TẠP CHÍ KHOA HỌC
TRƯỜNG ĐẠI HỌC SƯ PHẠM TP HỒ CHÍ MINH
Tập 23, Số 9 (2026): 2009-2020
Website: https://journal.hcmue.edu.vn
ISSN: 2734-9918

HO CHI MINH CITY UNIVERSITY OF EDUCATION
JOURNAL OF SCIENCE
Vol. 23, No. 9 (2026): 2009-2020
https://doi.org/10.54607/hcmue.js.23.9.5534(2026)

Research Article

A PREPROCESSING PIPELINE FOR IMAGE CONSTRUCTION IN CNN-BASED MULTICLASS INTRUSION DETECTION

***Ha Thanh Dung*[1*], *Nguyen Hong Son*[2]**

[1]*Faculty of Information Technology, Saigon University, Vietnam*

[2]*Faculty of Information Technology, Posts and Telecommunications Institute of Technology, Vietnam*

[*]*Corresponding author: Ha Thanh Dung – Email: htdung@sgu.edu.vn*

Received: January 27, 2026; Revised: April 10, 2026; Accepted: June 20, 2026

ABSTRACT

*Recent studies have applied deep learning methods, such as convolutional neural networks (CNNs), to structured network traffic by representing the data as images for intrusion detection. Multiclass intrusion detection using these representations faces challenges because class distributions are imbalanced and features differ in scale. We propose a sequential preprocessing pipeline to convert network traffic data into images for CNN-based intrusion detection systems. The pipeline combines categorical variable encoding, adaptive synthetic sampling (ADASYN) to address class imbalance, and base-256 decomposition to preserve precision in large numerical fields. Evaluation using the NSL-KDD dataset and the independ*ent KDDTest+ test set shows that systematic preprocessing improves multiclass detection performance.

Keywords: Data preprocessing; NSL-KDD; CNN

1. Introduction

The trend of digital transformation is global, and information systems support industrial production, business, and social services on almost a daily basis. The increased connectivity of infrastructures also brings cybersecurity threats, for example, through the use of artificial intelligence techniques in social engineering and malware-as-a-service activities (Patel, 2023). Intrusion detection systems (IDSs) continue to play an important role in the network defense architecture.

Whereas signature-based and statistical models were less effective in detecting unknown attacks, machine learning models for intrusion detection were developed and implemented. Support vector machines (SVMs), random forests (RFs), self-organizing maps (SOMs), and artificial neural networks (ANNs) have been used for detecting attacks in several published data-driven IDSs (Tama et al., 2019; Thaseen & Kumar, 2017). However,

Cite this article as: Ha, T. D., & Nguyen, H. S. (2026). A preprocessing pipeline for image construction in CNN-based multiclass intrusion detection. *Ho Chi Minh City University of Education Journal of Science, 23*(9), 2009-2020. https://doi.org/10.54607/hcmue.js.23.9.5534(2026)

the volume and diversity of network traffic data make detection and classification with a low false-positive rate challenging (Liu & Lang, 2019; Zhou et al., 2017).

Deep learning–based intrusion detection is a well-researched area. Several research studies have experimented with deep neural networks (DNNs), recurrent neural networks (RNNs), and hybrid DNN–RNN models on benchmark datasets and reported promising results (Al-Qatf et al., 2018; Vinayakumar et al., 2019; Yin et al., 2017). Some have opted to apply transfer learning with pretrained CNNs (such as VGG) to improve representation. Masum et al. (2021) found binary intrusion detection less complex than multiclass intrusion detection.

Deep learning has been extensively studied in medical image analysis and smart control tasks (Huynh et al., 2021; Tran et al., 2022) and for learning hierarchical representations in multimodal data. Another promising approach is to convert network data into images, which allows the use of CNNs for the task. Network data imaging can be defined as the transformation of structured tabular network records into matrices that can be interpreted by CNNs. Although CNNs are useful for image analysis, differences between natural images and network traffic complicate the direct application of image-based deep learning approaches to network traffic.

