Detection of computer attacks based on sonification of network traffic*

Bohdan Semeniuk[1,†], Antonina Kashtalian[1,*,†], Dmytro Martiniuk [1,*,†], Andriy Drozd[1,†] and Abdel-Badeeh M. Salem[2,†]

[1] *Khmelnitsky National University, Khmelnitsky, Instytutska street 11, 29016, Ukraine*
[2] *Ain Shams University, El-Khalyfa El-Mamoun Street Abbasya, Cairo, Egypt*

Abstract

Proposed is a model for detecting computer attacks based on the sonification of network traffic. This work examines a "sonification" approach for network traffic to detect attacks (Intrusion Detection System – IDS). The proposed model converts feature vectors into a pseudo-audio signal (PCM), which is then processed via the Short-Time Fourier Transform (STFT) to obtain a two-dimensional (time × frequency) representation. Based on the resulting spectrograms, a 2D-CNN is built to perform binary classification into "normal vs. attack." It is shown that sonification enables the use of advanced audio-analytical methods and yields results comparable to traditional 1D-based approaches. The paper also discusses the automated "enhancement" of individual attributes in the audio domain to improve detection metrics. Experimental studies confirm that combining PCM+STFT with a two-dimensional convolutional neural network (2D-CNN) can detect attack signatures with accuracy levels close to those of traditional IDS solutions, while also offering new possibilities for visualizing and analyzing network traffic. The presented results illustrate the feasibility of further research on sonification in information security applications.

Keywords

intrusion detection system, convolutional neural network, pulse-code modulation, short-time Fourier transform

1. Introduction

In today's environment of increasingly heavy network traffic and the constant emergence of new threats, the timely detection of attacks (Intrusion Detection System, IDS [1]) becomes especially relevant. Traditional security measures such as antivirus software, firewalls, and classic IDS [21, 22, 23, 24, 25] remain important elements of cybersecurity, yet they often suffer from a significant number of false alarms and an insufficient capability to detect unknown or modified attacks. This creates a need for more flexible and accurate methods capable of promptly identifying a wide range of threats.

The development of machine learning methods and deep neural networks [2,3,4,5] has opened new opportunities for building IDS capable of automatically extracting relevant features from large volumes of network traffic. Such deep-learning-based solutions substantially reduce dependence on expert security knowledge and demonstrate high detection accuracy even under conditions of rapid emergence of new attack types. However, despite successes in this field, challenges remain regarding the representation of input data, model regularization, and the processing of extensive information in real time.

Intelitsis'25: The 6th International Workshop on Intelligent Information Technologies & Systems of Information Security, April 04, 2025, Khmelnytskyi, Ukraine
* Corresponding author.
† These authors contributed equally.

faludore@gmail.com (B. Semenyuk); yantonina@ukr.net (A. Kashtalian); martiniyuk.dim14@gmail.com (D. Martiniuk); andriydrozdit@gmail.com (A. Drozd); abmsalem@yahoo.com (Abdel-Badeeh M. Salem);
0009-0001-8831-8835 (B. Semeniuk); 0000-0002-4925-9713 (A. Kashtalian); 0009-0002-3524-872X (D. Martiniuk); 0009-0008-1049-1911 (A. Drozd); 0000-0003-0268-6539 (Abdel-Badeeh M. Salem);

One promising direction for improving IDS is the "sonification" of network traffic, in which the original feature vectors are converted into a quasi-audio signal (PCM), subsequently analyzed by standard audio technologies such as the Short-Time Fourier Transform (STFT). This "audio-domain" representation enables the use of well-established approaches from speech and audio recognition—particularly 2D-convolutional networks (2D-CNN)—which effectively operate on two-dimensional spectrograms (time × frequency). The subsequent sections examine how exactly "sonification" of network traffic offers additional opportunities for data visualization and analysis, as well as how it impacts the final accuracy of attack detection.

Hence, the goal of this study is to justify and experimentally investigate the "PCM + STFT + 2D-CNN" approach for IDS. We compare the results with conventional methods that directly operate on feature vectors (1D-CNN or MLP), analyze the strengths and weaknesses of the "audio-based" representation, and discuss prospects for its improvement (in particular, enhancement of certain "frequency" attributes and the automation of weighting coefficients).

Thus, this work simultaneously continues the tradition of applying deep learning to attack detection and proposes an original approach at the intersection of audio analytics and IDS technologies.

2. Analysis of known solutions and research on machine-learning-based IDS

By its nature, machine learning is a data-driven approach that underscores the importance of in-depth understanding of the underlying data as a first step [26, 27]. In this context, the data source type becomes the central link for classification. This section explores various strategies for integrating machine learning into the development of intrusion detection systems (IDS) for different data types. Since the characteristics of these data types reflect specific attack patterns—ranging from host-based activity recorded in system logs to network-level activity captured in traffic—the choice of the right data source is critically important. For instance, a Denial of Service (DOS) attack, characterized by a massive number of packets sent within a short time, is best detected using flow-based data, while covert channels relating to data leakage between specific IP addresses require session-level analysis.

