I. J. Computer Network and Information Security, 2023, 5, 63-72
Published Online on October 8, 2023 by MECS Press (http://www.mecs-press.org/)
DOI: 10.5815/ijcnis.2023.05.06

Detection of DDOS Attacks on Cloud Computing Environment Using Altered Convolutional Deep Belief Networks

S. Sureshkumar*
Karpagam Academy of Higher Education, Department of Computer Science and Engineering, Coimbatore, 641021, India
E-mail: sureshkumarcse2022@gmail.com
ORCID iD: https://orcid.org/0000-0002-5288-2039
*Corresponding Author

G. K. D. Prasanna Venkatesan
Karpagam Academy of Higher Education, Department of Computer Science and Engineering, Coimbatore, 641021, India
E-mail: dean.engineering@kahedu.edu.in
ORCID iD: https://orcid.org/0000-0002-0638-1938

R. Santhosh
Karpagam Academy of Higher Education, Department of Computer Science and Engineering, Coimbatore, 641021, India
E-mail: santhoshrd@gmail.com
ORCID iD: https://orcid.org/0000-0002-9287-5829

Received: 18 June 2022; Revised: 03 October 2022; Accepted: 01 December 2022; Published: 08 October 2023

Abstract: The primary benefits of Clouds are that they can elastically scale to meet variable demands and provide corresponding environments for computing. Cloud infrastructures require highest levels of protections from DDoS (Distributed Denial-of-Services). Attacks from DDoSs need to be handled as they jeopardize availability of networks. These attacks are becoming very complex and are evolving at rapid rates making it complex to counter them. Hence, this paper proposes GKDPCAs (Gaussian kernel density peak clustering techniques) and ACDBNs (Altered Convolution Deep Belief Networks) to handle these attacks. DPCAs (density peak clustering algorithms) are used to partition training sets into numerous subgroups with comparable characteristics, which help in minimizing the size of training sets and imbalances in samples. Subset of ACDBNs get trained in each subgroup where FSs (feature selections) of this work are executed using SFOs (Sun-flower Optimizations) which evaluate the integrity of reduced feature subsets. The proposed framework has superior results in its experimental findings while working with NSL-KDD and CICIDS2017 datasets. The resulting overall accuracies, recalls, precisions, and F1-scoresare better than other known classification algorithms. The framework also outperforms other IDTs (intrusion detection techniques) in terms of accuracies, detection rates, and false positive rates.

Index Terms: Cloud Computing, Distribute Denial of Service Attacks, Gaussian Kernel Density Peak Clustering Algorithm, Altered Convolutional Deep Belief Networks (ACDBN) and Sun-flower Optimization.

1. Introduction

E-governance has surely made citizens' life simpler and communications easier. Furthermore, e-government enhances public services and fosters democratic processes by allowing wider access to information and there have been significant shifts towards "paperless governments" using technologies. The use of clouds and allied services are steadily expanding [1]. The growing use of electronic technology and apps in government services has contributed significantly to citizen happiness and cost savings. Even if the move to digital governance offers numerous benefits for the quality of government services, it also comes with a slew of security risks. DoS (denial of service) assaults are one of the most

This work is open access and licensed under the Creative Commons CC BY 4.0 License.

serious risks and security challenges that e-government faces. DoS assaults have previously brought down several of the most popular e-government sites for many hours, resulting in massive losses and restoration expenses [2].

Cloud computing or Clouds are now regarded as the most recent computing paradigm, offering a variety of services that are flexible and consistent. Cloud computing infrastructures are being used by not just private firms and people, but also government departments to improve availability of services [3]. Clouds have flexibility in providing network accesses to computer resources or shared resources to users quickly whenever they demand with literally no need for management or involvement from service providers. According to NIST [4]. Pay-per-use, virtualizations, on-demand accesses, flexibility, low-cost hardware and maintenance expenses all contribute to cloud computing's appeal. Virtualizations have lately been implemented in a variety of levels, including networks, CPUs, memories, and storage devices.

They improve system availabilities while simultaneously lowering costs and presenting more flexible solutions. Attacks from DDoSs are a major source of downtime in these infrastructures. The attackers acquire powers to interrupt or entirely shut down network connections after compromising a significant number of agents or hosts. They then utilises these agents to execute depletion attacks on target networks. DDoSs mainly aim to prevent victims from accessing resources including Web servers, CPUs, Storage units, and other Network resources [5]. DDoSs can drastically degrade the performances of cloud services by destroying virtual servers.

These issues have slowed down Cloud computing adoptions and deployments. DDoSs prevent legitimate cloud computing users from accessing shared services or resources. The targeted servers experience traffic from multiple sources at the same time and exhaust their resources in managing these sudden loads [6]. IIDSs (Intelligent Intrusion Detection Systems) that can identify and detect unusual activities in network traffics are required. The proposed technique uses Clustering, training, and testing where NSL-KDD and CICIDS2017 datasets are split into training and testing sets. Then, from the specified input databases, training datasets are extracted which are then grouped using GKDPCAs for decreased complexity. Subsequently ACDBNs get trained on the subset. SFOs are utilized in this work to choose the best features thus assisting in minimizing time complexities and improving detection accuracies. Finally, data is categorized as benign or attacks using ACDBNs.

The remainder of the paper is laid out as follows: In Section 2, a brief review of investigations related to the suggested approach is presented. The authors discuss the research's background and offer DDOS attack detection in Section 3. Section 4 explains the complete experimental results and comments. In Section 5, the conclusion is summarized, and recommendations for further work are presented.

