Attention-Enhanced CNN–RNN Framework with Adaptive Coyote Optimization–Based Feature Selection for High-Dimensional Data Mining and Anomaly Detection

Priyanga.P[1], Dr.K. Saraswathi[2]
[1]*Ph.D. Research Scholar, PG & Research Department of Computer Science, Government Arts College - Coimbatore-18.*
[2]*Associate Professor, PG & Research Department of Computer Science, Government Arts College - Coimbatore-18.*

***Abstract*— Data mining (DM) is fundamental for extracting meaningful knowledge from large-scale, high-dimensional datasets, were redundant attributes and noisy features often hinder classification and anomaly detection. In order to produce a smaller but more informative feature set, the suggested ACOA technique uses a process of eliminating features that are superfluous or redundant. This method produces small, extremely informative subsets. The spatial and sequential dependencies in data can be effectively captured by the application of an attention based- hybrid CNN–RNN (A-CNN-RNN). For anomaly detection study, and DM analysis, a widely accessible dataset named KDD Cup 1999 dataset is utilized to assess suggested method. When compared this suggested model with some standard deep learning (DL) models (CNN-only, RNN-only, CNN–RNN) and traditional machine learning (ML) techniques (Random Forest (RF), XGBoost), this suggested model executes well than other methods, and it was demonstrated by the outcomes of simulation. High scores in prediction accuracy (ACC), precision (P), recall (R), F1-score, and AUC-ROC were attained using ACOA. Additionally, feature selection (FS) was used to minimise the input feature space by more than 50%. By effectively finding pertinent features, metaheuristic-guided feature selection (FS) improves classification and anomaly detection in high-dimensional data. It facilitates Strong pattern recognition and extracting valuable insights over a wide range of DM backgrounds**

***Index Terms*— Data Mining; Anomaly Detection; High-Dimensional Data; CNN–RNN; Attention Mechanism; Adaptive Coyote Optimization (ACOA); Feature Selection; KDD Cup 1999**

I. INTRODUCTION

Massive, complex datasets (big data) are produced by the growth of digital ecosystems in healthcare, finance, e-commerce, and industry, opening doors for efficiency and innovation. Data mining plays a vital role in this context, aiming to uncover patterns, classify instances, and detect anomalies that may represent fraud, faults, or rare events. Among these tasks, anomaly detection is especially critical, as identifying unusual patterns in massive datasets can provide early warnings and support decision-making. An essential component of DM is anomaly detection. Data outliers, aberrant activity, and system failure are only a few examples of anomalous appearance. Both rarity and type diversity are key problems in anomaly identification because of the large range of anomaly kinds (point, collective, contextual, etc.) and the infrequency of anomalies, which makes it hard to gather enough labelled samples for training models. In a high-dimensional space, anomalous characteristics are more hidden, while they are more likely to appear in a low-dimensional region. While the majority of anomaly detection techniques directly or implicitly rely on distance contrast [3], increasing dimensionality causes the distance contrast across data to become similar [1] [2]. For these distance contrast-based anomaly detection techniques, high dimensionality eventually results in failure. Additionally, it is difficult to distinguish between normal and abnormal cases due to the data's sparse distribution in a high-dimensional

space. [4]. Anomaly detection for the data in a high-dimensional space is difficult in this situation.

DL-based methods, distance metric-based techniques, and deep hybrid-based methods are important categories of anomaly detection. K-nearest neighbour (K-NN) [5] and random distances [6] are examples of distance metric-based techniques that do not obtain data distribution. Such methods are susceptible to the adverse impacts of high dimensionality, as distance measurements grow increasingly comparable as data dimensionality increases. Deep features of the data [7] [8] can be learnt and understood by DL-based methods, like the Bayesian Variational Autoencoder (BVAE) [9] and those in [10] [11]. Both supervised and unsupervised detection techniques are included in this category. The unsupervised detection techniques, such as Deep One-class Classification (DOC) [13], Generative Adversarial Network (GAN) [14], etc., do not depend on data labels, they are very susceptible to noise and missing data [12]. Anomaly detection accuracy may typically be lower than that of supervised detection methods since the objective function of unsupervised detection methods is primarily used for data compression or dimensionality reduction. Because supervised detection techniques, like those in [15], rely on data labels, they perform better in detection than unsupervised techniques.

If there is a lot of data or if the data is multi-dimensional, labelling it can be difficult. As a result, it is challenging to modify supervised anomaly detection techniques for large-scale and high-dimensional data anomaly detection. Deep hybrid-based techniques combine deep and conventional detection techniques, as seen in Deep (NN) Neural Networks-(SVM) Support Vector Machine (DNN-SVM) [16], Deep Autoencoder, and Ensemble KNN (DAE-KNN) [17]. As a result, they adopt the traits of both conventional and deep detection techniques. They have inherent advantages in anomaly detection; however, precision of calculation and processing cost must be traded off. However, anomaly detection in high-dimensional data remains difficult due to redundancy, irrelevant attributes, and nonlinear dependencies that degrade classifier performance and increase computational burden.

DL approaches have shown notable potential for mining such complex data. Convolutional NN (CNNs) are effective at learning spatial and hierarchical features, while Recurrent NN (RNNs) capture sequential and contextual dependencies. Their combination provides rich feature representations well-suited for anomaly detection. Moreover, attention mechanisms further enhance these models by dynamically prioritizing the most relevant information, thereby improving interpretability and robustness. Despite these advantages, applying DL directly to high-dimensional data is inefficient, as redundant attributes may introduce noise, overfitting, and scalability issues.

FS is crucial method in anomaly detection and classification methods, and this implementation may support in overcoming those limitations. The most relevant attributes are retained, and dimensionality was reduced by the FS. Thus, improved efficiency, better generalization, and faster model convergence may result from this. But non-linear and large-scale data was not effectively managed by conventional FS methods, more adaptable and effective method was offered by the metaheuristic optimization algorithms.