2.1. Packet-based attack detection

Packets, as the fundamental units of network communication, provide detailed information about each interaction. They consist of binary data requiring parsing to be understandable. Typically, a packet includes a header and application-level payload. The header holds structured fields like IP addresses, port numbers, and protocol-specific details, while the payload contains the actual data of application-level protocols. Employing packets as a data source for IDS offers three main advantages:

- The payload may reveal U2R (User-to-Root) or R2L (Remote-to-Local) attacks.
- IP addresses and timestamps allow precise identification of the attack's origin.
- Packets can be processed in real time without the need for caching.

However, since a single packet does not reflect the complete context or state of the communication session, certain attacks (e.g., DDoS) may not be as effectively detected. Packet-based detection methods typically break down into packet-parsing and payload-based analysis approaches.

2.1.1. Parsing-based packet detection

In parsing-based detection, diverse network protocols (e.g., HTTP, DNS) are examined with a focus on their header fields. A common practice is using tools like Wireshark or Bro to extract these fields, after which the most relevant header values are transformed into feature vectors for classification. For example, Mayhew et al. proposed an SVM-based method combined with K-means clustering for packet detection. Their approach involved capturing packets in a corporate network, parsing them via Bro, then grouping packets by protocol type. They subsequently used K-means++ to cluster similar packets, and the extracted features were used to train SVM models, achieving 99.6% accuracy for HTTP, 92.9% for TCP, 99% for Wiki, 96% for Twitter, and 93% for email. Additionally, unsupervised learning is often employed to reduce false alarms. For instance, Hu et al. applied fuzzy C-means clustering to Snort-processed packets from the DARPA 2000 dataset. Repeated clustering reduced false positives by 16.58% and missed detections by 19.23%.

2.1.2. Payload-based detection

Alternatively, payload-based detection methods focus on application-layer data contained in packets rather than their headers. This approach is less protocol-specific and effective for diverse protocols, provided that the payload is not encrypted. Whereas shallow models require manual feature engineering (which can be labor-intensive and pose privacy concerns), deep learning methods are capable of automatically learning features from raw payload data. For example, Min et al. [6] implemented a text-based CNN for attack detection on the ISCX 2012 dataset, merging payloads from different packets, encoding them using a skip-gram word-embedding model, then combining statistical header features with CNN-extracted content features, and classifying with random forest at 99.13% accuracy. Furthermore, combining multiple deep models can enhance feature extraction. Zeng et al. [7] employed CNN, LSTM, and stacked autoencoders simultaneously to extract local, sequential, and textual features, achieving 99.22% overall accuracy. Unsupervised approaches, such as convolutional autoencoders, have also proven effective, with Yu et al. [8] reporting ~98.4% in terms of accuracy, recall, and F-measure on the CTU-UNB dataset.

2.2. Flow-based attack detection

Flow-based data, aggregating packets over a given time interval, constitutes one of the most common data sources for IDS. Flow data offers two main advantages: first, a broad view of the network environment—especially useful for detecting DOS and Probe attacks—and second, simpler preprocessing steps compared to packet-level or session-level reconstruction. However, flow data generally lacks packet-level details, limiting its ability to detect U2R and R2L attacks, and the need to buffer packets to generate flow data may introduce delays. Flow-based detection methods typically fall under feature-engineering-based or deep-learning-based approaches; due to heterogeneity in flow data, traffic is often partitioned into more homogeneous subsets to improve detection accuracy.

2.2.1. Feature-engineering-based detection

Traditional machine learning algorithms cannot directly handle raw flow data, hence a pivotal preprocessing step is to convert flow data into interpretable feature vectors. Among common features are average packet length, variance in packet length, TCP/UDP ratio, and the proportion of specific TCP flags. Although such methods can achieve high detection accuracy, they often come with a significant rate of false positives. Some researchers therefore combine multiple weak classifiers. For instance, Goeschel et al. introduced a hybrid solution combining SVM, decision trees, and Naïve Bayes. Their approach first segregated data into normal vs. anomalous with an SVM, employed a decision tree to identify known attack types, and applied Naïve Bayes for unknown attacks, reaching 99.62% accuracy with a 1.57% false positive rate on KDD99. Other studies improved efficiency, e.g., Kuttranont et al. developed an optimized KNN with GPU-based parallel processing, achieving 99.30% on KDD99 with ~30× speedup over CPU-only computations. Unsupervised methods like improved K-means (enhanced initialization and mini-batch processing, as in Peng et al. [9]) also yield gains in both accuracy and computational efficiency.

2.2.2. Deep-learning-based detection

Deep learning method enable direct operation on raw flow data, automatically learning features "end-to-end," reducing reliance on manual feature extraction. Potluri et al. proposed a CNN approach where feature vectors were converted into images by one-hot encoding nominal features (expanding the dimensionality from 41 to 464) and splitting them into 8-byte chunks forming 8×8 pixel images. Their three-layer CNN surpassed deeper architectures (ResNet50, GoogLeNet), scoring 91.14% on NSL-KDD and 94.9% on UNSW-NB15. A two-stage approach—first extracting features via an unsupervised model (e.g., a sparse autoencoder), then classifying with XGBoost—also proved effective (Zhang et al.) on NSL-KDD. To address small or imbalanced datasets, adaptive learning (e.g., GAN-based data augmentation) has been adopted, significantly improving detection performance across various attack classes.

