A Hybrid Multiobjective RBF-PSO Method for Mitigating DoS Attacks in Named Data Networking

Amin Karami*, Manel Guerrero-Zapata

Computer Architecture Department (DAC), Universitat Politècnica de Catalunya (UPC), Campus Nord, C. Jordi Girona 1-3. 08034 Barcelona, Spain

Abstract
Named Data Networking (NDN) is a promising network architecture being considered as a possible replacement for the current IP-based (host-centric) Internet infrastructure. 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. This paper presents a new intelligent hybrid algorithm for proactive detection of DoS attacks and adaptive mitigation reaction in NDN. In the detection phase, a combination of multiobjective evolutionary optimization algorithm with PSO in the context of the RBF neural network has been applied in order to improve the accuracy of DoS attack prediction. Performance of the proposed hybrid approach is also evaluated successfully by some benchmark problems. In the adaptive reaction phase, we introduced a framework for mitigating DoS attacks based on the misbehaving type of network nodes. 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.

Keywords: Named Data Networking, DoS attacks, Intelligent Hybrid Algorithm, RBF neural networks, Particle Swarm Optimization, NSGA II

1. Introduction
In recent years there have been several efforts to design new network architectures for a viable and vital replacement for the current IP-based Internet [1, 2, 3]. These new architectures are designed to better cope with the fundamental limitations of the Internet in supporting today’s content-oriented services [4, 5]. In particular, there have been significant efforts regarding security, privacy, better mobility, scalability and efficient content distribution [6, 7]. Strong security has been one of the main design requirements for these architectures [8, 9]. Named Data Networking (NDN) [10] is one of these architectures as an ongoing research effort that aims to move the Internet into the future with a content-centric approach. NDN is a prominent example of Content-Centric Networking (CCN, also referred to as Information-Centric Networking (ICN) or Data-Centric Networking (DCN)) design. In CCN, content -rather than hosts, like in IP-based design- plays the central role in the communications [11, 12]. In contrast to today’s Internet, a key goal of the NDN project is "security by design" [13, 14, 15]. 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 [16, 17]. However, with this new paradigm, new kinds of attacks and anomalies -from Denial of Service (DoS) to privacy attacks- will arise [18, 19]. The big question is how resilient will this new NDN architecture be against DoS/DDoS attacks [15, 20]. 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 [20, 21].
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 [22, 21]. 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 [23, 24, 25]. 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 [20, 26]. 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 [16, 27]. The most popular approach for DOS/DDoS attacks prediction is using Artificial Neural Networks (ANNs) classification [28, 29, 30]. ANNs have become one of the most vital and valuable tools in solving many complex practical problems [31, 32], 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 [33, 34, 35]. 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 [32, 36, 37]. In order to find the most appropriate parameters, an optimization algorithm can be used [38, 39]. 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 [38, 40]. Hence, to improve the prediction accuracy and robustness of the RBF network, network parameters (centers, widths and weights) should be simultaneously tuned [32]. Some of the existing algorithms to achieve that are given in [32, 36, 41, 42, 43]. 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) [36, 43, 44, 45]. However, MSE is not suitable for determining the optimal position of basis function centers. Since the MSE decreases, the number of centers increases [46]. 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 [47]. It is generally accepted that well-separated (external separation of) centers and their local optimization (internal homogeneity) are conflicting objectives [46, 48]. This trade-off is a well-known problem as the Multiobjective Optimization Problem (MOP) [49, 50, 51, 52]. This paper 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 [53, 54, 55, 56]. 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 [57, 58, 59]. 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 paper is summarized in three objectives. The first objective of this paper 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. The rest of the paper is organized as follows. Section 2 describes the background materials of the NDN, and follows by the explanation of the related DoS attacks in Section 3. Section 4 provides the related work. Section 5 presents RBF neural networks. Section 6 describes the PSO algorithm. In Section 7, NSGA II is presented.
in detail. Section 8 describes the proposed hybrid intelligent method. Section 9 examines the detection (classification) phase of the proposed hybrid algorithm on several benchmark problems. Evaluation environment is described in Section 10 in detail. Section 11 proposes our countermeasure including proactive detection and adaptive reaction mechanism against DoS attacks in NDN. Detailed analysis and discussions are explained in Section 12. Finally, Section 13 draws conclusions.

2. NDN overview

NDN is a novel next-generation Internet architecture based on the principle of CCN (Content-Centric Networking) paradigm, where contents are retrieved by their names instead of by the network addresses where they are hosted [60]. Data names in NDN are hierarchically structured, e.g., eighth fragment of a youtube video (file) would be named /youtube/videos/A7m5I6n8kVUw/8. NDN is one of the five NSF-sponsored Future Internet Architecture (FIA) [21]. It supports two types of messages: Interest and content (Fig. 1). A consumer asks for a content by routing an Interest request using name prefixes instead of today’s IP prefixes. Interest packet is routed towards the location of the origin content where it has been published. Any router and middle 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 middle nodes store a copy of content in their caches to answer to probable same Interest requests from subsequent requests [16]. Each NDN router maintains three major data structures including Forwarding Information Base (FIB), Pending Interest Table (PIT) and Content Store (CS) or buffer memory. FIB is a lookup table used to determine interfaces for forwarding incoming Interests packets which contains [name prefix, interface number] entries. Another lookup table is PIT which contains outstanding entries [Interest prefix, arrival interface]. 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 [12, 61].

3. 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 [20, 62]:

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.

3.1. 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 [20]. There are three types of Interest flooding attacks, based on the type of content requested [20]:

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.

4. **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. [20] 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. [15] 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. [21] 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 and explained the effect of Interest flooding DoS 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.

