Journal of Big Data

RESEARCH Open Access

Network intrusion detection using feature fusion with deep learning

Abiodun Ayantayo[1], Amrit Kaur[1], Anit Kour[1], Xavier Schmoor[1,2], Fayyaz Shah[2], Ian Vickers[2], Paul Kearney[1] and Mohammed M. Abdelsamea[3,4*]

*Correspondence:
m.abdelsamea@exeter.ac.uk

[1] School of Computing and Digital Technology, Birmingham City University, Birmingham, UK
[2] METCLOUD LTD, Birmingham, UK
[3] Faculty of Computers and Information, Assiut University, Assiut, Egypt
[4] Department of Computer Science, University of Exeter, Exeter, UK

Abstract

Network intrusion detection systems (NIDSs) are one of the main tools used to defend against cyber-attacks. Deep learning has shown remarkable success in network intrusion detection. However, the effect of feature fusion has yet to be explored in how to boost the performance of the deep learning model and improve its generalisation capability in NIDS. In this paper, we propose novel deep learning architectures with different feature fusion mechanisms aimed at improving the performance of the multi-classification components of NIDS. We propose three different deep learning models, which we call early-fusion, late-fusion, and late-ensemble learning models using feature fusion with fully connected deep networks. Our feature fusion mechanisms were designed to encourage deep learning models to learn relationships between different input features more efficiently and mitigate any potential bias that may occur with a particular feature type. To assess the efficacy of our deep learning solutions and make comparisons with state-of-the-art models, we employ the widely accessible UNSW-NB15 and NSL-KDD datasets specifically designed to enhance the development and evaluation of improved NIDSs. Through quantitative analysis, we demonstrate the resilience of our proposed models in effectively addressing the challenges posed by multi-classification tasks, especially in the presence of class imbalance issues. Moreover, our late-fusion and late-ensemble models showed the best generalisation behaviour (against overfitting) with similar performance on the training and validation sets.

Keywords: Feature fusion, Deep learning, Fully-connected networks, Network intrusion detection

Introduction

According to the Gartner report, "Market Guide for AIOps Platforms" [1], the rapid growth in event data cannot wait for humans to derive insights. There is a need for automation and support from machine learning (ML) in IT security operations. The report also mentions that rule-based event correlation has given way to AI-based correlation due to the speed at which the correlation rules must be updated. In fact, any modern cybersecurity vendor website states that traditional signature-based solutions can be beaten by advanced threats such as polymorphic malware, hence the need for more adaptive solutions using ML.

One of the main tools used to defend against cyber-attacks is the Network Intrusion Detection System (NIDS). A NIDS is an appliance that monitors network traffic from a cybersecurity point of view. It is installed in a strategic location on a network, often just inside a perimeter firewall. It takes in a stream of packets and sends alerts to operators, Security Information and Event Management systems (SIEMs) and/or other devices and applications, whenever it detects events of potential cybersecurity significance. Devices that can take appropriate action in addition to detecting potentially malicious activity are known as Network Intrusion Prevention or Detection and Prevention Systems (NIPS or NIDPS) [2].

A NIDS commonly uses three complementary event detection methods, individually or in combination:

- Signature-based: the NIDS examines individual packets that comply with specified conditions, looking for textual patterns that are characteristic of known malicious activity. Such 'smoking gun' characteristics are known as Indicators of Compromise (IoC).
- Anomaly-based: the NIDS compares the traffic conditions it observes with known profiles representing the normal behaviour of entities such as users, hosts, network connections and applications. An alert is issued if the observed behaviour is significantly different from the profiles. Profiles are generated using machine learning techniques by observing traffic over time under typical usage conditions when malicious activity is not believed to be present.
- Stateful protocol analysis-based: The NIDS models the state of dialogues between hosts based on the packets they exchange. An alert is issued if an exchange deviates from what is generally expected for the protocol in question.

Signature-based approaches have a number of disadvantages, including the following.

- Cybercriminals come up with new attacks on a regular basis. Consequently, the IoC database needs to be updated continually.
- Attackers are aware of the detection techniques used and develop ways to circumvent them. For example, they apply random modifications to malware that do not affect its function but mean that the hash function will produce a different result. Consequently, attackers and defenders are in a never-ending arms race, with defenders usually having to respond to the attackers' innovations.
- A new attack must be observed, reported and confirmed before the appropriate IoC can be added to the signature database. During this time, the attacker has a window of opportunity to exploit the technique undetected.

Anomaly detection techniques are not dependent on signatures, but rather on the (reasonable) assumption that malicious activity will reveal itself in changes to the behaviour of the system being monitored. They should be able to detect novel attacks, not be fooled by variations in existing attacks, and not be subject to the time lag between the first use of an attack and the ability to detect it. They are complementary to signature-/rule-based techniques, each detecting attacks that the other may miss.

In cybersecurity applications for anomaly detection, unusual activity is identified as potentially malicious. This means high false-positive rates can result from atypical examples of legitimate behaviour that are not included in the dataset used for training. In contrast, the main concern with signature-based methods is false negatives.

Researchers have also applied classification-based ML techniques to the problem of distinguishing between malicious and benign traffic. These are trained using datasets containing a mixture of benign and malicious traffic, with records labelled accordingly. For multi-way classification tasks, the labels of the malicious records reflect the type of attack involved.