2. Related Works

Many studies have attempted to mitigate assaults from DDoSs while explaining their consequences in cloud computing environments. Sahi et al. [7] provided a novel classifier system for identifying and combating DDoS TCP flood assaults on public clouds called CS-DDoS. Their proposed CS-DDoS secured stored data by categorising incoming packets using least squares and SVMs (support vector machines) i.e. LS-SVM, NBs (nave Bayes), KNNs (K-nearest neighbours), and MLPs (multilayer Perceptrons). The proposed scheme made decisions based on the findings of classifications. Bhardwaj et al. [8] presented an architectural combination of AEs (Auto Encoders) for feature learning with DNNs (Deep Neural Networks) for benign network traffic categorizations and DDoSs traffic. The work by tweaking settings and using correctly devised methodologies, their proposed AE and DNNs were optimized for detection of assaults from DDoSs. Singh et al. [9] suggested IDSs (Intrusion Detection Systems) that blended COFSs (Cuckoo Optimization Feature Selections) with NBs where the former eliminated redundant and unnecessary characteristics before passing the processed features to NBs, which handled categorizations. Kasim[10]merged AEs with SVMs for effective and resilient DLTs (deep learning techniques). Their strategy based on self-taught learning identifies attacks from DDOSs in network traffics. AEs learnt features reduced dimensionalities. The study used SVMs in place of soft-max layer in their DLT. Akmakç et al., [11] proposed E-KOADs (Enhanced Kernel Online Anomaly Detections), a revolutionary detection technique for DDoSs using updated dictionaries of normal traffics to compute Mahalanob is distances. They demonstrated that their algorithm's detection accuracy with appropriate thresholds was unaffected by the time taken to resolve alarms. DDoSs were detected using proposed SaE-ELM (Self-adaptive evolutionary extreme learning machine) by Kushwah & Ranga [12] in their work. The study added 2 new features where most appropriate crossover operators were identified and followed by automatic finding of needed hidden layer neuron counts. These characteristics enhanced the model's learning and classification skills. To identify DDoSs, Kushwah & Ranga [13] suggested combining MLTs (machine learning techniques). The study’s hybrid MLTs were based on ELMs (extreme learning machines) and adaptive DEs (differential evolutions). DEs maximized weights of linked hidden layer inputs while biases of ELMs for output and hidden layers were analytically calculated. Verma et al. [14] proposed adaptive hybrid approaches for selecting attributes of of incoming communication information for classifications. The study’s pre-processed adaptive attribute selections, and prevented systems by their detections. The study’s tests on NSL-KDD datasets aided in the evaluation of their suggested method where results showed that MADs (Mean Absolute Deviations) in combination with RFs (Random Forests) called MAD-RF performed better than other options.

Most existing request categorizations are based on statistics which do not operate well under a variety of network

situations or for varied levels of attacks from DDoS.Adaptive strategies are pertinent to handle different incoming traffic. A similar technique finds rogue nodes in high-traffic environments, but the structure with a large number of nodes was not taken into account. Attacks from DDoSs will not be noticed in real time if the selected characteristics are not correctly extracted. To solve the aforementioned issues, we presented the DPCAs and ACDBNs classifiers for identifying attacks from DDoSs. When compared to state-of-the-art algorithms, ACDBNs has significant advantages in pre-training due to its fine-tuning learning approaches and multi-layer structures for extracting deep properties of training data.

As a result, ACDBNs overcome issues including poor training efficiency, local optimum, and detecting complicated network attacks. The major goal of this research is to present a system that uses the clustering approach followed by classification to efficiently identify assaults from DDoSs in any networked systems.

3. Proposed Methodology

The proposed model's main goal is to automatically detect attacks from DDoSs from input datasets. Figure 1 depicts the architecture of the developed detections using ACDBNs. Pre-processing, anomaly clustering, optimum feature selection, and classification are all steps of the proposed model. Data cleaning, label encoding, feature deletion, and normalization are all used in the pre-processing step. When employing GKDPCAs to create clusters, the cluster creation module prioritises highly scored characteristics. The suggested approach for finding unique features uses optimal feature selections. SFOs utilised in this work carry out feature selections. In addition, ACDBNs provide ideally gathered characteristics for identifying DDoSs. Furthermore, the suggested feature selections aim to reduce associations between selected characteristics and detection errors. As a result, the final classification model determines whether DDoSs are benign or malicious.

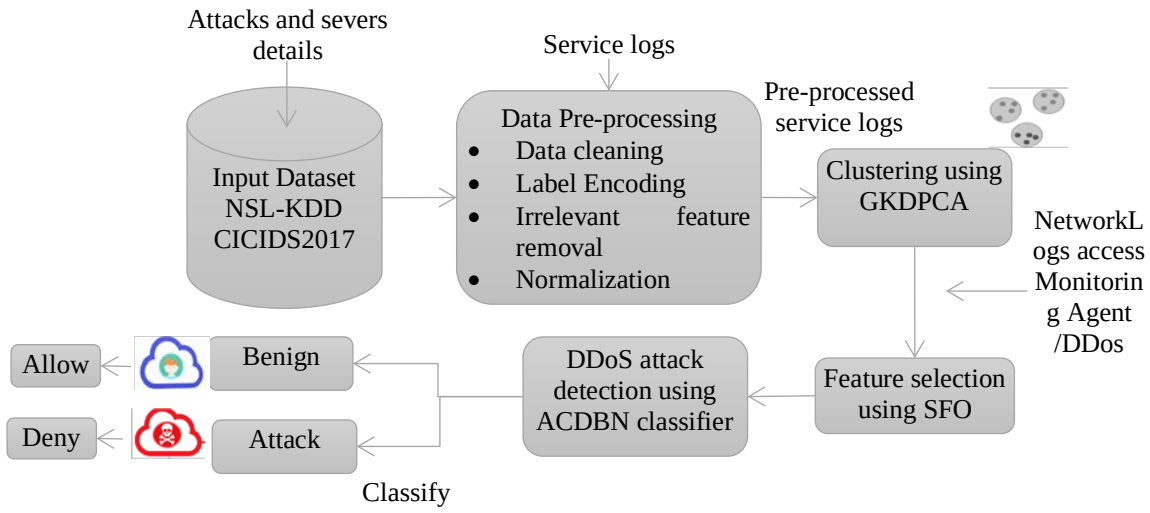

Fig.1. General framework diagram of ACDBNs based DDoSs detection

3.1. Data Pre-processing

Input Datasets: The University of New Brunswick's KDD'99 dataset has been updated, cleaned, and reworked into the NSL-KDD dataset [15]. In the NSL-KDD database, there are 43 varied features. More than 40 of these features are related to traffic, and the remaining two are the class designation (normal or attack) and the severity of the attack. DoS, probe, U2R, and R2L are just four of the 37 possible threats. The CICIDS dataset has been selected as the second dataset for our investigations. There are a total of 84 features in this labelled dataset [16]. As a final feature, a class label can be used to identify whether the traffic is malicious or non-harmful. The initial stage involves incrementing input data's qualities resulting in enhancing the efficiency of outcomes and mining procedure's performances. Data cleaning, Label encoding, feature elimination, and normalizing are the four preprocessing techniques that have been applied.

