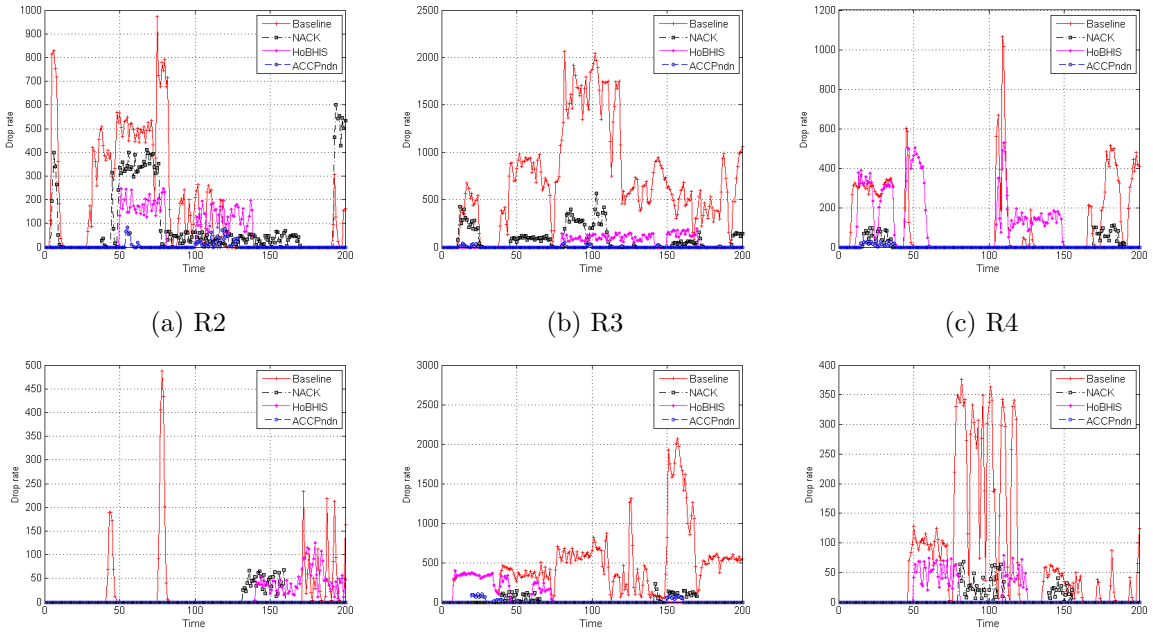


Figure 6-14: The sample solution area (fuzzy inference) of proposed fuzzy decision-making system

6.5.3 Results and Observations

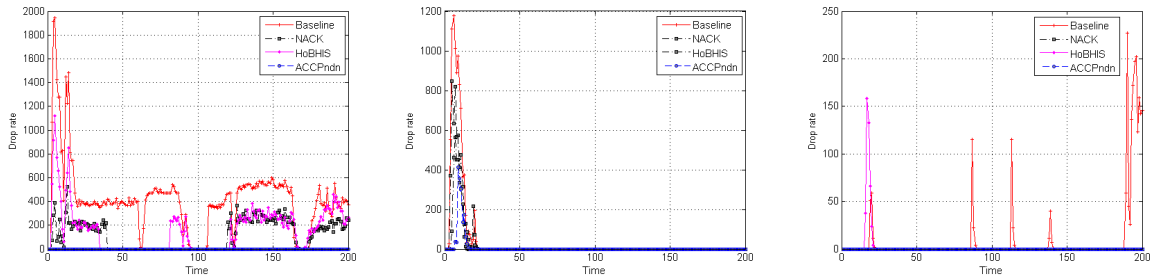
In this section, we demonstrate through simulations that ACCPndn satisfies applied performance criteria as compared to NACK [199] and HoBHIS [43] methods. The Interest NACK mechanism enables NDN routers to perform quick recovery per interface rate limit to avoid congestion on a local outbound interface. The Hop-by-hop Interest Shaping (HoBHIS) is also a congestion control mechanism by shaping the rate of the Interest which is currently sending towards content providers with routers. NACK, HoBHIS and ACCPndn also have a fundamental difference in the implementation of the algorithm. ACCPndn controls or avoids congestion traffic through an hybridization of TLFN, metahuristics and non-linear fuzzy logic-based control system to predict future PIT entries and perform an adaptive recovery whereas NACK and HoBHIS apply a rate limiting after arriving congestion traffic which prevents the link between the two nodes from being congested. The experimental results demonstrate that ACCPndn outperforms NACK and HoBHIS mechanisms sufficiently. In the training phase of ACCPndn we select the fourth, the first and the fifth time intervals configurations for DFN-like, the first domain and the second domain of Switch-like topologies, respectively. These time intervals perform better than others in (near) optimal configuration of TLFN + PSO-GA predictor based on the applied performance metrics within 20 runs.



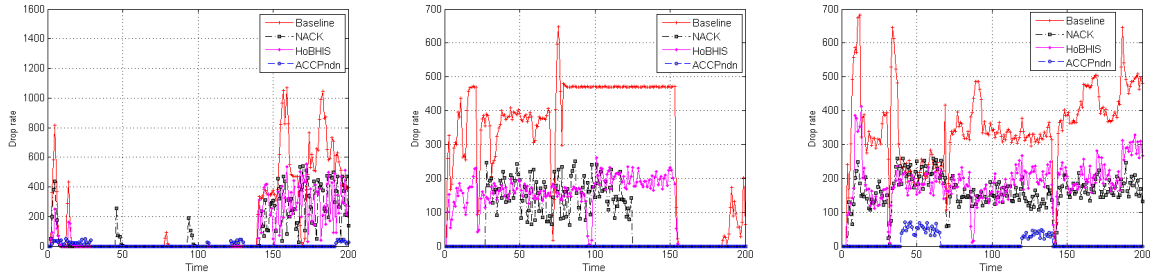
(a) R2 (b) R3 (c) R4

(d) R5 (e) R6 (f) R7

Figure 6-15: Average of Data drop in contributing routers' buffer in DFN-like topology



(a) R7 (b) R8 (c) R10



(d) R11 (e) R12 (f) R13

Figure 6-16: Average of Data drop in contributing routers' buffer in Switch-like topology

Table 6.5: statistics of packet drop in DFN-like topology (mean of 10 runs)

Routers	Methods	No. drop	Drop boundary	Mean	Std.	SEM (95%)	μ (95%)	σ (95%)
R2	Baseline	108	[1 971]	173.2488	237.2683	32.8018	[140.2479 206.2497]	[216.1195 263.0411]
	NACK	145	[1 600]	95.8657	151.1534	20.8966	[74.8422 116.8891]	[137.6805 167.5722]
	HoBHIS	69	[59 247]	52.5373	78.0588	10.7914	[41.6804 63.3943]	[71.101 86.5377]
	ACCPndn	37	[1 84]	5.1045	14.77	2.0419	[3.0502 7.1588]	[13.4535 16.3744]
R3	Baseline	179	[5 2063]	719.3085	536.6871	74.1958	[644.6624 793.9546]	[488.8498 594.9836]
	NACK	111	[1 565]	81.9303	116.5032	16.1063	[65.7263 98.1344]	[106.1188 129.1582]
	HoBHIS	83	[50 193]	46.7761	60.3131	8.3381	[38.3874 55.1649]	[54.9371 66.8644]
	ACCPndn	29	[1 39]	2.2289	7.2545	1.0029	[1.2199 3.2379]	[6.6078 8.0425]
R4	Baseline	89	[1 1065]	122.0796	193.2822	26.7208	[95.1966 148.9626]	[176.0541 214.2771]
	NACK	42	[15 109]	12.398	27.8455	3.8496	[8.5251 16.2709]	[25.3635 30.8701]
	HoBHIS	92	[1 528]	100.3881	146.412	20.2411	[80.0241 120.752]	[133.3616 162.3157]
	ACCPndn	23	[1 29]	1.8657	6.0495	0.83633	[1.0243 2.7071]	[5.5103 6.7067]
R5	Baseline	40	[1 487]	20.7413	67.6333	9.3502	[11.3344 30.1482]	[61.6048 74.9798]
	NACK	28	[13 68]	5.9801	16.0652	2.221	[3.7456 8.2146]	[14.6332 17.8102]
	HoBHIS	62	[3 125]	14.0348	26.4268	3.6534	[10.3592 17.7104]	[24.0712 29.2973]
	ACCPndn	0	-	-	-	-	-	-
R6	Baseline	194	[17 2066]	494.1692	404.1683	55.8754	[437.9547 550.3836]	[368.143 448.0703]
	NACK	52	[1 232]	20.6368	43.2485	5.979	[14.6215 26.6521]	[39.3936 47.9463]
	HoBHIS	57	[1 395]	71.1791	128.9646	17.8291	[53.2418 89.1164]	[117.4694 142.9731]
	ACCPndn	27	[12 98]	7.9652	22.1243	3.0586	[4.888 11.0424]	[20.1522 24.5275]
R7	Baseline	115	[1 376]	72.209	109.1693	15.0924	[57.0249 87.393]	[99.4386 121.0276]
	NACK	46	[1 68]	6.194	14.4975	2.0042	[4.1776 8.2104]	[13.2053 16.0722]
	HoBHIS	47	[21 79]	11.5522	22.5388	3.1159	[8.4174 14.6871]	[20.5298 24.9871]
	ACCPndn	0	-	-	-	-	-	-

This TLFN + PSO-GA runs iteratively (we set 1 sec) to gather historical information of PIT entries in contributing routers in defined sliding windows in order to predict the PIT entries in the next time interval (see sections 6.3.1 and 6.5.1). These amount of predictions are sent to the

Table 6.6: statistics of packet drop in Switch-like topology (mean of 10 runs)

Routers	Methods	No. drop	Drop boundary	Mean	Std.	SEM (95%)	μ (95%)	σ (95%)
R7	Baseline	182	[1 1943]	418.0249	294.9422	40.7751	[377.0023 459.0475]	[268.6527 326.9797]
	NACK	113	[3 521]	118.0498	120.7619	16.6951	[101.2534 134.8461]	[109.9979 133.8794]
	HoBHIS	123	[5 1120]	163.1791	184.8821	25.5595	[137.4644 188.8938]	[168.4028 204.9646]
	ACCPndn	0	-	-	-	-	-	-
R8	Baseline	21	[1 1177]	43.2786	185.2503	25.6104	[17.5127 69.0445]	[168.7382 205.3728]
	NACK	18	[2 564]	19.5224	85.344	11.7986	[7.6522 31.3926]	[77.7369 94.6143]
	HoBHIS	15	[1 848]	23.7861	116.7227	16.1366	[7.5515 40.0207]	[106.3187 129.4014]
	ACCPndn	10	[5 414]	7.6915	46.6156	6.4445	[1.2079 14.1752]	[42.4606 51.6791]
R10	Baseline	27	[1 227]	11.0896	38.4164	5.311	[5.7463 16.4328]	[34.9922 42.5893]
	NACK	0	-	-	-	-	-	-
	HoBHIS	6	[4 158]	2.1045	15.5169	2.1452	[-0.053718 4.2627]	[14.1338 17.2024]
	ACCPndn	0	-	-	-	-	-	-
R11	Baseline	78	[2 1068]	178.796	282.9273	39.1141	[139.4446 218.1475]	[257.7088 313.6597]
	NACK	79	[1 542]	98.2687	159.9374	22.111	[76.0235 120.5138]	[145.6815 177.3103]
	HoBHIS	74	[2 559]	94.7811	157.2892	21.7449	[72.9042 116.658]	[143.2693 174.3744]
	ACCPndn	48	[11 49]	6.7413	13.1428	1.817	[4.9133 8.5693]	[11.9713 14.5704]
R12	Baseline	172	[4 647]	316.8458	184.8103	25.5496	[291.1411 342.5505]	[168.3374 204.885]
	NACK	97	[62 250]	74.9851	85.8511	11.8687	[63.0443 86.9258]	[78.1988 95.1765]
	HoBHIS	155	[1 260]	126.3781	80.6666	11.152	[115.1585 137.5978]	[73.4764 89.4288]
	ACCPndn	0	-	-	-	-	-	-
R13	Baseline	198	[15 683]	350.1343	117.5996	16.2579	[333.7778 366.4909]	[107.1175 130.3737]
	NACK	197	[5 259]	160.6468	50.5236	6.9848	[153.6196 167.6739]	[46.0202 56.0116]
	HoBHIS	198	[9 413]	189.7811	64.9344	8.977	[180.7496 198.8126]	[59.1465 71.9877]
	ACCPndn	47	[22 70]	10.1642	19.6443	2.7158	[7.4319 12.8964]	[17.8933 21.7781]

corresponding routers to run second phase of ACCPndn, i.e., a nonlinear fuzzy control system per interface to control/avoid packet losses to mitigate congestion (see sections 6.3.2 and 6.5.2). When the controller runs initially, some time intervals are not available, that we set zero to those time intervals until their time reaches.