In computer vision, preprocessing usually means normalizing and augmenting data.

However, varying feature scales, wide value ranges, categorical features, and imbalanced class distributions in network traffic datasets complicate image construction. Despite existing systems, multiclass classifiers still perform poorly on minority attack categories in datasets such as NSL-KDD (Thaseen & Kumar, 2017). Hence, simple image transformation methods do not help improve the performance of CNNs, and may even reduce performance.

While studies of ensemble-based or lightweight IDS pipelines have appeared (Jang et al., 2023; Wisanwanichthan & Thammawichai, 2025), the effects of different preprocessing strategies have not been fully studied in image-based IDS pipelines. Furthermore, the effects of feature transformation, class balancing, and numerical encoding strategies combined in multiclass IDSs remain open questions.

This paper fills the research gap by proposing a systematic preprocessing pipeline for converting raw network traffic data into images. The pipeline includes categorical feature encoding, class imbalance handling using ADASYN, and base-256 decomposition. In particular, the last method preserves information in large numerical attributes before converting them into images (Ha & Nguyen, 2023).

The contributions of this study are as follows:

(1) A data preprocessing pipeline that converts network traffic input into a format suitable for a CNN intrusion detection system.

(2) On the NSL-KDD dataset and its independent KDDTest+ subset (Tavallaee et al., 2009), the proposed method performs comparably to other multiclass detection methods

based on machine learning or deep learning.

(3) Through a component-wise analysis, we show the importance of class-imbalance handling and structured numerical encoding for multiclass intrusion detection.

The rest of the paper is organized as follows: We present the dataset and our preprocessing method in Section 2, detail our experimental setup and results in Section 3, and provide our conclusions and future work in Section 4.

2. Materials and methods

2.1. Dataset Description

We use the NSL-KDD dataset (Tavallaee et al., 2009), which is a refined version of the KDD Cup 99 dataset and reduces redundant records. This version also improves the evaluation process. The complete NSL-KDD dataset is divided into several subsets. The KDDTrain+ and KDDTest+ datasets used for training and testing contain 125,973 and 22,544 records, respectively. Each data record contains 41 traffic attributes and a class label. The traffic data are divided into normal, denial of service (DoS), probe, remote to local (R2L), and user to root (U2R) classes for multiclass intrusion detection tasks. Different class proportions between training and testing in the NSL-KDD dataset complicate model generalization.

The NSL-KDD dataset contains an imbalance in the R2L and U2R classes, so preprocessing is required to train the model.

2.2. Proposed Preprocessing Pipeline

To construct image-compatible representations for training CNNs, we propose a three-stage preprocessing pipeline (Figure 1):

1. **Categorical feature encoding**
2. **Class imbalance handling**
3. **Base-256 encoding for large numerical attributes**

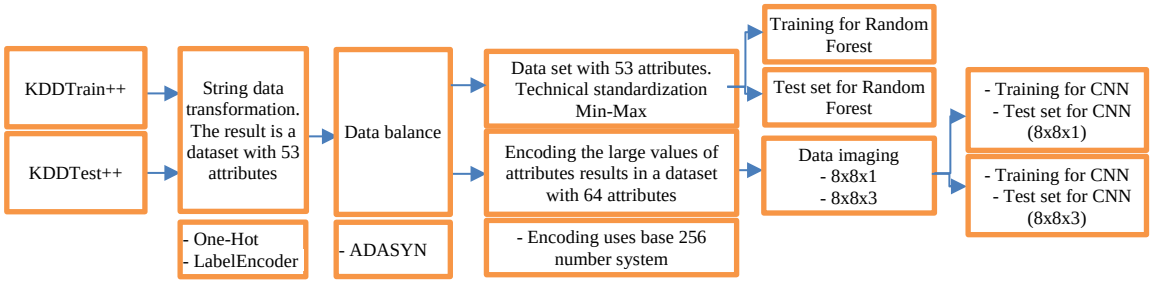


Figure 1. *Data preprocessing process*

2.2.1. Categorical Feature Encoding

The NSL-KDD dataset has 41 features, of which protocol, service, and flag are categorical features. However, machine learning algorithms work on numeric values. As a result, these categorical features need to be converted to numeric features before further preprocessing and deep learning model construction.