2.2.3. Traffic grouping for detection

Because flow data may be highly heterogeneous, training ML models on these data can lead to overfitting. One effective strategy is grouping (clustering) traffic to reduce variability. Teng et al. [10] split KDD99 into TCP, UDP, and ICMP subsets, choosing distinct features for each protocol before training separate SVM models, yielding 89.02% average accuracy. Alternatively, grouping by data characteristics via clustering is also effective, e.g., Ma et al. [11] used spectral clustering to split the dataset into six homogeneous clusters, after which separate deep neural networks were trained per cluster, reaching ~92.1% accuracy on KDD99 and NSL-KDD.

2.3. Session-based attack detection

Sessions, reflecting communication between two endpoints, offer high-level semantics and are generally defined by a five-tuple (client IP, client port, server IP, server port, protocol). Session-based detection has two main advantages: first, it is well-suited to identifying attacks targeting specific IP pairs (e.g., covert or Trojan attacks); second, the detailed communication within a session aids in pinpointing the attack's origin. However, because session duration can vary widely, analysis often requires caching large numbers of packets, potentially introducing latency. Session-based detection methods often focus on statistical vs. sequential features.

2.3.1. Statistical feature-based detection

Statistical session-based detection extracts features such as header fields, packet counts, or traffic direction ratios to form a feature vector suitable for classic classifiers. While high-level session semantics lend themselves well to rule-based or decision-tree methods, they may overlook the sequential nature of network sessions. Supervised methods that rely on statistical features can be fairly accurate but commonly suffer from higher computational costs. For instance, Ahmim et al. presented a hierarchical decision tree approach that uses independent sub-classifiers (decision tree, rule-based model) on partial feature sets, then a random forest to analyze the combined set of features, demonstrating competitive results for 8 out of 15 classes on the CICIDS 2017 dataset.

2.3.2. Sequential feature-based detection

Unlike flow data, sessions inherently maintain the packet order, facilitating the extraction of sequential features like packet length sequences and inter-arrival times. However, most conventional ML algorithms lack built-in mechanisms for sequential data, so RNN-based solutions are less frequent but can be effective. For example, Yuan et al. utilized LSTM for DDoS detection on the UNB ISCX 2012 dataset, embedding 20-dimensional packet features via a "bag-of-words" approach, then assembling them into variable-sized matrices. A CNN extracted local patterns, while LSTM handled the temporal dimension, achieving ~97.6% in accuracy, precision, recall, and F-measure. Similar CRNN structures have also emerged, employing hierarchical layers (CNN + RNN) to handle both spatial (packet-level) and temporal features.

2.4. Log-based attack detection

Logs—generated by operating systems or applications—provide semantically rich data vital for detecting attacks like SQL injection, U2R, or R2L. Although logs supply valuable context (user actions, timestamps) and aid in tracing attackers, their analysis can be highly dependent on specialized knowledge, and differences in log format hamper scalability. Log-based detection methods generally split into hybrid rule-based/ML, feature-extraction-based, and text-analysis-based approaches.

2.4.1. Hybrid rule-based + machine learning methods

Hybrid detection leverages the strengths of rule-based systems (e.g., Snort) which often produce large numbers of alerts, most of them benign, and feeds those alerts into ML models for filtering. Meng et al. [15], for instance, applied a KNN approach to prioritize alerts in a real network, cutting their volume by 89%. Similarly, McElwee et al. [16] developed a deep neural network to filter McAfee-generated alerts; the most critical alerts were then examined by security experts, thus reducing analyst workload while boosting accuracy.

2.4.2. Feature extraction from logs

This approach involves using domain knowledge to extract relevant features from logs for standard ML classifiers. A common technique is sliding-window analysis to capture contextual event sequences with low detection latency. For example, Tran et al. [17] employed CNN on system-call logs in NGIDS-DS and ADFA-LD datasets, applying a sliding window to detect local patterns indicative of intrusions. Tuor et al. [18] combined a DNN with RNN to classify logs from the CERT Insider Threat dataset, cutting analysis load by 93.5% at a detection rate of ~90%, with an added bonus of interpretability at the component (behavioral) level.

2.4.3. Text-based log detection

Text-based methods treat logs as unstructured text, drawing on established NLP techniques like n-grams to derive semantic features. For instance, Uwagbole et al. [19] tackled SQL injection for IoT by building a dictionary of high-frequency words from logs (including SQL syntax), then applying n-gram analysis plus Chi-square-based feature selection; final classification by SVM reached 98.6% accuracy, 97.4% precision, 99.7% recall, and 98.5% F-measure. In scenarios where normal activity vastly outnumbers anomalies, one-class classification (e.g., isolation forest) is effective. Vartouni et al. [20] used an isolation forest on HTTP logs from CSIC 2010—after n-gram feature extraction and dimensionality reduction via autoencoder—attaining 88.32% accuracy in detecting anomalous web activity.