In this work, the Adaptive Coyote Optimization Algorithm (ACOA) is employed exclusively for FS. The social dynamics of coyote packs are the motivation of the COA. The exploration and exploitation are effectively adaptively balanced by the ACOA. Thus, optimal feature subsets are identified, and premature convergence are avoided by this algorithm. In order to produce a smaller but more informative feature set, the suggested ACOA technique uses a process of eliminating features that are superfluous or redundant. This method produces small, extremely informative subsets. This allows a small but highly discriminative feature set to be used by the DL classifier. For anomaly detection study, and DM analysis, a widely accessible dataset named KDD Cup 1999 dataset is utilized to assess the suggested method. For anomaly detection study, and DM analysis, a widely accessible dataset named KDD Cup 1999 dataset is utilized to assess suggested method.

II. RELATED WORKS

Anomaly detection and classification in high-dimensional data mining have been extensively studied using both classical ML and advanced deep learning approaches. Traditional models such as RF, SVM, and Naïve Bayes (NB) have been employed for detecting anomalies, but their performance often degrades with large-scale, nonlinear, and redundant

data features. To overcome these challenges, recent research has increasingly integrated feature selection (FS) with deep learning (DL) and metaheuristic optimization.

In order to solve data imbalance in anomaly detection, integration of an improved RF with the Synthetic Minority Over-sampling Technique (SMOTE) was suggested by Wu et al. (2022). Their method achieved high training accuracy but showed limited generalization on testing data, highlighting the importance of FS and robust model design [18]. Similarly, Kasongo and Sun (2020) applied FS using XGBoost on the UNSW-NB15 dataset, demonstrating that selecting optimal attributes improves accuracy by 2.72% compared to raw high-dimensional features [19].

On the DL front, for anomaly detection, Halbouni et al. (2022) presented a hybrid CNN–LSTM (Long Short-Term Memory) system. Both temporal and spatial dependencies in sequential data were captured by the suggested method. While effective, the model faced scalability issues when applied to high-dimensional inputs without FS [20]. For dimensionality reduction, Principal Component Analysis (PCA) and Mutual Information were used with LSTM by Laghrissi et al. (2021), reporting accuracy above 99% on the KDD99 dataset, but PCA's linearity limited its adaptability to nonlinear data [21].

Metaheuristic algorithms have also been integrated with deep models to improve FS. Malibari et al. (2022) proposed an enhanced Arithmetic Optimization Algorithm with Deep Wavelet Neural Networks (DWNN), validated on CIC-IDS2017. State-of-the-art (SOTA) anomaly detection methods are outperformed by the suggested technique [22].

Murali Mohan et al. (2022) employed a hybrid metaheuristic (SMSLO) with Deep Belief Networks (DBNs) for cloud anomaly detection, showing notable improvements in detection rates [23]. However, both works emphasized IDS-specific applications and did not generalize to broader data mining contexts.

Generative models have also been explored. In order to increase classification robustness, Using Generative Adversarial Networks (GANs), Ding et al. (2022) introduced a hybrid approach for imbalanced anomaly detection tasks that blends KNN with samples produced by GAN. Their work validated the potential of synthetic augmentation but noted computational overheads as a limitation [24].

Despite these advancements, most existing approaches either rely on traditional FS methods or apply DL models directly to raw high-dimensional data, which leads to redundancy, overfitting, and inefficiency. Few works have investigated metaheuristic-guided FS exclusively integrated with CNN–RNN–Attention architectures for anomaly detection. This gap motivates the present study, where the Adaptive Coyote Optimization Algorithm (ACOA) is employed to refine feature subsets before classification, and a hybrid A-CNN-RNN with attention is used to capture spatial, sequential, and contextual patterns effectively.

III. PROPOSED METHODOLOGY

To detect anomalies in high-dimensional datasets, this paper suggests an optimised deep DM framework that combines hybrid DL classification with metaheuristic-based FS. The methodology consists of four primary phases: data preprocessing, feature selection using Adaptive Coyote Optimization Algorithm (ACOA), classification with CNN–RNN–Attention (A-CNN-RNN), and performance evaluation. Figure 1 illustrates the overall framework.

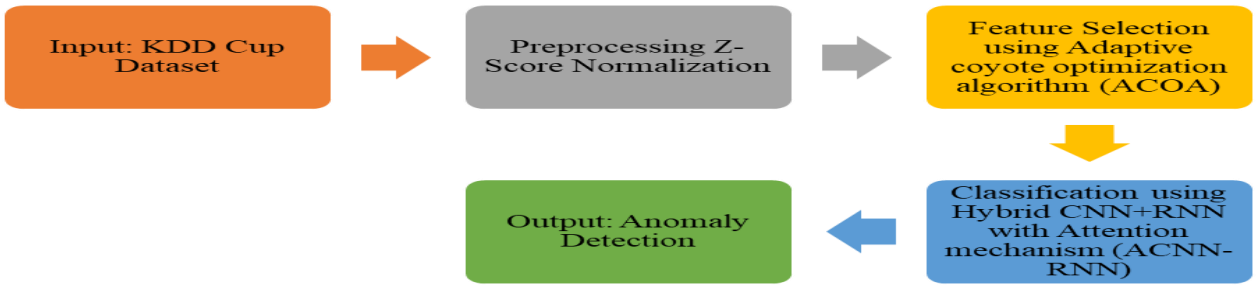

Figure 1: Overview diagram of suggested DL model for ID

3.1.1. ZSN for Preprocessing the KDD Cup data

The KDD dataset (KDD Cup 1999) is a famous data mining dataset used mainly for network intrusion detection research. It was prepared for the Knowledge Discovery and Data Mining (KDD) Cup competition in 1999 and is one of the earliest large-scale datasets in this field. With 41 attributes such protocol type, service, duration, and error counts, each record represents a network connection. DoS (Denial of Service), Probe, Remote to Local (R2L), and User to Root (U2R) are the four categories into which the data is classified, either as regular traffic or as one of numerous attack kinds.