5. **RBF neural networks**

Radial Basis Function (RBF) is a kind of feed-forward neural networks, which were developed by Broomhead and Lowe in 1998 [64]. This type of neural networks use a supervised algorithm and have been broadly employed for classification and interpolation regression [65]. 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 [66, 67]. 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 [34, 37]. The real output in output layer is given by:

\[
y_i(X) = \sum_{j=1}^{l} w_j \phi \left( \frac{\| P - C_j \|}{\sigma_j} \right) \quad \text{for} \quad 1 \leq s \leq l \quad (1)
\]

Where \( y_i \) is \( s \)-th network output, \( P \) is an input pattern, \( w_j \) 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 \( \sigma_j \) is the width of the \( j \)-th unit in the hidden layer. The \( \phi \) denotes to an basis
ity of

Where \( X \) noted by vector \( \text{Gbest} \) find the global best position (known as the mathematical best, a velocity vector and has a memory which helps it in the search space according to a simple mathematical formula over the particle’s position and velocity to initialize a flock of birds randomly over the search space to mimicking a flock of birds randomly over the search space [70]. Each particle is a candidate solution to the problem which is assigned a velocity vector and has a memory which helps it in remembering its previous best position (known as local best, \( lbest \)). 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 (known as \( Gbest \)). 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. Usually a fitness function is the objective function to be minimized or maximized. Hereafter, a record of the best position of particle based on the fitness function value is kept in process. The best previously visited position of the particle \( i \) at current stage is denoted by vector \( lbest_i = (lbest_{i1}, lbest_{i2}, ..., lbest_{in}) \) as the personal best. The position of the best objective function value until the current stage is also recorded as the global best position denoted by \( Gbest = (gbest_1, gbest_2, ..., gbest_n) \).

The velocity and position of particle \( i \) at iteration \( k+1 \) can be calculated according the following equations:

\[
V_i(k + 1) = \omega V_i(k) + c_1 r_1 (lbest_i(k) - X_i(k)) + c_2 r_2 (gbest(k) - X_i(k))
\]

(3)

\[
X_i(k + 1) = X_i(k) + V_i(k + 1)
\]

(4)

Where \( \omega \) is the inertia weight, \( c_1 \) (cognitive parameter) and \( c_2 \) (social parameter) are constants which control the search space between the local best position and the global best position (generally \( c_1 = c_2 = 2 \) [48]). Parameters \( r_1 \) and \( r_2 \) are random numbers uniformly distributed within \([0 \ 1]\). Since larger \( \omega \) performs more efficient global search and smaller one performs more effective local search, Eberhart and Shi [71] used Eq. (5) that properly balances between exploration (global search) and exploitation (local search) to avoid premature convergence to a local optimum.

\[
\omega = \omega_{\text{max}} - \frac{(\omega_{\text{max}} - \omega_{\text{min}})}{T} \cdot t
\]

(5)

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

6. Particle Swarm Optimization (PSO)

PSO was initially introduced in 1995 by James Kennedy and Russell Eberhart as a global optimization technique [69]. 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 [70]. Each particle is a candidate solution to the problem which is assigned a velocity vector and has a memory which helps it in remembering its previous best position (known as local best, \( lbest \)). 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 (known as \( Gbest \)). 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. Usually a fitness function is the objective function to be minimized or maximized. Hereafter, a record of the best position of particle based on the fitness function value is kept in process. The best previously visited position of the particle \( i \) at current stage is denoted by vector \( lbest_i = (lbest_{i1}, lbest_{i2}, ..., lbest_{in}) \) as the personal best. The position of the best objective function value until the current stage is also recorded as the global best position denoted by \( Gbest = (gbest_1, gbest_2, ..., gbest_n) \).

The velocity and position of particle \( i \) at iteration \( k+1 \) can be calculated according the following equations:

\[
V_i(k + 1) = \omega V_i(k) + c_1 r_1 (lbest_i(k) - X_i(k)) + c_2 r_2 (gbest(k) - X_i(k))
\]

(3)

\[
X_i(k + 1) = X_i(k) + V_i(k + 1)
\]

(4)

Where \( \omega \) is the inertia weight, \( c_1 \) (cognitive parameter) and \( c_2 \) (social parameter) are constants which control the search space between the local best position and the global best position (generally \( c_1 = c_2 = 2 \) [48]). Parameters \( r_1 \) and \( r_2 \) are random numbers uniformly distributed within \([0 \ 1]\). Since larger \( \omega \) performs more efficient global search and smaller one performs more effective local search, Eberhart and Shi [71] used Eq. (5) that properly balances between exploration (global search) and exploitation (local search) to avoid premature convergence to a local optimum.

\[
\omega = \omega_{\text{max}} - \frac{(\omega_{\text{max}} - \omega_{\text{min}})}{T} \cdot t
\]

(5)

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

7. 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 [72]. Fig. 2 shows the NSGA II procedure. Generally, NSGA II can be roughly detailed as following steps [72, 73, 74]:

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:

\[
d_{ij} = \sum_{m=1}^{M} \frac{j_m^{p+1} - j_m^{p-1}}{f_m^{p+1} - f_m^{p-1}}
\]

(6)

Where, \( d_{ij} \) is crowded distance of \( j \)-th solution, \( M \) is number of objectives, \( f_m^{p+1} \) and \( f_m^{p-1} \) are values of \( m \)th
Figure 3: The overview of the proposed DoS mitigation method in NDN

8. The proposed hybrid intelligent method for DoS mitigation in NDN

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 8.1) and the second one adaptive reaction (see section 11.2). The proposed predictor in the first phase is a global framework so that we can use the predictor in other networks. In this paper, 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. 3.

8.1. 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. 4. Each phase is given in the next subsections.