3. The main part

3.1. Problem of attack detection in networks

Modern computer networks comprise huge amounts of diverse traffic, making analysis highly challenging. Traditional methods—heuristics, signature-based rules, or static filters—decline in effectiveness under dynamic conditions. Identifying sophisticated or unknown threats is crucial for ISPs, corporate segments, and industrial networks. In building IDS, researchers confront not only accuracy issues but also a substantial rate of false positives, which can "clutter" the security system. In the classic IDS model, we receive a feature vector $x \in \mathbb{R}^F$ (where F is the number of traffic attributes, e.g., src_bytes, dst_bytes, count). The task is to classify each vector as "normal" or "attack."

3.2. Sonification of network traffic

An alternative approach involves the "sonification" of network traffic—converting the feature vector into a pseudo-audio signal (PCM). The rationale lies in leveraging the wealth of audio-recognition developments (speech, music, etc.), which might be beneficial for network analysis.

The basic scheme:

1. Normalize and prepare $x \in [-1,1]^F$.
2. Form a PCM signal of NNN samples (e.g., 0.4 s at 8000 Hz ⟶ 3200 samples). Each attribute x_i corresponds to a time block Δt, producing a sinusoidal fragment with $\mathrm{freq}_i = f_{min} + x_i \cdot (f_{max} - f_{min})$, and amplitude $\mathrm{amp}_i = x_i($ або $|x_i|)$.
3. The resulting PCM is processed by audio-signal methods, especially STFT, to produce a 2D "time × frequency" representation.

3.3. Comparing 1D-CNN and "PCM + STFT" (2D-CNN)

In a classic deep-learning scenario for feature vectors, a **1D-CNN** is used. Given $x \in \mathbb{R}^F$:

- x is interpreted as a 1D sequence of length F.
- Each step applies convolution with $k \times 1$ filters.
- After several convolutions/poolings, we move to either a fully connected layer or global pooling, then produce the final classification (sigmoid/softmax).

For a **2D-CNN**, a 2D tensor $\mathbb{R}^{H\times W}$is needed. If we apply sonification, we get a pseudo-audio wave, then STFT yields $\mathrm{Spec}(f,t)$, where f and t correspond to frequency and time windows. Hence, Spec is treated as an image (H,W), and 2D-CNN effectively detects local "patches" in this 2D domain, akin to image processing.

3.4. Mathematical basis of PCM transformation

Let $x \in [-1,1]^F$. Denote $\mathrm{N} = \text{ sample rate } \times \text{ duration } \text{sec}$. For each attribute i ,

$$\Delta n_i = \left\lfloor \frac{N}{F} \right\rfloor. \tag{1}$$

If $\sum_i \Delta n_i < N$, the remainder $R = N - \sum_i \Delta n_i$ may be filled with zeros. Δn_i, for $k = 0..(\Delta n_i - 1)$:

$$\text{wave [start } + k] = \mathrm{amp}_i \cdot \sin\left(2\pi \text{ freq }_i t_{\text{local}}\right), \tag{2}$$

with

$$\mathrm{freq}_i = f_{min} + x_i(f_{max} - f_{min}), \mathrm{amp}_i = x_i. \tag{3}$$

It is basic formulas.

3.5. Short-time Fourier transform (STFT)

Given the PCM signal wave $[n], n = 0, \dots, N-1\}$ over duration_sec, we split it via a window of size L (e.g., 256 samples) with overlap Δ. For the m-th window:

$$\widetilde{w}_m[k] = \text{ window } [k] \cdot \text{ wave } [m(L-\Delta)+k], k = 0..L-1. \tag{4}$$

We compute DFT

$$Zxx_m(\omega) = \sum_{k=0}^{L-1} \tilde{w}_m[k] e^{-\frac{j2\pi\omega k}{L}} \tag{5}$$

Taking $|Zxx_m(\omega)|$ for magnitude, we then apply $\log(1 + |Z|)$ for numerical stability and better visualization. The final STFT matrix $\text{Spec} \in \mathbb{R}^{(\text{freq\_bins} \times \text{time\_frames})}$ serves as the 2D-CNN input.

3.6. Implementation example

To illustrate, we developed a code snippet generating waves for two mock samples A and B (Fig. 1). Each has 41 attributes in [−1..1]. After forming a PCM signal (~3200 samples, 0.4 s, 8 kHz), STFT with nperseg=256 and noverlap=128 yields ~129×25 matrices. Fig. 2 shows color maps for Spec(A) and Spec(B).