Data cleaning: One of the most important phases in data preparation is data cleansing. The process began with the deletion of features having the same values. The accuracy of the ACDBNs model is unaffected by features having identical values as they are managed by Impute Missing module's data cleaning part replaces all inf values with NA. DLTs do not consider inf values, making this step a necessity. At the completion of data cleaning, Impute Missing module finds best acceptable values for missing values in the dataset. Though eliminating such values is another way of handling missing data, it removes entire rows. Incomplete data raise processing costs and have large impacts and hence ACDBNs use lower features for detections in order to speed up attack detection processes. The developed missing value imputation module uses LRs (linear regressions) to predict the missing values. Furthermore, three pre-processing approaches have been used: label encoding, feature deletion, and normalisation [16].

3.2. GKDPCAs

The main goal of DPCAs is to discover cluster centres based on two conditions where the first is the assumption that cluster centre's neighbours have low densities. The second being cluster centres which are far from other cluster centres with higher local densities. DPCAs use local densities and distances to compute cluster centres as it is necessary to compute distances between two data points. The default Euclidean distances are used by original DPCAs to determine distances between data points. When datasets are complex and linearly inseparable, Euclidean distances result in major misclassifications. GKs (Gaussian kernels) were used to improve DPCAs by automatically transforming raw data into high-dimensional feature spaces by GKDPCAs.

The equivalent index set for the dataset $D = \{\overrightarrow{x_i}\}_{i=1}^{N}$, is $Ind_D = \{1,2,\dots,N\}$, and the distances between data points $\overrightarrow{x_i}$ and $\overrightarrow{x_j}$ was marked using $dist(\overrightarrow{x_i},\overrightarrow{x_j})$. These distances can be represented as kernel function distances where GKDPCAs use these measurements to compute distances between data points $\overrightarrow{x_i}$ and $\overrightarrow{x_j}$ as:

$$dist_{ij} = \sqrt{2\left(1 - GK\left(\overrightarrow{x_i},\overrightarrow{x_j}\right)\right)} \tag{1}$$

GKs can be formulated using Equation (2):

$$GK\left(\overrightarrow{x_i},\overrightarrow{x_j}\right) = \exp\left(-\frac{\left\|\overrightarrow{x_i}-\overrightarrow{x_j}\right\|}{2\epsilon^2}\right) \tag{2}$$
$$\epsilon > 0 \text{ (User defined value)}$$

In a dataset D, the density of localdata points ($\overrightarrow{x_i}$) which include GKs can be defined as ϑ_i.

$$\vartheta_i = \sum_{j \in Ind_D \setminus \{i\}} e^{-\left(\frac{dist_{ij}}{dist_c}\right)^2} \tag{3}$$

where $dist_c$ stands for cut-off distances that need to be defined in advance, and ϑ_i is a continuous variable in Equation (3). As a result, continuous variables have lesser chances of coming into conflicts with cut-off distances i.e.data points have same local densities. The distancesς_i between data pointsx_i and othershaving higher local densities can be defined as the shortest distances. Considering maximum distances between $\overrightarrow{x_i}$and other points with lower local densities for x_i with higher local densities and assuming$\{d_i\}_{i=1}^{N}$represents indices of x sets of local density sets set$\{\vartheta_i\}_{i=1}^{N}$ in descending orders, the following conditions are satisfied:

$$\vartheta_{d_1} \geq \vartheta_{d_2} \geq \cdots \geq \vartheta_{d_N} \tag{4}$$

The distance ς_i of $\overrightarrow{x_i}$ in Dcan be computed using:

$$\varsigma_{d_i} = \begin{cases} \min\limits_{j<i}\left\{dist_{d_{i,j}}\right\} & i \geq 2 \\ \max\limits_{j \geq 2}\left\{\varsigma_{d_j}\right\} & i = 1 \end{cases} \tag{5}$$

Cluster centers are points with higher values of ς_i and ϑ_i. For finding cluster centers, assuming $\vartheta_i$and $\varsigma_i$define values for χ_i, this combined value for each point $\overrightarrow{x_i}$can be calculated as follows:

$$\chi_i = \vartheta_i \varsigma_i,\ i \in Ind_D \tag{6}$$

Defining $\{c_i\}_{i=1}^{N}$ as descending ordersof indices for set of$\{\chi_{i,j}\}_{i=1}^{N}$as:

$$\chi_{c_1} \geq \chi_{c_2} \geq \cdots \geq \chi_{c_N} \tag{7}$$

Data points with higher values of χ_i have more chances of being cluster centers and assuming $\mathcal{K}$ stands for the count of clusters in D, selections of $\mathcal{K}$'s corresponding data points as maximum values of $\{\chi_{i,j}\}_{i=1}^{N}$become clustering centers. These selections can be denoted as:

$$CC = \{CC_i | CC_i = x_{c_i}\}, i = 1,2,\dots,\mathcal{K} \tag{8}$$

Where CC_i stands for ith cluster's center, $\mathcal{K}$ stands for counts of clusters. Once clusters and their centers are found, GKDPCAs assign remaining points to clusters as neighbours with higher densities. Subsequently, the dataset is

partitioned into $\mathcal{K}$ after getting labels of clusters and datapoints as $DCC_1, DCC_2, \dots, DCC_{\mathcal{K}}$.These$\mathcal{K}$subsets of D are trained using different ACDBNs where optimal feature selections are done by SOA. The algorithmic part of GKDPCAs is described as Algorithm 1.

Algorithm 1. The steps of GKDPCAs

Input: Training dataset$D = \{\overrightarrow{x_i}\}_{i=1}^{N}$, the number of clusters $\mathcal{K}$, parameters for GKs.
Output: the $\mathcal{K}$ subsets $DCC_1, DCC_2, \dots, DCC_{\mathcal{K}}$.