ML methods can broadly be classified as either classical/statistical or deep learning. Classical/statistical methods often require extensive pre-processing of data records to get them into optimal form. Such 'feature engineering' requires both domain knowledge and ML expertise to be successful. With the availability of massive data, which often involves multiple data types, deep learning [3] has been a catalyst for solving complex problems. Unlike classical/statistical machine learning methods, deep learning models can learn the features themselves in an automated and hierarchical fashion. This provides an efficient and effective solution to eliminate the traditional way of manually hand-crafting or selecting features that are then fed to a machine learning model.

NIDSs are confronted with diverse data modalities, necessitating effective handling of this heterogeneity. Ramachandram et al. [4] explore various deep multimodal learning architectures capable of integrating features from distinct modalities or signals from diverse sensors. Their work provides a comprehensive overview of the impact of incorporating learned multimodal representations into deep learning models. Leveraging the availability of multiple data modalities, deep learning exhibits substantial promise in addressing intricate machine learning challenges. Consequently, there is a research gap regarding the development of novel deep learning architectures and feature fusion mechanisms that are custom-tailored to NIDSs, enabling the effective management of heterogeneous data modalities and enhancing the multi-classification performance. To address this gap, our study presents pioneering deep learning architectures employing distinct feature fusion mechanisms, with the specific aim of augmenting the performance of NIDSs. These architectures incorporate aggregation layers that combine disparate features and facilitate the learning of their interdependencies, akin to models employed in multimodal learning architectures. Notably, our feature fusion mechanisms have been meticulously designed to encourage the deep learning model to capture the relationships between inputs across different stages and levels during the training process. Consequently, further investigation and evaluation are warranted to ascertain the efficacy and superiority of our proposed models in comparison to existing approaches within the realm of NIDSs. For illustrative purposes, we utilise the deep fully connected architecture as an exemplar to showcase the effectiveness of our proposed solutions.

The main contributions can be summarised as follows:

1. Early-fusion scenario
    - Integration of various feature types into a unified feature vector, serving as input for a fully-connected network.

- A three-step process is employed: initial signal transformation using a processing layer, concatenation of transformed signals, and subsequent feeding into a fully connected network.
- Facilitation of signal fusion within a single network to learn shared local and hierarchical features.

2. Late-fusion scenario

- Introduction of a late-fusion mechanism to address potential biases in the early-fusion model.
- Utilisation of multiple fully connected networks, each associated with a specific feature subset.
- Fusion of high-level representations for subsequent usage as input to the output/classification layer.

3. Ensemble learning scenario

- Exploitation of both early-fusion and late-fusion models through ensemble learning.
- Integration of high-level representations from both models into a consolidated feature vector.

Related work

There is considerable interest in applying machine learning techniques to improve threat detection performance, and many publicly available datasets have been generated to facilitate this research, including 1998 DARPA, KDDCup99, NSL-KDD, and UNSW-NB15 [5–8]. In [9], researchers propose a generic layered framework for applying machine learning to threat detection and discuss the different unsupervised and supervised machine learning techniques that can be applied. In [10], three different machine learning techniques (support vector machine, decision trees, and deep belief network) were used in several cybersecurity datasets for malware detection, spam detection, and intrusion detection. It has been pointed out that there is no silver bullet to tackle cyberthreats, different machine learning techniques need to be used depending on the type of threat, and there is a need for more open data, which currently lacks diversity and advanced attacks. Ref. [11] proposes a combination of supervised and unsupervised machine learning using random forest and k-means on an intrusion detection dataset. The research suggests that this hybrid approach yields better results than traditional methods. Ref. [8] used the NSL-KDD dataset to evaluate different supervised machine learning approaches, including Random Forest Classifier (RFC), comparing performance metrics such as precision, recall, and F1-Score to determine the most effective classifier for intrusion detection. According to the results of the experiment for the NSL-KDD dataset and the set of parameters, the random forest classifier outperformed other statistical machine learning classifiers. Ref. [12] proposed to use a Random Effects Logistic Regression (RELR) model to forecast the discovery of anomalies. It employed a random-effects model to account for network environment features and unaccounted for uncertainty, introducing a wrapper feature selection phase based on fixed-effects logistic

regression (FELR). The UNSW-NB15 dataset has been used to conduct a study to determine the types of cyber attacks that have occurred, using both k-means and correlation analysis for feature selection, followed by Naive Bayes and decision trees for classification [13]. As a result of this hybrid feature selection procedure, there was a noticeable improvement in the Naives Bayes classifier's accuracy, but the decision trees performed similarly with or without feature selection. The proposed approach was able to identify more uncommon threats, including BackDoor, Shellcode, and Worms. Ref. [14] provides a novel technique based on the k-Nearest Neighbour classifier approach for simulating program behaviour in intrusion detection systems. Early trials using 1998 DARPA BSM audit data indicate that this technique is capable of detecting invasive software activity efficiently. Compared to previous approaches that use short system call sequences, the kNN classifier does not need distinct profiles of short system call sequences for various programs, significantly reducing the work required to identify new program activity. Furthermore, the findings demonstrate that a low false positive rate is achievable. While this conclusion may not hold for more complicated datasets, text classification approaches seem to be well suited for use in the intrusion detection sector.