The protocol attribute (TCP, UDP, and ICMP) and the flag attribute, which has 11 connection states, are one-hot encoded to avoid imposing arbitrary ordinal relationships

among attribute values.

Because the service attribute has 70 different categories, label encoding is used to reduce the dimensionality of the feature set while maintaining the ability to distinguish between different services. Therefore, the resulting feature space expands from 41 to 53 attributes and provides the basis for subsequent preprocessing and image construction.

2.2.2. Class Imbalance Handling

The KDDTrain+ subset in NSL-KDD has an imbalanced class distribution. The normal class contains 67,343 instances, while the R2L and U2R classes contain only 995 and 52 instances, respectively. Consequently, this could lead to a bias toward majority classes and poor detection performance for infrequent types of attack. The adaptive synthetic sampling (ADASYN) algorithm (He et al., 2008) can be used to generate synthetic samples for minority classes. The ADASYN algorithm focuses on areas with low data density in feature space and generates synthetic samples in the areas with low minority-instance density. The total training dataset consists of 125,973 examples and increases to 336,771 after ADASYN. The ADASYN preprocessing serves to balance the representation of the five classes in the dataset.

Note that ADASYN is applied only to the training set. The KDDTest+ set does not undergo this transformation to avoid data leakage when evaluating classifiers trained on the training set.

2.2.3. Base-256 Encoding for Large Numerical Attributes

To represent the data as images, all feature values must be mapped to pixels with values in [0, 255]. While the majority of NSL-KDD dataset features are integer-valued, some features that are based on rates are real numbers in the [0, 1] range.

In the NSL-KDD dataset, the continuous-valued attributes used contain at most two decimal places. Therefore, to allow attributes to be represented with the same positional encoding while preserving their value, each continuous-valued attribute x is transformed as follows:

$$x_{int} = x \times 10^2$$

Each of these rate features, which had at most two decimal digits of precision, was multiplied by a power of 10 to convert it into an integer. The same power-of-10 multiplier was applied to both the training and test features.

The integer scaling step ensures that all of the attributes in the dataset are non-negative integers.

Base-256 Decomposition:

Attributes such as duration, src_bytes, dst_bytes, num_compromised, num_root, count, and srv_count might take values greater than 255, but since pixel values are limited to [0, 255], the exact value cannot be represented by a single pixel.

For each of these high-value attributes (X), we retain its format and apply base-256

positional encoding:

$$X = \sum_{i=0}^{n-1} a_i \cdot 256^i \quad (1)$$

where:

- $a_i \in [0,255]$
- Let n be the number of components.
- Each a_i component is compatible with a pixel value.

The number of components *n* is determined by finding the maximum attribute value in the training set:

In practice, *n* should be chosen to be$n = \lfloor \log_{256}(\max(X)) \rfloor + 1$ (2)

where $\lfloor \cdot \rfloor$ is the floor function.

Thus, each attribute is represented by an n-dimensional vector:

$(X) \rightarrow (a_0, a_1, \dots, a_{n-1})$

As all components are valid pixel intensities in [0, 255], these values can be used directly in image compositing without any further processing.

For example, the maximum value of the src_bytes attribute in the training set is 1,379,963,888. The required number of base-256 components is calculated using Equation (2), yielding$n = \lfloor \log_{256}(1{,}379{,}963{,}888) \rfloor + 1 = 4$
components in the encoding step.