We show the experimental results in four conditions (Baseline, NACK, HoBHIS and ACCPndn) in the bottleneck links to confirm the effectiveness and efficiency of ACCPndn in terms of the applied performance metrics. Figs. 6-15 and 6-16 demonstrate the average Data packet drop within 10 runs in DFN-like and Switch-like topologies, respectively. As shown in these figures, there is a considerable benefits of the proposed countermeasure implemented by ACCPndn in reduction of the packet drop. Tables 6.5 and 6.6 illustrate the statistics of packet drop rate in DFN-like and Switch-like topologies, respectively. According to these tables, the average number of packet drop and its boundary have considerably decreased by ACCPndn as compared to the baseline, NACK and HoBHIS within 10 runs. We also show the benefit of the ACCPndn by Mean, Standard Deviation (Std.), Standard Error of the Mean (SEM) and lower and upper boundaries of the 95% confidence interval in probability distribution of the amount of packet drop in contributing routers.

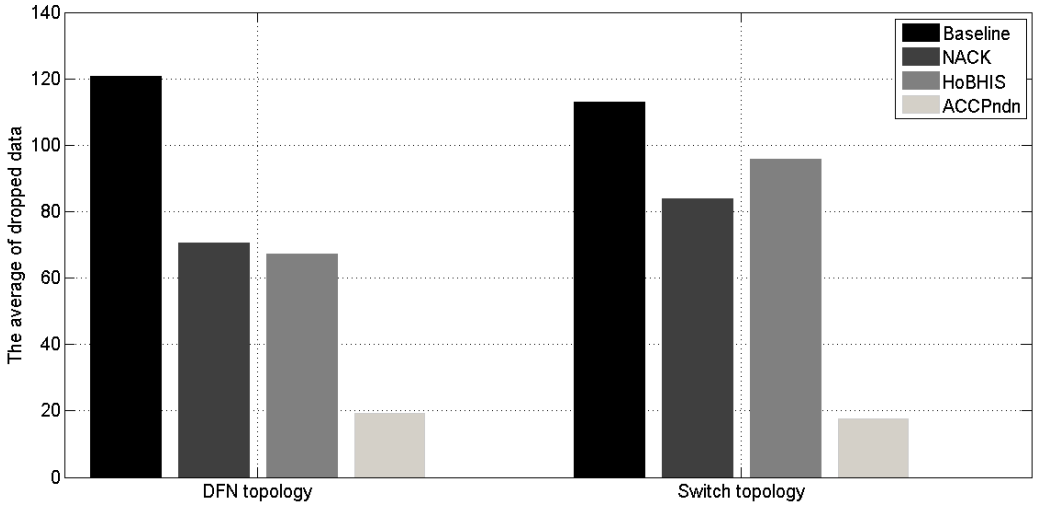
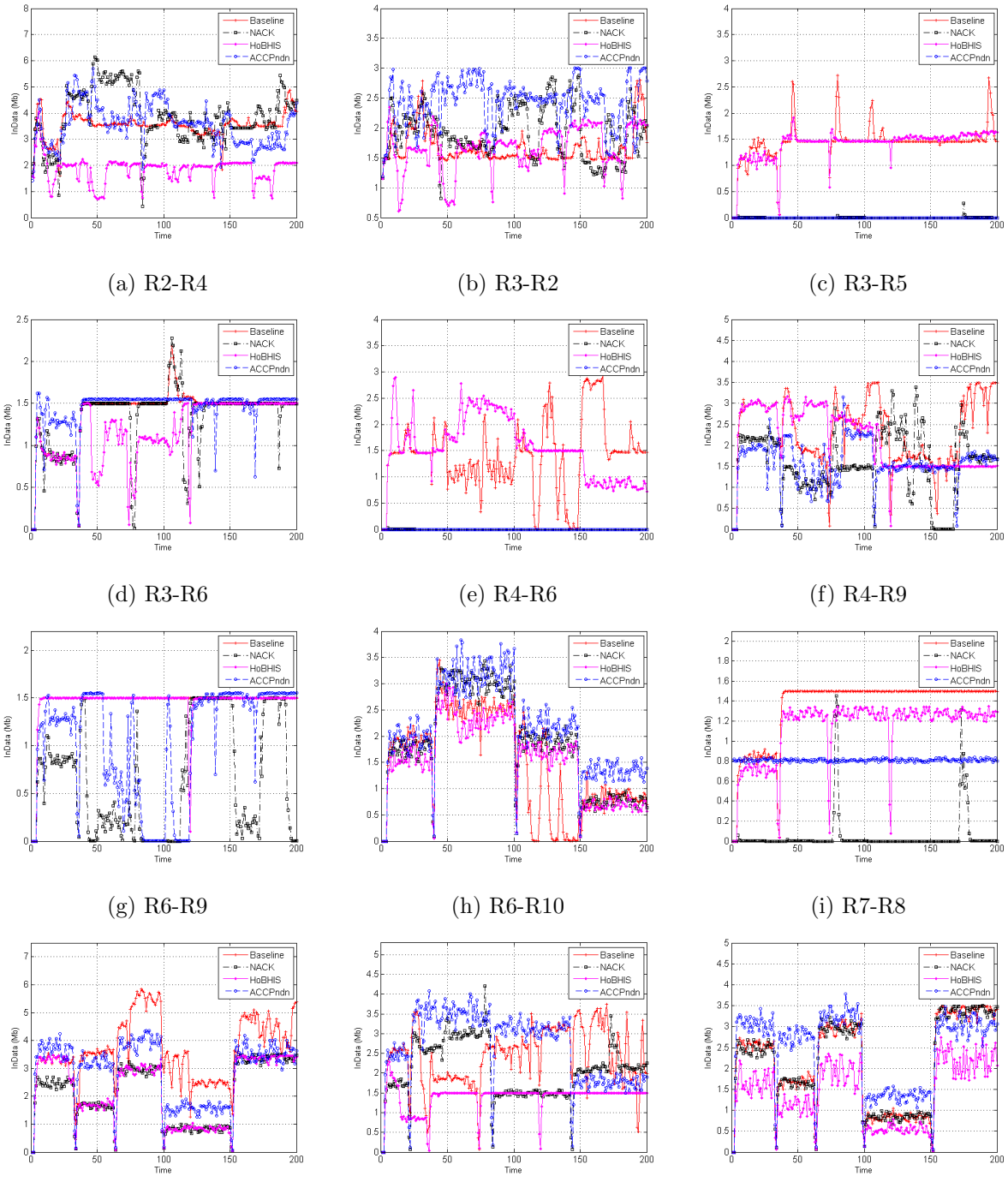


Figure 6-17: The total average packet drop rate in both considered topologies

The total average packet drop rate in both considered topologies is illustrated in Fig. 6-17. In Figs. 6-18 and 6-19 we show the average utilization of the bottleneck links and retrying alternative links in four conditions. According to the statistics of the average of packet drop in Tables 6.5 and 6.6, we observe that ACCPndn achieves the highest and the better average utilization and retrying alternative links as compared to NACK and HoBHIS.

These highlights confirm that the ACCPndn is effective and efficient in presence of bottleneck links and congestion problems, and outperforms the NACK and the HoBHIS sufficiently.



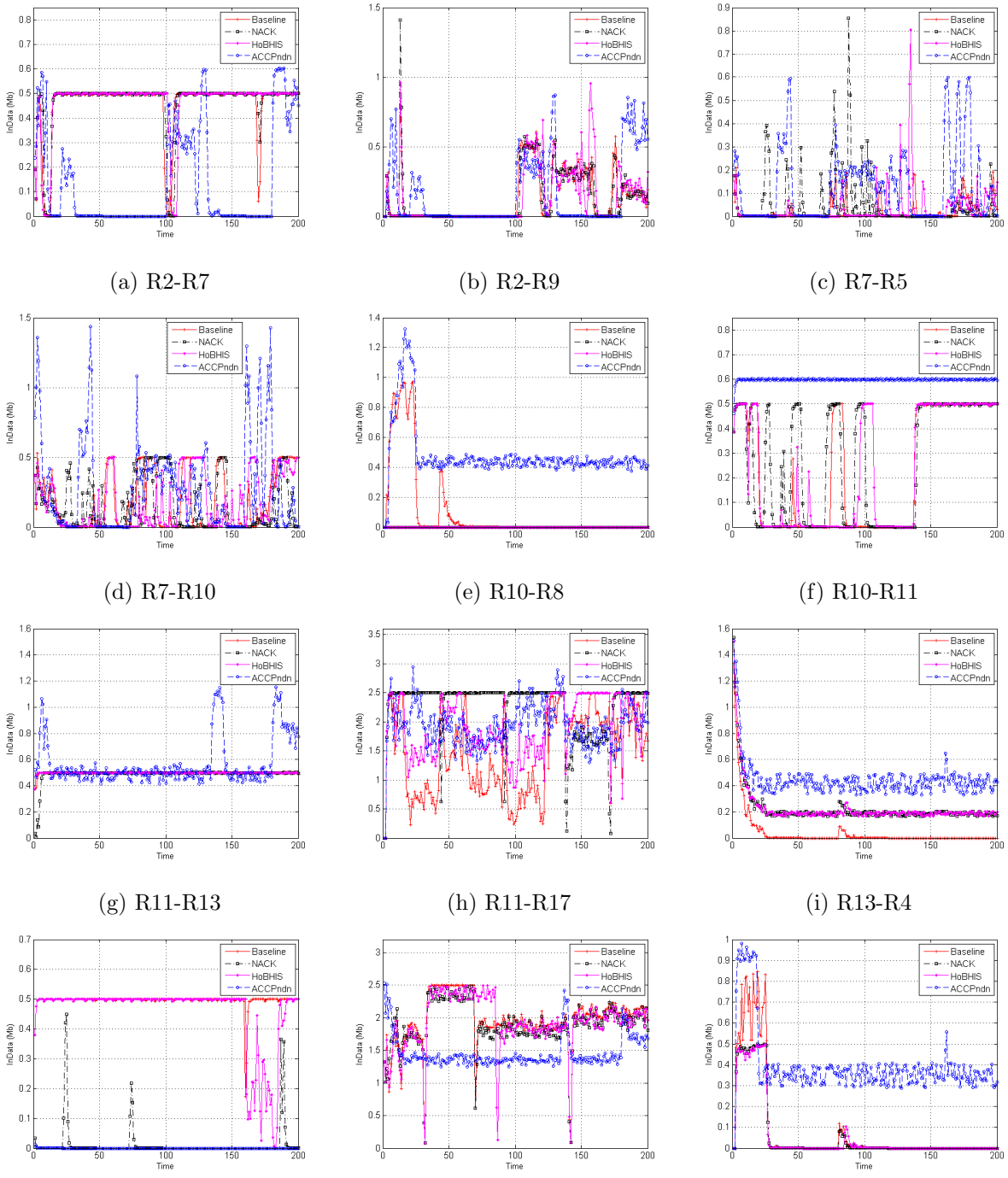
(a) R2-R4 (b) R3-R2 (c) R3-R5

(d) R3-R6 (e) R4-R6 (f) R4-R9

(g) R6-R9 (h) R6-R10 (i) R7-R8

(j) R8-R1 (k) R8-R4 (l) R8-R11

Figure 6-18: Average of InData in contributing routers in DFN-like topology



(a) R2-R7 (b) R2-R9 (c) R7-R5

(d) R7-R10 (e) R10-R8 (f) R10-R11

(g) R11-R13 (h) R11-R17 (i) R13-R4

(j) R13-R12 (k) R13-R16 (l) R13-R17

Figure 6-19: Average of InData in contributing routers in Switch-like topology

6.6 Conclusion

Our main contribution is to develop an Adaptive Congestion Control Protocol in Named Data Networking (ACCPndn) that works in two phases. The first phase -adaptive training- forecasts the source of congestion together with the amount of congestion in NDN routers with a Timed-Lagged Feedforward Network (TLFN) optimized by hybridization of PSO and GA. The second phase -fuzzy avoidance- employs a non-linear fuzzy logic-based control system based on the outcomes of first phase, which it makes a proactive decision in each router per interface to control and/or prevent packet drop well enough in advance. Extensive simulations and experimental results show that ACCPndn sufficiently satisfies the performance metrics and outperforms two previous proposals such as NACK and HoBHIS in terms of the minimal packet drop and high-utilization (retrying alternative paths) in bottleneck links to mitigate congestion traffics. In addition, it is found to be scalable with respect to varying bandwidths, delays, packet generation, and replacement policies in cache and PIT table.

Chapter 7

Mining and Visualizing Uncertain Data Objects and NDN Traffics by Fuzzy Self-Organizing Map

Uncertainty is a frequent issue in data analysis. The various factors that lead to data uncertainty include: approximate measurement, data sampling fault, transmission error or latency, data integration with noise, data acquisition by device error, and so on [229] [230]. These factors produce vague and imprecise data. Visualizing uncertain data is one of the new challenges in the uncertain databases [231].