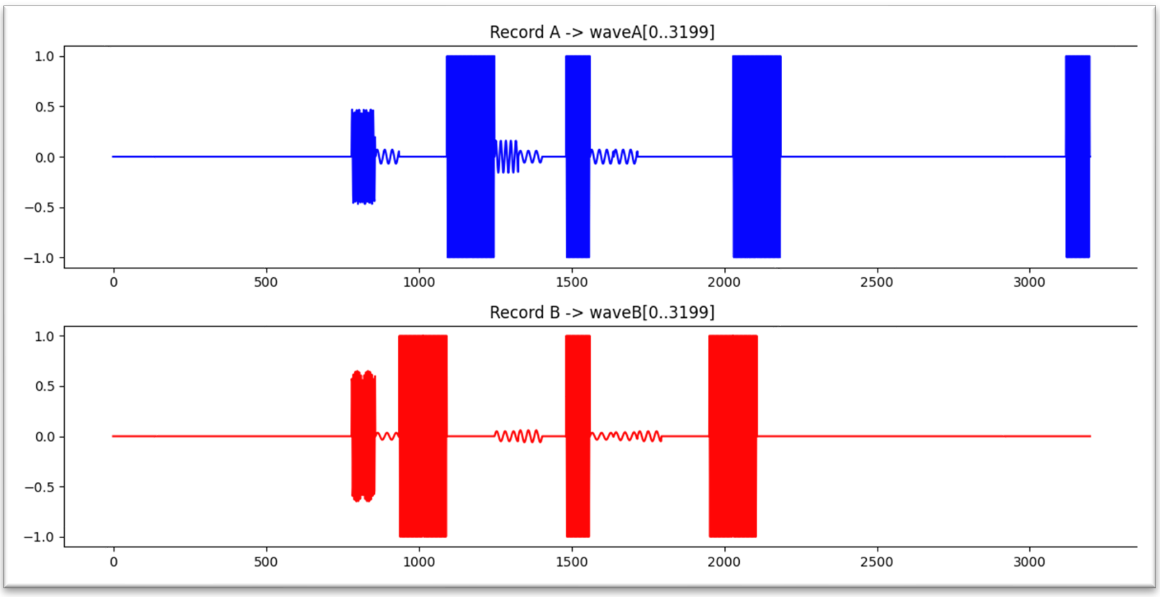

Figure 1: Performance graph for recording A and B after PCM conversion.

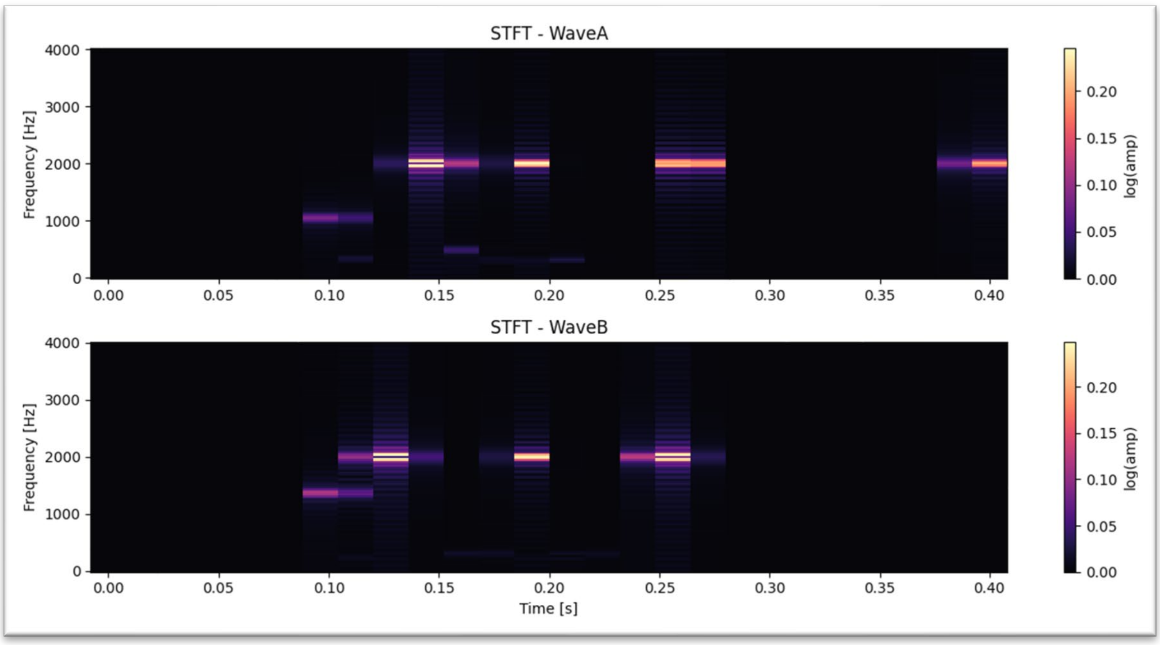

Figure 2: STFT spectrograms of two different recordings.

3.7. Advantages and disadvantages

Advantages:

- Allows using off-the-shelf audio-based techniques (2D-CNN, CRNN, spectral features).
- Intuitive visualization (spectrogram plots).
- If a genuine temporal structure exists, it may expose patterns not visible to 1D analysis.

Disadvantages:

- Artificial "time" axis: if the input vector's attributes lack a natural sequence, the "synthetic signal" adds limited benefit.
- Higher computational cost: generating and STFT-transforming 100k+ samples (each 0.4s) is expensive.
- In many cases, a simple 1D-CNN or MLP on raw features can yield comparable accuracy at lower overhead.

3.8. 2D-CNN training stage

For Spec $\in \mathbb{R}^{H \times W}$, we build a 2D-CNN:

$$\begin{cases} \text{Conv2D } (16{,}3 \times 3), \\ \text{MaxPooling2D } (2 \times 2), \\ \text{Conv2D } (32{,}3 \times 3) \\ \text{GlobalMaxPooling2D } (), \\ \text{Dense } (32, \text{relu }) \\ \text{Dense}(1, \text{sigmoid }) \end{cases} \tag{6}$$

This architecture convolves the 2D "image" (the spectrogram), detects local time–frequency patterns, then performs final classification ("attack" vs. "normal"). After 5–10 epochs, accuracy typically reaches ~75–80%, contingent on hyperparameters.