8.1.1. 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 ($Ite_{Max}$), crossover percentage ($pCrossover$), number of parents (offspring) after crossover operator ($nCrossover = 2 \times \text{round}(pCrossover \times N)$), mutation percentage ($pMutation$), number of mutants after mutation ($nMutation = \ldots$)
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 \( P_1, P_2, \ldots, P_N \).
   2-2- Calculate the two conflicting cost functions as DBI and MSE (presented in section 8.1.3) 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. 6).
   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 (this paper adopts the two-point crossover).
      b. Execute mutation operator, PopMutation (A Gaussian distributed random number with mean zero and variance 1 is used [75, 76]).
   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 \( PF_1 \).
   3-6- Repeat Step 3 until termination condition (\( IterMax \)) is reached.
   3-7- Keep the final \( PF_1 \) 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 ($\sigma$) and weight ($w$) variables.
1-3- Adjust the PSO parameters: inertia weight ($\omega$) which is linearly decrease by Eq. 5, acceleration coefficients ($c1 = c2 = 2$), and two random numbers ($r1$ and $r2$) which distributed uniformly in [0 1]
2. Initialize population for each particle:
2-1- Generate the initial populations (particle positions) including $Particle(1), Particle(2), ..., Particle(N)$:
\[
Particle(i).Position_{\sigma} \text{ and } Particle(i).Position_w.
\]
\[
i = 1, 2, ..., N.
\]
\[
particle(i).Position_{\sigma} = \text{Continuous uniform random numbers between } \sigma \text{.Lower and } \sigma \text{.Upper in size of nKernel.}
\]
\[
particle(i).Position_w = \text{Continuous uniform random numbers between } w \text{.Lower and } w \text{.Upper in size of nKernel.}
\]
2-2- Initialize velocity vectors in a feasible space for each particle:
\[
particle(i).Velocity_{\sigma} \text{ a } nKernel \text{ size zero matrix.}
\]
\[
particle(i).Velocity_w \text{ a } nKernel \text{ size zero matrix.}
\]
2-3- Evaluate each particle by Gaussian basis function in each RBF units (Eq. 1). Calculate Gaussian basis function with two tuned parameters ($\sigma$ -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. 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. 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.

8.1.2. 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 [77]. 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 pseudocode of classification computation is given below:
1- Define some input parameters:
\[
LowEdge = \text{lower bound of target class.}
\]
\[
UpEdge = \text{upper bound of target class.}
\]
\[
NumBins= \text{number of target classes in training data set.}
\]
\[
BinEdges= \text{Generate linearly spaced vectors between } LowEdge \text{ and } UpEdge \text{ in the size of } NumBins, \text{ where the bin range is equal to the number of target class.}
\]
2- Assign input patterns into the closest index of specified bin range. The index of bin range is the predicted classes of input patterns.

<table>
<thead>
<tr>
<th>Data set</th>
<th>No. of features</th>
<th>No. of classes</th>
<th>No. of patterns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wine</td>
<td>13</td>
<td>3</td>
<td>178</td>
</tr>
<tr>
<td>Iris</td>
<td>4</td>
<td>3</td>
<td>150</td>
</tr>
<tr>
<td>Ionosphere</td>
<td>34</td>
<td>2</td>
<td>351</td>
</tr>
<tr>
<td>Zoo</td>
<td>17</td>
<td>7</td>
<td>101</td>
</tr>
</tbody>
</table>
8.1.3. 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 \( c \) (MSE):

\[ \text{MSE} = \frac{1}{N} \sum_{i=1}^{N} d(x_i, c_i)^2 \] (7)

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

Based on our experiments [46], we have found it quite reliable. DBI [78] 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:

\[ \frac{1}{NC} \sum_{i} \max_{j \neq i} \left[ \frac{1}{n_i} \sum_{c_i} d(x, c_i) + \frac{1}{n_j} \sum_{c_j} d(x, c_j) \right] \] (8)

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 \( c_i \).

Table 2: adjusting RBF units’ centers in Wine

<table>
<thead>
<tr>
<th>n</th>
<th>Pop.</th>
<th>Iter</th>
<th>MSE</th>
<th>Std.</th>
<th>SEM</th>
<th>CI (95%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSO:</td>
<td>20</td>
<td>20</td>
<td>1500</td>
<td>0.19224</td>
<td>0.1235</td>
<td>0.0101</td>
</tr>
<tr>
<td></td>
<td>40</td>
<td>30</td>
<td>2000</td>
<td>0.16474</td>
<td>0.1207</td>
<td>0.0104</td>
</tr>
<tr>
<td></td>
<td>70</td>
<td>35</td>
<td>2500</td>
<td>0.14989</td>
<td>0.1013</td>
<td>0.0088</td>
</tr>
<tr>
<td>GA:</td>
<td>20</td>
<td>20</td>
<td>1500</td>
<td>0.19423</td>
<td>0.1242</td>
<td>0.0104</td>
</tr>
<tr>
<td></td>
<td>40</td>
<td>30</td>
<td>2000</td>
<td>0.16532</td>
<td>0.1222</td>
<td>0.0105</td>
</tr>
<tr>
<td></td>
<td>70</td>
<td>35</td>
<td>2500</td>
<td>0.3729</td>
<td>0.1072</td>
<td>0.0093</td>
</tr>
<tr>
<td>ICA:</td>
<td>20</td>
<td>20</td>
<td>1500</td>
<td>0.41448</td>
<td>0.1421</td>
<td>0.0123</td>
</tr>
<tr>
<td></td>
<td>40</td>
<td>30</td>
<td>2000</td>
<td>0.3396</td>
<td>0.1235</td>
<td>0.0107</td>
</tr>
<tr>
<td></td>
<td>70</td>
<td>35</td>
<td>2500</td>
<td>0.30012</td>
<td>0.1242</td>
<td>0.0107</td>
</tr>
</tbody>
</table>

Table 3: adjusting RBF units’ centers in Iris

<table>
<thead>
<tr>
<th>n</th>
<th>Pop.</th>
<th>Iter</th>
<th>MSE</th>
<th>Std.</th>
<th>SEM</th>
<th>CI (95%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSO:</td>
<td>25</td>
<td>35</td>
<td>2000</td>
<td>0.00957</td>
<td>0.0309</td>
<td>0.0036</td>
</tr>
<tr>
<td></td>
<td>35</td>
<td>50</td>
<td>2500</td>
<td>0.00661</td>
<td>0.0346</td>
<td>0.0033</td>
</tr>
<tr>
<td></td>
<td>40</td>
<td>70</td>
<td>3000</td>
<td>0.00541</td>
<td>0.0362</td>
<td>0.0032</td>
</tr>
<tr>
<td>GA:</td>
<td>25</td>
<td>35</td>
<td>2000</td>
<td>0.01078</td>
<td>0.0394</td>
<td>0.0037</td>
</tr>
<tr>
<td></td>
<td>35</td>
<td>50</td>
<td>2500</td>
<td>0.00975</td>
<td>0.0439</td>
<td>0.0041</td>
</tr>
<tr>
<td></td>
<td>40</td>
<td>70</td>
<td>3000</td>
<td>0.00598</td>
<td>0.0365</td>
<td>0.0033</td>
</tr>
<tr>
<td>ICA:</td>
<td>25</td>
<td>35</td>
<td>2000</td>
<td>0.02359</td>
<td>0.0666</td>
<td>0.0063</td>
</tr>
<tr>
<td></td>
<td>35</td>
<td>50</td>
<td>2500</td>
<td>0.01497</td>
<td>0.0438</td>
<td>0.0041</td>
</tr>
<tr>
<td></td>
<td>40</td>
<td>70</td>
<td>3000</td>
<td>0.01332</td>
<td>0.0368</td>
<td>0.0035</td>
</tr>
</tbody>
</table>