Among the many visualization techniques, the Self-Organizing Map (SOM) [97] is widely and successfully applied due to its good result. SOM is a very popular unsupervised learning algorithm based on the classical set theory (see section 2.3.9). An important application of SOM is discovering the topological relationship among multidimensional input vectors and mapping them to a low dimensional output which is easy for further analysis by experts [232] [233]. The process of SOM training requires a certain and an unambiguous input data either belongs or not belong to a weight vector (cluster), where the membership evaluation is boolean. In contrast, uncertain and vague input vectors are not either entirely belong or not belong to a weight vector. A data may be considered vague and imprecise where some things are not either entirely true nor entirely false and where the some things are somehow ambiguous. For instance, fuzzy location in the right side of Fig. 7-1 is a way to represent the item of vague information: the object is *approximately* at position (4, 3), in which the grey levels indicate membership values with white representing 0 and

black representing 1. In contrast, the left side of Fig. 7-1 shows the exact position of a certain data where the membership evaluation of centers (weights) is boolean. There has been a lot of

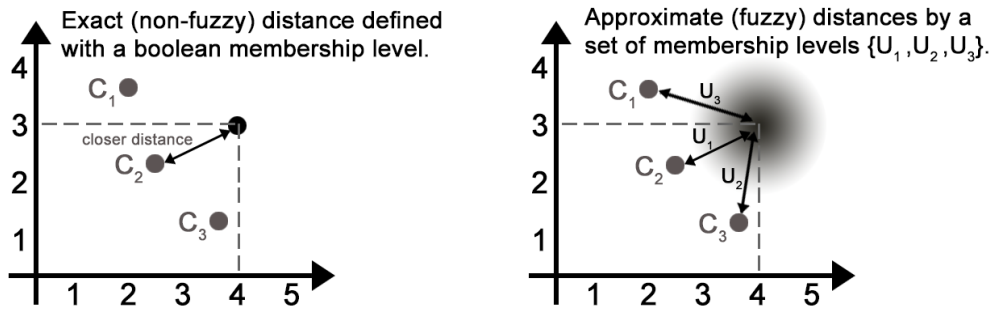


Figure 7-1: An example of exact (non-fuzzy) and approximate (fuzzy) distances in a 2-D space for a certain and vague data.

research in the application of Fuzzy sets theory to model vague and uncertain information [234]. The Fuzzy set (FS) theory introduced by Zadeh [72] is a more flexible approach than classical set theory, where objects belong to sets (clusters) with certain degree of membership ranging [0..1]. This makes FS theory suitable for representing and visualizing uncertain data [235]. Therefore, a combination of SOM and FS is able to illustrate dependencies in the uncertain data sets in a very intuitive manner.

SOM is indeed originally intended as a classification method, not a visualization method so there are a few additions to apply SOM for visualization. Li et al. [231] proposed a mining and visualizing algorithm for uncertain data, called USOM which combines fuzzy distance function and SOM to mine and visualize the uncertain data. In this research work, we employ the FS theory through the application of Fuzzy C-mean (FCM) clustering algorithm in the context of SOM algorithm to mine and visualize the uncertain objects in the uncertain databases. Experimental results over four classic benchmark problems and a new network architecture as Named Data Networking (NDN) show that the proposed method outperforms standalone SOM and USOM [231] in terms of the applied performance metrics. The proposed method and its findings are at an early stages of uncertainty management in NDN and there are some improvements needed. We initially plan to help to foster discussions and new research ideas among our readers as a future work.

7.1 The Proposed Method

The procedure of the proposed method, application of fuzzy set theory in the context of SOM for mining and visualizing uncertainties is as follows. A diagram of the proposed method is shown in Fig. 7-2.

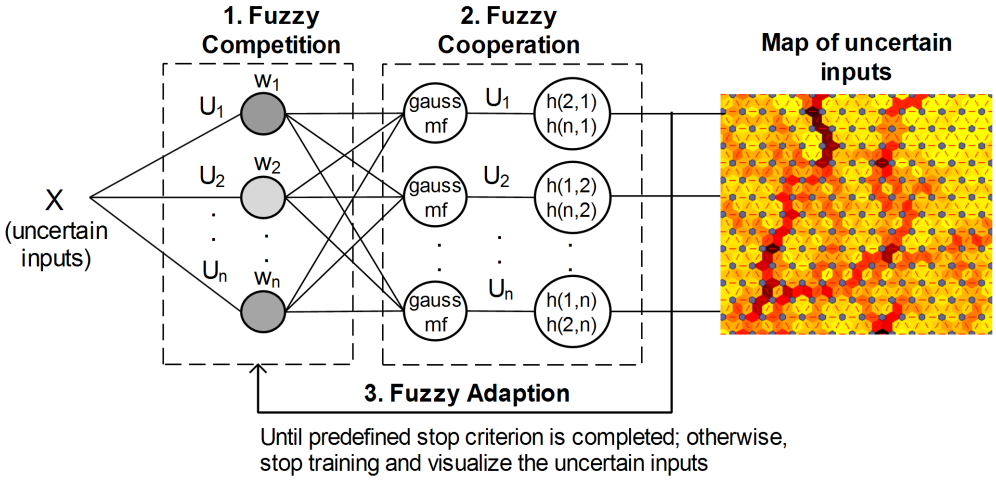


Figure 7-2: The proposed method for mining and visualizing uncertainties. The color of neurons in the competition phase indicates the membership value in which the darker color represents highest value, while the lighter color represents smallest value.

1. Fuzzy competition: in *hard competition*, the input vector is divided into distinct weights (clusters), where each input element belongs to exactly one weight. In *fuzzy competition*, input vector can belong to more than one weight, and associated with each element by a set of membership levels. Fuzzy c-means (FCM) [236] method allows one piece of input data to belong to two or more clusters (weights). The standard function is:

$$U_x = \frac{1}{\sum_j \left(\frac{d(weight_k, x)}{d(weight_j, x)}\right)^{\frac{2}{m-1}}} \tag{7.1}$$

Where, U_x is the membership value of each input vector x to all weights, $j = 1, 2, ..., w$, and m is the level of cluster fuzziness which is commonly set to 2. By the fuzzy competition all the neurons are wining neurons (called Best-Matching Units (BMU)) with the membership degree ranging [0..1].

2. Fuzzy cooperation: in fuzzy cooperation, all wining neurons cooperate with their neighboring neurons in terms of the membership degree by Eq. 7.2. For the size of the neighborhood, we

employed the Gaussian function that shrinks on each iteration until eventually the neighborhood is just the BMU itself.

$$h(j,i) = U_{xi} \times exp(\frac{-{d_{j,i}}^2}{2\sigma^2}) \quad i,j = 1,2,..,n; \quad i \neq j \tag{7.2}$$

Where, i is the number of the wining neurons including all the neurons with different membership degrees, j is the number of the cooperating neighbor neurons. U_{xi} is the membership value of input vector x from i^{th} wining neuron. $h(j,i)$ is the topological area centered around the wining neuron i and the cooperating neuron j. The size σ of the neighborhood needs to decrease with time. A popular time dependence is an exponential decay by:

$$\sigma(t) = \sigma_0 exp(\frac{-t}{\lambda}) \tag{7.3}$$

Where, $\sigma(t)$ is the width of the lattice at time t, σ_0 is the width of the lattice at time t_0, and λ is the time constant.

3. Fuzzy adaption: the adaption phase is the weight update by:

$$w_j = w_j + U_j \times (\eta h(j,i) \times (x - w_j)) \quad i,j = 1,2,..,n; \quad i \neq j \tag{7.4}$$

Where, U_j is the membership value of input x from neuron j.

These three phases are repeated, until the maximum number of iterations is reached or the changes become smaller than a predefined threshold.

7.2 Experimental Results

The proposed method, USOM and SOM were implemented by the MATLAB on an Intel Pentium 2.13 GHz CPU, 4 GB RAM running Windows 7 Ultimate.

7.2.1 Uncertain data modeling

To assess the accuracy and performance of the proposed method, four classic benchmark problems with different dimensions from the UCI machine learning repository [132] are applied. The selected data sets are Iris (4-D), Glass (9-D), Wine (13-D), and Zoo (17-D) in Table 7.1. In practice,

uncertainties are usually modeled in the form of Gaussian distributions [230]. For some attributes in data sets, we add a Gaussian noise with a zero mean and the standard deviation with the normal distribution $[0, 2 * f]$, where, f is an integer parameter from the set of {1, 2, 3} to define different uncertain levels.

Table 7.1: The four applied benchmark data sets

Data set	No. of features	No. of classes	No. of patterns
Iris	4	3	150
Glass	9	6	214
Wine	13	3	178
Zoo	17	7	101

7.2.2 Assessing the quality of visualizations

To assess the quality of the proposed method, several measures have been applied. The applied performance metrics are Quantization Error (QE), Topographic Error (TE), Trustworthiness of a Visualization, and Continuity of the Neighborhoods [237].

7.2.3 Visualization results

The experiments on each method were repeated 10 times independently. We evaluate the several SOM network structures on applied uncertain data sets which the optimal ones are Iris with 16x16 nodes, Glass with 16x16 nodes, Wine with 17x17 nodes, and Zoo with 15x15 nodes.

Table 7.2: Performance improvements achieved by the proposed scheme

Data	SOM				USOM				Proposed Method			
			Time				Time				Time	
	QE	TE	Exe.	Inc.	QE	TE	Exe.	Inc.	QE	TE	Exe.	Inc.
Iris (16x16)	0.024	0.0404	9.6	5.45	0.023	0.034	11.34	4.16	0.02	0.0267	16.43	2.17
Glass (16x16)	0.066	0.0312	21.17	15.45	0.042	0.02	23.11	14.23	0.028	0.0174	26.82	10.1
Wine (17x17)	0.072	0.0381	18.87	12.05	0.06	0.022	19.12	10.16	0.049	0.0102	22.07	7.74
Zoo (15x15)	0.067	0.0215	15.54	11.23	0.046	0.016	18.51	10.36	0.039	0.0103	21.34	8.52

Table 7.2 shows that our proposed method outperforms SOM and USOM methods in terms of the Quantization Error (QE) and Topographic Error (TE). The proposed method seems to be more time consuming (with Exec.) than the other methods due to the application of fuzzy set theories in the context of the SOM, in which all the neurons are winner with different membership grading. However, the proposed method can find a better solution with less times of increment on computational time (with Inc.) than the other methods due to its fast convergence speed. The trustworthiness and continuity values for K={1, 10, 20} are shown in Tables 7.3 and 7.4,

respectively. The trustworthiness and continuity measures show that the proposed method obtains the better results as compared to SOM and USOM. The results show that the proposed method with the application of fuzzy set theory in the context of the SOM yields high accuracy as compared to other methods without very much computational cost.

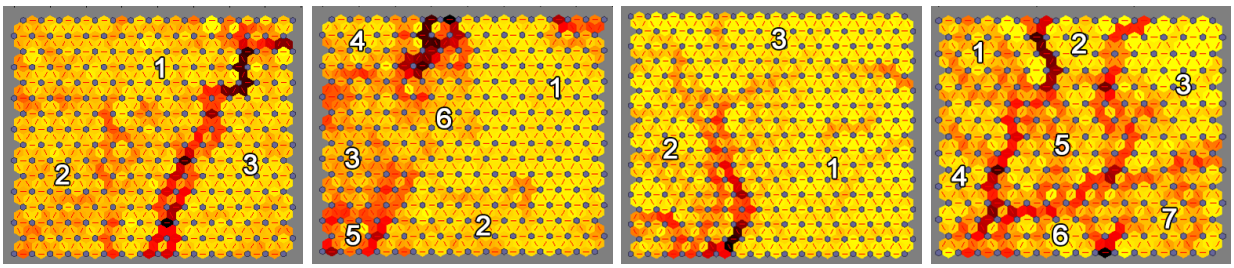
Table 7.3: The quality measurement by Trustworthiness

Data	SOM			USOM			Proposed Method		
	K=1	K=10	K=20	K=1	K=10	K=20	K=1	K=10	K=20
Iris (16x16)	0.97	0.94	0.93	0.95	0.962	0.968	0.962	0.968	0.974
Glass (16x16)	0.923	0.903	0.898	0.914	0.921	0.933	0.915	0.93	0.939
Wine (17x17)	0.931	0.921	0.904	0.924	0.941	0.953	0.925	0.951	0.962
Zoo (15x15)	0.961	0.96	0.96	0.962	0.963	0.968	0.963	0.97	0.978

Table 7.4: The quality measurement by Continuity

Data	SOM			USOM			Proposed Method		
	K=1	K=10	K=20	K=1	K=10	K=20	K=1	K=10	K=20
Iris (16x16)	0.945	0.901	0.892	0.961	0.964	0.966	0.97	0.974	0.982
Glass (16x16)	0.911	0.898	0.873	0.914	0.916	0.92	0.92	0.931	0.937
Wine (17x17)	0.921	0.892	0.883	0.93	0.931	0.935	0.935	0.939	0.941
Zoo (15x15)	0.86	0.841	0.812	0.89	0.898	0.902	0.91	0.918	0.927

Since our proposed method performs well as compared to SOM and USOM, we visualize uncertainties in the applied uncertain data sets. To facilitate the interpretation of results, we use the U-Matrix (unified distance matrix) where visualize the high-dimensional uncertain data into a 2-D space in Fig. 7-3. In this figure, the blue hexagons represent the neurons (weights).