The value can be expressed as:

$1{,}379{,}963{,}888 = 82 \cdot 256^3 + 64 \cdot 256^2 + 147 \cdot 256^1 + 240 \cdot 256^0$

Thus, the original attribute is replaced by a four-dimensional vector:

$(82{,}64{,}147{,}240)$

This is done for every attribute whose value exceeds 255. In the NSL-KDD dataset, the seven attributes to which this step is applied are duration, src_bytes, dst_bytes, num_compromised, num_root, count, and srv_count. The total number of attributes in the dataset rises from 53 to 64 after the decomposition. The same transformation is applied to both the training and test sets to ensure that the features are on the same scale and that there is no leakage of information from the training set to the test set. Base-256 retains information about magnitude without the information compression associated with more extreme scaling.

2.3. Image Construction and Model Architecture

Each preprocessed record, containing 64 attributes, was represented as an image.

- **Case 1:** 8 × 8 × 1 (grayscale image)
- **Case 2:** 8 × 8 × 3 (color image)

The CNN architecture consists of the following:

- Two 3 × 3 ReLU convolutional blocks (32 filters each)
- Two convolution blocks (64 filters)
- Three convolution blocks (128 filters)
- Max-pooling layers

- Two fully connected (dense) layers with 512 and 64 units, respectively.
- Softmax output layer (5 classes)

Alternatively, a random forest classifier is also trained on the tabular data using min–max scaling:

$$x_i = \frac{x_i - Min}{Max - Min} \quad (3)$$

Each of the network records, after preprocessing and reshaping, can be viewed as an 8 × 8-pixel image as illustrated for each traffic class in Figure 2. Although the images are different from natural images, the structure of the features can be captured by convolutional layers through the fixed positions of these features. In this representation, the features are ordered consistently across samples, thereby preserving the relative position and ordering of the features during the training process.

The spatial structure does not represent the proximity of the features in the semantic space, but the invariance of the transformation allows convolutional filters to learn local feature correlations.

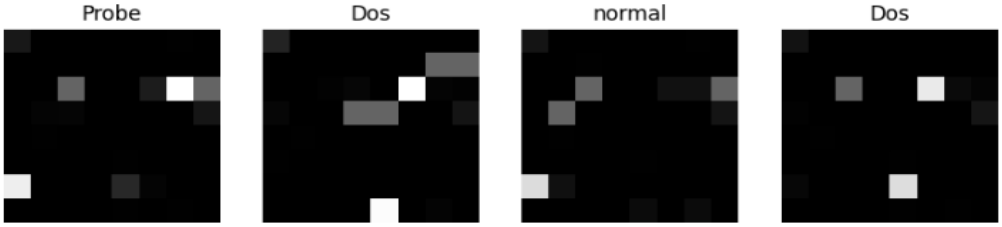


Figure 2. *Examples of Network Traffic Records Represented as Images*

2.4. Evaluation Metrics

The KDDTest+ test set is an independent dataset, and model performance is evaluated using accuracy, precision, recall, and F-score:$Accuracy = \frac{TP+TN}{TP+FP+TN+FN}$ (4)

$$Precision = \frac{TP}{TP+FP} \quad (5)$$

$$Recall = \frac{TP}{TP+FN} \quad (6)$$

$$F\text{-}score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (7)$$

For multiclass evaluation, macro-averaged versions of the metrics are reported, which treat all classes equally.

3. Results and discussion

All experiments were run on a workstation powered by an Intel® Xeon® CPU (2.30 GHz), eight cores, 64 GB of RAM, and an NVIDIA Tesla T4 GPU with 15 GB of memory.

3.1. Baseline Performance Without the Full Preprocessing Pipeline

First, we report baseline model performance with categorical encoding applied but without class imbalance handling (ADASYN) or base-256 decomposition.

In this configuration, we trained three models: the RF classifier, a CNN classifier using

grayscale images of shape 8 × 8 × 1, and a CNN classifier using color images of shape 8 × 8 × 3. The results are shown in Table 1.