Although it became a benchmark in DM and ML, the dataset has shortcomings like duplicates, imbalance, and outdated attack patterns, which is why improved versions like NSL-KDD and newer datasets (e.g., UNSW-NB15, CIC-IDS2017) are now widely used. The KDD Cup 1999 dataset is selected as the benchmark, containing 41 features and one class label with normal and anomalous instances (DoS, U2R, R2L, Probe). Preprocessing includes the ZSN was used on the dataset to guarantee that features from all scales contribute uniformly during model training. ZSN scales to unit variance and eliminates the mean to standardise the features. It is especially helpful for algorithms like distance-based or gradient-based models that are sensitive to feature magnitude.

For each feature x, the normalized value z was computed using the formula:

$z = \frac{x-\mu}{\sigma}$ (1)

- Here, the standard deviation is denoted by σ
- The mean of the feature across the training set is denoted as μ.

This ensures uniform contribution of all features during training.

- Categorical Encoding: Symbolic attributes (e.g., protocol type, service, flag) are encoded using one-hot encoding.

3.1.2. FS using ACOA

In particular, when working with HD datasets, FS is crucial for enhancing the classification efficiency of ML frameworks. By removing unnecessary or redundant features, a proper FS strategy improves model generalisation while significantly lowering computing costs.

Here, the Adaptive Coyote Optimization Algorithm (ACOA) was utilized for optimal feature subset selection. ACOA, an enhanced version of the Coyote Optimization Algorithm (COA), dynamically adjusts its search behaviour during optimization to effectively balance energy, leading to a more efficient search for minimal, highly-informative feature subsets.

A. Overview of COA

The social relationships within Canis latrans packs have been used as inspiration for the population-based and nature-inspired algorithm (NIA) known as COA, and it was suggested by Piezan et al. (2019). Determining the initial population of coyotes, that implies on $N_p \in N^*$ packs with $N_c \in N^*$ coyotes each, is the first stage in the process. The division of the initial packs is chosen at random. "Social condition ($\mathcal{SC}$)" refers to the decision variables (DV), and the c^{th} coyote of the p^{th} pack in the j^{th} dimension has the following values:

$\mathcal{SC}_{c,j}^{p,t} = lb_j + random_j \cdot (ub_j - lb_j)$ (2)

Here, the limits of the j^{th} DV are represented by lb_j and ub_j. A random number that are uniformly distributed is denoted as $random_j \in [0, 1]$. Calculating the objective function (OF) values for every set of DV is the second stage

$fit_c^{p,t} = f(\mathcal{SC}_c^{p,t})$ (3)

A coyote's transition between packs is controlled by a particular probability (Pr) and is impacted by N_c.

$Pr_e = 0.005.N_c^2$ (4)

Additionally, the coyote with best OF cost value in the p^{th}pack of the t^{th} timestamp is referred as alpha coyote, and it is considered by the COA. It is described below:

$alpha^{p,t} = \{\mathcal{SC}_c^{p,t} | argc = \{1,2,..,N_c\} minf(\mathcal{SC}_c^{p,t})\}$ (5)

Each pack has an estimate of the cultural trend $\mathcal{CT}$, and it is also taken into consideration by the COA. This is how the calculation is done:

$$\mathcal{CT}_j^{p,t} = \begin{cases} O_{\frac{(N_c+1)}{2},j}^{p,t}, & N_c \text{ is odd} \\ \frac{o_{\frac{N_c}{2},j}^{p,t} + o_{\left(\frac{N_c}{2}+1\right)}^{p,t}}{2}, & \text{otherwise} \end{cases} \quad (6)$$

In the range $[1, D]$, the ranking $\mathcal{SC}$ of all coyotes of the p^{th} pack in the t^{th} instant of time for each j is denoted as $O^{p,t}$. The dimension of the search space is D.

Based on the values of the OF and the coyotes' ages (which are calculated in years and are defined as

$\text{age}_c^{p,t} \in N$, the algorithm synchronises the coyotes' births and deaths. In Algorithm 1, this method is explained. Here, the group of coyotes with the worst OF score is denoted by ω. The total coyotes count in this group is denoted by φ. The OF values of the young coyote (pup) and every coyote in the pack are compared to determine the group of ω coyotes.

The of two randomly selected parents is combined with an environmental element to create the pups. Regardless of their social condition, parents are chosen. Consequently, the pups are described as follows:

$$\text{pup}_j^{p,t} = \begin{cases} \text{soc}_{dc_1,j}^{p,t}, & \text{random}_j < P_g \text{or} j = j_1 \\ \text{soc}_{dc_2,j}^{p,t}, & \text{random}_j \geq P_d + P_g \text{or } j = j_2 \\ R_j, & \text{otherwise} \end{cases} \quad (7)$$

Here, the two designated coyotes from the p^{th} pack are dc_1 and dc_2. The optimisation problem has two random dimensions, j_1 and j_2. The scatter probability is P_s. The association probability is P_a. An arbitrary value that can only be found within the j^{th} dimension's DV bounds is denoted as R_j. A uniformly generated random number ∈ [0,1] is called rnd_j.

P_s and P_a determine the coyote diversity.

The following formula is used to determine these values:

$P_s = 1/D$ and (8)

$P_a = (1 - P_d)/2$ (9)

Here, the influence impact on both parents is balanced by P_a.

The alpha influence (δ_a) and whole pack influence (δ_t) on the coyotes are expressed as follows:

$\delta_t = ct^{p,t} - \text{soc}_{cr_1}^{p,t}$ and (10)

$\delta_a = \text{alpha}^{p,t} - \text{soc}_{cr_2}^{p,t}$ (11)

Here, the cultural difference from a random coyote is indicated by cr_1. The cultural difference between the alpha coyote and a random coyote in the pack is shown by cr_2.