(a) Iris 16x16 SOM (b) Glass 16x16 SOM (c) Wine 17x17 SOM (d) Zoo 15x15 SOM

Figure 7-3: U-Matrix of the applied benchmark problems.

The darker colors in the regions between neurons represent larger distance, while the lighter colors represent smaller distances. Fig. 7-3a shows that the constructed 4-D uncertain Iris SOM network has been clearly clustered into three distinct groups. The Glass SOM network (Fig. 7-3b) has been apparently classified 9-D uncertain data objects into six distinct types of glass. Figs. 7-3c and 7-3d show the three and the seven distinct groups of 13-D and 17-D uncertain data from Wine

and Zoo data sets, respectively. The results confirm that the proposed method performs well in mining and visualizing uncertain data into somewhat expected distinct groups.

7.2.4 Visualizing uncertain traffics in Named Data Networking

After evaluating the robustness and the accuracy of our proposed method with some benchmark problems, we apply the proposed method for visualizing uncertain traffics in Named Data Networking (NDN). NDN [6] is a promising network architecture being considered as a possible replacement to overcome the fundamental limitations of the current IP-based Internet. Traffic uncertainty refers to traffic volumes belong to more than one pattern (i.e., normal and attack), and associated with each pattern by a set of membership levels. Fuzzy approach can increase the detection rate and reduce the false positive rate with higher reliability in identifying the pattern of traffic volumes, due to any uncertain attack (or normal) data may be similar to some normal (or attack) patterns [12]. Whereas certain attack (or normal) data is exactly similar to a specific attack (or normal) pattern. Therefore, our aim is to visualize uncertain traffic volumes in NDN in a low dimensional output to be much easier for further analysis by network security experts. We conduct the same testbed configuration from papers [12] [19]. The employed features for traffic generation come from paper [12] as well as the ratio of (1) cache hit, (2) dropped Interest packet, (3) dropped data packets, (4) satisfied Interest packet, and (5) timed-out Interest packets in each 1 sec time interval. The structure of the traffic generated is shown in Table 7.5.

Table 7.5: NDN traffic generation

Type of traffic		Frequency	Pattern
Normal (526 records)		[100..500]	Exponential
Attack (211 records)	Cache pollution	[200..800]	Locality-disruption attacks uniformly
	DoS attacks	[400..1500]	Interest flooding attacks for non-existent and existent content uniformly and exponentially

We modeled uncertainties for some attributes in NDN traffic samples in the form of Gaussian distributions similar to Section 7.2.1. Fig. 7-4 maps the 11-D uncertain traffic samples to the 2-D space through our proposed method. This figure shows that the our proposed method performs somewhat well in mining and visualizing uncertainties into predefined distinct groups. Fig. 7-4 illustrates that there are some small groups of clustered data points with the lighter regions. These small clusters may contain some normal or attack data that try to be incorrectly placed in the neighboring regions, due to their uncertain nature. The experiments on each method were repeated 10 times independently with SOM 18 $\times$ 18 neurons. The results in Table 7.6 show that

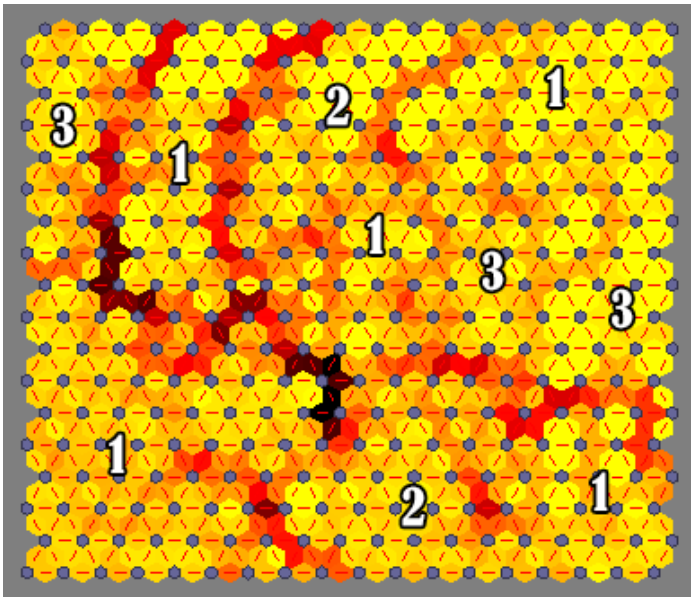


Figure 7-4: U-Matrix of the NDN traffic. 1: normal, 2: DoS attack, 3: cache pollution attack

our proposed method offers the best performance and outperforms sufficiently other preexisting methods.

Table 7.6: Comparing results of visualizing NDN traffic samples

Criteria		Methods		
		SOM	USOM	Proposed Method
Quantization Error		0.042	0.029	0.0125
Topographic Error		0.074	0.053	0.031
Trustworthiness	K=1	0.91	0.95	0.968
	K=15	0.905	0.943	0.954
	K=30	0.877	0.925	0.942
Continuity	K=1	0.914	0.922	0.954
	K=15	0.893	0.931	0.941
	K=30	0.867	0.917	0.936

7.3 Conclusion

In this work, we propose a new hybrid algorithm for mining and visualizing the uncertain data objects. We investigate the implementation of fuzzy set theory through the application of Fuzzy C-means clustering algorithm in the design of SOM neural network in order to improve the accuracy of visualizing uncertain data bases. The experimental results over the uncertain benchmarking data sets and the uncertain traffics in Named Data Networking (NDN) show that the proposed method is effective and precise in terms of the applied performance criteria.

The proposed method and its findings are at an early stages and there are some improvements needed. For instance, we plan to improve the proposed method for various uncertain models and big uncertain network traffic data in NDN. We suggest this research work to help to foster discussions and new research ideas among our readers. We leave further investigation of this initial idea to future work.

Chapter 8

Conclusion

8.1 Summary

Information-Centric Networking (ICN), in particular, Named Data Networking (NDN) has been proposed as a solution for a vital replacement for the current IP-based Internet due to the fundamental limitations of the Internet in supporting today's content-oriented services. Strong security has been one of the main design requirements for these architectures. However, there are still many missing pieces to be filled in to make NDN a fully working system at Internet scale. In this dissertation, we study the most important security issues in NDN in order to defense against new forms of (unknown) attacks, ensure privacy, achieve high availability and block network traffics belong to the attackers or at least limit their effectiveness. We fill in the four most important missing pieces for security in NDN, i.e., anomaly detection, mitigating DoS/DDoS attacks, congestion control, and mitigatin cache pollution attacks. In this context, the characteristics of Computational Intelligence (CI) methods make them suitable to be applied to the problems of NDN security. Hence, we suggest new hybrid CI-based methods to make NDN a more reliable and viable architecture for the future Internet.

In CCN/NDN as a possible future Internet, new kinds of attacks and anomalies will arise. Hence, the contents should be resilient against both anomalies and new forms of (unknown) attacks or at least limit their effectiveness. In this dissertation, we proposed a novel fuzzy anomaly detection system based on the hybridization of PSO and K-means clustering algorithms. Experimental results and analysis show that the proposed method in the training phase is very effective in determining the optimal number of clusters, and has a very high detection rate and a very low false positive rate at the same time. In the detection phase, the proposed method clearly outperforms other applied

method in terms of AUC (area under the ROC curve), accuracy, sensitivity and specificity. In addition, the times of increment on computational time of proposed method are relatively smaller than the other considered methods.

The ubiquitous in-network caching is a key NDN feature. However, pervasive caching strengthens security problems namely cache pollution attacks including cache poisoning and cache pollution. In this dissertation, we proposed a novel ANFIS-based cache replacement method to mitigate two generic cache pollution attacks namely false-locality and locality-disruption. Simulation results showed that the proposed method provides very accurate results as compared to LRU and LFU algorithms independently and in conjunction with CacheShield scheme without very much computational cost. Experimental results and analysis show the proposed ANFIS-based cache replacement method is very effective in determining and mitigating the fake content, and has a very high detection rate of locality-disruption attacks to replace them when new content is added to a full cache in a timely manner.

NDN can overcome the fundamental limitations of the current Internet, in particular, Denial-of-Service (DoS) attacks. However, NDN can be subject to new type of DoS attacks namely Interest flooding attacks and content poisoning. These types of attacks exploit key architectural features of NDN. We examined the most current instances of DoS/DDoS attacks to show that an adversary with limited resources can serve service degradation for legitimate users. We then introduced our intelligent hybrid algorithm for proactive detection (i.e., a hybrid multiobjective RBF-PSO method) and adaptive reaction (i.e. enforcing explicit limitations against adversaries) against DoS/DDoS attacks. Our extensive analysis shows that the proposed countermeasure against DoS/DDoS attacks performed well with the robust recovery from network failures and accuracy more than 90% in terms of the average of Interest satisfaction ratio for legitimate users, the PIT usage, the number of received contents (throughput), and a very low false positive rate over 10 simulation runs.

NDN is subject to congestion when the number of data packets that reach one or various routers in a certain period of time is so high than its queue gets overflowed. When this happens a high data packet loss and increase in the end-to-end delay occur affecting negatively on the performance, stability and robustness of the network. To address this problem many congestion control protocols have been proposed in literature which, however, they are too high sensitive to their control parameters as well as unable to predict congestion traffic well enough in advance. Hence, we developed an Adaptive Congestion Control Protocol in Named Data Networking (ACCPndn). It first

forecasts the source of congestion together with the amount of congestion in NDN routers with a Timed-Lagged Feedforward Network (TLFN) optimized by hybridization of PSO and GA. Then, we employ a non-linear fuzzy logic-based control system to make a proactive decision in each router per interface to control and/or prevent packet drop well enough in advance. Extensive simulations and experimental results show that ACCPndn sufficiently satisfies the performance metrics and outperforms some preexisting proposals in terms of the minimal packet drop and high-utilization (retrying alternative paths) in bottleneck links to mitigate congestion traffics. In addition, it is found to be scalable with respect to varying bandwidths, delays, packet generation, and replacement policies in cache and PIT table.

Finally, we provided a new research direction into the visualization of uncertain traffics in NDN. We motivated our work by pointing out the problems of uncertain traffics. The uncertain traffics belong to more than one pattern (normal or attack) which are associated by a set of membership levels. We proposed a new hybrid algorithm, called fuzzy self-organizing map for mining and visualizing uncertain objects. We investigate the implementation of fuzzy set theories in the design of SOM neural network in order to improve the accuracy of visualization in uncertain data sets. The experimental results over four uncertain benchmarking data sets and uncertain network traffics in NDN show that the proposed method is effective and precise in terms of the applied performance criteria. The proposed method and its findings are at an early stages and there are some improvements needed. We leave further investigation of this idea to future work.

8.2 Future Work

We hereby point out a few possible directions to further extend the current dissertation:

For anomaly detection purposes, we are currently working on several improvements of the presented approach (see Chapter 3) with the application of computational intelligence methodologies (such as multi-objective optimization techniques) to propose a robust method to improve the accuracy of detection rate and reduce the false positive rate over different NDN traffics.

For mitigating cache pollution attacks, future work includes devising several improvements to the approach presented in Chapter 4 and its use in larger and more complex network topologies.

According to the extensive analysis and the experimental results concerning DoS/DDoS attacks in NDN (see Chapter 5), two future works are suggested. The first work is the classification of legitimate users' traffics as either good (non-malicious), bad (malicious) or low and high prone to

attack traffics (non-malicious, but with the same properties as malicious traffics). The second work is investigating inter-domain DoS attacks.

Our next objective for congestion control/avoidance in the future work is to verify the properties of ACCPndn (see Chapter 6) analytically in many arbitrary NDN topologies.

Finally, we proposed a visualization method for uncertain traffic data in NDN (see Chapter 7) to help to foster discussions and new research ideas among our readers as a future work. Hence, there are some improvements needed, such as improving the proposed approach for various uncertain models and big uncertain network traffic data in the future.

Bibliography

[1] M. Acar Boyacioglu and D. Avci. An adaptive network-based fuzzy inference system (anfis) for the prediction of stock market return: The case of the istanbul stock exchange. *Expert Systems with Applications*, 37(12):7908 – 7912, 2010.

[2] M. Conti, P. Gasti, and M. Teoli. A lightweight mechanism for detection of cache pollution attacks in named data networking. *Computer Networks*, 57(16):3178 – 3191, 2013.

[3] G. Tselentis, J. Domingue, A. Galis, A. Gavras, and D. Hausheer. *Towards the Future Internet: A European Research Perspective*. IOS Press, Amsterdam, The Netherlands, The Netherlands, 2009.

[4] M. ĎAmbrosio B. Ahlgren, C. Dannewitz, A. Eriksson, J. Golić, B. Grönvall, D. Horne, A. Lindgren, O. Mämmelä, M. Marchisio, J. Mäkelä, S. Nechifor, B. Ohlman, K. Pentikousis, S. Randriamasy, T. Rautio, E. Renault, P. Seittenranta, O. Strandberg, B. Tarnauca, V. Vercellone, and D. Zeghlache. Second netinf architecture description. Technical Report 4WARD EU FP7 Project, Deliverable D-6.2 v2.0, Apr. 2010, fP7-ICT-2007-1-216041-4WARD / D-6.2, 2010.