Table 1. *Baseline performance with minimal preprocessing (five-class classification)*

Model	Accuracy (%)	Precision (%)	Recall (%)	F-score (%)
Random Forest	74.49	88.35	74.49	78.51
CNN (Grayscale)	74.94	70.96	74.94	70.02
CNN (Color)	76.18	79.17	76.18	71.96

Baseline models achieved multiclass detection accuracy of 74.49%–76.18%. The random forest model had high precision but low recall and F-score, particularly for minority classes. Thus, creating images directly, without systematic preprocessing, was insufficient for effective multiclass intrusion detection.

3.2. Performance of the Proposed Preprocessing Pipeline

Next, we evaluate the full preprocessing pipeline, which consists of:

- Categorical encoding
- ADASYN-based class balancing
- Base-256 decomposition for large numerical attributes.

After preprocessing, the dataset is divided into training (80%) and validation (20%) subsets. The CNN model trained on 8 × 8 × 3 color images converges quickly.

In Figure 3, both training and validation accuracy continue to improve and both training and validation loss continue to decrease until convergence with limited divergence. The near-perfect accuracy of 99.95% on the training set is not due to overfitting but occurs because of convergence on the augmented and balanced training set. However, generalization is evaluated on the independent KDDTest+ test set.

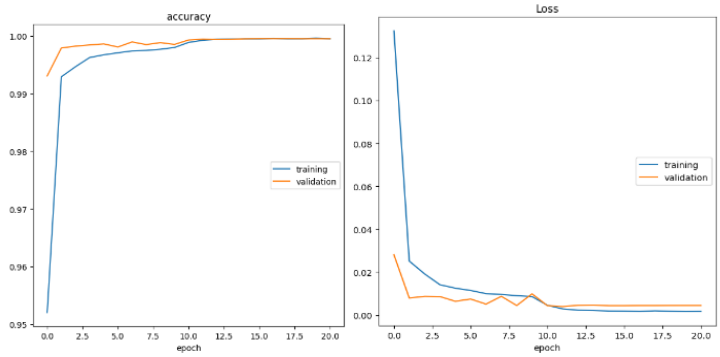


Figure 3. *Training Results Using the Dataset Preprocessed According to the Proposed Method*

On KDDTest+, the CNN (color) model in its final configuration achieved 82.60% accuracy, exceeding the baseline configuration. These results imply that using the full preprocessing pipeline improves CNN accuracy by approximately six percentage points relative to the baseline in Table 1. This shows that systematic preprocessing is a key factor for enabling CNN architectures to learn from network data represented as images.

Additionally, as shown in Table 2, the proposed method achieves similar multiclass

detection performance to deep learning-based and traditional machine learning-based methods reported in the literature.

Table 2. *Comparison with existing methods on NSL-KDD (five-class classification)*

Model	Accuracy (%)	Precision (%)	Recall (%)	F-score (%)
MCNN (Al-Turaiki & Altwaijry, 2021)	81.10	83.00	81.00	80.00
RNN-IDS (Yin et al., 2017)	81.29	–	–	–
SVM (Al-Qatf et al., 2018)	80.48	–	–	–
Random Forest (This study)	81.05	88.02	81.30	83.02
CNN (Grayscale, This study)	81.01	84.00	81.01	79.76
CNN (Color, This study)	**82.60**	**83.80**	**82.50**	**80.90**

3.3. Component-wise Analysis of the Preprocessing Pipeline

We can measure the impact of individual preprocessing steps based on the prediction performance of the pipeline as described in Sections 3.1 and 3.2.

However, the difference in performance between minimal preprocessing and the full preprocessing pipeline can be explained by the severe underrepresentation of minority classes such as R2L and U2R in the NSL-KDD training set. In this scenario, the model's decision boundary would have largely shifted toward the class with the most records. This leads to poor macro-averaged performance; however, by generating samples of the hard-to-learn minority class examples, a classifier's performance on an imbalanced dataset can be improved at the class level.

Base-256 decomposition appears to be primarily a representational improvement, given that it yields a smaller performance improvement than the imbalance-handling step. This is not surprising because base-256 encoding bounds the components of high-valued numerical attributes within the pixel-value range. Thus, image construction avoids excessive compaction and preserves information about magnitude. It is an auxiliary component performing numerical encoding with little effect on the distribution of the classes.