1: Calculate the distance $dist_{ij}$between data points $\overrightarrow{x_i}$ and $\overrightarrow{x_j}$according to Equation (1).
2: Assigning cut-off distances$dist_c$.
3: Computing local densities $\{\vartheta_i\}_{i=1}^{N}$ of data points$\overrightarrow{x_i}$based on Equation (3).
4: do step 1
5: Calculate$\chi_i = \vartheta_i \varsigma_i$, $i \in Ind_D$, selecting data points that correspond to maximum values of$\mathcal{K}$ $\{\chi_{i,j}\}_{i=1}^{N}$ as the cluster centersCC using Eq.(8)
6: Assigning remaining data points to relative clusters as center's nearest neighbours with higher densities, Partitioning training dataset Dinto$\mathcal{K}$ subsets $DCC_1, DCC_2, \dots, DCC_{\mathcal{K}}$.
7: **return** the $\mathcal{K}$ subsets $DCC_1, DCC_2, \dots, DCC_{\mathcal{K}}$ as the final result

3.3. SFOs

DBNs DBNs (Deep Belief Networks) address this issue by learning meaningful knowledge from a corpus of incoming data through their several hidden layers. Unfortunately, for large-scale applications, DBNs is a time-consuming and computationally expensive operation. ACDBNs with Feature Selection are developed in this paper. The complexity of the vocabulary input is reduced by using SFO-based feature selection since certain unimportant characteristics are filtered out, making the learning phase of ACDBNs more efficient.

SFOs: SFOs are a natural phenomenon first described by Gomes et al in [17]. They move like sunflowers which have two stages in their life cycle: sun rise facing east and sunset facing west. Sunflowers go in opposite directions at night to prepare tracking the sun next morning. Majority of SFO's approaches are based on radiation laws, which stipulates quantity of radiations with distances inversely proportional. More heat is gathered when distances from the sun and blooms are shortest. To move closer to global optimums in this stage, a few actions are necessary. When the distances between flowers and sun are greater, intensity of received radiations are low, necessitating bigger steps to reach global optimums. The quantity of heat H_i gathered by the plants (features) from solar radiations can be represented as:

$$H_i = \frac{RP}{4\pi d_i^2} \tag{9}$$

whereRPrepresents sun's radiation power and $d_i$represents distance between plant i's current position $X_i$and andX^*. Tracing sunfower$i$using the Sun's position can be achieved by:

$$\overrightarrow{Dir_i} = \frac{X^* - X_i}{\|X^* - X_i\|} \tag{10}$$

The number of steps$Steps$ of the sunflowersto the sun, on the direction $\overrightarrow{Dir_i}$ is given by the following equation:

$$Steps_i = z \times PP_i(\|X_i + X_{i-1}\|) \times \|X_i + X_{i-1}\| \tag{11}$$

where z represents a constant value characterizing displacements of plants while $PP_i(\|X_i + X_{i-1}\|)$ stands for probability of pollinations. PP between sunflowers i nearest neighbors$i - 1$result in new individuals with random positions. The maximum steps for individuals can be defined as:

$$Step_{max} = \frac{\|X_{max} - X_{min}\|}{2N_p} \tag{12}$$

where X_{min}stands for upper limits and X_{max}represents lower limits. $N_p$represents the size of the population. Plant positions are updated and for obtaining final optimal features and can be defined as follows:

$$\tilde{X}_i = \tilde{X}_i + Steps_i \cdot \overrightarrow{Dir_i} \tag{13}$$

Feature selections based on SFOsare detailed below.

Algorithm 2. The steps of the Feature selection based on SFOs

Initialization process such as population size as features of dataset

Generate randomly initial N_p plants
Find the best position from the initial population orientation towards the sun using Eq.(10)
While $Iteration \leq Max_{iteration}$do // $Max_{iteration}$ is the maximum number of iterations
Compute the step vector for every plant using Eq.(11)
Adjust the position of every plant using Eq.(13)
Remove mor(%) plants which further away from the sun //m is the mortality rate
Find the best position of every plant and return as optimal features
End while

3.4. DDoSs Attack Detection Using ACDBNs

This work first used CRBMs (Convolution restricted Boltzmann machines) as components of CDBNs (convolution deep belief networks) for detecting attacks from DDoSs. This work uses [18]'s formulations and applies it to one-dimensional situations. Assuming all inputs have ideal feature data then CRBMs are Convolution-based versions of "standard" RBMs where hidden and visible unit weights are shared across all hidden layer locations The CRBMs are divided into two layers: visible input layer V and hidden input layer H. Units that are visible have binary or real values, but the concealed units have binary values. Considering input layers, which consist of optimal features generated from SFOs, visible layer are first created, then the concealed layers. Considering n-dimensional filter weights $\mathcal{W}^k$ (bases), hidden layers are made up of K "groups" of n-dimensional arrays (where $nH, nV - nW + 1$), each have their own set of weights. There is also a shared biasb_k for each group and a shared bias c for visible units. Energy functions of CRBM scan then be defined as:

$$E(v,h) = -\sum_{k=1}^{K}\sum_{j=1}^{nH}\sum_{r=1}^{nW} h_j^k W_r^k v_{j+r-1} - \sum_{k=1}^{K} b_k \sum_{j=1}^{nH} h_j^k - c\sum_{i=1}^{nV} v_i \tag{14}$$

Similarly, the energy function of ACRBMs with Gaussian visible units for enhancing CRBM models. The energy functions of this modified model are defined in equation (15):

$$E(gv,h) = \frac{1}{2}\sum_{i,j=1}^{nV} gv_{i,j}^2 - \sum_{k=1}^{K}\sum_{i,j=1}^{nH}\sum_{r,s=1}^{nW} h_{i,j}^k W_{r,s}^k v_{i+r-1,j+s-1} - \sum_{k=1}^{K} b_k \sum_{i,j=1}^{nH} h_j^k - c\sum_{i,j=1}^{nV} gv_{i,j} \tag{15}$$

Where b_k indicates the value of each hidden layer unit bias and $c$is bias-term of visible units. Unlike the standard CRBM, the conditional probabilitiesCP of ACRBM are

$$\begin{aligned} CP_{new}\left(h_{i,j}^k = 1|gv\right) &= \sigma^2\left((W^k * gv)_{i,j} + b_k\right) \\ CP_{new}\left(gv_{i,j}^k = 1|h\right) &= \mu\left(\sum_{k,i,j}(W^k * h^k)_{i,j} + c, 1\right) \end{aligned} \tag{16}$$