In [15], a multilevel hybrid intrusion detection model (based on a support vector machine (SVM) and an extreme learning machine) has been designed to improve the efficiency of network intrusion detection, where the KDDCup99 dataset was used to evaluate the performance of the model. An SVM-based intrusion detection system was previously proposed in [16] that employs a hierarchical clustering for feature selection and SVM model, to speed up the training time on the KDDCup99 dataset. Motivated by many reported shortcomings in the KDDCup99 datasets, such as the curse of high dimensionality, in [17], different machine learning models have been used with multiple sets of features to study the importance of features and improve intrusion detection rates in the UNSW-NB15 and KDDCup99 datasets. In [18], many generative and discriminative approaches (such as XGBoost, Support Vector Machine, k-Nearest-Neighbour, Logistic Regression, Artificial Neural Network, and Decision Tree) have been used with/without feature selection on the UNSW-NB15 dataset, studying the effect of feature selection stage. An integrated classification-based IDS was proposed in [19], where the performance has been evaluated on the UNSW-NB15 dataset showing better accuracy compared to traditional models such as the decision tree model. With the aim of reducing the efficiency of detecting attacks and increasing false alarm rate, features significance and characteristics of UNSW-NB15 and KDDCup99 datasets were examined in [20] using An Association Rule data mining approach. Similarly, an Association Rule approach under rough set theory [21] was proposed to model IDSs. In [22], a fuzzy rule-based system was proposed, which is designed to find the optimal feature subset using a genetic feature selection wrapper and providing interpretable fuzzy IF-THEN rules on the KDDCup99 dataset. In [23], a multilevel semi-supervised ML (MSML) approach was proposed to cope with the network traffic class imbalance problem in the KDDCup99 dataset. In [24], a dual ensemble model has been proposed that combines bagging and gradient boosting decision tree (GBDT) techniques, showing its superiority in reducing false alarms and increasing detection rates for anomaly-based intrusion detection systems compared to existing approaches. Moreover, [25] provides a comprehensive overview of how ensemble learners are utilised in intrusion detection systems (IDSs) through

a systematic mapping study, analysing 124 publications and presenting an empirical investigation of a new classifier ensemble approach called stack of ensemble (SoE), which combines random forest, gradient boosting machine, and extreme gradient boosting machine in a parallel architecture to improve performance metrics such as Matthews correlation coefficients, accuracies, false positive rates, and area under ROC curve metrics. In [26], an IDS driven by statistical analysis and autoencoder (AE) techniques was proposed, which leverages data analytics, statistical methodologies, and recent advances in machine learning theory to extract highly optimised and strongly correlated features, yielding superior classification performance compared to the state-of-the-art approaches, as evidenced by comparative experimental results utilising the NSL-KDD database. This comparison study includes multiple classical models such as Linear-Support Vector Machine (L-SVM), Quadratic-Support Vector Machine (Q-SVM), Linear Discriminant Analysis (LDA) and Quadratic Discrimination Function (QDA), and the Long Short-Term Memory (LSTM) models.

The above-mentioned studies all apply classical/statistical ML techniques. Deep learning has also demonstrated excellent success in many cybersecurity-related tasks due to its ability to learn high-level features from large datasets with complex distributions. This is achieved by automatically learning hierarchical feature representations by passing the data through several hidden layers, eliminating the need for feature crafting. Consequently, the significance of new data can be grasped without the need for subject expertise [27]. In [28], different deep learning techniques, including malware classification and anomaly detection, were investigated against eight types of cyberthreat. In [29], network intrusion data are classified based on two techniques, a deep learning approach called 'self-taught learning' that first applies a sparse autoencoder to unlabelled data and then uses a neural network classifier on labelled data, and a soft-max regression model applied without feature learning. The 'self-taught learning' approach outperformed the other method. A deep learning framework has been developed in [30] using a Restricted Boltzmann Machine and a deep belief network (DBN) to improve the detection speed and accuracy in the KDDCup99 dataset. Unlike DBN, the convolutional neural network (CNN) was designed to extract features from images and was used in [31] to identify multiple attack classes. A nonsymmetric deep autoencoder (NDAE) for unsupervised feature learning has been proposed in [32] and has been evaluated on the KDDCup99 dataset, as a deep learning solution for intrusion detection. In [33], a highly scalable and hybrid framework called scale-hybrid-IDS-AlertNet is proposed for real-time monitoring and proactive alerting of cyberattacks. The study evaluates the performance of various machine learning algorithms on publicly available malware datasets, selecting optimal network parameters and topologies for the deep neural network (DNN) models, and concludes that DNNs outperform classical machine learning classifiers. Altwaijry et al. [34] address the increasing cybersecurity threats and the need for an Anomaly Detection Based Network Intrusion Detection System (ADNIDS) by introducing a multi-classification DNN (MDNN) with good accuracy and recall compared to other deep learning approaches and classification models. Similarly, in [35], a deep learning model based on convolutional neural network (CNN) is proposed for binary and multi-class classification of network attacks, achieving good performance in terms of accuracy and recall, surpassing similar models in the literature. Yin et al. [36] propose a hybrid

feature selection method, IGRF-RFE, which combines information gain (IG) and random forest (RF) filter methods with recursive feature elimination (RFE) wrapper method, demonstrating its effectiveness in improving anomaly detection accuracy on the UNSW-NB15 dataset. However, the study only considers six classes during training and removes minority classes in the reprocessing stage. Salim et al. [37] propose a novel deep learning strategy using bidirectional long short-term memory (LSTM) and a symmetric logarithmic loss function to address limitations of current intrusion detection systems (IDS) on the Internet of Things (IoT), achieving high accuracy rates on benchmark datasets such as NSL-KDD and UNSW-NB15, but limited to binary classification tasks.