[5] A. Detti, N. Blefari Melazzi, S. Salsano, and M. Pomposini. Conet: a content centric inter-networking architecture. In *Proceedings of the ACM SIGCOMM workshop on Information-centric networking*, ICN '11, pages 50–55, New York, NY, USA, 2011. ACM.

[6] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard. Networking named content. In *Proceedings of the 5th international conference on Emerging networking experiments and technologies*, CoNEXT '09, 2009.

[7] T. Koponen, M. Chawla, B.-G. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica. A data-oriented (and beyond) network architecture. *SIGCOMM Comput. Commun. Rev.*, 37(4):181–192, 2007.

[8] L. Zhang, D. Estrin, J. Burke, V. Jacobson, J. D. Thornton, D. K. Smetters, G. Tsudik B. Zhang, K. Claffy, D. Krioukov, D. Massey C. Papadopoulos, T. Abdelzaher, L. Wang P. Crowley, and E. Yeh. Named data networking (ndn) project. In *In Proceedings of the ACM SIGCOMM workshop on Information-centric networking*, number PARC TR-2010-3, pages 68–73, 2010.

[9] H. Lee and A. Nakao. User-assisted in-network caching in information-centric networking. *Computer Networks*, 57(16):3142 – 3153, 2013.

[10] G. Rossini and D. Rossi. Evaluating ccn multi-path interest forwarding strategies. *Computer Communications*, 36(7):771–778, 2013.

[11] Ch. Li, W. Liu, and K. Okamura. A greedy ant colony forwarding algorithm for named data networking. In *Proceedings of the Asia-Pacific Advanced Network*, volume 34, pages 17–26, 2012.

[12] A. Karami and M. Guerrero-Zapata. A fuzzy anomaly detection system based on hybrid pso-kmeans algorithm in content-centric networks. *Neurocomputing*, 149, Part C:1253–1269, 2015.

[13] G. q. Wang, T. Huang, J. Liu, J. y. Chen, and Y. j. Liu. Modeling in-network caching and bandwidth sharing performance in information-centric networking. *The Journal of China Universities of Posts and Telecommunications*, 20(2):99 – 105, 2013.

[14] Ch. Dannewitz, D. Kutscher, B. Ohlman, S. Farrell, B. Ahlgren, and H. Karl. Network of information (netinf) an information-centric networking architecture. *Computer Communications*, 36(7):721 – 735, 2013.

[15] X. Jiang and J. Bi. Technical report: Named content delivery network. Technical report, 2013.

[16] S. Tarkoma, D. Trossen, and M. Särelä. Black boxes: making ends meet in data driven networking. In *Proceedings of the 3rd international workshop on Mobility in the evolving internet architecture*, MobiArch '08, pages 67–72, New York, NY, USA, 2008. ACM.

[17] A. Afanasyev, P. Mahadevan, I. Moiseenko, E. Uzun, and Lixia Zh. Interest flooding attack and countermeasures in named data networking. In *IFIP Networking Conference, 2013*, pages 1 – 9, 2013.

[18] J. Ran, N. Lv, D. Zhang, Y. Ma, and Zh. Xie. On performance of cache policies in named data networking. In *International Conference on Advanced Science and Electronics Information (ICACSEI)*, pages 668 – 671. Atlantis Press, 2013.

[19] A. Karami. Data clustering for anomaly detection in content-centric networks. *International Journal of Computer Applications*, 81(7):1–8, November 2013. Published by Foundation of Computer Science, New York, USA.

[20] G. Carofiglio, M. Gallo, and L. Muscariello. On the performance of bandwidth and storage sharing in information-centric networks. *Computer Networks*, 57(17):3743–3758, 2013.

[21] B. Ahlgren, Ch. Dannewitz, C. Imbrenda, D. Kutscher, and B. Ohlman. A Survey of Information-Centric Networking (Draft). In *Information-Centric Networking*, number 10492 in Dagstuhl Seminar Proceedings, Dagstuhl, Germany, 2011. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, Germany.

[22] P. Louvieris, N. Clewley, and X. Liu. Effects-based feature identification for network intrusion detection. *Neurocomputing*, 121:265–273, 2013.

[23] P. Gasti, G. Tsudik, E. Uzun, and L. Zhang. Dos and ddos in named-data networking. *CoRR*, abs/1208.0952, 2012.

[24] H.-J. Liao, Ch.-H. Richard Lin, Y.-Ch. Lin, and K.-Y. Tung. Intrusion detection system: A comprehensive review. *Journal of Network and Computer Applications*, 36(1):16–24, 2013.

[25] C. Kolias, G. Kambourakis, and M. Maragoudakis. Swarm intelligence in intrusion detection: A survey. *Computers and Security*, 30(8):625–642, 2011.

[26] S. Peddabachigari, A. Abraham, C. Grosan, and J. Thomas. Modeling intrusion detection system using hybrid intelligent systems. *Journal of Network and Computer Applications*, 30(1):114–132, 2007.

[27] A. Karami and M. Guerrero-Zapta. A fuzzy anomaly detection system based on hybrid pso-kmeans algorithm in content-centric networks. In *Advanced Research on Hybrid Intelligent Techniques and Applications*. IGI Global, Hershey, Pennsylvania (USA), undergoing editing for 2015.

[28] Y. Kim and I. Yeom. Performance analysis of in-network caching for content-centric networking. *Computer Networks*, 57(13):2465 – 2482, 2013.

[29] W. K. Chai, D. He, I. Psaras, and G. Pavlou. Cache less for more in information-centric networks (extended version). *Computer Communications*, 36(7):758 – 770, 2013.

[30] G. Xylomenos, Ch. N. Ververidis, V. A. Siris, N. Fotiou, Ch. Tsilopoulos, X. Vasilakos, K. V. Katsaros, and G. C. Polyzos. A survey of information-centric networking. *IEEE Communications Surveys and Tutorials*, 99:1–26, 2013.

[31] T. Lauinger, N. Laoutaris, P. Rodriguez, T. Strufe, E. Biersack, and E. Kirda. Privacy risks in named data networking: what is the cost of performance? *ACM SIGCOMM Computer Communication Review*, 42(5):54 – 57, 2012.

[32] M. Xie, I. Widjaja, and H. Wang. Enhancing cache robustness for content-centric networking. In *INFOCOM*, pages 2426–2434, 2012.

[33] H. Park, I. Widjaja, and H. Lee. Detection of cache pollution attacks using randomness checks. In *IEEE International Conference on Communications (ICC)*, pages 1096 – 1100, 2012.

[34] C. Ghali, G. Tsudik, and E. Uzun. Needle in a haystack: Mitigating content poisoning in named-data networking. In *Proceedings of NDSS Workshop on Security of Emerging Networking Technologies (SENT)*, 2014.

[35] A. Karami and M. Guerrero-Zapata. An anfis-based cache replacement method for mitigating cache pollution attacks in named data networking. *Computer Networks*, 2015.

[36] M. Bari, S. Chowdhury, R. Ahmed, R. Boutaba, and B. Mathieu. A survey of naming and routing in information-centric networks. In *Communications Magazine, IEEE*, volume 50, pages 44–53, 2012.

[37] M. Wählisch, T. C. Schmidt, and M. Vahlenkamp. Backscatter from the data plane — threats to stability and security in information-centric networking. *CoRR*, abs/1205.4778, 2012.

[38] S. Choi, K. Kim, S. Kim, and B. h. Roh. Threat of dos by interest flooding attack in content-centric networking. In *International Conference on Information Networking (ICOIN)*, pages 315–319, 2013.

[39] L. Zhang, D. Estrin, J. Burke, V. Jacobson, J. D. Thornton, D. K. Smetters, G. Tsudik, B. Zhang, K. Claffy, D. Krioukov, D. Massey, C. Papadopoulos, T. Abdelzaher L. Wang P. Crowley, and E. Yeh. Named data networking (ndn) project. In *In Proceedings of the ACM SIGCOMM workshop on Information-centric networking*, pages 68–73. PARC TR-2010-3, 2010.

[40] P. Gasti A. Compagno, M. Conti and G. Tsudik. Poseidon: Mitigating interest flooding ddos attacks in named data networking. In *38th Annual IEEE Conference on Local Computer Networks (LCN)*, pages 630 – 638, 2013.

[41] A. Karami and M. Guerrero-Zapata. A hybrid multiobjective rbf-pso method for mitigating dos attacks in named data networking. *Neurocomputing*, 151, Part 3:1262–1282, 2015.

[42] A. Karami and M. Guerrero-Zapta. A hybrid multiobjective rbf-pso method for mitigating dos attacks in named data networking. In *Advanced Research on Hybrid Intelligent Techniques and Applications*. IGI Global, Hershey, Pennsylvania (USA), undergoing editing for 2015.

[43] N. Rozhnova and S. Fdida. An effective hop-by-hop interest shaping mechanism for ccn communications. In *IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 322 – 327, 2012.

[44] L. Saino, C. Cocora, and G. Pavlou. Cctcp: A scalable receiver-driven congestion control protocol for content centric networking. In *Communications (ICC), 2013 IEEE International Conference on*, pages 3775 – 3780, 2013.

[45] X. Qian, Y. Jing, and J. Tian. Network congestion avoidance strategy with particle filter. *Computer Communications*, 31(9):1723 – 1726, 2008.

[46] M. Lestas, A. Pitsillides, P. Ioannou, and G. Hadjipollas. Adaptive congestion protocol: A congestion control protocol with learning capability. *Computer Networks*, 51(13):3773 – 3798, 2007.

[47] Q. Xu and J. Sun. A simple active queue management based on the prediction of the packet arrival rate. *Journal of Network and Computer Applications*, 42:12 – 20, 2014.

[48] F. Li, J. Sun, M. Zukerman, Zh. Liu, Q. Xu, S. Chan, G. Chen, and K.-T. Ko. A comparative simulation study of tcp/aqm systems for evaluating the potential of neuron-based {AQM} schemes. *Journal of Network and Computer Applications*, 41:274 – 299, 2014.

[49] A. Karami and M. Guerrero-Zapata. Accpndn: Adaptive congestion control protocol in named data networking. *Journal of Network and Computer Applications, (under review)*, 2015.

[50] M. Hovaidi Ardestani, A. Karami, P. Sarolahti, and J. Ott. Congestion control in content-centric networking using neural network. In *CCNxCon 2013*. PARC, a Xerox company, 2013.

[51] Sh. Xiaonan Wu and W. Banzhaf. the use of computational intelligence in intrusion detection systems: A review. *Applied Soft Computing*, 10:1–35, 2010.

[52] B.M. Wilamowski. Methods of computational intelligence. In *IEEE International Conference on Industrial Technology*, ICIT'04, pages 8–10, 2004.

[53] A. Afanasyev, I. Moiseenko, and L. Zhang. ndnsim: Ndn simulator for ns-3. Technical Report NDN-0005, NDN, October 2012.

[54] J. Choi, J. Han, E. Cho, T. Kwon, , and Y. Choi. A survey of content-oriented networking for efficient content delivery. *IEEE Communications Magazine*, 49(3):121–127, 2011.

[55] N. Fotiou, P. Nikander, D. Trossen, and G. C. Polyzos. Developing information networking further: From psirp to pursuit. In *Broadband Communications, Networks, and Systems*, volume 66 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, pages 1–13. Springer Berlin Heidelberg, 2012.

[56] D. Poole, A. Machworth, and R. Goebel. *Computational Intelligence-A Logical Approach*. Oxford University Press, Oxford, UK, 1998.

[57] J. C. Bezdek. *What is Computational Intelligence? Computational Intelligence Imitating Life*. IEEE Press, New York, USA, 1994.

[58] L. C. Jain, Sh. Ch. Tan, and Ch. P. Lim. An introduction to computational intelligence paradigms. *Computational Intelligence Paradigms*, 2008.

[59] A. P. Engelbrecht. *Computational Intelligence: An Introduction*. Wiley; 2 edition (November 19).

[60] W. Duch. What is computational intelligence and where is it going? *Challenges for Computational Intelligence*, 63:1–13, 2007.

[61] M. Laszlo and S. Mukherjee. A genetic algorithm that exchanges neighboring centers for k-means clustering. *Pattern Recognition Letters*, 28(16):2359–2366, 2007.

[62] R. J. Kuo and L. M. Lin. Application of a hybrid of genetic algorithm and particle swarm optimization algorithm for order clustering. *Decision Support Systems*, 49(4):451 – 462, 2010.

[63] A. Chamkalani, A. Mae'soumi, and A. Sameni. An intelligent approach for optimal prediction of gas deviation factor using particle swarm optimization and genetic algorithm. *Journal of Natural Gas Science and Engineering*, 14:132 – 143, 2013.

[64] M. H. Moradi and M. Abedini. A combination of genetic algorithm and particle swarm optimization for optimal {DG} location and sizing in distribution systems. *International Journal of Electrical Power & Energy Systems*, 34(1):66 – 74, 2012.

[65] J. Kennedy and R. Eberhart. Particle swarm optimization. In *Proceedings in IEEE International Conference Neural Networks*, volume 4, pages 1942–1948, 1995.