The component-wise comparison suggests that the performance gains are mainly obtained through the correction of class imbalance and the stabilization of the tabular-to-image transformation through structured numeric encoding.

3.4. Discussion

The results also provide insight into the design of preprocessing techniques for image-based intrusion detection.

First, our findings provide evidence that multiclass intrusion detection performance on NSL-KDD is influenced by its class distribution. When the class distribution is imbalanced, optimization dynamics tend to favor the optimization of majority classes within the training data. The improvement in performance after applying ADASYN shows that it is a key factor in multiclass evaluation of this benchmark.

Second, base-256 encoding should be interpreted as a representation. For example, if a numerical variable with a wide range is being mapped to a pixel's brightness, the structural

magnitude can be downscaled. Splitting these attributes into finite positional components through base-256 encoding supports image construction and preserves their magnitudes. While its effect on global accuracy is moderate, it adds stability.

Another observation is the gap between convergence in training and performance on the test set. Thus, while the training accuracy converges to near-perfect accuracy, the generalization performance on the KDDTest+ set is determined by the characteristics of the test dataset and the similarities and differences between the training and test distributions. This emphasizes the need for independent evaluation in cybersecurity research.

More generally, this study shows that, for image-based intrusion detection, accuracy may not necessarily depend on novel architectures but on how preprocessing methods determine where data in structured tabular form are mapped onto the spatial structure of the input space of a CNN. Data preprocessing methods have a non-negligible impact on a CNN's behavior.

This study has several limitations. The first limitation is that it was evaluated solely on NSL-KDD, and the dataset may not reflect current network traffic. In addition, the computational efficiency and scalability were not evaluated. The preprocessing increases the dimensionality of the data from 41 to 64 attributes, but the intermediate representation of the 8×8 image remains compact. Further, this work uses a small CNN architecture and small input image sizes to limit inference time. In the future, the proposed preprocessing framework can be evaluated on other recent datasets and its performance in real-time applications and on large-scale datasets can be explored.

The results show the general importance of preprocessing steps, such as balancing and systematic numerical encoding, for effective multiclass intrusion detection in image-based deep-learning models.

4. Conclusion

We propose a tabular-to-image preprocessing pipeline to transform tabular network traffic datasets into images. The pipeline consists of categorical feature encoding, synthetic instance generation by the adaptive synthetic sampling (ADASYN) algorithm, base-256 decomposition of high-valued numeric features, and conversion of the tabular dataset to an image format.

Experiments on the NSL-KDD dataset show that multiclass detection performance can be considerably improved under the proposed preprocessing configuration for image-based CNN detection. The component-wise analysis of the proposed preprocessing configuration indicates that class imbalance handling is the main contributor to performance improvements, while base-256 encoding is responsible for preserving the numerical consistency of the image transformation.

In summary, our findings highlight the importance of preprocessing strategies for structured nonimage network traffic tasks using deep learning models. Proper feature

transformation based on certain data properties and class balancing have the potential to considerably improve detection robustness and class-balanced detection performance, rather than relying only on the architecture used. Further work may examine the generalization of the proposed preprocessing framework to recent intrusion detection datasets and real-world network traffic, as well as its integration with lightweight or stream-based intrusion detection system architectures.

❖ ***Conflict of Interest:*** *Authors have no conflict of interest to declare.*

❖ ***Acknowledgments:*** *This work is a part of the research project CSB2023-33 (Contract No. 348/HĐ-ĐHSG, dated 11 April 2024), funded by Saigon University.*

REFERENCES

Aldweesh, A., Derhab, A., & Emam, A. Z. (2019). Deep learning approaches for anomaly-based intrusion detection systems: A survey, taxonomy, and open issues. *Knowledge-Based Systems, 189*, 105124.

Al-Qatf, M., Lasheng, Y., Al-Habib, M., & Al-Sabahi, K. (2018). Deep learning approach combining sparse autoencoder with SVM for network intrusion detection. *IEEE Access, 6*, 52843–52856.