9. 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 [79]. Table 1 shows the main characteristics of these data sets. In the experiments, 70% of data set 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.),
Table 6: Classification of Wine data set based on RBF-PSO optimization algorithm

<table>
<thead>
<tr>
<th>n</th>
<th>Pop.</th>
<th>Iter.</th>
<th>Training data set</th>
<th>Test data set</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>MSE</td>
<td>Std.</td>
</tr>
<tr>
<td>20</td>
<td>25</td>
<td>2000</td>
<td>0.00838</td>
<td>0.0912</td>
</tr>
<tr>
<td>40</td>
<td>30</td>
<td>2500</td>
<td>0.00586</td>
<td>0.08024</td>
</tr>
<tr>
<td>70</td>
<td>40</td>
<td>3000</td>
<td>0.00519</td>
<td>0.07145</td>
</tr>
</tbody>
</table>

Table 7: Classification of Iris data set based on RBF-PSO optimization algorithm

<table>
<thead>
<tr>
<th>n</th>
<th>Pop.</th>
<th>Iter.</th>
<th>Training data set</th>
<th>Test data set</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>MSE</td>
<td>Std.</td>
</tr>
<tr>
<td>20</td>
<td>25</td>
<td>2000</td>
<td>0.00848</td>
<td>0.0943</td>
</tr>
<tr>
<td>40</td>
<td>30</td>
<td>2500</td>
<td>0.00598</td>
<td>0.08227</td>
</tr>
<tr>
<td>70</td>
<td>40</td>
<td>3000</td>
<td>0.00525</td>
<td>0.07254</td>
</tr>
</tbody>
</table>

Table 8: Classification of Ionosphere data set based on RBF-PSO optimization algorithm

<table>
<thead>
<tr>
<th>n</th>
<th>Pop.</th>
<th>Iter.</th>
<th>Training data set</th>
<th>Test data set</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>MSE</td>
<td>Std.</td>
</tr>
<tr>
<td>20</td>
<td>25</td>
<td>2000</td>
<td>0.01159</td>
<td>0.10808</td>
</tr>
<tr>
<td>40</td>
<td>30</td>
<td>2500</td>
<td>0.00906</td>
<td>0.09555</td>
</tr>
<tr>
<td>70</td>
<td>40</td>
<td>3000</td>
<td>0.00647</td>
<td>0.08002</td>
</tr>
</tbody>
</table>

Standard Error of Mean (SEM), Confidence Interval (CI) by 95% and the number of incorrect classification (Cls. 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 [80, 81, 82, 83, 84, 85], Genetic Algorithm (GA) [86, 87, 88, 89, 90, 91], Imperialist Competitive Algorithm (ICA) [92, 93, 94] and Differential Evolution (DE) [95, 96, 97, 98]. The experiments on each algorithm were repeated 20 times independently to find
Table 9: Classification of Zoo data set based on RBF-PSO optimization algorithm

<table>
<thead>
<tr>
<th>n</th>
<th>Pop.</th>
<th>Iter</th>
<th>MSE</th>
<th>Std</th>
<th>CI (95%)</th>
<th>SEM</th>
<th>Cls. err.</th>
<th>MSE</th>
<th>Std</th>
<th>CI (95%)</th>
<th>SEM</th>
<th>Cls. err.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Units’ centers by PSO:</td>
<td>30</td>
<td>50</td>
<td>2000</td>
<td>0.00156</td>
<td>0.03974</td>
<td>[-0.077 0.079]</td>
<td>0.00455</td>
<td>3</td>
<td>0.00394</td>
<td>0.0552</td>
<td>[-0.113 0.104]</td>
<td>0.01104</td>
</tr>
<tr>
<td></td>
<td>40</td>
<td>70</td>
<td>2500</td>
<td>0.00063</td>
<td>0.03024</td>
<td>[-0.059 0.059]</td>
<td>0.00366</td>
<td>1</td>
<td>0.00471</td>
<td>0.07</td>
<td>[-0.14 0.135]</td>
<td>0.01401</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>90</td>
<td>3000</td>
<td>0.00095</td>
<td>0.03114</td>
<td>[-0.061 0.061]</td>
<td>0.00335</td>
<td>3</td>
<td>0.00607</td>
<td>0.07197</td>
<td>[-0.157 0.131]</td>
<td>0.01439</td>
</tr>
<tr>
<td>Units’ centers by GA:</td>
<td>30</td>
<td>50</td>
<td>2000</td>
<td>0.00222</td>
<td>0.47477</td>
<td>[-0.093 0.094]</td>
<td>0.00544</td>
<td>4</td>
<td>0.00904</td>
<td>0.08952</td>
<td>[-0.212 0.139]</td>
<td>0.0179</td>
</tr>
<tr>
<td></td>
<td>40</td>
<td>70</td>
<td>2500</td>
<td>0.00141</td>
<td>0.03783</td>
<td>[-0.074 0.074]</td>
<td>0.00433</td>
<td>5</td>
<td>0.00595</td>
<td>0.07695</td>
<td>[-0.167 0.134]</td>
<td>0.01539</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>90</td>
<td>3000</td>
<td>0.00115</td>
<td>0.03422</td>
<td>[-0.067 0.067]</td>
<td>0.00392</td>
<td>4</td>
<td>0.00628</td>
<td>0.06858</td>
<td>[-0.147 0.122]</td>
<td>0.01383</td>
</tr>
<tr>
<td>Units’ centers by ICA:</td>
<td>30</td>
<td>50</td>
<td>2000</td>
<td>0.00211</td>
<td>0.04628</td>
<td>[-0.089 0.093]</td>
<td>0.0053</td>
<td>5</td>
<td>0.00706</td>
<td>0.08286</td>
<td>[-0.155 0.17]</td>
<td>0.01657</td>
</tr>
<tr>
<td></td>
<td>40</td>
<td>70</td>
<td>2500</td>
<td>0.00111</td>
<td>0.0334</td>
<td>[-0.065 0.066]</td>
<td>0.00383</td>
<td>2</td>
<td>0.00487</td>
<td>0.06209</td>
<td>[-0.155 0.17]</td>
<td>0.01241</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>90</td>
<td>3000</td>
<td>0.00102</td>
<td>0.03228</td>
<td>[-0.063 0.064]</td>
<td>0.0037</td>
<td>2</td>
<td>0.00826</td>
<td>0.08935</td>
<td>[-0.2 0.151]</td>
<td>0.01787</td>
</tr>
<tr>
<td>Units’ centers by DE:</td>
<td>30</td>
<td>50</td>
<td>2000</td>
<td>0.00159</td>
<td>0.04024</td>
<td>[-0.079 0.078]</td>
<td>0.00461</td>
<td>3</td>
<td>0.00514</td>
<td>0.07046</td>
<td>[-0.158 0.119]</td>
<td>0.01409</td>
</tr>
<tr>
<td></td>
<td>40</td>
<td>70</td>
<td>2500</td>
<td>0.00136</td>
<td>0.03712</td>
<td>[-0.073 0.073]</td>
<td>0.00425</td>
<td>4</td>
<td>0.00848</td>
<td>0.08664</td>
<td>[-0.206 0.134]</td>
<td>0.01733</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>90</td>
<td>3000</td>
<td>0.00113</td>
<td>0.03385</td>
<td>[-0.066 0.066]</td>
<td>0.00388</td>
<td>2</td>
<td>0.00635</td>
<td>0.074</td>
<td>[-0.155 0.135]</td>
<td>0.0148</td>
</tr>
</tbody>
</table>