The weights of $\delta_t$and δ_a are denoted by r_t and r_a. Uniformly distributed random values $\in [0, 1]$ are represented by r_t and r_a.

$$\text{new}_{\text{soc}_c^{p,t}} = \text{soc}_c^{p,t} + r_t.\delta_t + r_a.\delta_a \quad (12)$$

The following is an expression of the new social condition:

$$\text{new\_fit}_c^{p,t} = f(\text{new\_soc}_c^{p,t}) \quad (13)$$

Reserving the best social condition

$$\text{soc}_c^{p,t+1} = \begin{cases} \text{new\_soc}_c^{p,t}, & \text{new\_fit}_c^{p,t} < \text{fit}_c^{p,t} \\ \text{soc}_{c,}^{p,t} & \text{otherwise} \end{cases} \quad (14)$$

Out of all the packs, the algorithm's final solution is the best solution.

Adaptive Mechanism in COA (ACOA Enhancement):

While the traditional COA uses static update parameters, the ACOA introduces a dynamic learning adjustment mechanism. In ACOA, the influence coefficients r_t and r_a are no longer purely random but are adaptively scaled based on an adaptive learning rate γ, which evolves according to:

$\gamma = \gamma_0 \times \exp(-\beta \times \text{Iteration})$ (15)

where:

- The initial learning rate is denoted as γ_0,
- β is the decay constant controlling the rate of exploitation increase,
- Iteration is the current optimization step.

Thus, the influence weights in the social condition update are modified as:

$r_t = \gamma_t \times \text{random}$ and $r_a = \gamma_a \times \text{random}$

Initially, the algorithm favors exploration with larger adaptive steps, encouraging broad search. As iterations progress, the steps become smaller, shifting toward precise exploitation of promising regions. Furthermore, if a new social update results in fitness improvement, the learning rate is temporarily boosted to encourage further exploration; otherwise, it is reduced to favor local search convergence.

This adaptive adjustment mechanism allows ACOA to:

- Dynamically balance exploration and exploitation,
- Improve convergence speed,
- Enhance robustness against getting stuck in local optima,
- Select more informative and compact feature subsets for classification tasks.

The time complexity of the COA and ACOA remains similar to traditional evolutionary algorithms, expressed as:

$O(N_p \times N_c \times D + N_p \times N_c \times f)$ (16)

Here, D is the search space dimension and f is the cost of the OF evaluation. Consequently, ACOA offers a very efficient method for choosing a small but extremely informative subset of features during the FS phase for the both datasets. ACOA lowers computing expenses and increasing classification accuracy. This ensures compact feature subsets with maximum

predictive power. ACOA successfully reduces dimensionality by more than 50%, retaining only the most relevant attributes, thus lowering computational cost and boosting classification efficiency

Algorithm: ACOA Feature Selection on KDD Cup 1999

Inputs (KDD-specific)

- Dataset: KDD Cup 1999 (full or 10% subset).
- Task: choose Binary (Normal vs Anomalous) or Multi-class (Normal + categories).
- Split ratios: e.g., Train 70% / Val 15% / Test 15% (no overlap).
- Preprocessing choices:
  - Encoding for 3 categorical features: protocol_type, service, flag (use one-hot).
  - Scaling for numeric features (z-score on training stats).
  - Imbalance strategy (class weights or balanced sampling).
- ACOA params: number of packs, coyotes per pack, max iterations, optional migration probability.
- Scoring model: fast version of your A-CNN-RNN (few epochs) or a light surrogate (e.g., small MLP) used only to score subsets inside ACOA.

Outputs

- Selected feature subset (names/indices) for KDD (post-encoding groups or original 41 fields; see note below).
- Reduced train/val/test matrices built from the selected subset.
- Final metrics after retraining the full A-CNN-RNN on the reduced features (ACC, P, R, F1, AUC (Area under the Curve) - ROC (Receiver Operating Characteristic) can explain.

KDD Cup 1999 Preprocessing (before ACOA)

1. Load data (full or 10% file).

3.1.3. Classification using Hybrid CNN+RNN with Attention mechanism (A-CNN-RNN) After ACOA-based feature selection (FS), the reduced feature set is classified with a hybrid CNN–RNN model augmented by an attention mechanism. The design exploits the complementary strengths of each component:

- CNN captures local cross-feature interactions and hierarchical patterns.
- RNN (BiLSTM) models long-range dependencies across feature groups.
- Attention highlights the most informative blocks for each record, improving accuracy and interpretability.

The model outputs either binary (normal vs. anomalous) or multi-class predictions (normal + anomaly categories), depending on the experimental setting.

A. MaxPooling based CNN

In the initial step of the ACNN-RNN model, the selected features are processed via several convolutional layers (CL). The input feature vector is reshaped, if necessary, into a two-dimensional format compatible with convolutional (Conv) operations. In order to capture local patterns and hierarchical feature representations, the CL applies a number of learnable filters throughout the input. A common method for adding non-linearity to a Conv operation is to use a non-linear (NL) activation function (AF), such as the Rectified Linear Unit (ReLU).

To further reduce dimensionality while maintaining the most important information, the feature maps (FM) are downsampled using pooling layers like MaxPooling.

The Conv operation is CNNs' primary function, where a small learnable kernel (or filter) slides over the input FM to generate a set of FM. The formal calculation of the convolutional output at a given spatial position (i, j) for the kth filter, given an input feature matrix X and a filter W, is as follows:

$$f_{i,j}^{(k)} = \sigma\left(\sum_m \sum_n W_{m,n}^{(k)} \times X_{i+m,j+n} + b^{(k)}\right) \quad (17)$$

Here:

- At location (i, j) in the k^{th} FM, the output feature value is $f_{i,j}^{(k)}$.
- In the k^{th} filter, $W_{m,n}^{(k)}$ is the weight of kernel at position (m, n).
- The input value at the relocated position is $X_{i+m,j+n}$.
- The bias connected to the k^{th} filter is $b^{(k)}$.
- The AF that adds non-linearity to the network is $\sigma(\cdot)$. It is also known as the ReLU, and it is described as $\sigma(x) = \max(0, x)$.