Having reviewed the state-of-the-art related models, it is evident that despite the notable success of classical machine learning and deep learning in network intrusion detection, feature selection remains a challenging problem and feature fusion mechanisms have not been explored explicitly. Due to the presence of different feature types, it is still not clear how the different features affect the resulting accuracy of deep learning models. As a consequence, this work focuses on guiding the learning process of deep learning models using features of the same type with different feature fusion mechanisms to encourage the deep learning model to learn the relationships between the different types of features, as detailed in the following section.

Material and methods

In this section, we describe, in detail, our proposed deep learning approaches for network intrusion detection. Starting with a description of the datasets used in this work, the section then gives an overview of the architecture of the three proposed deep learning models and discusses their training settings.

Datasets

In this work, we used the UNSW-NB15 and NSL-KDD datasets to validate and evaluate the performance of our deep learning solutions.

UNSW-NB15 contains real normal network traffic and synthetic attack behaviours. Packet data representing the mixture of normal and abnormal traffic was passed through a network audit tool (Argus) to produce flow data (a flow represents a collection of packets exchanged between two end-points), and a network traffic analyser (Bro-IDS, now known as Zeek) to produce connection-related data. The connections and flows were aligned, resulting in a total of 35 packet-based and flow-based features, from which a further 12 features were extracted using bespoke algorithms. Of the 47 features, 2 are timestamps, 2 are binary, 5 are strings, 28 are integers, and 10 are floating-point. In addition, each entry has a binary label according to whether it represents normal or abnormal traffic, and a string label categorising it as Normal or as being an example of one of 9 types of attack (Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode, Worms). This work uses a version of the dataset in which the source/destination addresses/ports and start/end times features have been removed and a new float feature (e.g. rate) has been introduced to describe the average packet transmission rate. Consequently, a total of 42 features have been selected to train and validate our models: 11 float features, 28 integer features (including two binary features), and 3 string features, see Fig. 1. The target variable has 10 unique classes, see Fig. 2, which highlights

that the dataset has imbalanced classes, with the "Normal" being the most represented with over 90,000 observations and "Worm" being the least represented with fewer than 50 observations.

NSL-KDD is an edited version of the KDD-Cup99 intrusion detection dataset. NSL-KDD addresses problems present in KDD-Cup99 such as duplicated records that could lead to high bias when classifiers are trained on it. The original dataset was created based on information obtained during the DARPA 1998 IDS assessment program. It includes 41 features, among which 3 are strings, 23 are integers (including 5 binary features), and 15 are floating-point. Each record is labelled as one of 39 distinct attack types. These attack types fit into one of four categories, which we use as the target variable: Denial of Service (DoS), Unauthorized Access to Local Superuser Privileges (U2R), Unauthorized Access from a Remote Machine (R2L), and Probing or Surveillance (probe), see Fig. 3. Normal traffic is also represented in the data, labelled as "normal". The dataset is provided already split between training and testing sets with distinct distributions. The training set contains 125,973 observations and the test set 22,544 observations. An interesting characteristic of the test set is the presence of 17 attack types absent from the training set, making 16.6% of the test set or 3750 observations. To observe the impact of these "unknown" attack types on our classifier, we created a distinct test set not including these observations.

The proposed deep learning models

In this work, we propose three different deep learning architectures based on multi-level information fusion mechanisms to cope with the multi-classification problem. They all share an initial step in which a normalisation layer transforms the floating point feature values into a distribution centred at 0 with a standard deviation of 1, and lookup layers for the integer and string signals convert categorical data values to their corresponding indices by means of look-up tables. More precisely, unlike traditional fully-connected networks, we process the input features separately according to the type of the feature. For example, the data set is divided into subsets (based on the type of data), and each subset is prepared by a different pre-processing layer. In contrast, in the traditional fully-connected network, the raw data are directly fused and used as input to the network. The idea behind our proposed pre-processing step is to simplify the local structure of the data where examples of the same subset are sharing the same datatype, and consequently the fully connected network can easily learn the relationships between the different types of the features.

Early fusion deep learning model

In the early-fusion model, we integrate the different types of feature into a single feature vector that can act as input to a fully-connected network. Our early fusion model is made up of three steps, as shown in Fig. 4.

Once the different input signals are transformed using our processing layer, the transformed signal is concatenated and fed into a fully connected network (of three layers), where each layer is followed by a ReLU activation function to add nonlinearity to the network. Also, two dropout layers have been added to improve the generalisation of the network. Finally, a classification layer is added with a Softmax classifier to generate the

predictions. The early-fusion model can cope with the different raw input features by fusing the different pre-processed signals into a single network to learn shared local and hierarchical features for the classification of the different classes. Algorithm 1 represents the Early Fusion Model, which aims to generate classification predictions for input data consisting of different signals $S_1, S_2, ..., S_n$. The algorithm begins by applying various processing layers, including the integer layer, normalisation layer, and string lookup layer, to transform each input signal. These transformed signals are then concatenated into a single vector. Subsequently, a fully connected network is created with three fully connected layers, each using a ReLU activation function and a dropout layer for regularisation. Finally, a classification layer with a softmax classifier is added, and the algorithm returns the classification predictions for the input data.