3.9. Discussion: when 2D-CNN is beneficial

1. If real temporal correlations exist (e.g., each attribute captures a chunk of traffic at different time points), STFT may uncover frequency-based patterns.
2. If attributes do not reflect "adjacent frames," 2D methods may not surpass 1D.
3. Nevertheless, sonification is useful for visualization, as system administrators can "play" (or at least view) the spectrum to spot anomalies quickly.

As a result, future improvements might include:

- Using Mel-spectrograms rather than raw STFT,
- Enhancing certain "frequency" attributes,
- Combining 2D-CNN with recurrent blocks (CRNN) for extended time ranges.

Hence, we have described the process of converting the feature vector into a pseudo-audio waveform, applying STFT for a 2D time–frequency representation, and constructing a 2D-CNN. Mathematical foundations are provided for PCM signal generation, along with example code for two sample records from NSL-KDD. While reasonably straightforward, the approach introduces additional computational overhead and does not always significantly surpass traditional methods. The following sections analyze its effectiveness and propose directions for refinement.

4. Experiments

4.1. Experimental setup

To evaluate the effectiveness of the "PCM+STFT+2D-CNN" approach, we used the NSL-KDD dataset (KDDTrain+ and KDDTest+), a widely recognized benchmark for IDS. As discussed in previous sections, each network record is first transformed into a feature vector $\mathbf{x} \in \mathbb{R}^F$ (with F=122 after One-Hot Encoding). Our experiment can be broken down into three stages:

Normalization: Using MinMaxScaler(feature_range=(-1,1)), attributes are mapped into $[-1,1]$. Thus each of the F=122 attributes of x resides in that same range.

Forming Pseudo-Audio Signals (PCM):

- We set sample_rate=2000 Hz (to reduce computational overhead) and record_dur=0.4 s, producing 800samples_per_record=800.
- Each of the 122 attributes takes ⌊800/122⌋≈6\lfloor 800/122\rfloor \approx 6⌊800/122⌋≈6 samples (any remainder is filled with zeros). For the i-th attribute x_i, $\text{freq}_i = f_{min} + x_i(f_{max} - f_{min})$, and each sample is generated as $\sin\left(2\pi \,\text{freq}_i t\right) \cdot x_i$.

Computing Spectrograms (STFT):

- We apply scipy.signal.stft with nperseg=256 and noverlap=128 to each PCM signal (~800 samples). This produces a 2D matrix of size (freq_bins,time_frames)≈(129,8).
- A log scaling $\log(1 + |Z|)$is applied for numerical stability and enhanced visibility of low-amplitude components.

Figure 1 schematically shows: input vector x ⟶ PCM wave ⟶ STFT matrix. For each training sample in NSL-KDD, we obtain 2D data of shape (129,8) aggregated into a four-dimensional tensor (N,129,8,1), where N=125973 is the number of training examples.

4.2. 2D-CNN configuration and training procedure

For spectrogram classification, we construct a simple 2D-CNN (Code 2):

- Conv2D(16, 3×3) + MaxPooling2D(2×2),
- Conv2D(32, 3×3) + GlobalMaxPooling2D(),
- Dense(32, relu), then Dense(1, sigmoid) for binary classification.

We use Adam (learning_rate=0.0005), train for 5 epochs, batch_size=128, and validation_split=0.2 (20% of the training set).

4.3. Training results

From the result logs:

Train Accuracy ~96% over 5 epochs, Val Accuracy ~96.7%.

On the test set (N=22544), the model obtains Test Accuracy ≈ 76.54% (Test Loss ~0.65).

$$\text{Test Accuracy } \approx 0.7654, \text{ Test Loss } \approx 0.6538$$

Confusion Matrix:

$$\begin{pmatrix} 9349 & 362 \\ 4927 & 7906 \end{pmatrix},$$

where:

- Class 0 (normal): precision=0.65, recall=0.96,

- Class 1 (attack): precision=0.96, recall=0.62.

Hence, the network distinguishes “normal” quite effectively (low false negatives for class 0) but often misclassifies “attack” as normal (62% recall for attacks). Overall accuracy is ~76–77%, aligning with other studies on this dataset absent additional balancing or deeper tuning.