[66] N.-J. Li, W.-J. Wang, Ch.-Ch. James Hsu, W. Chang, H.-G. Chou, and J.-W. Chang. Enhanced particle swarm optimizer incorporating a weighted particle. *Neurocomputing*, 124:218–227, 2014.

[67] M. Settles. *An Introduction to Particle Swarm Optimization*. Department of Computer Science, University of Idaho, Moscow, 2005.

[68] Y. Shi and R. Eberhart. A modified particle swarm optimizer. In *IEEE World Congress on Computational Intelligence*, pages 69–73, 1998.

[69] R. C. Eberhart and Y. Shi. Comparing inertia weights and constriction factors in particle swarm optimization. In *Proceedings of the Evolutionary Computation*, volume 1, pages 84–88, 2000.

[70] N. Padhye, K. Deb, and P. Mittal. Boundary handling approaches in particle swarm optimization. In *BIC-TA (1)*, pages 287–298, 2012.

[71] U. Shivakumar, V. Ravi, and G. R. Gangadharan. Ranking cloud services using fuzzy multi-attribute decision making. In *2013 IEEE International Conference on Fuzzy Systems (FUZZ)*, pages 1–8, 2013.

[72] Loti A. Zadeh. Fuzzy sets. *Information Control*, 8:338 – 353, 1965.

[73] Z. A. Baig and S. A. Khan. Fuzzy logic-based decision making for detecting distributed node exhaustion attacks in wireless sensor networks. In *Second International Conference on Future Networks (ICFN '10)*, pages 185 – 189, 2010.

[74] H. Izakian and W. Pedrycz. Agreement-based fuzzy c-means for clustering data with blocks of features. *Neurocomputing*, 127:266–280, 2014.

[75] W. Chimphlee, A. H. Abdullah, S. Chimphlee, and S. Srinoy. Unsupervised clustering methods for identifying rare events in anomaly detection. In *6th international Enformatika Conference*, pages 26–28, 2005.

[76] H. t. He, X. n. Luo, and B. l. Liu. Detecting anomalous network traffic with combined fuzzy-based approaches. In *International Conference on Intelligent Computing (ICIC)*, pages 433–442, 2005.

[77] H. C. Cho, S. M. Fadali, and H. Lee. Adaptive neural queue management for tcp networks. *Computers and Electrical Engineering*, 34:447 – 469, 2008.

[78] R. j. Kuo, S. Y. Hung, and W. C. Cheng. Application of an optimization artificial immune network and particle swarm optimization-based fuzzy neural network to an rfid-based positioning system. *Information Sciences*, 262:78 – 98, 2014.

[79] G. Ruan and Y. Tan. A three-layer back-propagation neural network for spam detection using artificial immune concentration. *Soft Computing*, 14(2):139 – 150, 2009.

[80] G. Kirubavathi Venkatesh and R. Anitha Nadarajan. Http botnet detection using adaptive learning rate multilayer feed-forward neural network. In *International Conference on Information Security Theory and Practice: Security, Privacy and Trust in Computing Systems and Ambient Intelligent Ecosystems*, pages 38 – 48. Springer-Verlag, 2012.

[81] A. Gomathy and B. Lakshmipathi. Network intrusion detection using genetic algorithm and neural network. In *Advances in Computing and Information Technology Communications in Computer and Information Science*, volume 198, pages 399 – 408. Springer-Verlag, 2011.

[82] M. Barabas, G. Boanea, and V. Dobrota. Multipath routing management using neural networks-based traffic prediction. In *Proceedings of the 3rd International Conference on Emerging Network Intelligence*, pages 118 – 124. IARIA, 2011.

[83] D. S. Broomhead and D. Lowe. Multivariable functional interpolation and adaptive networks. *Complex Systems*, 2:321–355, 1998.

[84] A. Hamad, D. Yu, J. B. Gomm, and M. S. Sangha. Radial basis function neural network in fault detection of automotive engines. In *International Journal of Engineering, Science and Technology*, volume 2, pages 1–8, 2010.

[85] R. Neruda and P. Kudová. Learning methods for radial basis function networks. *Future Generation Computer Systems*, 21(7):1131–1142, 2005.

[86] K.-K. Tan and K.-Z. Tang. Adaptive online correction and interpolation of quadrature encoder signals using radial basis functions. *IEEE Transactions on Control Systems Technology*, 13(3):370–377, 2005.

[87] S. N. Qasem and S. M. Shamsuddin. Radial basis function network based on time variant multi-objective particle swarm optimization for medical diseases diagnosis. *Applied Soft Computing*, 11(1):1427–1438, 2011.

[88] Y. Bai and L. Zhang. Genetic algorithm based self-growing training for rbf neural networks. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN '02)*, volume 1, pages 840–845, 2002.

[89] J.-S. Roger Jang. Anfis: adaptive-network-based fuzzy inference system. *IEEE Transactions on Systems, Man and Cybernetics*, 23(3):665 – 685, 1993.

[90] R. Singh, A. Kainthola, and T. N. Singh. Estimation of elastic constant of rocks using an anfis approach. *Applied Soft Computing*, 12(1):40 – 45, 2012.

[91] H. M. Jiang, C. K. Kwong, W. H. Ip, and T. C. Wong. Modeling customer satisfaction for new product development using a pso-based anfis approach. *Applied Soft Computing*, 12(2):726 – 734, 2012.

[92] A. Fuat Güneri, T. Ertay, and A. Yücel. An approach based on anfis input selection and modeling for supplier selection problem. *Expert Systems With Applications*, 38:14907 – 14917, 2011.

[93] J.-Sh. Jang. *Neuro-fuzzy modeling: Architectures, analyses, and applications*. PhD thesis, University of California, Berkeley, 1992.

[94] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.

[95] Y. Yusoff, M. Salihin Ngadiman, and A. Mohd Zain. Overview of nsga-ii for optimizing machining process parameters. *Procedia Engineering*, 15:3978–3983, 2011.

[96] E. Fallah-Mehdipour, O. Bozorg Haddad, M. M. Rezapour Tabari, and M. A. Mari no. Extraction of decision alternatives in construction management projects: Application and adaptation of nsga-ii and mopso. *Expert Systems with Applications*, 39(3):2794–2803, 2012.

[97] T. Kohonen. *Self-Organizing Maps*. Springer, Berlin, Heidelberg, 1995.

[98] H. Wu, T Gedeon, and D. Zhu. Spherical topology self-organizing map neuron network for visualization of complex data. A report submitted for the degree of master science, The Australian National University, 2011.

[99] A. Patcha and J.-M. Park. An overview of anomaly detection techniques: existing solutions and latest technological trends. *Computer Networks*, 51(12):3448–3470, 2007.

[100] F. Palmieri and U. Fiore. Network anomaly detection through nonlinear analysis. *computers & security*, 29:737–755, 2010.

[101] R. Perdisci, D. Ariu, P. Fogla, G. Giacinto, and W. Lee. Mcpad: A multiple classifier system for accurate payload-based anomaly detection. *Computer Networks*, 53:864–881, 2009.

[102] M. A. Faysel and S. S. Haque. Towards cyber defense: research in intrusion detection and intrusion prevention systems. *International Journal of Computer Science and Network Security (IJCSNS)*, 10(7):316–325, 2010.

[103] B. Krawczyk and M. Woźniak. Diversity measures for one-class classifier ensembles. *Neurocomputing*, 126:36–44, 2014.

[104] U. Fiore, F. Palmieri, A. Castiglione, and A. D. Santis. Network anomaly detection with the restricted boltzmann machine. *Neurocomputing*, 122(25):13–23, 2013.

[105] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: a survey. *ACM Computer Survey*, 41(3):15:1–15:58, 2009.

[106] G. Corral, E. Armengol, A. Fornells, and E. Golobardes. Explanations of unsupervised learning clustering applied to data security analysis. *Neurocomputing*, 72(13-15):2754–2762, 2009.

[107] Q. Wang and V. Megalooikonomou. A performance evaluation framework for association mining in spatial data. *Intelligent Information Systems*, 35(3):465–494, 2010.

[108] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: A review. *ACM Computing Surveys*, 31(3):264–323, 1999.

[109] A. Karami and R. Johansson. Choosing dbscan parameters automatically using differential evolution. *International Journal of Computer Applications*, 91(7):1–11, 2014. Published by Foundation of Computer Science, New York, USA.

[110] Y. T. Kao, E. Zahara, and I. W. Kao. A hybridized approach to data clustering. *Expert Systems with Applications*, 34(3):1754–1762, 2008.

[111] K. R. Zalik. An efficient k-means clustering algorithm. *Pattern Recognition Letters*, 29:1385–1391, 2008.

[112] C.-Y. Chen and F. Ye. Particle swarm optimization algorithm and its application to clustering analysis. In *Proceedings of the IEEE International Conference on Networking, Sensing and Control*, pages 789–794, 2004.

[113] Sh. Z. Selim and M. A. Ismail. K-means-type algorithms: A generalized convergence theorem and characterization of local optimality. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(1):81–87, 1984.

[114] M. C. Naldi and R. J. G. B. Campello. Evolutionary k-means for distributed data sets. *Neurocomputing*, 127:30–42, 2014.

[115] M. R. Anderberg. *Cluster Analysis for Applications*. Academic Press, Inc., New York, NY, 1973.

[116] H. Quan, D. Srinivasan, and A. Khosravi. Particle swarm optimization for construction of neural network-based prediction intervals. *Neurocomputing*, 127:172–180, 2014.

[117] A. Carlisle and G. Dozier. An off-the-shelf pso. In *Proceedings of the Particle Swarm Optimization Workshop*, page 16, 2001.

[118] J. Kennedy and R. C. Eberhart. *Swarm Intelligence*. Morgan Kaufmann, San Francisco, CA, 2001.

[119] B. S. Everitt. *Cluster Analysis*. 3rd edition, London, Edward Arnold / Halsted Press, 1993.

[120] L. Kaufman and P. J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. New York: John Wiley Sons, 1990.

[121] I. Kärkkäinen and P. Fränti. Minimization of the value of davies-bouldin index. In *Proceedings of the LASTED International Conference signal proccessing and communications*, pages 426–432, Marbella, Spain, 2000.

[122] J. Chen. Hybrid clustering algorithm based on pso with the multidimensional asynchronism and stochastic disturbance method. *Journal of Theoretical and Applied Information Technology*, 46(1):434–440, 2012.

[123] P. Zhenkui, H. Xia, and H. Jinfeng. The clustering algorithm based on particle swarm optimization algorithm. In *Proceedings of the International Conference on Intelligent Computation Technology and Automation*, ICICTA '08, pages 148–151, Washington, DC, USA, 2008. IEEE Computer Society.

[124] X. Cui and T. E. Potok. Document clustering analysis based on hybrid pso+k-means algorithm. *Journal of Computer Sciences*, pages 27–33, 2005.

[125] D. W. Van Der Merwe and A. P. Engelbrecht. Data clustering using particle swarm optimization. In *Proceedings of the IEEE Congress on Evolutionary Computation (CEC)*, pages 215–220, Canberra, Australia, 2003.

[126] L. Xiao, Zh. Shao, and G. Liu. K-means algorithm based on particle swarm optimization algorithm for anomaly intrusion detection. In *Proceedings of the 6th World Congress on Intelligent Control and Automation*, pages 5854–5858, Dalian, China, 2006.

[127] A. Gersho and R. M. Gray. *Vector Quantization and Signal Compression*. Dordrecht: Kluwer Academic Publishers, 1992.

[128] D. L. Davies and D. W. Bouldin. A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-1(2):224–227, 1979.

[129] A. Karami. Utilization and comparison of multi attribute decision making techniques to rank bayesian network options. master thesis, University of Skövde, Skövde, Sweden, 2011.

[130] A. Karami and R. Johansson. Utilization of multi attribute decision making techniques to integrate automatic and manual ranking of options. *Journal of Information Science and Engineering*, 30(2):519 – 534, 2014.

[131] J. P. Verma. *Data Analysis in Management with SPSS Software*, chapter 10. Springer, 2013.

[132] A. Asuncion and D. J. Newman. UCI machine learning repository, 2007.

[133] W. Lee and S. J. Stolfo. A framework for constructing features and models for intrusion detection systems. *ACM transactions on Information and system security (TiSSEC)*, 3(4):227–261, 2000.

[134] G. Münz, S. Li, and G. Carle. Traffic anomaly detection using k-means clustering. In *Proceeding of performance, reliability and dependability evaluation of communication networks and distributed systems, 4 GI / ITG Workshop MMBnet*. Hamburg, Germany, 2007.

[135] A. P. Bradley. The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30(7):1145–1159, 1997.

[136] M. Mohri C. Cortes. Confidence intervals for the area under the roc curve. In *Proceedings Advances in Neural Information Processing Systems (NIPS)*, 2004.

[137] H. Chen, Y. Xiao, and S. V. Vrbsky. An update-based step-wise optimal cache replacement for wireless data access. *Computer Networks*, 57(1):197 – 212, 2013.

[138] L. Deng, Y. Gao, Y. Chen, and A. Kuzmanovic. Pollution attacks and defenses for internet caching systems. *Computer Networks*, 52(5):935–956, 2008.