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

Figure 5: Optimal Pareto fronts of Wine data set

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

Figure 6: Optimal Pareto fronts of Iris data set

the optimal considered performance criteria. Tables 2-5 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.

Secondly, for adjusting the RBF units’ widths and output weights, we integrate the optimal placement of centers from four applied optimization algorithm (from Tables 2-5) with PSO. The obtained results are shown in Tables 6-9. The classification error (Cls. err.) is calculated based on 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 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 centers. To confirm convincingly this claim, this paper presents a multi-objective approach to find the (near) optimal placement of centers. According to the first part of the proposed method (see Fig. 4), 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. 5-8 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. Afterward, we integrate the optimal placement of units’ centers (obtained by our two-objective approach in Figs. 5-8) with the PSO (see second step of the first phase in Fig. 4) 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 10-13. As seen in these tables, the first five optimal Pareto solutions outperform significantly the other methods by single-objective approach in Tables 6-9 based on the MSE, Std. and the number 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
<table>
<thead>
<tr>
<th>n</th>
<th>Pop.</th>
<th>Iter.</th>
<th>NSGA II MSE</th>
<th>DBI MSE</th>
<th>Training data set MSE</th>
<th>Std. CI (95%)</th>
<th>SEM</th>
<th>Cls. err.</th>
<th>Test data set MSE</th>
<th>Std. CI (95%)</th>
<th>SEM</th>
<th>Cls. err.</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.2252</td>
<td>0.566</td>
<td>0.00657</td>
<td>0.08015</td>
<td>[0.137 0.158]</td>
<td>0.00892</td>
<td>0</td>
<td></td>
<td>0.01042</td>
<td>0.0331</td>
<td>[0.222 0.191]</td>
<td>0.01708</td>
</tr>
<tr>
<td>20</td>
<td>25</td>
<td>2500</td>
<td>0.2303</td>
<td>0.566</td>
<td>0.00702</td>
<td>0.08134</td>
<td>[0.169 0.171]</td>
<td>0.00759</td>
<td>1</td>
<td>0.01055</td>
<td>0.0433</td>
<td>[0.232 0.201]</td>
</tr>
<tr>
<td>40</td>
<td>30</td>
<td>2500</td>
<td>0.2305</td>
<td>0.5362</td>
<td>0.00734</td>
<td>0.08196</td>
<td>[0.162 0.171]</td>
<td>0.00759</td>
<td>1</td>
<td>0.01062</td>
<td>0.0871</td>
<td>[0.230 0.21]</td>
</tr>
<tr>
<td>70</td>
<td>40</td>
<td>3500</td>
<td>0.2004</td>
<td>0.399</td>
<td>0.00499</td>
<td>0.07017</td>
<td>[0.138 0.138]</td>
<td>0.00607</td>
<td>0</td>
<td>0.01074</td>
<td>0.0322</td>
<td>[0.221 0.184]</td>
</tr>
<tr>
<td>100</td>
<td>50</td>
<td>3500</td>
<td>0.2095</td>
<td>0.3799</td>
<td>0.00527</td>
<td>0.07952</td>
<td>[0.155 0.157]</td>
<td>0.00678</td>
<td>0</td>
<td>0.01316</td>
<td>0.1134</td>
<td>[0.251 0.198]</td>
</tr>
<tr>
<td>130</td>
<td>70</td>
<td>3500</td>
<td>0.2074</td>
<td>0.3646</td>
<td>0.00556</td>
<td>0.07985</td>
<td>[0.156 0.157]</td>
<td>0.00702</td>
<td>0</td>
<td>0.01315</td>
<td>0.1134</td>
<td>[0.259 0.185]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>n</th>
<th>Pop.</th>
<th>Iter.</th>
<th>NSGA II MSE</th>
<th>DBI MSE</th>
<th>Training data set MSE</th>
<th>Std. CI (95%)</th>
<th>SEM</th>
<th>Cls. err.</th>
<th>Test data set MSE</th>
<th>Std. CI (95%)</th>
<th>SEM</th>
<th>Cls. err.</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.0136</td>
<td>0.2759</td>
<td>0.00465</td>
<td>0.0852</td>
<td>[0.134 0.134]</td>
<td>0.00644</td>
<td>1</td>
<td></td>
<td>0.01183</td>
<td>0.0909</td>
<td>[0.198 0.23]</td>
<td>0.01939</td>
</tr>
<tr>
<td>25</td>
<td>35</td>
<td>2500</td>
<td>0.0129</td>
<td>0.3903</td>
<td>0.00452</td>
<td>0.07929</td>
<td>[0.142 0.144]</td>
<td>0.00866</td>
<td>2</td>
<td>0.01285</td>
<td>0.0272</td>
<td>[0.168 0.235]</td>
</tr>
<tr>
<td>35</td>
<td>50</td>
<td>3000</td>
<td>0.0111</td>
<td>0.1911</td>
<td>0.00336</td>
<td>0.05651</td>
<td>[0.131 0.131]</td>
<td>0.00531</td>
<td>1</td>
<td>0.01397</td>
<td>0.0926</td>
<td>[0.153 0.191]</td>
</tr>
<tr>
<td>40</td>
<td>70</td>
<td>3500</td>
<td>0.0105</td>
<td>0.2122</td>
<td>0.00381</td>
<td>0.06083</td>
<td>[0.142 0.133]</td>
<td>0.00608</td>
<td>1</td>
<td>0.01348</td>
<td>0.0913</td>
<td>[0.146 0.212]</td>
</tr>
</tbody>
</table>