Al-Turaiki, I., & Altwaijry, N. (2021). A convolutional neural network for improved anomaly-based network intrusion detection. *Cybersecurity*, *4*(3).

Altwaijry, N., Alqahtani, A., & Al-Turaiki, I. (2020). A deep learning approach for anomaly-based network intrusion detection. In Y. Tian, T. Ma, & M. K. Khan (Eds.), *Proceedings of the First International Conference on Big Data and Security*. Springer.

Dhanabal, L., & Shantharajah, S. (2015). A study on NSL-KDD dataset for intrusion detection system based on classification algorithms. *International Journal of Advanced Research in Computer and Communication Engineering, 4*(6), 446–452.

Ha, T. D., & Nguyen, H. S. (2023). A data preprocessing method prior to imaging for CNN-based intrusion detection. In *Proceedings of the 26th National Conference on Electronics, Communications and Information Technology (REV-ECIT 2023)* (pp. 311–316). https://rev-ecit.vn/vi/content/ky-yeu-hoi-nghi-0

Ferrag, M. A., Maglaras, L., Moschoyiannis, S., & Janicke, H. (2019). Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study. *Journal of Information Security and Applications, 50*, 102419.

He, H., Bai, Y., Garcia, E. A., & Li, S. (2008). ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In *Proceedings of the IEEE International Joint Conference on Neural Networks* (pp. 1322–1328). IEEE.

Huynh, A. T., Nguyen, B.-T., Nguyen, H.-T., Vu, S., & Nguyen, H. D. (2021). A method of deep reinforcement learning for simulation of autonomous vehicle control. In *Proceedings of the 16th International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE 2021)* (pp. 372–379). SciTePress. https://doi.org/10.5220/0010478903720379

Iftikhar, N., Rehman, M. U., Shah, M. A., Alenazi, M. J. F., & Ali, J. (2025). Intrusion detection in NSL-KDD dataset using hybrid self-organizing map model. *Computer Modeling in Engineering & Sciences*, *143*(1), 639–671.

Jang, J., An, Y., Kim, D., & Choi, D. (2023). Feature importance-based backdoor attack in NSL-KDD. *Electronics*, *12*(24), 4953. https://doi.org/10.3390/electronics12244953

Liu, H., & Lang, B. (2019). Machine learning and deep learning methods for intrusion detection systems: A survey. *Applied Sciences*, *9*(20), 4396.

Masum, M., Shahriar, H., & Haddad, H. M. (2021). A transfer learning with deep neural network approach for network intrusion detection. *International Journal of Intelligent Computing Research*, *12*(1), 1087–1095.

NSL-KDD Dataset. (n.d.). *NSL-KDD intrusion detection dataset*. https://www.unb.ca/cic/datasets/nsl.html

Patel, R. (2023, August 9). *Protecting against sophisticated cyberattacks requires layered defenses*. Forbes Technology Council.

Potdar, K. (2017). A comparative study of categorical variable encoding techniques for neural network classifiers. *International Journal of Computer Applications*, *175*(4).

Sherin, V. J. I. J., & Radhika, N. (2022). Stacked ensemble IDS using NSL-KDD dataset. *Journal of Pharmaceutical Negative Results*, *13*(3).

Tama, B. A., Comuzzi, M., & Rhee, K.-H. (2019). TSE-IDS: A two-stage classifier ensemble for intelligent anomaly-based intrusion detection system. *IEEE Access*, *7*, 94497–94507.

Tavallaee, M., Bagheri, E., Lu, W., & Ghorbani, A. A. (2009). A detailed analysis of the KDD Cup 99 data set. In *Proceedings of the IEEE Symposium on Computational Intelligence for Security and Defense Applications* (pp. 1–6). IEEE.

Thaseen, I. S., & Kumar, C. A. (2017). Intrusion detection model using fusion of chi-square feature selection and multiclass SVM. *Journal of King Saud University – Computer and Information Sciences*, *29*(4), 462–472.