Without requiring manual feature engineering, the CL allows the network to autonomously learn spatially invariant features including edges, patterns, and higher-order feature combinations.

Following the convolution operation, it is common practice to apply a pooling operation to further process

the feature maps. By summarising feature responses in local neighbourhoods, pooling introduces a degree of translation invariance and lowers computational cost by reducing the spatial dimensions of the FM.

The most commonly used pooling strategy is MaxPooling. In MaxPooling, a local neighborhood $R(i,j)$ around the position (i,j) is considered. In this region, the max value is chosen. The MaxPooling operation is mathematically defined as:

$$P_{i,j} = \max_{(m,n) \in R(i,j)} f_{m,n} \quad (18)$$

Here,

- $P_{i,j}$ is the pooled output at position (i,j),
- $R(i,j)$ denotes the receptive field (i.e., local window) centered at (i,j),
- $f_{m,n}$ are the convolution output values within the neighborhood.

While keeping the most important features, pooling drastically reduces the FM's size, which lowers the number of parameters and computational load. Additionally, it helps to control overfitting by preventing the network from memorizing specific details of the training data.

The resulting high-level FM are flattened into one-dimensional sequences following several phases of convolution, activation, and pooling. These sequences encapsulate the most significant spatial patterns present in the input feature set, forming a rich representation that can be passed into sequential modeling layers such as RNNs or LSTMs for further processing.

Thus, the Convolutional Neural Network (CNN) feature extraction (FE) phase transforms the original input features into compact, informative, and spatially-aware representations that facilitate more effective downstream classification.

Sequential Learning with RNN and Attention

For RNN processing, the resultant FM are flattened and transformed into sequential data after the spatial features are extracted. First, flattening a sequence and then feeding it into the RNN layer to capture temporal correlations and dependencies. The variant like LSTM is selected in this study, because it has the potential in managing long-term dependencies and mitigate vanishing gradient problems (VGP). An internal hidden state is used by the RNN, and it detects the feature sequence's historical context, when it processes the sequential input step-by-step.

By serving as an advanced focussing tool, the attention mechanism enables the model to improve the sequential data that the RNN processes. The Attention mechanism assigns dynamic importance weights to different time steps or hidden states produced by the RNN. Instead of analysing each segment of the sequence equally, the model can concentrate more on features or temporal patterns that are more important for precise classification according to the attention mechanism. Each RNN hidden state is projected into a common space, its relevance is determined, and attention weights are formed by normalisation using a softmax function. This process yields the attention scores. A weighted sum of all hidden states makes up the context vector. Then, it is computed to highlight the most informative parts of the sequence.

Classification Layer and (LF) Loss Function

After an attention mechanism processes an input and produces a context vector, fully connected (or dense) layers are applied to transform these features into higher-level abstractions. Here, the probability distribution over the class labels are generated by a softmax output layer. This layer is capable of distinguishing normal and attack types. The final output of the framework represents the class label with highest probability.

Using an appropriate optimisation technique, like Adam, the entire ACNN-RNN model is trained end-to-end by minimising a cross-entropy (CE) loss (CEL) function between the predicted labels and the ground truth labels. Techniques like batch normalisation (BN) and dropout regularisation are used during training to stabilise the process and avoid overfitting.

By combining local feature extraction (CNN), temporal modeling (RNN), and dynamic focusing (Attention), the suggested ACNN-RNN framework achieves superior classification performance compared to conventional classifiers and standalone DL models. It not only improves overall classification accuracy but also provides better generalization to unseen network traffic patterns, making it highly effective for practical ID scenarios.

The input to the ACNN is the optimized feature set $X \in \mathbb{R}^{n \times d}$, here, d is the count of chosen features, and n is the count of student instances. The initial layers of the ACNN consist of one or more Convolutional Layers (CLs), which apply a set of learnable filters $W_k \in \mathbb{R}^{f \times f}$ to extract local feature patterns.

The Conv operation for a specified input window x is expressed as follows:

$y_{i,j}^{(k)} = \sum_{m=0}^{f-1} \sum_{n=0}^{f-1} x_{i+m,j+n} . \omega_{m,n}^{(k)} + b^{(k)}$ (19)

Here, $y_{i,j}^{(k)}$ is the activation at location (i, j) for filter k, and $b^{(k)}$ is the corresponding bias term. Non-linearity is then introduced by passing the resultant FM through an NL AF, usually ReLU:

$\mathrm{ReLU}(x) = \max(0, x)$ (20)

To reduce spatial dimensionality and computational cost, pooling layers such as max-pooling are optionally applied. Once the hierarchical representations are built, the output is forwarded into the attention mechanism, which is the core innovation in ACNN.

To represent each feature vector's significance in the final decision, the attention mechanism dynamically assigns weights to each feature vector. For each input vector x_i, a score e_i is computed using a learned function (e.g., dot-product or feed-forward scoring):

$e_i = \mathrm{score}(x_i)$ (21)

The attention weights α_i are then derived using the softmax function:

$\alpha_i = \frac{\exp(e_i)}{\sum_{j=1}^{n} \exp(e_i)}$ (22)

The model's confidence in the relative importance of each feature or location is represented by these weights. The attended output representation is the weighted sum of the feature vectors.

$\hat{x} = \sum_{i=1}^{n} \propto_i . x_i$ (23)

This mechanism allows the framework to "emphasis" on the most informative attributes (e.g., number of forum views or assignment scores) while down-weighting irrelevant ones. After an output layer, the attended feature vector $\hat{x}$ is processed by one or more dense layers.