Evaluation environment

We use simulations to quantify effects of DoS attacks and their countermeasures. In this work, we used the open-source ndnSIM [99] package, which implements NDN protocol stack for NS-3 network simulator (http://www.nsnam.org/), to run simulations.
### Table 12: Classification of Ionosphere data set based on proposed method

<table>
<thead>
<tr>
<th>n</th>
<th>Pop.</th>
<th>Iter.</th>
<th>NSGA II Training data set</th>
<th>Test data set</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MSE</td>
<td>Std</td>
<td>CI (95%)</td>
<td>SEM Cls. err.</td>
</tr>
<tr>
<td>30</td>
<td>60</td>
<td>2000</td>
<td>1.2472 0.2147 [-0.443 0.439] 0.01346 12</td>
<td>0.05129 0.2064 [-0.371 0.438] 0.02139 2</td>
</tr>
<tr>
<td>40</td>
<td>80</td>
<td>2500</td>
<td>1.2266 0.2210 [-0.442 0.442] 0.01351 11</td>
<td>0.05392 0.2026 [-0.366 0.472] 0.02957 2</td>
</tr>
<tr>
<td>50</td>
<td>90</td>
<td>3000</td>
<td>1.2539 0.2147 [-0.443 0.442] 0.01353 12</td>
<td>0.05308 0.2116 [-0.389 0.439] 0.02079 2</td>
</tr>
</tbody>
</table>

### Table 13: Classification of Zoo data set based on proposed method

<table>
<thead>
<tr>
<th>n</th>
<th>Pop.</th>
<th>Iter.</th>
<th>NSGA II Training data set</th>
<th>Test data set</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MSE</td>
<td>Std</td>
<td>CI (95%)</td>
<td>SEM Cls. err.</td>
</tr>
<tr>
<td>50</td>
<td>90</td>
<td>3000</td>
<td>0.7917 0.3458 [-0.044 0.045] 0.00261 0</td>
<td>0.00382 0.0522 [-0.132 0.075] 0.01204 1</td>
</tr>
<tr>
<td>60</td>
<td>120</td>
<td>6000</td>
<td>0.8269 0.3325 [-0.048 0.049] 0.0028 0</td>
<td>0.0055 0.0512 [-0.126 0.075] 0.01125 1</td>
</tr>
<tr>
<td>70</td>
<td>150</td>
<td>9000</td>
<td>0.7474 0.2903 [-0.054 0.055] 0.00319 0</td>
<td>0.00413 0.0598 [-0.121 0.114] 0.01097 2</td>
</tr>
<tr>
<td>80</td>
<td>180</td>
<td>12000</td>
<td>0.7383 0.2921 [-0.057 0.057] 0.00332 0</td>
<td>0.00304 0.0563 [-0.112 0.108] 0.01126 2</td>
</tr>
</tbody>
</table>

The proposed method first phase was implemented by the MATLAB software on the Intel Pentium 2.13 GHz CPU, 4 GB RAM running Windows 7 Ultimate. The algorithm deployed to C++ project integrating as a C++ shared library using the MATLAB Compiler. Then, this C++ proposed adaptive reaction was also implemented with C++ in adHSM environment. We demonstrate through simulations that our countermeasure satisfies consideration for evaluating the performance of considered mitigating method, adHSM simulation environment reproduces the basic structures of a NDN node (i.e., CS, PIT, FIB, strategy layer, and so on).
ably applied performance metrics as compared to two recently applied DoS attack mitigation methods namely satisfaction-based pushback and satisfaction-based Interest acceptance [15]. We perform 10 times simulation runs to calculate the average performance metrics. Our experiments are performed over two topologies shown in Figs. 9 and 10. Fig. 9 corresponds to DFN-like (Deutsche Forschungsnetz as the German Research Network) [100], and Fig. 10 corresponds to the AT&T network [101]. We use the symbols $C_x$, $P_x$, $R_x$, and $A_x$ 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 [2]. 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 3):

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 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 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 14.

11. 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.