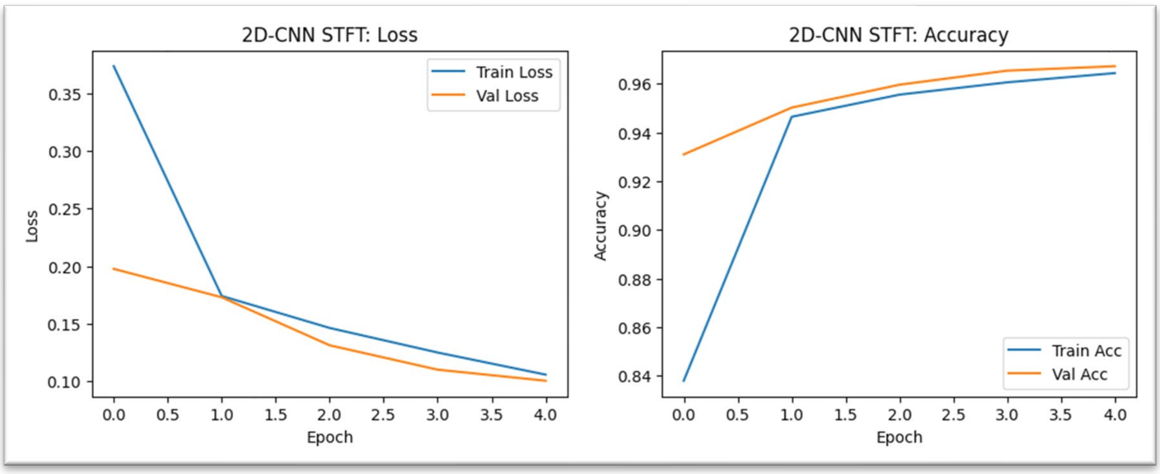

Figure 3: Test results.

4.4. Comparison with 1D-CNN

Comparing ~76.5% accuracy to a baseline 1D-CNN on raw features often yields similar or slightly better accuracy (78–80%). Thus, converting to 2D spectrograms does not guarantee an accuracy boost. Still, “sonification” offers:

- More convenient spectrogram-based visualization,
- Opportunities to utilize audio-based methods (CRNN).

On the other hand, the computational cost over large datasets increase, and improvements in attack detection remain modest without specialized optimizations (e.g., oversampling or attribute weighting).

4.5. Discussion

- Performance: Generating PCM+STFT for 125k samples is computationally expensive, which may pose practical limitations.
- Class Imbalance: NSL-KDD typically has varying proportions of normal vs. attack, and 62% recall for attacks indicates a need for oversampling (SMOTE, GAN).
- Potential Enhancements: Using Mel-spectrograms, deeper 2D-CNN, or conditional CRNN might raise metrics but require additional investigation.

5. Conclusions

This study explored a “sonification” approach to network traffic for building an Intrusion Detection System (IDS). We proposed a method wherein the input feature vector is transformed into a PCM signal, then processed by the Short-Time Fourier Transform (STFT) and subsequently classified via a 2D-CNN. Experiments on the NSL-KDD dataset demonstrated that the model achieves ~76.5% test accuracy—comparable to classic 1D-CNN on vector features.

Main findings:

- Sonification allows leveraging advanced audio-analytics techniques (2D-CNN, CRNN, Mel-spectrograms) for IDS and provides an intuitive means of visualizing spectrograms.
- In the absence of actual temporal structure in the data, the advantage over a conventional 1D approach may be limited, typically around ~75–80% accuracy.
- Computational overhead rises due to generating PCM and STFT for large sample sets, potentially becoming a bottleneck.
- Future refinements may involve oversampling (SMOTE/GAN) to address class imbalance, deeper architecture tuning (2D-CNN or recurrent modules), or amplifying certain "key" attributes in the audio domain.

Consequently, "PCM + STFT + 2D-CNN" is an intriguing experimental direction for attack detection, unifying audio-processing techniques with cybersecurity tasks. Further research might focus on speed optimization and improving recall for rare attack types, for instance through conditional GAN or automated weighting ("enhancement") of attributes.

Declaration on Generative AI

The author(s) have not employed any Generative AI tools.

References

[1] H. Liu, B. Lang, Machine learning and deep learning methods for intrusion detection systems, Applied Sciences 9 (2022) 4396. doi:10.3390/app9204396.

[2] I. Mbona, J.H.P. Eloff, Detecting zero-day intrusion attacks using semi-supervised machine learning approaches, IEEE Access 10 (2022) 3187116. doi:10.1109/ACCESS.2022.3187116.

[3] A.L. Buczak, E. Guven, A survey of data mining and machine learning methods for cyber security intrusion detection, IEEE Communications Surveys & Tutorials 18 (2016) 1153–1176. doi:10.1109/COMST.2015.2494502.

[4] P. Mishra, V. Varadharajan, U. Tupakula, E.S. Pilli, A detailed investigation and analysis of using machine learning techniques for intrusion detection, IEEE Communications Surveys & Tutorials 21 (2019) 686–728. doi:10.1109/COMST.2018.2847722.

[5] A. Aldweesh, A. Derhab, A.Z. Emam, Deep learning approaches for anomaly-based intrusion detection systems, Knowledge-Based Systems 189 (2020) 105124. doi:10.1016/j.knosys.2019.105124.

[6] E. Min, J. Long, Q. Liu, J. Cui, W. Chen, TR-IDS: Anomaly-based intrusion detection through text-convolutional neural network and random forest, Security and Communication Networks 2018 (2018) 4943509. doi:10.1155/2018/4943509.

[7] Y. Zeng, H. Gu, W. Wei, Y. Guo, Deep-Full-Range: A deep learning-based network encrypted traffic classification and intrusion detection framework, IEEE Access 7 (2019) 2908225. doi:10.1109/ACCESS.2019.2908225.