For binary classification (BC) of student performance (e.g., Pass/Fail), the final output is computed using the sigmoid activation function:

$\hat{y} = \frac{1}{1+e^{-z}}$ (24)

Where z is the linear combination of weights and biases from the final dense layer. In situations with many classes (such as Low, Medium, and High performance), the softmax function is employed instead:

$\hat{y}_i = \frac{e^{z_i}}{\sum_{j=1}^{c} e^{z_j}}, \quad i \in \{1, \ldots C\}$(25)

For binary output, the model is trained to minimise the binary cross-entropy loss (CEL):

$\mathcal{L}_{\mathrm{binary}} = -[y . \log(\hat{y}) + (1 - y) . \log(1 - \hat{y})]$ (26)

Or the categorical CEL for multiclass output:

$\mathcal{L}_{\mathrm{categorical}} = -\sum_{i=1}^{C} y_i . \log(\hat{y}_i)$27)

Using gradient descent (GD) techniques like Adam or Stochastic GD (SGD), model parameters (filters, attention weights, and dense layer weights) are updated during backpropagation. Performance metrics like ACC, P, R, F-measure, and AUC-ROC are employed for evaluating the framework, guaranteeing that the classification is reliable and performs well when applied to new data.

Pseudocode: Suggested Procedure for ID using ACOA + A-CNN-RNN

Step 1: Data Preprocessing

Input:

- Data: KDD Cup 1999, split into Train / Val / Test
- Features: Reduced feature blocks selected by ACOA (e.g., Basic, Content, Time, Host, Protocol, Service, Flag)
- Labels: Binary (normal vs anomalous) or multi-class (normal + attack categories)
- Hyperparameters: batch size, epochs, learning rate, dropout, class weights (if imbalanced)
- Architecture settings: embedding size, #Conv layers (kernel 3–5), BiLSTM hidden size/layers, attention type (additive/dot), head size

Output:

- Trained model weights (A-CNN-RNN)
- Test metrics: ACC, P, R, F1-score, AUC-ROC (micro/macro for multi-class)
- Per-record predictions: class probabilities + predicted label
- Attention weights: block-level importance (per record) for interpretability

1.1 Load dataset

1.2 For each feature x in the dataset:

Compute mean (mu) and standard deviation (sigma) on training set

Normalize: z = (x - mu) / sigma

If sigma == 0: add small constant epsilon to avoid separation by 0

Step 2: Feature Selection using ACOA

Input: Normalized Data

Output: Selected Feature Subset

2.1 Initialize ACOA parameters:

Number of packs (Np), number of coyote (Nc), max iterations
Initialize social conditions (SC) for each coyote randomly within bounds

2.2 For each iteration:
For each pack p:
Evaluate fitness for each coyote: fit_c = f(SC_c)
Determine alpha^(p): best coyote in pack
Compute cultural trend CT^(p)
Generate pup using social mix of two coyotes + environment factor
Replace worst coyote if pup is better (elitism)
For each coyote c in pack p:
Compute influence from trend (delta_t) and alpha (delta_a)
Update SC: SC_new = SC + r_t * delta_t + r_a * delta_a
If new fitness better: accept SC_new
Adapt learning rate gamma:
gamma = gamma_0 * exp (-beta * iteration)
r_t = gamma_t * random, r_a = gamma_a * random
2.3 Return best SC as optimal feature subset

Step 3: Classification using A-CNN-RNN
Input: Feature-Selected Data
Output: Class Labels

3.1 Reshape selected features as 2D input for CNN
3.2 CNN Layer:
Apply Conv2D -> ReLU -> MaxPooling (multiple times)
3.3 Flatten CNN output to 1D sequence
3.4 RNN Layer:
Use LSTM to capture temporal patterns from sequence
3.5 Attention Layer:
For each timestep:
Compute attention score e_i = score(h_i)
Compute weight alpha_i = softmax(e_i)
Context vector: sum(alpha_i * h_i)
3.6 Fully Connected Layers:
Apply dense layer(s) to context vector
Use Softmax/Sigmoid for final classification

Step 4: Training and Evaluation
4.1 Define LF:
Binary: Binary CE
Multiclass: Categorical CE
4.2 Train model using Adam optimizer
4.3 Evaluate on test set using:
ACC, P, R, F1-score, AUC-ROC

IV. EXPERIMENTAL RESULTS

Experiments were conducted on KDD Cup 1999 with duplicate removal, stratified Train/Val/Test split, one-hot encoding for protocol_type, service, flag, and z-score scaling on numeric features in NS2 (Network Simulator 2). ACOA was used only for feature selection, operating on feature groups (38 numeric + 3 categorical groups). The classifier is A-CNN-RNN with Attention (BiLSTM), trained with Adam and early stopping. Metrics reported on the held-out test set: ACC, P, R, F1-score, AUC-ROC.
Below are the standard performance metric formulas (ACC, P, R, and F-measure) along with their typical interpretations in the context of IDS. In this context:

- True Positive (TP): The framework accurately identifies an intrusion as an intrusion.
- True Negative (TN): The framework precisely identifies normal traffic (non-intrusion) as normal.
- False Positive (FP): The system in accurately identifies non-intrusion as intrusion.
- False Negative (FN): The intrusion is not detected by the system. (i.e., it is an intrusion but classified as normal).

P: P shows the proportion of those that were real attacks. When evaluating false alarms, P is essential. If a traffic is classified as attack by the system, it is probably accurate because a high P indicates fewer false alarms.