Require: Different input signals $S_1, S_2, ..., S_n$.
Ensure: Classification predictions for input data.
Transform the input signals using the processing layer:
for $i \leftarrow 1$ to n **do**
    Apply the integer layer to input signal S_i.
    Apply the normalisation layer to input signal S_i.
    Apply the string lookup layer to input signal S_i.
end for
Concatenate the transformed signals into a single vector.
Create a fully connected network:
for $i \leftarrow 1$ to 3 **do**
    Add a fully connected layer with ReLU activation function.
    Add a dropout layer.
end for
Add a classification layer with softmax classifier.
return Classification predictions for input data.

Late-fusion deep learning model

Early fusion could lead the fully-connected network to be biased towards a subset of the fused feature vectors. To alleviate this issue, we propose a late-fusion mechanism that allows the fully connected network to learn high-level representations of the different subsets (see Fig. 5). After the pre-processing stage, this model employs multiple fully connected networks associated with the different feature subsets. The multiple fully-connected networks can generate high-level representations separately from the different types of features. These high-level representations are then fused and used as input to the output/classification layer of the model. The algorithmic steps of the late fusion model are described in Algorithm 2, which aims to generate classification predictions for input data. It takes pre-processed feature subsets $S_1, S_2, ..., S_n$ as input. For each feature subset, a fully connected network is created with ReLU activation functions and dropout layers for regularisation. The network is trained using the corresponding feature subset. The high-level representations of each feature subset are combined into a single vector,

which is then fed into a classification layer with a softmax classifier to generate predictions. The number of output nodes in the classification layer corresponds to the number of classes in the classification problem. Finally, the algorithm returns the classification predictions for the input data.

Require: Pre-processed feature subsets $S_1, S_2, ..., S_n$.
Ensure: Classification predictions for input data.
for $i \leftarrow 1$ to n **do**
    Create a fully connected network for feature subset S_i.
    Each layer should have a ReLU activation function.
    Include dropout layers to enhance generalisation.
    Train the fully connected network using feature subset S_i.
end for
Combine the high-level representations of each feature subset into a single vector.
Feed the fused vector into a classification layer.
Include a softmax classifier layer to generate predictions.
The number of output nodes should match the number of classes in your classification problem.
return Classification predictions for input data.

Late-Ensemble deep learning model

Here, we propose a different learning scenario using ensemble learning by exploiting the power of both the early and late fusion models. The idea behind our ensemble model is to improve the diversity of the classification model by integrating the high-level representations from both the early-fusion and late-fusion models into a single feature vector that can be fed into a fully-connected network to better learn the relationships between the different representations. Our model not only learns the features from early- and late-fusion models but also uses a late-fusion mechanism to mitigate any bias problem that might occur during the integration of their decisions. As shown in Fig. 6, the ensemble model combines the deep features encoded by the early and late fusion models through an averaging operation. Then a fully connected three-layer network is used to learn the relationship between the fused features. This is followed by a classification layer with a Softmax classifier. The algorithmic steps of the late fusion model are described in Algorithm 3. As illustrated in Algorithm 3, the Ensemble Fusion (or late-ensemble) Model takes an early-fusion model and a late-fusion model as inputs. It initialises an empty array to store fused features and generates high-level representations using both models. Then, for each sample, it concatenates the corresponding features from both models and computes the average of the concatenated feature vector. These averaged feature vectors are stored in an array. Next, a fully connected network is created, consisting of three fully connected layers with ReLU activation and dropout layers. Finally, a classification layer with a softmax classifier is added, and the algorithm returns the classification predictions for the input data.

Require: Early-fusion model, Late-fusion model.
Ensure: Classification predictions for input data.
Initialise an empty array F to store fused features.
Generate high-level representations using the early-fusion model.
Generate high-level representations using the late-fusion model.
for $i \leftarrow 1$ to number of samples **do**
    Initialise an empty array V to store fused vector for sample i.
    for all features f in the high-level representations **do**
        Concatenate feature f from early-fusion model and late-fusion model.
        Add the concatenated feature to array V.
    end for
    Compute the average of all features in array V.
    Add the averaged feature vector to array F.
end for
Create a fully connected network:
for $i \leftarrow 1$ to 3 **do**
    Add a fully connected layer with ReLU activation function.
    Add a dropout layer.
end for
Add a classification layer with softmax classifier.
return Classification predictions for input data.

Training settings

In the training stage of all the proposed deep learning models, a stochastic gradient descent approach called Adam optimisation is used to minimise a categorical cross-entropy loss function. Adam optimisation is computationally and memory efficient, invariant to diagonal rescaling of gradients, and well suited for extensive data and parameter tuning situations.