Tran, N. N. D., Nguyen, H. D., Huynh, N. T., Tran, N. P., & Nguyen, L. V. (2022). Segmentation on chest CT imaging in COVID-19 based on the improvement Attention U-Net model. In *New trends in intelligent software methodologies, tools and techniques* (Vol. 355, pp. 596–606). IOS Press. https://doi.org/10.3233/FAIA220288

Vinayakumar, R., Alazab, M., Soman, K. P., Poornachandran, P., Al-Nemrat, A., & Venkatraman, S. (2019). Deep learning approach for intelligent intrusion detection system. *IEEE Access*, *7*, 41525–41550.

Wisanwanichthan, T., & Thammawichai, M. (2025). A lightweight intrusion detection system for IoT and UAV using deep neural networks with knowledge distillation. *Computers*, *14*(7). https://doi.org/10.3390/computers14070291

Wu, Y., & Hu, X. (2022). An intrusion detection method based on fully connected recurrent neural network. *Scientific Programming*, *2022*, 1–11. https://doi.org/10.1155/2022/7777211

Yin, C., Zhu, Y., Fei, J., & He, X. (2017). A deep learning approach for intrusion detection using recurrent neural networks. *IEEE Access*, *5*, 21954–21961.

Zhou, L., Pan, S., Wang, J., & Vasilakos, A. V. (2017). Machine learning on big data: Opportunities and challenges. *Neurocomputing*, *237*, 350–361.

QUY TRÌNH TIỀN XỬ LÍ DỮ LIỆU CHO XÂY DỰNG BIỂU DIỄN ẢNH TRONG PHÁT HIỆN XÂM NHẬP ĐA LỚP DỰA TRÊN CNN

Hà Thanh Dũng[1*], Nguyễn Hồng Sơn[2]

[1]Khoa Công nghệ Thông tin, Trường Đại học Sài Gòn, Việt Nam

[2]Khoa Công nghệ Thông tin II , Học viện Công nghệ Bưu chính Viễn thông, Việt Nam

[*]Tác giả liên hệ: Hà Thanh Dũng – Email: htdung@sgu.edu.vn

Ngày nhận bài: 21-01-2026; Ngày nhận bài sửa: 10-4-2026; Ngày duyệt đăng: 20-6-2026

TÓM TẮT

Trong những năm gần đây, học sâu ngày càng được nghiên cứu và ứng dụng rộng rãi trong phát hiện tấn công mạng, với xu hướng chuyển đổi dữ liệu lưu lượng mạng có cấu trúc thành các biểu diễn dạng ảnh nhằm khai thác các mạng nơ-ron tích chập (CNN). Tuy nhiên, việc phát hiện xâm nhập đa lớp trên các bộ dữ liệu mạng được biểu diễn dưới dạng ảnh vẫn còn nhiều thách thức, đặc biệt khi dữ liệu bị mất cân bằng nghiêm trọng và các đặc trưng có phạm vi giá trị rất khác nhau. Trong bài báo này, chúng tôi đề xuất một quy trình tiền xử lí có cấu trúc nhằm chuẩn bị dữ liệu lưu lượng mạng trước khi xây dựng ảnh đầu vào cho các hệ thống phát hiện xâm nhập dựa trên CNN. Phương pháp kết hợp mã hóa đặc trưng phân loại, xử lí mất cân bằng lớp bằng ADASYN và cơ chế phân rã cơ số 256 đối với các thuộc tính số có giá trị lớn nhằm bảo toàn thông tin trong quá trình chuyển đổi sang dạng ảnh. Phương pháp được đánh giá trên bộ dữ liệu NSL-KDD với tập kiểm thử độc lập KDDTest+. Kết quả cho thấy quy trình tiền xử lí có hệ thống giúp cải thiện hiệu năng phát hiện đa lớp so với các thiết lập tiền xử lí tối thiểu.

Từ khóa: tiền xử lí dữ liệu; NSL-KDD; CNN