Where μ is the mean and σ^2 is variance. The ACRBM model can be seen as a three-layer network containing a Gaussian visible layer GV, a binary hidden layer H and a pooling layer P. The pooling and hidden layers both have K groups ($\{H^k\}, \{CP\}^k, k \in \{1,2,\dots,K\}$). Each group size in the pooling layer is $N_p \times N_p$. The kth hidden group H^k is divided into several small blocks of size $C \times C$, and C is usually set to 1, 2 or 3. The training of the ACRBM aims to determine the optimal parameter set $\{W^k\}$. Similar to the standard CRBM, contrastive divergence (CD) learning algorithm based on Gibbs sampling is applied to update the connection weights

$$\begin{aligned} [W^k]_{(q+1)} &= [W^k]_{(q)} + \delta[\Delta W^k]_{(q+1)} \\ \Delta W^k &= \left\{\langle h_{i,j}^k v_{i+r-1,j+s-1}\rangle_{train} - \langle h_{i,j}^k v_{i+r-1,j+s-1}\rangle_{recon}\right\} \end{aligned} \tag{17}$$

whereδ is the learning rate, and $[W^k]_{(q)}$denotes the kth weight matrix at the qth iteration and q represents the current iteration number.$\langle . \rangle_{train}$ and $\langle . \rangle_{recon}$refers to the training data and reconstructed data expectations, respectively. As indicated in Eq. 1, only values from preceding iterations are used for weight updates in standard learning approaches, while past weights are disregarded as in Equation (17). In real-world applications, Gibbs sampling steps counts are severely limited, and are not constant, lowering consistencies and precisions of learned weights. A convDBN is a multilayer generative model. Each layer of the model can be trained in a greedy layer-wise fashion by treating each pair of adjacent layers as a convolutional restricted Boltzmann machine (convRBM). A convRBM is a probabilistic graphical model composed of layers of visible units (v) and hidden units (h)whose probabilistic relationships can be expressed in terms of convolutions. Traditional CRBMs learn with flaws like error oscillations and sluggish convergences. Smoothing of weights have been an effective strategy for overcoming these flaws and improving neural network's generalisations. As a result, ACDBNs based on EMAs (exponential moving averages) for weight smoothing are used in this work. For updating and smoothing current weights, ACDBNs consider all learned earlier round parameters. The formulae for updating data changes where W^k is:

$$[W^k]_{(q+1)} \leftarrow [W^k]_{(q)} + \delta[\Delta W^k]_{(q+1)}$$
$$[W^k]_{(q+1)} \leftarrow \frac{2\times[W^k]_{(q+1)}+q[W^k]_{(q)}}{q+2}, q \geq 1 \quad (18)$$

To avoid over completeness of learnt features, each hidden unit of the ACRBMs are regularised for sparsity, which can restrict information contents of feature maps and described as:

$$\sum_{i,j=1}^{nH}\left(\mathrm{P} - \frac{1}{T}\sum_{t=1}^{T}\sum_{k=1}^{K}\mathcal{CE}\left[h_{i,j}^{k}\right]v^{(t)}\right)^2 \quad (19)$$

where $\mathcal{CE}[\cdot]$ is the conditional expectation operator, $v^{(t)}$ is a sample in the training set with T total samples. γ is a regularization constant, and P is a constant controlling the sparseness of the hidden units $h_{i,j}^{k}$. Updated CRBMs have been developed and building of ACDBNs can be performed by consecutive training of individual ACRBM models, similar to Hinton's suggested training processes for ordinary DBNs (from lowest to highest) [19]. The following is a summary of the ACRBM learning algorithm, which is depicted in Fig.2.