The categorical cross-entropy loss function that we used for all the proposed models is defined as

$$E_{CCE}\left(y^i, y'(x^j, W)\right) = -\sum_{i=1}^{c} y^i \ln y'\left(x^j, W\right), \tag{1}$$

where x^j is the set of input examples (or rows), y^i is the class labels, c is the number of classes, while $y'\left(x^j, W\right)$ is the predicted output from a Softmax function, where W is the converged weight matrix associated to the deep learning model.

To evaluate the performance of our deep learning models, we adopted Accuracy, Precision, and Recall metrics for the multi-class confusion matrix, where the input sample can be classified into one of (c) classes. As a consequence, the matrices are defined as follows:

$$\text{Accuracy} = \frac{1}{c}\sum_{i=1}^{c} \frac{TP_i + TN_i}{TP_i + TN_i + FP_i + FN^i}, \tag{2}$$

$$\text{Precision} = \frac{1}{c}\sum_{i=1}^{c}\frac{TP_i}{TP_i + FP_i}, \tag{3}$$

$$\text{Recall} = \frac{1}{c}\sum_{i=1}^{c}\frac{TP_i}{TP_i + FN_i}, \tag{4}$$

where c is the number of classes in the dataset (i.e., 10 different classes), TP is the true positive and TN is the true negative, while FP and FN are incorrect model predictions. The TP, TN, FP, and FN are defined based on a specific class i as:

$$TP_i = \sum_{i=1}^{n} x_{ii} \tag{5}$$

$$TN_i = \sum_{j=1}^{c}\sum_{k=1}^{c} x_{jk}, j \neq i, k \neq i \tag{6}$$

$$FP_i = \sum_{j=1}^{c} x_{ji}, j \neq i \tag{7}$$

$$FN_i = \sum_{j=1}^{c} x_{ij}, j \neq i, \tag{8}$$

where x_{ii} is an element in the diagonal of the multi-classes confusion matrix (as pointed out in [38]).

Finally, for a fair comparison, although the deep learning architectures of our models were different, the number of neurons in each layer was generated by the same hyperparameter tuning mechanism. The Keras tuner package was used in this work to investigate the best hyper-parameters for deep learning models. Furthermore, the L2 regularisation method (at 0.01) was used with all models and each model was trained with 100 epochs and a batch size of 32.

Experimental results

This section analyses and discusses the performance of the proposed deep learning models demonstrates the effectiveness of our data fusion approaches and the robustness of combining multiple models over a single deep learning model and finally compares the performance of our models to the state-of-the-art models.

Performance on the UNSW-NB15 dataset

As illustrated in Table 1, the precision obtained by the deep learning model of early fusion in the training set was 90.9%, 84.4% for the validation set, and 76.4% for the test set. However, the precision obtained by the late fusion model in training was 85.9%, 83.9% for validation, and 77.1% in the testing set. As also demonstrated by Fig. 7, the gap between the accuracy/loss curves on the training and validation is significantly less in

Table 1 The performance of our deep learning models on the UNSW-NB15 dataset

DL model	Train	Validation Performance (%)	Test
Early-fusion model			
Accuracy	90.99	84.44	76.47
Precision	96.29	90.0	83.53
Recall	87.46	80.87	71.59
Late-fusion model			
Accuracy	85.92	83.92	77.09
Precision	93.54	91.49	86.04
Recall	81.45	79.20	69.50
Late-Ensemble model			
Accuracy	83.67	83.09	76.84
Precision	91.75	90.88	85.92
Recall	78.55	77.64	68.18

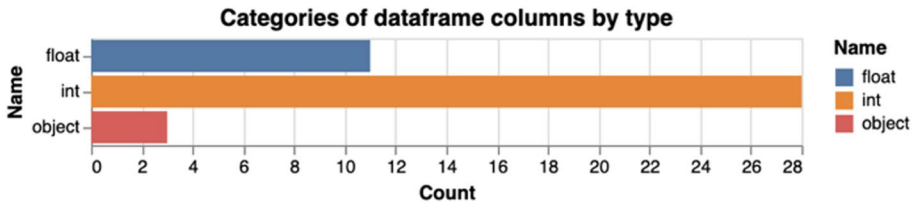

Fig. 1 Categorising UNSW-NB15 dataset based on data types

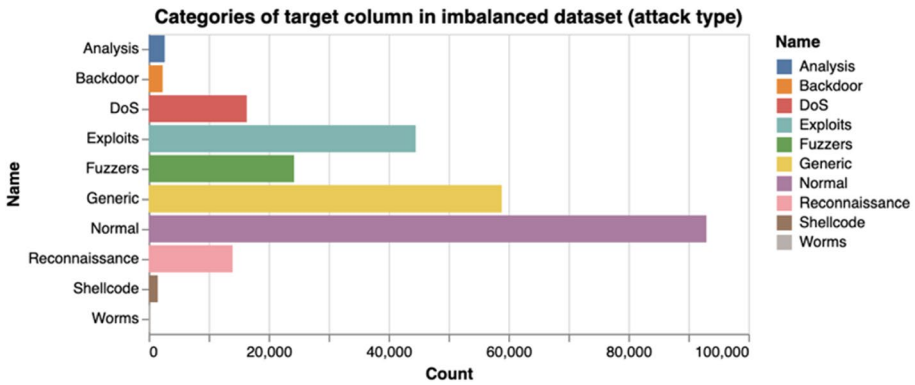

Fig. 2 Categorising UNSW-NB15 dataset based on target classes

our late-fusion model than in the early-fusion model. This confirms the capability of the late-fusion model to learn more generic features. It also shows better performance on the testing set.