11.1. Detection Phase

This step adopts our proposed intelligent classifier from section 8.1. We choose the DFN-like topology (Fig. 9) in the training phase with the recommended parameter settings in Table 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 figures.
Figure 10: AT&T topology

Table 14: Network parameters considered

<table>
<thead>
<tr>
<th>Node Distribution</th>
<th>Pattern</th>
<th>Frequency</th>
<th>Run time (minute)</th>
<th>Producer</th>
<th>Goal</th>
</tr>
</thead>
<tbody>
<tr>
<td>DFN-like topology (Fig. 9)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C1</td>
<td>randomize</td>
<td>uniform</td>
<td>[100 500]</td>
<td>0-40</td>
<td>P1</td>
</tr>
<tr>
<td>C2</td>
<td>randomize</td>
<td>exponential</td>
<td>[100 500]</td>
<td>2-40</td>
<td>P2</td>
</tr>
<tr>
<td>C3</td>
<td>Zipf-Mandelbort ($\alpha = [0.5 0.9]$)</td>
<td>exponential</td>
<td>[100 500]</td>
<td>3-40</td>
<td>P3</td>
</tr>
<tr>
<td>C4</td>
<td>randomize</td>
<td>exponential</td>
<td>[100 500]</td>
<td>5-40</td>
<td>P6</td>
</tr>
<tr>
<td>C5</td>
<td>Zipf-Mandelbort ($\alpha = [0.5 0.9]$)</td>
<td>exponential</td>
<td>[100 500]</td>
<td>3-40</td>
<td>P2, P3</td>
</tr>
<tr>
<td>C6</td>
<td>randomize</td>
<td>uniform</td>
<td>[100 500]</td>
<td>3-40</td>
<td>P3</td>
</tr>
<tr>
<td>C7</td>
<td>randomize</td>
<td>uniform</td>
<td>[100 500]</td>
<td>7-16, 22-31</td>
<td>P6, P4</td>
</tr>
<tr>
<td>C8</td>
<td>randomize</td>
<td>exponential</td>
<td>[100 500]</td>
<td>8-18, 25-40</td>
<td>P1</td>
</tr>
<tr>
<td>C9</td>
<td>randomize</td>
<td>uniform</td>
<td>[1500 3000]</td>
<td>0-50</td>
<td>P0, P1</td>
</tr>
<tr>
<td>C10</td>
<td>randomize</td>
<td>exponential</td>
<td>[200 600]</td>
<td>2-50</td>
<td>P0</td>
</tr>
<tr>
<td>C11</td>
<td>randomize</td>
<td>exponential</td>
<td>[200 600]</td>
<td>3-50</td>
<td>P1</td>
</tr>
<tr>
<td>C12, C13</td>
<td>randomize</td>
<td>uniform</td>
<td>[200 600]</td>
<td>5-40</td>
<td>P0, P1</td>
</tr>
<tr>
<td>C14, C15</td>
<td>randomize</td>
<td>uniform</td>
<td>[200 600]</td>
<td>7-16</td>
<td>P6</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>AT&amp;T topology (Fig. 10)</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>C0, C7</td>
<td>randomize</td>
<td>uniform</td>
<td>[200 600]</td>
<td>0-50</td>
<td>P0, P1</td>
</tr>
<tr>
<td>C1, C8</td>
<td>randomize</td>
<td>exponential</td>
<td>[200 600]</td>
<td>2-50</td>
<td>P0</td>
</tr>
<tr>
<td>C2, C9</td>
<td>randomize</td>
<td>exponential</td>
<td>[200 600]</td>
<td>3-50</td>
<td>P1</td>
</tr>
<tr>
<td>C3, C10</td>
<td>randomize</td>
<td>uniform</td>
<td>[200 600]</td>
<td>4-50</td>
<td>P1</td>
</tr>
<tr>
<td>C4, C11</td>
<td>Zipf-Mandelbort ($\alpha = [0.5 0.9]$)</td>
<td>exponential</td>
<td>[200 600]</td>
<td>5-40</td>
<td>P0, P1</td>
</tr>
<tr>
<td>C5, C12, C13</td>
<td>randomize</td>
<td>uniform</td>
<td>[200 600]</td>
<td>6-50</td>
<td>P0, P1</td>
</tr>
<tr>
<td>C14, C15</td>
<td>randomize</td>
<td>uniform</td>
<td>[200 600]</td>
<td>8-50</td>
<td>P1</td>
</tr>
<tr>
<td>A0</td>
<td>randomize</td>
<td>uniform</td>
<td>[1000 3000]</td>
<td>7-25</td>
<td>P1</td>
</tr>
<tr>
<td>A1</td>
<td>Zipf-Mandelbort ($\alpha = [0.5 0.9]$)</td>
<td>exponential</td>
<td>[500 1000]</td>
<td>7-25</td>
<td>P0</td>
</tr>
<tr>
<td>A2</td>
<td>Zipf-Mandelbort ($\alpha = [0.5 0.9]$)</td>
<td>uniform</td>
<td>[1000 3000]</td>
<td>30-45</td>
<td>P1</td>
</tr>
</tbody>
</table>

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 [15, 20]. Hence, we employed simple intrinsic features from the network which is shown in Table 15 (i.e., the input features in the RBF neural network).

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 16. The optimal Pareto front solutions by NSGA II are also depicted in Fig. 11. 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. 12 and 13, respectively. As seen in these Figures and Table 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).

As we expected (based on our proof in section 9), This phase constructs an optimized and 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 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.

### Table 15: Feature construction

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>InInterests</td>
<td>a number of arrival Interest in an interface</td>
</tr>
<tr>
<td>InData</td>
<td>a number of arrival data in an interface</td>
</tr>
<tr>
<td>InSatisfiedInterests</td>
<td>a number of satisfied Interests where interface was part of the incoming set</td>
</tr>
<tr>
<td>InTimedOutInterests</td>
<td>a number of timed out Interests where interface was part of the incoming set</td>
</tr>
<tr>
<td>OutInterests</td>
<td>a number of sent Interest from an interface</td>
</tr>
<tr>
<td>OutData</td>
<td>a number of sent data from an interface</td>
</tr>
<tr>
<td>OutSatisfiedInterests</td>
<td>a number of satisfied Interests where interface was part of the outgoing set</td>
</tr>
<tr>
<td>OutTimedOutInterests</td>
<td>a number of timed out Interests where interface was part of the outgoing set</td>
</tr>
<tr>
<td>DropInterests</td>
<td>a number of dropped Interest in an interface</td>
</tr>
<tr>
<td>DropData</td>
<td>a number of dropped data in an interface</td>
</tr>
<tr>
<td>SatisfiedInterests</td>
<td>a total number of satisfied Interests</td>
</tr>
<tr>
<td>TimedOutInterests</td>
<td>a total number of timed out Interests</td>
</tr>
</tbody>
</table>