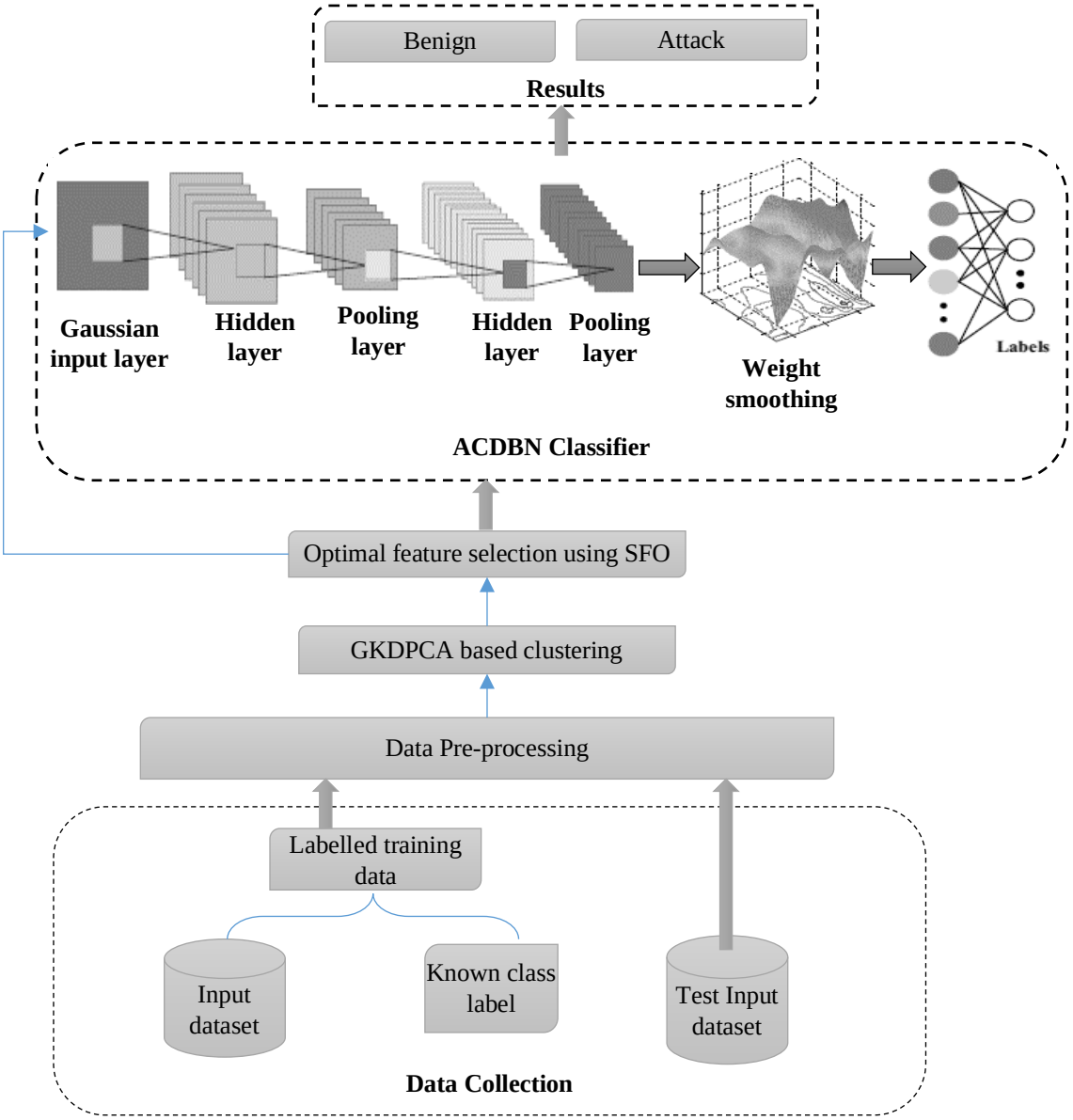

Fig.2.The framework of the proposed ACDBNs model for DDoSs detection

4. Experimental Results and Discussion

The proposed research work is implemented in NS2 Simulator. The NSL-KDD and CICIDS2017 network intrusion detection datasets were utilised in this research. The NSL-KDD dataset is vulnerable to DoSs, Probes, R2Ls,

and U2Rs, four types of assaults. In addition, each of these groups contains 37 additional attack kinds. As a result, the dataset includes Back, Land, Neptune, and Pod DDoS attacks; Smurf; Teardrop; Mailbomb; Process table; Udp storm; Apache2; and Worm DDoS attacks. After extraction, the KDD Train and KDD Test datasets have 148517 records and 43 feature columns. One-hot encoding is used to convert category data into numerical values when there are three columns. 43 feature columns are mapped to 124 at this level. The min-max scaler is used to normalise data between 0 and 1. Following that, a subset of the input training dataset is used to train the model. DNNs to avoid overfitting. To reduce generalisation error, the DNNs is trained on a random sample of 25000 data. ABOA generates data that is optimally encoded. DNNs employ these ideal data to categorise attack and non-attack traffic. There are two portions to the database: training and testing. A 75 percent training set with 40 input features is supplied to DNNs. Table 6 lists the settings used to run DNNs for NSL-KDD. There are 225720 data records in CICIDS2017, with 85 feature columns. They are omitted from the flow ID because they are not fed to AE. These functionalities have been identified as useful and will be made available to DNNs directly. At random, 15,000 records are chosen.

The min-max scaler is used to standardise the column values to (0, 1). This gives the AE 24994 records to train and 78 columns. The AE splits these 78 columns into 23. The ideal characteristics are provided to DNNs for categorization. The dataset is split into 75 percent training and 25 percent testing. DNNs is fed a 75% training set containing input features. The tests were done on a Windows 10 64-bit PC with 16 GB RAM and an Intel(R) Core-i7 CPU. VMware Workstation was used to create a cloud server with several virtual machines. The detection system's performance is defined by how well it can classify incoming communications. The proposed approach SFO+ACDBN was compared to current methods such as ABOA+DNN, AE+SVM, Naive AE+DNN and optimized AE+DNN. These measures are described below.

Recall: calculates the amount of useful class predictions based on positive examples in the database:

$$Recall = \frac{TP}{TP+FN}$$

Precision: calculates the percentage of positive class predictions that are truly positive:

$$Precision = \frac{TP}{TP+FP}$$

F-measure: calculates a single score that accounts both for precision and recall issues:

$$F - Measure = \frac{(2 * Precision * Recall)}{(Precision + Recall)}$$

Accuracy: It's a ratio of the total samples that were correctly classified to total number of samples:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$$

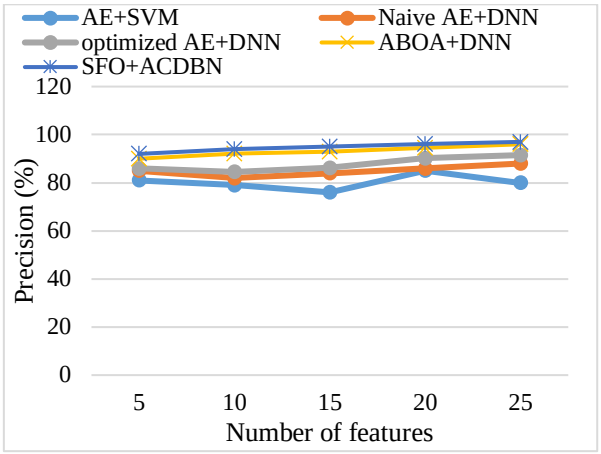

Fig.3. Precision performance comparison

4.1. Precision Result Comparison

The precision comparison results between suggested SFO+ACDBN and conventional ABOA+DNN, AE+SVM, Naive AE+DNN, and refined AE+DNN classifiers are shown in Fig.3. According to the graph, the proposed method has a high precision rate when compared to existing methods. It is an efficient method of detecting attacks with a high precision rate of 97%. When comparing the precision of existing approaches, ABOA+DNN, AE+SVM, Naive

AE+DNN, and Optimized AE+DNN provide good precision rates of 96%, 80%, 88.5%, and 91.5 %, respectively, which is lesser than the SFO+ACDBN. When dealing with feature data, deep learning algorithms have clearly greater accuracies than standard methods. The major reason for this is that ACDBNs approaches may adaptively learn useful information from input data via various feature modifications, improving accuracy.

4.2. F-measure Result Comparison

The F-measure comparison findings of suggested SFO+ACDBN, ABOA+DNN, AE+SVM, Naive AE+DNN, and refined AE+DNN classifiers are shown in Fig.4. According to the results, the proposed ABOA+DNN achieves a high F-measure rate of 92%. When comparing the F-measure rate between the existing approaches, ABOA+DNN, AE+SVM, Naive AE+DNN, and optimized AE+DNN provide lower rates of 91%, 73.15 %, 84.5 %, and 87 %, respectively, demonstrating that the suggested scheme can provide good attack identification outcomes than the previous techniques. The justification for this is that the SFO+ACDBN network has the input features from the SFO method that select the optimal features in order avoid local optimum also complexity of the vocabulary input of ACDBNs.