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}} \quad (28)$$

F-measure: The F-measure balances FP and FN by taking the harmonic mean of P and R.

$$\text{F} - \text{measure} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision+Recall}} \quad (29)$$

R: It is defined as the number of real attacks that the system detects. R is essential to ensuring that no attack is missed. Less missed attacks (FN) are indicated by a high R.

$$\text{Recall} = \frac{\text{TP}}{\text{TP+FN}} \quad (30)$$

ACC: ACC assess the ratio of total instances (both normal and attack) that are accurately identified by the IDS.

$$\text{Accuracy} = \frac{\text{TP+TN}}{\text{TP+TN+FP+FN}} \quad (31)$$

The Table 1 shows the A-CNN-RNN + ACOA-FS attains an optimal efficiency over all metrics, demonstrating both stronger separability (AUC-ROC 99.3) and a better precision–recall balance (F1 98.85). Relative to the strongest deep baseline without attention/FS, it gains +0.62 Accuracy (99.12 vs 98.5), +0.50 Recall, +0.55 F1, and +0.5 AUC-ROC—showing that ACOA-based feature selection removes redundancy and that attention helps focus on the most informative blocks. Compared to XGBoost, the proposed model improves Accuracy by +1.72 and AUC-ROC by +1.3, highlighting the advantage of jointly modeling local patterns (CNN), long-range dependencies (RNN), and per-record importance (attention) in high-dimensional data mining.

Table 1: Performance Comparison of the various approaches using several metrics on KDD Data

Method	Metrics (%)				
	Accuracy	Precision	Recall	F1-score	AUC-ROC
Random Forest	96.3	96.2	96.0	96.1	97.0
XGBoost	97.4	97.3	97.1	97.2	98.0
CNN-only	97.9	97.8	97.6	97.7	98.4
RNN-only	97.5	97.4	97.2	97.3	98.2
CNN–RNN (no Attention, no FS)	98.5	98.4	98.3	98.3	98.8
A-CNN-RNN + ACOA-FS (Proposed)	99.12	98.9	98.8	98.85	99.3

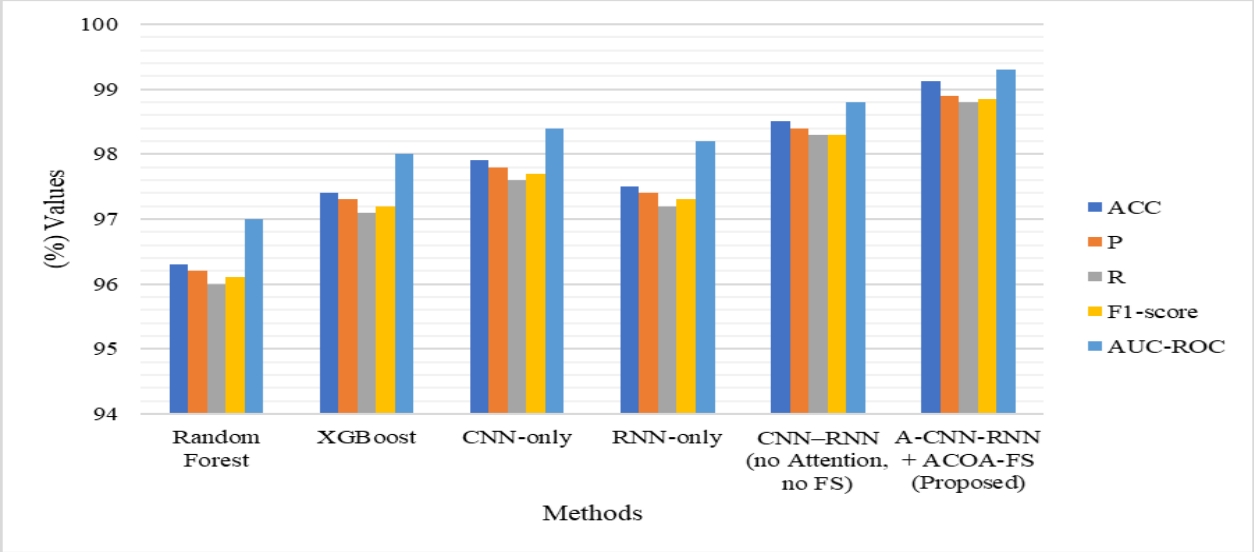

Figure 2: Comparison graph using suggested and current methods

Figure 2 presents the result analysis of the suggested and current methods using various metrices. Accuracy. The proposed A-CNN-RNN + ACOA-FS tops the chart (99.12%), a +0.62-p gain over the best deep baseline without attention/FS (98.5%). This indicates overall correctness improved after removing redundant features and adding attention.

Precision. At 98.9%, the proposed method generates fewer false alarms than all baselines. That's important in anomaly detection, where unnecessary alerts can overwhelm analysts.

Recall. With 98.8%, the model misses fewer true anomalies than competitors. The attention layer helps the network focus on informative blocks (e.g., time/host statistics, service, flag), lifting sensitivity.

F1-score. The best F1 = 98.85% reflects a balanced trade-off among P and R. Compared to the no-attention/no-FS hybrid, F1 rises by ~0.55 pp, showing that both ACOA-FS and attention contribute to better balance.

AUC-ROC. The highest AUC = 99.3% demonstrates strong separability across thresholds, i.e., the proposed

model ranks anomalous vs. normal connections more reliably than others.

V. CONCLUSION

In this work presented a data-mining–centric framework for high-dimensional anomaly detection that pairs Adaptive Coyote Optimization (ACOA) for feature selection only with an Attention-augmented CNN–RNN (A-CNN-RNN) classifier. Using KDD Cup 1999, we treated features as semantically grouped blocks (basic, content, time-based, host-based, and categorical groups), embedded them, learned local and long-range interactions via CNN and BiLSTM, and used attention to emphasize the most informative blocks per record. With ~50% feature reduction from ACOA, the suggested framework attained 99.12% ACC, 98.9% P, 98.8% Recall, 98.85% F1, and 99.3% AUC-ROC, outperforming classical baselines (Random Forest, XGBoost) and deep baselines (CNN-only, RNN-only, CNN–RNN without attention/FS). Beyond higher scores, we observed faster, more stable convergence and improved sensitivity to minority patterns, while attention weights provided transparent, per-record explanations. The current study applies the model to anomaly detection. Future work should extend its applicability to other core DM tasks, like clustering, association rule mining, regression, and recommendation systems. Exploring how metaheuristic-driven FS benefits these tasks would broaden its impact.