Our ensemble model showed an accuracy of 83.6% on the training set, 83.09% on the validation set, and 76.8% on the testing set. Although the late-fusion model achieved the highest accuracy in the test set, the ensemble deep learning model showed more generalisation capability, with the behaviour of the model almost the same for both training and validation sets (see Fig. 7).

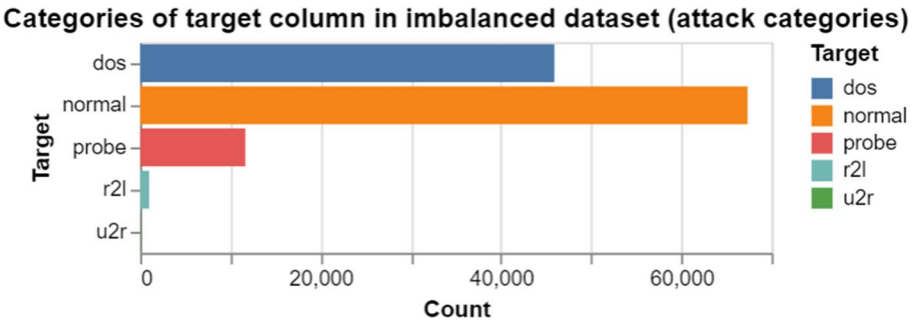

Fig. 3 Categorising NSL-KDD dataset based on target classes

In terms of precision and recall, the late fusion model outperformed all other models with a testing precision of 86.04%, while the early fusion model showed the highest recall of 71.5%. Moreover, the late-ensemble model showed the best generalisation behaviour (against over-fitting) with similar performance (in terms of precision and recall) on the training and validation sets while the early-fusion model is the most sensitive to the over-fitting problem.

To demonstrate the sensitivity of our models to the individual classes, Figs. 8, 9, and 10 show the ROC curves of our models for the 10 classes. Both early- and late-fusion models show robust behaviour towards all the classes but they are less robust to the (Analysis) class while the late-ensemble model shows low performance toward both Analysis and Worms classes.

Ablation study

To better demonstrate the behaviour of our proposed deep learning models, we applied the same feature-fusion mechanisms without the proposed processing layer (resulting in three special versions of the early-fusion, late-fusion, and late-ensemble models). We also compared the performance of our models with the traditional fully connected network. We report the results of the four models in Table 2. The ROC curves for the four models are demonstrated in Fig. 11 to demonstrate the behaviour of the four models on the individual classes.

Performance on the NSL-KDD dataset

Additionally, to further validate our three proposed models, namely early-fusion, late-fusion, and late-ensemble, we conducted experiments using another dataset (NSL-KDD) and employed the Keras Tuner library to select the optimal set of hyperparameters. The results, presented in Table 3, demonstrate that the deep learning model utilising early fusion outperformed the late-fusion model on the testing set. On the contrary, the late ensemble model achieved the highest performance, exhibiting accuracy, precision and recall rates of 86.81%, 86.86%, and 86.80%, respectively. Moreover, we generated the ROC curves using the late-ensemble model to show the sensitivity to the individual classes; see Fig. 12.

Table 2 The performance of special versions of our deep learning models and other architectures

Model	Train	Validation Performance (%)	Test
Early-fusion (w/o processing layer)			
Accuracy	84.27	84.33	73.55
Precision	92.41	92.18	83.45
Recall	78.39	78.32	66.04
Late-fusion (w/o processing layer)			
Accuracy	84.32	84.48	74.09
Precision	93.8	94.02	84.07
Recall	77.32	77.37	64.8
Late-Ensemble (w/o processing layer)			
Accuracy	84.19	84.26	73.97
Precision	92.38	92.46	82.05
Recall	78.62	78.58	65.48
Traditional FCN (w/o processing layer)			
Accuracy	71.88	68.25	31.93
Precision	75.24	71.98	31.93
Recall	68.56	64.94	31.90

Table 3 The performance of our deep learning models on the NSL-KDD dataset

DL model	Train	Validation Performance (%)	Test
Early-fusion model			
Accuracy	99.74	99.66	84.50
Precision	99.76	99.69	84.98
Recall	99.72	99.63	84.31
Late-fusion model			
Accuracy	99.50	99.51	83.73
Precision	99.57	99.57	84.01
Recall	99.41	99.42	83.52
Late-Ensemble model			
Accuracy	99.55	99.43	86.81
Precision	99.56	99.46	86.86
Recall	99.54	99.41	86.80

Comparison to existing models

Subsequently, a comparison is conducted between our deep learning models and existing models, ensuring fairness by employing models previously proposed for the multi-classification task of distinguishing between various attacks within the testing sets of the UNSW-NB15 and NSL-KDD datasets. According to the results presented in Table 4, the Adaboost method exhibits the lowest performance on the UNSW-NB15 test set, while the decision tree with genetic algorithm (DT & GA) achieves the highest accuracy of 84.33%. However, both precision and recall demonstrate relatively low values, indicating a limited ability to address the class imbalance issue. In contrast, our proposed model attains higher precision and recall rates of 86.04% and 69.50%, respectively, showcasing its potential to handle the challenging dataset