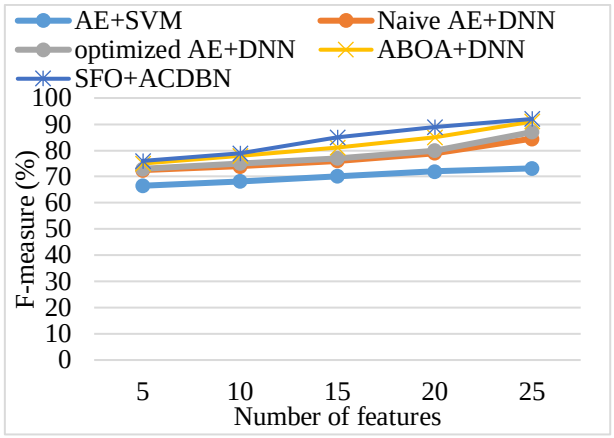

Fig.4. F-measure performance comparison

4.3. Recall Result Comparison

The recall comparison results for suggested SFO+ACDBN, ABOA+DNN, AE+SVM, Naive AE+DNN, and enhanced AE+DNN classifiers are shown in Fig.5. The proposed approach offers an extremely high recall rate of 96 %. According to the results, the suggested SFO+ACDBN has a high recall rate value, showing a good attack recognition accuracy. When comparing the recall rates of the existing approaches, ABOA+DNN, AE+SVM, Naive AE+DNN, and optimized AE+DNN provide recall rates of 96%, 84 %, 92 %, and 93 %, respectively, demonstrating that the suggested scheme can provide good attack recognition outcomes than the previous techniques. ACDBNs are built for enhanced feature learning of compressed datausing Gaussian visible units. Furthermore, EMA methods alter learning processes, allowing the generated deep learning model's generalization performances to be improved even further.

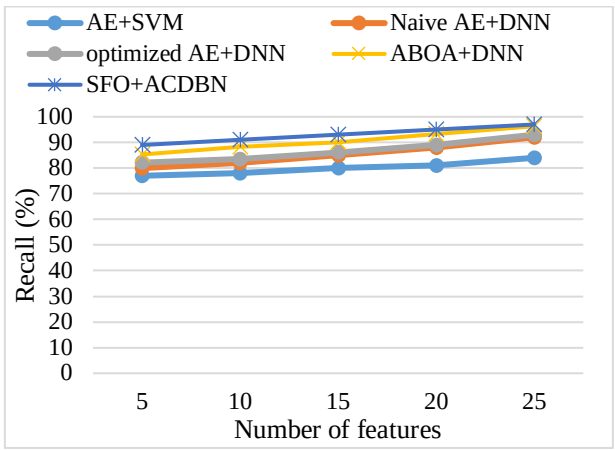

Fig.5. Recall performance comparison

4.4. Accuracy Result Comparison

The graph in Fig.6 below illustrates the accuracy comparison for attack detection. Methods such as SFO+ACDBN, AE+SVM, Naive AE+DNN, Optimized AE+DNN, and ABOA+DNN multiclass classifiers are used. ABOA+DNN is

an excellent method for obtaining accurate predictions, with a high accuracy rate of 99.05 %. When comparing the accuracy of previous techniques such as AE+SVM, Naive AE+DNN, and optimized AE+DNN, the rates are as follows: 80%, 95%, and 98 %, respectively. ABOA+DNN learning methods are relatively resistant to noise in training data, allowing for higher accuracy while eliminating the local optima issue. Furthermore, SFO has a faster convergence capability than other algorithms while avoiding premature convergence, and the effective feature samples provide excellent information categorization and intrusion detection accuracy. On the basis of traffic characteristics, large-scale network data is divided into several training subsets of diverse clustering centres. By addressing the imbalance of multiclass records and minimising the complexity of training subsets, GKDPCAs assist the model in achieving the best detection performances and reducing the complexity of training subsets.

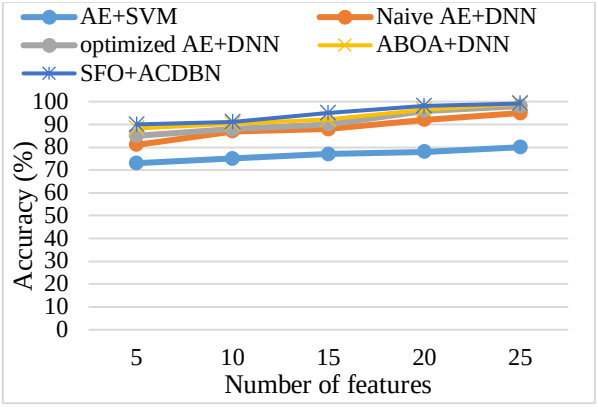

Fig.6. Accuracy performance comparison

5. Conclusions and Future Works

This work presents hybrid IDTs that combines GKDPCAs with ACDBNs in this paper. GKDPCAs are used to detect aspects in complicated and large-scale network data that are related. ACDBNs can identify high-level abstract features from training subseta automatically and reduce data dimensions to avoid the dimensionality curse and without the use of many heuristic rules or manual experiences. ACDBNs incorporate feature extractions and classification modules into systems that can automatically extract and classify features using a DLTs. with the best features picked using SFOs to avoid the problem of local optima. This is a good technique to boost your detecting performance. The findings show that the suggested technique is more successful than standard methods in eliminating the need for manual feature extraction. More deep learning algorithms are highly intriguing to develop. Because of the various hidden layers, computing cost is a challenge during implementation. As a result, the near future scope is to investigate the possibilities of inventing new optimization algorithms or other swarm algorithms to combine with DBNs in order to improve accuracy and detection rate while reducing computing cost. ABOA+DNN is an excellent method for obtaining accurate predictions, with a high accuracy rate of 99.05 %. When comparing the accuracy of previous techniques such as AE+SVM, Naive AE+DNN, and optimized AE+DNN, the rates are as follows: 80%, 95%, and 98 %, respectively. ABOA+DNN learning methods are relatively resistant to noise in training data, allowing for higher accuracy while eliminating the local optima issue. Furthermore, SFO has a faster convergence capability than other algorithms while avoiding premature convergence, and the effective feature samples provide excellent information categorization and intrusion detection accuracy. When compare to COFSs method, this proposed Convolutional Deep Belief Networks good perform effectively. Comparative trials of various optimizers, other networks (such as recurrent neural networks and generative adversarial networks), and attention-based methods will be done in future studies.