[139] Cuneyd C. Kaya, G. Zhang, Y. Tan, and Vijay S. Mookerjee. An admission-control technique for delay reduction in proxy caching. *Decision Support Systems*, 46(2):594 – 603, 2009.

[140] S. Romano and H. ElAarag. A neural network proxy cache replacement strategy and its implementation in the squid proxy server. *Neural Computing and Applications*, 20(1):59 – 78, 2011.

[141] W. A. Ahmed and S. M. Shamsuddin. Neuro-fuzzy system in partitioned client-side web cache. *Expert Systems with Applications*, 38:14715 – 14725, 2011.

[142] A. Bagheri, H. Mohammadi Peyhani, and M. Akbari. Financial forecasting using anfis networks with quantum-behaved particle swarm optimization. *Expert Systems With Applications*, 41(14):6235 – 6250, 2014.

[143] V. R. Budyal and S. S. Manvi. Anfis and agent based bandwidth and delay aware anycast routing in mobile ad hoc networks. *Journal of Network and Computer Applications*, 39:140 – 151, 2014.

[144] S. Moayer and P. A. Bahri. Hybrid intelligent scenario generator for business strategic planning by using anfis. *Expert Systems With Applications*, 36:7729 – 7737, 2009.

[145] S. Guillaume. Designing fuzzy inference systems from data: an interpretability-oriented review. *IEEE Transactions on Fuzzy Systems*, 9(3):426 – 443, 2001.

[146] L. Naderloo, R. Alimardani, M. Omid, F. Sarmadian, P. Javadikia, M. Y. Torabi, and F. Alimardani. Application of anfis to predict crop yield based on different energy inputs. *Measurement*, 45(6):1406 – 1413, 2012.

[147] A. Vakali. Evolutionary techniques for web caching. *Distributed and Parallel Databases*, 11(1):93 – 116, 2002.

[148] Y. Gao, L. Deng, A. Kuzmanovic, and Y. Chen. Internet cache pollution attacks and countermeasures. In *Proceedings of the 14th IEEE International Conference on Network Protocols (ICNP)*, pages 54 – 64, 2006.

[149] Sh. Shanbhag. Design and implementation of parallel anomaly detection, 2007.

[150] Yaochu Jin. *Multi-objective machine learning*, volume 16. Springer, 2006.

[151] O. Heckmann, M. Piringer, J. Schmitt, and R. Steinmetz. On realistic network topologies for simulation. In *Proceedings of the ACM SIGCOMM Workshop on Models, Methods and Tools for Reproducible Network Research*, MoMeTools '03, pages 28 – 32, New York, NY, USA, 2003. ACM.

[152] G. Oke, G. Loukas, and E. Gelenbe. Detecting denial of service attacks with bayesian classifiers and the random neural network. In *International Fuzzy Systems Conference, FUZZ-IEEE 2007*, pages 1–6, 2007.

[153] H.-G. Han, Q. l. Chen, and J.-F. Qiao. An efficient self-organizing rbf neural network for water quality prediction. *Neural Networks*, 24:717–725, 2011.

[154] M. Lotfi Shahreza, D. Moazzami, B. Moshiri, and M. R. Delavar. Anomaly detection using a self-organizing map and particle swarm optimization. *Scientia Iranica*, 18(6):1460–1468, 2011.

[155] X.-L. Li, Ch. Jia, D.-X. Liu, and D.-W. Ding. Nonlinear adaptive control using multiple models and dynamic neural networks. *Neurocomputing*, 136:190–200, 2014.

[156] E. Michailidis, S. K. Katsikas, and E. Georgopoulos. Intrusion detection using evolutionary neural networks. In *Informatics, 2008. PCI '08. Panhellenic Conference on*, pages 8–12, 2008.

[157] M. Gan, H. Peng, and X. p. Dong. A hybrid algorithm to optimize rbf network architecture and parameters for nonlinear time series prediction. *Applied Mathematical Modelling*, 36:2911–2919, 2012.

[158] Z. Zhang, T. Wang, and X. Liu. Melt index prediction by aggregated rbf neural networks trained with chaotic theory. *Neurocomputing*, 131:368–376, 2014.

[159] Ch.-M. Lee and Ch.-N. Ko. Time series prediction using rbf neural networks with a nonlinear time-varying evolution pso algorithm. *Neurocomputing*, 73(1-3):449–460, 2009.

[160] G. E. Tsekouras and J. Tsimikas. On training rbf neural networks using input-output fuzzy clustering and particle swarm optimization. *Fuzzy Sets Systems*, 221:65–89, 2013.

[161] S. Sieniutycz and J. Jeowski. Brief review of static optimization methods. In *Energy Optimization in Process Systems and Fuel Cells (Second Edition)*, pages 1–43. Elsevier, Amsterdam, second edition edition, 2013.

[162] Y. Sun, L. Zhang, and X. Gu. A hybrid co-evolutionary cultural algorithm based on particle swarm optimization for solving global optimization problems. *Neurocomputing*, 98(3):76–89, 2012.

[163] H. Du and N. Zhang. Time series prediction using evolving radial basis function networks with new encoding scheme. *Neurocomputing*, 71(7-9):1388–1400, 2008.

[164] V. Fathi and Gh. A. Montazer. An improvement in rbf learning algorithm based on pso for real time applications. *Neurocomputing*, 111:169–176, 2013.

[165] Gh. A. Montazer, H. Khoshniat, and V. Fathi. Improvement of rbf neural networks using fuzzy-osd algorithm in an online radar pulse classification system. *Applied Soft Computing*, 13(9):3831–3838, 2013.

[166] G. E. Tsekouras. A simple and effective algorithm for implementing particle swarm optimization in rbf network's design using input-output fuzzy clustering. *Neurocomputing*, 108:36–44, 2013.

[167] I. Kokshenev and A. P. Braga. A multi-objecitve approach to rbf network learning. *Neurocomputing*, 71:1203–1209, 2008.

[168] R. J. Kuo, Y. J. Syu, Zhen-Yao Chen, and F. C. Tien. Integration of particle swarm optimization and genetic algorithm for dynamic clustering. *Informaton Sciences*, 195:124–140, 2012.

[169] M. M. Etghani, M. H. Shojaeefard, A. Khalkhali, and M. Akbari. A hybrid method of modified nsga-ii and topsis to optimize performance and emissions of a diesel engine using biodiesel. *Applied Thermal Engineering*, 59:309–315, 2013.

[170] P.-Ch. Chang and Sh.-H. Chen. The development of a sub-population genetic algorithm ii (spga ii) for multi-objective combinatorial problems. *Applied Soft Computing*, 9(1):173–181, 2009.

[171] S. O. Mert, Z. Özçelik, Y. Özçelik, and I. Dinçer. Multi-objective optimization of a vehicular pem fuel cell system. *Applied Thermal Engineering*, 31(13):2171–2176, 2011.

[172] M. H. Zangooei, J. Habibi, and R. Alizadehsani. Disease diagnosis with a hybrid method svr using nsga-ii. *Neurocomputing*, 136:14–29, 2014.

[173] V. Lefort, C. Knibbe, G. Beslon, and J. Favrel. Simultaneous optimization of weights and structure of an rbf neural network. In *Proceedings of the 7th international conference on Artificial Evolution*, EA'05, pages 49–60, Berlin, Heidelberg, 2006. Springer-Verlag.

[174] I. Kokshenev and A. P. Braga. An efficient multi-objective learning algorithm for rbf neural network. *Neurocomputing*, 73:2799–2808, 2010.

[175] S. N. Qasem, S. M. Shamsuddin, S. Z. Mohd Hashim, M. Darus, and E. Al-Shammari. Memetic multiobjective particle swarm optimization-based radial basis function network for classification problems. *Information Sciences*, 239:165–190, 2013.

[176] H. S. Urade and R. Patel. Dynamic particle swarm optimization to solve multi-objective optimization problem. *Procedia Technology*, 6:283–290, 2012.

[177] H. Dai, Y. Wang, J. Fan, and B. Liu. Mitigate ddos attacks in ndn by interest traceback. In *2nd IEEE International Workshop on Emerging Design Choices in Name-Oriented Networking (NOMEN 2013)*, Turin, Italy, 2013.

[178] J. Chen, Z. Ren, and X. Fan. Particle swarm optimization with adaptive mutation and its application research in tuning of pid parameters. In *Proc. 1st International Symposium on Systems and Control in Aerospace and Astronautics*, pages 990–994, 2006.

[179] S. Kazemzadeh Azad and A. Jayant Kulkarni. Structural optimization using a mutation-based genetic algorithm. *International journal of optimization in civil engineering*, 2(1):80–100, 2012.

[180] M. Shokrian and K. Ann High. Application of a multi objective multi-leader particle swarm optimization algorithm on nlp and minlp problems. *Computers & Chemical Engineering*, 60:57–75, 2014.

[181] A. Khare and S. Rangnekar. A review of particle swarm optimization and its applications in solar photovoltaic system. *Applied Soft Computing*, 13(5):2997–3006, 2013.

[182] M. Thida, H.-L. Eng, D. N. Monekosso, and P. Remagnino. A particle swarm optimisation algorithm with interactive swarms for tracking multiple targets. *Applied Soft Computing*, 13(6):3106–3117, 2013.

[183] Y. Marinakis, G.-R. Iordanidou, and M. Marinaki. Particle swarm optimization for the vehicle routing problem with stochastic demands. *Applied Soft Computing*, 13(4):1693–1704, 2013.

[184] S. Espezua, E. Villanueva, and C. D. Maciel. Towards an efficient genetic algorithm optimizer for sequential projection pursuit. *Neurocomputing*, 123:40–48, 2014.

[185] Ch.-W. Tsai, Ch. Shih, and J. Wang. Using genetic algorithms to calibrate the user-defined parameters of iist model for sbloca analysis. *Annals of Nuclear Energy*, 63:499–505, 2014.

[186] M. Thakur. A new genetic algorithm for global optimization of multimodal continuous functions. *Journal of Computational Science*, 5(2):298–311, 2014.

[187] E. Atashpaz-Gargari and C. Lucas. Imperialist competitive algorithm: an algorithm for optimization inspired by imperialistic competition. In *Evolutionary Computation, 2007. CEC 2007. IEEE Congress on*, pages 4661–4667. IEEE, 2007.

[188] S. M. Goldansaz, F. Jolai, and A. H. Zahedi Anaraki. A hybrid imperialist competitive algorithm for minimizing makespan in a multi-processor open shop. *Applied Mathematical Modelling*, 37(23):9603–9616, 2013.

[189] R. Enayatifar, H. Javedani Sadaei, A. Hanan Abdullah, and A. Gani. Imperialist competitive algorithm combined with refined high-order weighted fuzzy time series (rhwftsica) for short term load forecasting. *Energy Conversion and Management*, 76:1104–1116, 2013.

[190] V. V. de Melo and G. L. C. Carosio. Investigating multi-view differential evolution for solving constrained engineering design problems. *Expert Systems with Applications*, 40(9):3370–3377, 2013.

[191] A. Wagdy Mohamed, H. Zaher Sabry, and T. Abd-Elaziz. Real parameter optimization by an effective differential evolution algorithm. *Egyptian Informatics Journal*, 14(1):37–53, 2013.

[192] W. Zhu, Y. Tang, J. a. Fang, and W. Zhang. Adaptive population tuning scheme for differential evolution. *Information Sciences*, 223:164–191, 2013.

[193] Dfn-verein: Dfn-noc (network operation center), 2013. retrieved Jun. 2013.

[194] O. Heckmann. *The competitive Internet service provider: network architecture, interconnection, traffic engineering and network design*. Wiley series in communications networking & distributed systems. J. Wiley, 2006.

[195] T. Fu, Y. Li, T. Lin, H. Tan, H. Tang, and S. Ci. An effective congestion control scheme in content-centric networking. In *Parallel and Distributed Computing, Applications and Technologies (PDCAT), 2012 13th International Conference on*, pages 245 – 248, 2012.

[196] D. Saucez, Luigi A. Grieco, and Ch. Barakat. Aimd and ccn: Past and novel acronyms working together in the future internet. In *Proceedings of the ACM Workshop on Capacity Sharing*, CSWS '12, pages 21 – 26, New York, NY, USA, 2012.

[197] L. Muscariello G. Carofiglio, M. Gallo and L. Papalini. Multipath congestion control in content-centric networks. In *Proceedings of the IEEE INFOCOM NOMEN*, 2013.

[198] Ch. Xia, M. Xu, and Y. Wang. A loss-based tcp design in icn. In *Wireless and Optical Communication Conference (WOCC)*, pages 449 – 454, 2013.

[199] Ch. Yi, A. Afanasyev, I. Moiseenko, L. Wang, B. Zhang, and L. Zhang. A case for stateful forwarding plane. *Computer Communications*, 36(7):779–791, 2013.

[200] M. Barabas, G. Boanea, A. B. Rus, V. Dobrota, and J. Domingo-Pascual. Evaluation of network traffic prediction based on neural networks with multi-task learning and multiresolution decomposition. In *IEEE International Conference on Intelligent Computer Communication and Processing (ICCP)*, pages 95 – 102, 2011.