[8] Y. Yu, J. Long, Z. Cai, Network intrusion detection through stacking dilated convolutional autoencoders, Security and Communication Networks 2017 (2017) 4184196. doi:10.1155/2017/4184196.

[9] K. Peng, V.C. Leung, Q. Huang, Clustering approach based on mini batch k-means for intrusion detection system over big data, IEEE Access 6 (2018) 2810267. doi:10.1109/ACCESS.2018.2810267.

[10] S. Teng, N. Wu, H. Zhu, L. Teng, W. Zhang, SVM-DT-based adaptive and collaborative intrusion detection, IEEE/CAA Journal of Automatica Sinica 4 (2017) 7510730. doi:10.1109/JAS.2017.7510730.

[11] T. Ma, F. Wang, J. Cheng, Y. Yu, X. Chen, A hybrid spectral clustering and deep neural network ensemble algorithm for intrusion detection in sensor networks, Sensors 16 (2016) 1701. doi:10.3390/s16101701.

[12] O.U. Olouhal, T.S. Yange, G.E. Okerekel, F.S. Bakpol, Cutting edge trends in deception-based intrusion detection systems, Journal of Information Security 12 (2021) 124014. doi:10.4236/jis.2021.124014.

[13] B. Radford, L. Apolonio, A. Trias, J. Simpson, Network traffic anomaly detection using recurrent neural networks, 2018. doi:10.48550/arXiv.1803.10769.

[14] Wang, W., Sheng, Y., Wang, J., Zeng, X., Ye, X., Huang, Y., Zhu, M. HAST-IDS: Learning hierarchical spatial-temporal features using deep neural networks to improve intrusion detection. IEEE Access 2017, *6*, 1792–1806. URL: https://doi.org/10.1109/ACCESS.2017.2780250.

[15] Meng, W., Li, W., Kwok, L.F. Design of intelligent KNN-based alarm filter using knowledge-based alert verification in intrusion detection, Proceedings of 20th International Symposium on Methodologies for Intelligent Systems (ISMIS 2012), 2012, pp. 115–124.

[16] S. McElwee, J. Heaton, J. Fraley and J. Cannady, Deep learning for prioritizing and responding to intrusion detection alerts, MILCOM 2017 - 2017 IEEE Military Communications Conference (MILCOM), Baltimore, MD, USA, 2017, pp. 1-5, doi: 10.1109/MILCOM.2017.8170757.

[17] N.N. Tran, R. Sarker, J. Hu, An approach for host-based intrusion detection system design using convolutional neural network, Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 235 (2018) 107–122. doi:10.1007/978-3-319-90775-8_10.

[18] A. Tuor, S. Kaplan, B. Hutchinson, N. Nichols, S. Robinson, Deep learning for unsupervised insider threat detection in structured cybersecurity data streams, 2017. doi:10.48550/arXiv.1710.00811.

[19] S. O. Uwagbole, W. J. Buchanan and L. Fan, Applied Machine Learning predictive analytics to SQL Injection Attack detection and prevention, 2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM), Lisbon, Portugal, 2017, pp. 1087-1090, doi: 10.23919/INM.2017.7987433.

[20] A.M. Vartouni, S.S. Kashi, M. Teshnehlab, An anomaly detection method to detect web attacks using Stacked Auto-Encoder, Proceedings of the 2018 6th Iranian Joint Congress on Fuzzy and Intelligent Systems (CFIS), Kerman, Iran, 2018, pp. 131–134.

[21] S. Lysenko, O. Savenko, K. Bobrovnikova, DDoS botnet detection technique based on the use of the semi-supervised fuzzy c-means clustering, in: CEUR Workshop Proceedings, 2104 (2018) 688–695.

[22] S. Lysenko, O. Savenko, K. Bobrovnikova, A. Kryshchuk, B. Savenko, Information technology for botnets detection based on their behaviour in the corporate area network, Communications in Computer and Information Science 718 (2017) 166–181. doi:10.1007/978-3-319-67162-8_13.

[23] S O. Savenko, S. Lysenko, A. Kryshchuk, Y. Klots. Botnet detection technique for corporate area network / / Proceedings of the 7-th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, Berlin (Germany), September 12–14, 2013. Berlin, pp. 363–368. ISBN 978-1-4799-1426-5.

[24] O. Pomorova, O. Savenko, S. Lysenko, A. Kryshchuk, Multi-Agent Based Approach for Botnet Detection in a Corporate Area Network Using Fuzzy Logic, in: Communications in Computer and Information Science 370 (2013) 243-254.

[25] O. Kyrychenko, Information technology for statistical cluster analysis of information in complex networks, Computer Systems and Information Technologies 4 (2022) 47–51. doi:10.31891/csit-2022-4-7.

[26] N. Doukas, P. Stavroulakis, N. Bardis, Review of artificial intelligence cyber threat assessment techniques for increased system survivability, in: Proceedings of the 2021 International Conference on Cybersecurity and Resilience, Springer, 2021, pp. 207–222. doi:10.1007/978-3-030-62582-5_7.

[27] I. Obeidat, M. AlZubi, Developing a faster pattern matching algorithm for intrusion detection system, International Journal of Computing 18 (2019) 278–284. doi:10.47839/ijc.18.3.1520.