References

[1] Kumar, A., & Sharma, A. (2016). Paradigm shifts from e-governance to s-governance. The human element of big data: issues, analytics, and performance, 213.
[2] Shaar, F., &Efe, A. (2018). DDoS attacks and impacts on various cloud computing components. International Journal of Information Security Science, 7(1).
[3] Mell, P., &Grance, T. (2011). The NIST definition of cloud computing. NIST special publication, vol. 800, no. 145, p. 7, 2011.
[4] Alarqan, M. A., Zaaba, Z. F., &Almomani, A. (2019, July). Detection mechanisms of DDoS attack in cloud computing environment: A survey. In International Conference on Advances in Cyber Security (pp. 138-152). Springer, Singapore.
[5] Bhardwaj, A., Mangat, V., &Vig, R. (2020). Hyperband Tuned deep neural network with well posed stacked sparse AutoEncoder for detection of DDoS attacks in cloud. IEEE Access, 8, 181916-181929.
[6] Kasim, Ö. (2020). An efficient and robust deep learning based network anomaly detection against distributed denial of service attacks. Computer Networks, 180, 107390.
[7] Çakmakçı, S. D., Kemmerich, T., Ahmed, T., & Baykal, N. (2020). Online DDoS attack detection using Mahalanobis distance and Kernel-based learning algorithm. Journal of Network and Computer Applications, 168, 102756.

[8] Kushwah, G. S., &Ranga, V. (2021). Optimized extreme learning machine for detecting DDoS attacks in cloud computing. Computers & Security, 105, 102260.
[9] Kushwah, G. S., &Ranga, V. (2022). Detecting DDoS Attacks in Cloud Computing Using Extreme Learning Machine and Adaptive Differential Evolution. Wireless Personal Communications, 1-24.
[10] Verma, P., Tapaswi, S., & Godfrey, W. W. (2020). An adaptive threshold-based attribute selection to classify requests under DDoS attack in cloud-based systems. Arabian Journal for Science and Engineering, 45(4), 2813-2834.
[11] Amma, N. G., &Selvakumar, S. (2021). Optimization of vector convolutional deep neural network using binary real cumulative incarnation for detection of distributed denial of service attacks. Neural Computing and Applications, 1-14.
[12] Bhardwaj, A., Mangat, V., &Vig, R. (2020). Hyperband Tuned deep neural network with well posed stacked sparse AutoEncoder for detection of DDoS attacks in cloud. IEEE Access, 8, 181916-181929.
[13] Gomes GF, da Cunha SS, Ancelotti AC (2018) A sunfower optimization (SFO) algorithm applied to damage identifcation on laminated composite plates. EngComput 2018:1–8
[14] Lee, H., Grosse, R., Ranganath, R., & Ng, A. Y. (2011). Unsupervised learning of hierarchical representations with convolutional deep belief networks. Communications of the ACM, 54(10), 95-103.
[15] Hinton, Geoffrey E. "Deep Belief Nets." (2010): 267-269.
[16] Bhardwaj, A., Mangat, V., &Vig, R. (2020). Hyperband Tuned deep neural network with well posed stacked sparse AutoEncoder for detection of DDoS attacks in cloud. IEEE Access, 8, 181916-181929.
[17] Atika Bansal, Divya Kapil, Anupriya, Sagar Agarwal, Vishan Kumar Gupta, " Analysis and Detection of various DDoS attacks on Internet of Things Network", International Journal of Wireless and Microwave Technologies, Vol.12, No.3, pp. 18-32, 2022.
[18] T. Raja Sree, S. Mary Saira Bhanu," Investigation of Application Layer DDoS Attacks Using Clustering Techniques", International Journal of Wireless and Microwave Technologies, Vol.8, No.3, pp. 1-13, 2018.
[19] Kaushik Sekaran, G.Raja Vikram, B.V. Chowdary, "Design of Effective Security Architecture for Mobile Cloud Computing to Prevent DDoS Attacks ", International Journal of Wireless and Microwave Technologies, Vol.9, No.1, pp. 43-51, 2019.

Authors' Profiles

Mr. S. Sureshkumar is a Research Scholar in Department of CSE at Karpagam Academy of Higher Education and Research, Coimbatore. Tamilnadu. He has completed his Undergraduate and Post graduate studies in Computer Science and Engineering in Anna University, Chennai. He has 14 years of academic experience in teaching. He has published 8 books and over 20 publications in highly cited Journals and Conferences. He is a professional body member of ISTE and IAENG.

Dr. G. K. D. Prasanna Venkatesan, Professor in the Department of Electronics and Communication Engineering, Karpagam Academy of Higher Education, Coimbatore, Tamil Nadu, India.He has completed his Ph.D in "Beyond 4G Networks" from Anna University, Chennai and obtained his M.Tech from SASTRA University, Thanjavur in 2002 and B.E (Electronics and Communication Engineering) from Madurai Kamaraj University, Madurai in 2000. His expertise in 5G Technologies, Machine Learning, Data analytics, Cloud Computing, IoT, Wireless sensor networks. He worked as a Specialist, in design and development at Tata Elxsi, Bangalore. He published more than 120 research articles in the international journals and Conferences. So far, he guided 12 doctorates. He serves as editorial board for various referred journals. He has chaired many IEEE conferences in India and abroad.

Dr. R. Santhosh received his B.Tech degree in Information Technology from K.S.R College of Technology in 2006, M.E degree in Software Engineering from Sri Ramakrishna Engineering College in 2009, M.B.A in Education Management from Alagappa University in 2011 and Ph.D in Computer Science and Engineering from Karpagam Academy of Higher Education in 2016. He is currently working as a Professor and Head in the Department of Computer Science and Engineering, Faculty of Engineering at Karpagam Academy of Higher Education. He has published more than 84 Research Articles in International Journals indexed in Scopus and Web of Science. He has published 2 International Patents. His current research interests include Cloud Computing, Distributed & Parallel Computing, Artificial Intelligence and Data Science.

How to cite this paper: S. Sureshkumar, G. K. D. Prasanna Venkatesan, R. Santhosh, "Detection of DDOS Attacks on Cloud Computing Environment Using Altered Convolutional Deep Belief Networks", International Journal of Computer Network and Information Security(IJCNIS), Vol.15, No.5, pp.63-72, 2023. DOI:10.5815/ijcnis.2023.05.06