REFERENCES

[1] Arias, L. A. S., Oosterlee, C. W., & Cirillo, P. (2023). AIDA: Analytic Isolation and Distance-based Anomaly Detection Algorithm. Pattern Recognition, 141, Article 109607.

[2] Zuo, Z., Li, Z., Cheng, P., & Zhao, J. (2023). A Novel Subspace Outlier Detection Method by Entropy-based Clustering Algorithm. Scientific Reports, 13(1), 15331.

[3] Zhang, X., et al. (2023). A Hybrid Anomaly Detection Method for High Dimensional Data. PeerJ Computer Science, 9, e1199.

[4] Soleimani H, Miller DJ. 2016. Atd: anomalous topic discovery in high dimensional discrete data. IEEE Transactions on Knowledge and Data Engineering 28(9):2267–2280

[5] Chehreghani MH. 2016. K-nearest neighbor search and outlier detection via minimax distances. In: SIAM international conference on data mining. 405–413.

[6] Wang H, Pang G, Shen C, Ma C. 2020b. Unsupervised representation learning by predicting random distances. In: Twenty-ninth international joint conference on artificial intelligence main track. 2950–2956.

[7] Yuan Y, Xun G, Ma F, Wang Y, Du N, Jia K, Su L, Zhang A. 2018. Muvan: a multiview attention network for multivariate temporal data. In: 2018 IEEE International Conference on Data Mining (ICDM). Piscataway: IEEE, 717–726.

[8] Roy, D. (2025). Bayesian Autoencoder for Medical Anomaly Detection. arXiv. Retrieved from https://arxiv.org/abs/2504.15562

[9] Roshan, K., & Zafar, A. (2021). Utilizing XAI Technique to Improve Autoencoder-Based Model for Computer Network Anomaly Detection with Shapley Additive Explanation (SHAP). arXiv. Retrieved from https://arxiv.org/abs/2112.08442.

[10] Grathwohl W, Wang K-C, Jacobsen J-H, Duvenaud D, Norouzi M, Swersky K. 2019. Your classifier is secretly an energy-based model and you should treat it like one. In: International conference on learning representations. 1–23.

[11] Grosnit A, Maraval AM, Tutunov R, Griffiths R-R, Cowen-Rivers AI, Yang L, Zhu L, Lyu W, Chen Z, Wang J, Peters J, Bou-Ammar H. 2022. High-dimensional Bayesian Optimisation with variational autoencoders and deep metric learning.

[12] Parchami M, Bashbaghi S, Granger E, Sayed S. 2017. Using deep autoencoders to learn robust domain-invariant representations for still-to-video face recognition. In: Advanced Video and Signal Based Surveillance (AVSS), 14th IEEE International Conference on 2017. Piscataway: IEEE, 1–6.

[13] Ruff L, Görnitz N, Deecke L, Ahmed Siddiqui S, Binder A, Müller E, Kloft M. 2018. Deep one-class classification. 4390–4399

[14] Li D, Chen D, Shi L, Jin B, Goh J, Ng S-K. 2019. MAD-GAN: multivariate anomaly detection for time series data with generative

adversarial networks. In: International Conference on Artificial Neural Networks (ICANN) 2019: artificial neural networks and machine learning—ICANN 2019: text and time series. 703–716.

[15] Metzen JH, Genewein T, Fischer V, Bischoff B. 2017. On detecting adversarial perturbations. ArXiv preprint. arXiv:1702.04267.

[16] Inoue J, Yamagata Y, Chen Y, Poskitt CM, Sun J. 2017. Anomaly detection for a water treatment system using unsupervised machine learning. In: Data Mining Workshops (ICDMW). Piscataway: IEEE, 1058–1065.

[17] Song H, Jiang Z, Men A, Yang B. 2017. A hybrid semi-supervised anomaly detection model for high-dimensional data. Computational Intelligence and Neuroscience 2017:8501683.

[18] Wu, T., Zhang, J., Wang, Y., & Li, H. (2022). Intrusion detection system combined enhanced Random Forest with SMOTE algorithm. EURASIP Journal on Advances in Signal Processing, 39, 1–12.

[19] Kasongo, S. M., & Sun, Y. (2020). Performance analysis of intrusion detection systems using a feature selection method on the UNSW-NB15 dataset. Journal of Big Data, 7(1), 105.

[20] Halbouni, A., Gunawan, T. S., Habaebi, M. H., & Kartiwi, M. (2022). CNN–LSTM: Hybrid deep neural network for anomaly detection. IEEE Access, 10, 99837–99849.

[21] Laghrissi, F., Douzi, S., & Hssina, B. (2021). Anomaly detection systems using long short-term memory (LSTM) with PCA-based feature reduction. Journal of Big Data, 8(1), 65.

[22] Malibari, A., Alharkan, I., Alshamrani, O., & Almotairi, S. (2022). A novel metaheuristics with deep learning enabled anomaly detection system for secured environments. Sustainable Energy Technologies and Assessments, 52, 102312.

[23] Murali Mohan, V., Balajee, R. M., Hiren, K. M., & Rajakumar, B. R. (2022). Hybrid machine learning approach for anomaly detection in cloud environments: A metaheuristic-assisted model. Multiagent and Grid Systems, 18(1), 21–43.

[24] Ding, H., Chen, L., Dong, L., Fu, Z., & Cui, X. (2022). Imbalanced data classification: A KNN and GAN-based hybrid approach for anomaly detection. Future Generation Computer Systems, 131, 240–254.