[201] T. Bonald, M. Martin, and J-C. Bolot. Analytic evaluation of red performance. In *Proceedings in Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies*, volume 3, pages 1415 – 1424, 2000.

[202] P. Cortez, M. Rio, M. Rocha, and P. Sousa. Multi-scale internet traffic forecasting using neural networks and time series methods. *Expert Systems*, 29:143 – 155, 2012.

[203] V. B. Dharmadhikari and J. D. Gavade. An nn approach for mpeg video traffic prediction. In *2nd International Conference on Software Technology and Engineering (ICSTE)*, volume 1, pages 57 – 61, 2010.

[204] Zh. Li, Q. Lei, X. Kouying, and Zh. Xinyan. A novel bp neural network model for traffic prediction of next generation network. In *Fifth International Conference on Natural Computation (ICNC'09)*, volume 1, pages 32 – 38, 2009.

[205] V. Alarcon-Aquino and J. A. Barria. Multiresolution fir neural-network-based learning algorithm applied to network traffic prediction. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 36(2):208 – 220, 2006.

[206] S. G. Makridakis, S. C. Wheelwright, and R. J. Hyndman. *Forecasting: Methods and Applications*. Wiley, New York, USA, 1997.

[207] O. Claveria and S. Torra. Forecasting tourism demand to catalonia: Neural networks vs. time series models. *Economic Modelling*, 36:220 – 228, 2014.

[208] P. Cortez, M. Rio, M. Rocha, and P. Sousa. Internet traffic forecasting using neural networks. In *International Joint Conference on Neural Networks (IJCNN'06)*, pages 4942 – 4949, 2006.

[209] J. Peralta Donate, P. Cortez, A. Sanchis de Miguel, and G. Gutierrez Sanchez. Evolving time-lagged feedforward neural networks for time series forecasting. In *Proceedings of the 13th Annual Conference Companion on Genetic and Evolutionary Computation*, GECCO '11, pages 163 – 164, New York, NY, USA, 2011. ACM.

[210] N. Kourentzes, D. K. Barrow, and S. F. Crone. Neural network ensemble operators for time series forecasting. *Expert Systems with Applications*, 41(9):4235 – 4244, 2014.

[211] Ch. Ren, N. An, J. Wang, L. Li, B. Hu, and D. Shang. Optimal parameters selection for {BP} neural network based on particle swarm optimization: A case study of wind speed forecasting. *Knowledge-Based Systems*, 56:226 – 239, 2014.

[212] M. Khashei, S. R. Hejazi, and M. Bijari. A new hybrid artificial neural networks and fuzzy regression model for time series forecasting. *Fuzzy Sets and Systems*, 159(7):769 – 786, 2008.

[213] G. Das, P. K. Pattnaik, and S. K. Padhy. Artificial neural network trained by particle swarm optimization for non-linear channel equalization. *Expert Systems with Applications*, 41(7):3491 – 3496, 2014.

[214] W. Wong and P. Nikander. Secure naming in information-centric networks. In *Proceedings of the Re-Architecting the Internet Workshop, (ReARCH'10)*, pages 1–12, 2010.

[215] S. Yogi, K. R. Subhashini, and J. K. Satapathy. A pso based functional link artificial neural network training algorithm for equalization of digital communication channels. In *Industrial and Information Systems (ICIIS), 2010 International Conference on*, pages 107 – 112, 2010.

[216] R. Malviya and D. K. Pratihar. Tuning of neural networks using particle swarm optimization to model {MIG} welding process. *Swarm and Evolutionary Computation*, 1(4):223 – 235, 2011.

[217] Ch.-H. Lee and Y.-Ch. Lee. Nonlinear systems design by a novel fuzzy neural system via hybridization of electromagnetism-like mechanism and particle swarm optimisation algorithms. *Information Sciences*, 186(1):59 – 72, 2012.

[218] C. Chrysostomou, A. Pitsillides, G. Hadjipollas, and A. Sekercioglu. Fuzzy logic congestion control in tcp/ip best-effort networks. In *Australian Telecommunications Networks and Applications Conference (ATNAC)*, pages 8 – 10, Melbourne, Australia, 2003.

[219] C. Chrysostomou, A. Pitsillides, and Y.A. Sekercioglu. Fuzzy explicit marking: A unified congestion controller for best-effort and diff-serv networks. *Computer Networks*, 53(5):650 – 667, 2009.

[220] R. J. Frank, N. Davey, and S. P. Hunt. Time series prediction and neural networks. *Journal of Intelligent and Robotic Systems*, 31(1-3):91 – 103, 2001.

[221] A. Gluszek, M. Kekez, and F. Rudzinski. Web traffic prediction with artificial neural networks. In *Photonics Applications in Astronomy, Communications, Industry, and High-Energy Physics Experiments III*, volume 5775 of *Society of Photo-Optical Instrumentation Engineers (SPIE)*, pages 520 – 525, 2005.

[222] A. Jain and A. M. Kumar. Hybrid neural network models for hydrologic time series forecasting. *Applied Soft Computing*, 7(2):585 – 592, 2007.

[223] G. S. D. S. Gomes and T. B. Ludermir. Optimization of the weights and asymmetric activation function family of neural network for time series forecasting. *Expert Systems with Applications*, 40(16):6438 – 6446, 2013.

[224] Y. Zhao, T. Weng, and M. Small. Response of the parameters of a neural network to pseudoperiodic time series. *Physica D: Nonlinear Phenomena*, 268:79 – 90, 2014.

[225] Stefan H. Steiner. Exponentially weighted moving average control charts with time varying control limits and fast initial response. *Journal of Quality Technology*, 31, 1999.

[226] X. Ding, S. Canu, and T. Denoeux. Neural network based models for forecasting. In *In proceedings of Applied Decision Technologies Conference (ADT'95)*, pages 243 – 252, Uxbridge, UK, 1995.

[227] K.-I. Funahashi. On the approximate realization of continuous mappings by neural networks. *Neural Networks*, 2(3):183 – 192, 1989.

[228] K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359 – 366, 1989.

[229] P. Milošević, B. Petrović, D. Radojević, and D. Kovačević. A software tool for uncertainty modeling using interpolative boolean algebra. *Knowledge-Based Systems*, 62:1 – 10, 2014.

[230] J. Ge, Y. Xia, and Y. Tu. A discretization algorithm for uncertain data. In *Proceedings of the 21st International Conference on Database and Expert Systems Applications: Part II*, pages 485 – 499, 2010.

[231] L. Li, X. Zhang, Zh. Yu, Z. Feng, and R. Wei. Usom: Mining and visualizing uncertain data based on self-organizing maps. In *International Conference on Machine Learning and Cybernetics (ICMLC)*, volume 2, pages 804 – 809, 2011.

[232] M. Khalilia and M. Popescu. Topology preservation in fuzzy self-organizing maps. *Advance Trends in Soft Computing*, 312:105 – 114, 2014.

[233] J. Vesanto and E. Alhoniemi. Clustering of the self-organizing map. *IEEE Transactions on Neural Networks*, 11(3):586 – 600, 2000.

[234] D. Herrero-Pérez, H. Martnez-Barberá, K. LeBlanc, and A. Saffiotti. Fuzzy uncertainty modeling for grid based localization of mobile robots. *International Journal of Approximate Reasoning*, 51(8):912 – 932, 2010.

[235] F. Qi and A.-X. Zhu. Comparing three methods for modeling the uncertainty in knowledge discovery from area-class soil maps. *Computers & Geosciences*, 37(9):1425 – 1436, 2011.

[236] J. C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algoritms*. Plenum Press, New York, 1981.

[237] G. A. Ruz and D. T. Pham. Nbsom: The naive bayes self-organizing map. *Neural Comput. Appl.*, 21(6):1319 – 1330, 2012.

[238] A. Karami and M. Guerrero-Zapata. Mining and visualizing uncertain data objects and network traffics by fuzzy self-organizing map. In *Proceedings of the AIC workshop on Artificial Intelligence and Cognition*, pages 156–163, 2014.

Chapter 9

Acronyms

AI Artificial Intelligence

ANFIS Adaptive Neuro-Fuzzy Inference System

ANN Artificial Neural Networks

AUC Area Under the Curve

BMU Best Matching Unit

BPNN Back-Propagation Neural Network

CCN Content-Centric Networking

CI Computational Intelligence

CS Content Store

DBI Davies Bouldin Index

DCN Data-Centric Networking

DDoS Distributed Denial of Service

DE Differential Evolution

DH Data Handler

DONA Data Oriented Network Architecture

DoS Denial of Service

DR Detection Rate

EA Evolutionary Algorithm

EMA Exponential Moving Average

EWMA Exponentially Weighted Moving Average

FCM Fuzzy C-mean

FIA Future Internet Architecture

FIB Forwarding Information Base

FIS Fuzzy Inference System

FPR False Positive Rate

FS Fuzzy Set

GA Genetic Algorithm

IA Intelligent Agents

ICA Imperialist Competitive Algorithm

ICN Information-Centric Networking

IDS Intrusion Detection Systems

IFA Interest Flooding Attack

IO Information Object

IP Internet Protocol

KDD Knowledge Discovery in Data

LFU Least Frequently Used

LRD Long-Range Dependence

LRU Least Recently Used

MAE Mean Absolute Error

MAPE Mean Absolute Percent Error

MLP Multilayer Perceptron

MOP Multiobjective Optimization Problem

MSE Mean Square Error

NDN Named Data Networking

NRS Name Resolution Service

NSF National Science Foundation

NSGA Non-dominated Sorting Genetic Algorithm

PIT Pending Interest Table

PSIRP Publish-Subscriber Internet Routing Paradigm

PSO Particle Swarm Optimization

QE Quantization Error

RBF Radial Basis Function

RMSE Root Mean Square Error

ROC Receiver Operating Characteristic

RSI Relative Strength Index

SAIL Scalable & Adaptive Internet soLutions

SEM Standard Error of Mean

SMAPE Symmetric Mean Absolute Percent Error

SOM Self-Organizing Map

TE Topographic Error

TLFN Timed-Lagged Feedforward Neural Network

TSF Time Series Forecasting

URLs Uniform Resource Locators

XC Xie-complex

Chapter 10

Appendix: Publications

- Journals and Conferences:

(1) **A. Karami** and M. Guerrero-Zapata (2015), *A Fuzzy Anomaly Detection System based on Hybrid PSO-Kmeans in Content-Centric Networks*, Neurocomputing, 149:Part C, pp. 1253-1269 [12]. **JCR-2013: 2.005 Q1 (28/121 Q1 COMPUTER SCIENCE, ARTIFICIAL INTELLIGENCE)**.

(2) **A. Karami**, M. Guerrero-Zapata (2015), *A Hybrid Multiobjective RBF-PSO Method for Mitigating DoS Attacks in Named Data Networking*, Neurocomputing, 151:Part 3, pp. 1262-1282 [41]. **JCR-2013: 2.005 Q1 (28/121 Q1 COMPUTER SCIENCE, ARTIFICIAL INTELLIGENCE)**.

(3) **A. Karami**, M. Guerrero-Zapata (2014), *An ANFIS-based cache replacement method for mitigating cache pollution attacks in Named Data Networking*, Computer Networks, undergoing second revision on 27th September 2014 [35]. **JCR-2013: 1.282 Q2 (17/50 Q2 COMPUTER SCIENCE, HARDWARE & ARCHITECTURE)**

(4) **A. Karami**, M. Guerrero-Zapata (2014), *ACCPndn: Adaptive Congestion Control Protocol in Named Data Networking*, Journal of Network and Computer Applications (JNCA), (under review) [49]. **JCR-2013: 1.772 Q1 (28/121 Q1 COMPUTER SCIENCE, ARTIFICIAL INTELLIGENCE)**.

(5) **A. Karami**, M. Guerrero-Zapata (2014), *Mining and Visualizing Uncertain Data Objects and Named Data Networking Traffics by Fuzzy Self-Organizing Map*. In: Proceedings of the Second International Workshop on Artificial Intelligence and Cognition (AIC 2014), CEUR Workshop Proceedings, Vol 1315, pp. 156-163. [238].

(6) M. H. Ardestani, **A. Karami**, P. Sarolahti and J. Ott (2013), *Congestion Control in Content-Centric Networking using Neural Network*, Talk and Presentation in CCNxCon 2013, 5-6th September 2013, Parc (Xerox Co.), California, USA [50].

- Book Chapters:

(7) **A. Karami** and M. Guerrero-Zapata (2015), *A Novel Fuzzy Anomaly Detection Algorithm based on Hybrid PSO-Kmeans in Content Centric Networking*, In Advanced Research on Hybrid Intelligent Techniques and Applications. IGI Global, Hershey, Pennsylvania (USA), undergoing editing for 2015 [27].

(8) **A. Karami** and M. Guerrero-Zapata (2015), *A Novel Intelligent Classifier based on Hybrid Multiobjective RBF-PSO for Mitigating DoS Attacks in Named Data Networking*, In Advanced Research on Hybrid Intelligent Techniques and Applications. IGI Global, Hershey, Pennsylvania (USA), undergoing editing for 2015 [42].