Figure 11: Optimal Pareto fronts of DFN-like training phase

### 11.2. Reaction Phase

Once a DoS attack from interface \( j \) of router \( i \) has been identified with the proposed proactive detector (see section 11.1), 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 de-
11.2.1. 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 [11]:

1. during an attack, the PTT 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. 14. Assuming in a time interval in router C the predictor reports a misbehaving traffic from a consumer (neigh-
Table 16: Classification of NDN data set based on proposed method

<table>
<thead>
<tr>
<th>Pop.</th>
<th>Iter.</th>
<th>NSGA II</th>
<th>Training data set</th>
<th>Test data set</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSE</td>
<td>DBI</td>
<td>MSE</td>
<td>Std. CI (95%)</td>
<td>SEM Cls. err.</td>
</tr>
<tr>
<td>0.0314</td>
<td>0.0979</td>
<td>0.00998</td>
<td>[-0.192 0.197]</td>
<td>0.00555 2</td>
</tr>
<tr>
<td>0.0643</td>
<td>0.0979</td>
<td>0.00994</td>
<td>[-0.186 0.206]</td>
<td>0.00555 2</td>
</tr>
<tr>
<td>0.0315</td>
<td>0.0914</td>
<td>0.00952</td>
<td>[-0.187 0.186]</td>
<td>0.00543 1</td>
</tr>
</tbody>
</table>

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 inter-
faces limit to 'announced reduced rate' × 'local unsatisfied rate' in each local inter-
face. 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 φ = 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 con-
sumed 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 traf-
fc 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 2counter. 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. 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.

```
Input: AlertMsg, timestamp of alert generation, reduced rate and wait time from interface j in router i (r_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
5.     foreach local interface j do
6.       new rate = unsatisfied rate of j × announced reduced rate;
7.       if φ < new rate then
8.         /* intensify the penalty */
9.         Decrease(interface j, new rate, announced wait time);
10.      else
11.         /* reset to original setting */
12.         Increase(interface j, original rate);
13.   else
14.     Drop(AlertMsg);
15.   end

/* section: Pushback alert message generation */
16. if (predictor module reports the adversary consumer (neighbor) in local interface j) then
17.   if (time from last sent AlertMsg to local interface r_i < current local time) then
18.     /* Pushback alert message generation */
19.     new time interval = 2^counter_j;
20.     AlertMsg = (current timestamp of alert generation, current unsatisfied rate in local interface j, new time interval);
21.     Send(AlertMsg to r_i);
22.     counter_j = counter_j + 1;
23. else
24.     /* reset to original setting */
25.     counter_j = 0;
26.     Increase(interface j, original rate);

Algorithm 1: Unsatisfied-based pushback algorithm
```

11.2.2. 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 11.2.1). 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 traffic. Then, the rate of incoming Interest packets should gradually decrease in upstream interface j based on our ascending penalty mechanism in previous subsection.

12. Experimental results and evaluation

In this section we report the experimental evaluation of countermeasures presented in Section 11. Our countermeasures are tested over two considered topologies in Figs. 9 and 10. Each router implements the proposed detection technique discussed in Section 11.1 and adaptive reaction technique discussed in 11.2. We report the results based on the five conditions: baseline, attack (no countermeasure), our proposed method, and two DoS mitigation methods applied in this work including satisfaction-based pushback and satisfaction-based Interest acceptance from [15]. Figs. 15 and 18 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 traffic and preventing legitimate users from service degradation by the accuracy more than 90% during the attack. Figs. 16 and 19 demonstrate the average PIT usage within 10 runs in five considered conditions in DFN-like
Figure 15: Interest satisfaction ratio for legitimate users in DFN

Figure 16: PIT usage with countermeasures in DFN

Figure 17: Effects of countermeasures in DFN (Throughput)
Figure 18: Interest satisfaction ratio for legitimate users in AT&T

Figure 19: PIT usage with countermeasures in AT&T

Figure 20: Effects of countermeasures in AT&T (Throughput)
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. 17 and 20 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.

12.1. 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 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 (see section 11.2.2). 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 11.2). Table 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. 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).

12.2. 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. 4) 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
Table 17: Comparison of false positive rate (mean of 10 runs)

<table>
<thead>
<tr>
<th>Topology</th>
<th>No countermeasure</th>
<th>Satisfaction-based Interest acceptance</th>
<th>Satisfaction-based pushback</th>
<th>Our method</th>
</tr>
</thead>
<tbody>
<tr>
<td>DFN</td>
<td>39.78%</td>
<td>21.05%</td>
<td>14.47%</td>
<td>6.44%</td>
</tr>
<tr>
<td>AT&amp;T</td>
<td>66.43%</td>
<td>24.29%</td>
<td>19.13%</td>
<td>9.26%</td>
</tr>
</tbody>
</table>

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 2-5 (training units’ centers) and Tables 6-9 (training units’ widths, output weights and calculating the misclassification error), and our conflicting two-objective approach in Figs. 5-8 (Pareto front of the units’ centers solutions) and Tables 10-13 (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 6-9 demonstrate the final results, using PSO, Genetic Algorithm (GA), Imperialist Competitive Algorithm (ICA) and Differential Evolution (DE). The proposed algorithm in this paper 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 11.1). 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 11.2). 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. 9) and AT&T (Fig. 10). The results were promising as compared to two recently proposed DoS mitigation methods from [15] 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.

13. 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 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.

We are currently investigating inter-domain DoS attacks. We leave further investigations to future work.

14. Acknowledgment

This work was partially supported by projects TIN2013-47272-C2-2 and SGR-2014-881.

References


[98] Heckmann, O.. The competitive Internet service provider: network architecture, interconnection, traffic engineering and network design. Wiley series in communications networking & distributed systems; J. Wiley; 2006. URL books.google.es/books?id=DyEzaQAkIAAJ.