Table 4 The testing performance of the state-of-the-art machine learning models on the multi-classification task of the UNSW-NB15 and NSL-KDD Dataset

Model	Refs.	Dataset	Accuracy (%)	Precision (%)	Recall (%)
Logistic Regression	[18]	UNSW-NB15	65.53	76.91	65.54
Support Vector Machine	[18]	UNSW-NB15	61.09	47.47	62.00
Decision Tree	[18]	UNSW-NB15	66.03	79.82	66.04
scale-hybrid-IDS-AlertNet	[33]	UNSW-NB15	66.00	62.30	66.00
MDNN	[34]	UNSW-NB15	62.87	76.00	63.00
MCNN	[35]	UNSW-NB15	69.46	84.00	69.00
Naive Bayes	[35]	UNSW-NB15	45.22	29.67	38.62
J48	[35]	UNSW-NB15	51.50	28.18	21.48
Random Forest	[35]	UNSW-NB15	68.09	62.51	35.15
Bagging	[35]	UNSW-NB15	51.45	32.85	21.45
Adaboost	[35]	UNSW-NB15	51.50	28.18	21.48
DT & GA	[39]	UNSW-NB15	84.33	53.20	52.23
Our proposed model	–	UNSW-NB15	77.84	86.04	69.50
scale-hybrid-IDS-AlertNet	[33]	NSL-KDD	78.50	81.00	78.50
MDNN	[34]	NSL-KDD	77.55	81.23	77.55
MCNN	[35]	NSL-KDD	81.1	83	81
Naive Bayes	[35]	NSL-KDD	72.73	76.1	72.7
J48	[35]	NSL-KDD	74.99	79.6	75.0
Random Forest	[35]	NSL-KDD	76.45	82.1	76.4
Bagging	[35]	NSL-KDD	74.83	78.3	74.8
AE	[26]	NSL-KDD	81.21	87.85	82.04
LSTM	[26]	NSL-KDD	67.17	82.34	72.49
MLP	[26]	NSL-KDD	68.26	85.05	73.96
L-SVM	[26]	NSL-KDD	69.73	86.77	74.09
Q-SVM	[26]	NSL-KDD	75.11	87.22	77.94
LDA	[26]	NSL-KDD	76.49	80.82	76.72
QDA	[26]	NSL-KDD	64.36	78.26	70.38
Our proposed model	–	NSL-KDD	86.81	86.86	86.80

without any preprocessing requirements. Regarding the NSL-KDD dataset, all the compared models exhibit comparable accuracy, with the quadratic discrimination function (QDA) method yielding the minimum accuracy of 64.36%, while the AE method attains the highest precision of 87.85%. However, our model surpasses all other methods in terms of both accuracy (86.81%) and recall (86.80%).

Discussion

In this section, we discuss practical considerations in making use of the ML-based approach described above to improve the effectiveness of a NIDS in detecting and characterising malicious activity. As mentioned above, the current NIDS combines detection mechanisms that are signature-based, anomaly-based, and protocol analysis-based. In contrast, the model we have proposed is a multi-way classifier, in which one of the classes represents normal behaviour and the others, different types of malicious behaviour.

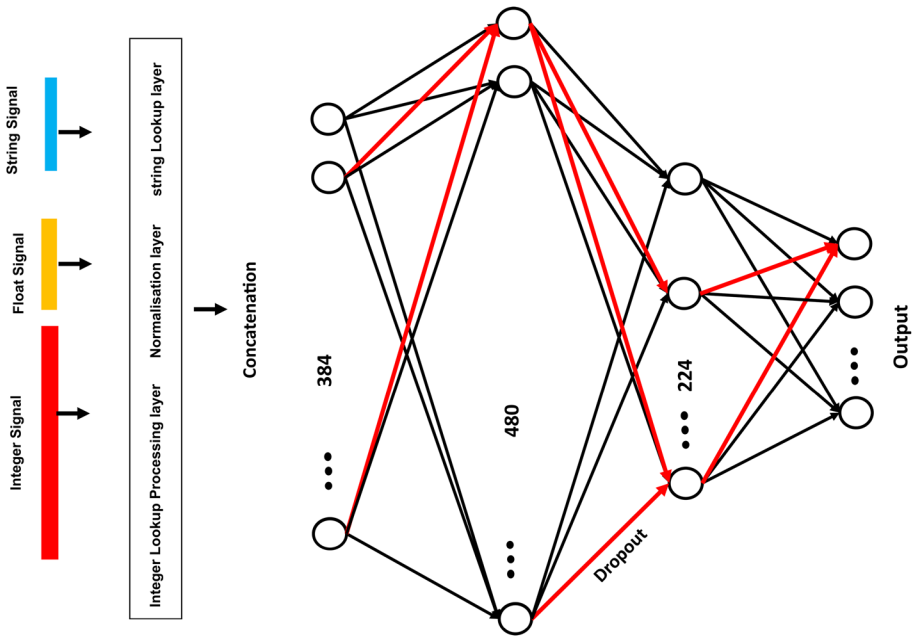

Fig. 4 The overall architecture of the early-fusion model, showing the connection between the different input signals, processing layer, the densely/fully connected layers (where neurons are actively connected using black arrows while red arrows are for the dropped out neurons or connections), and the classification layer

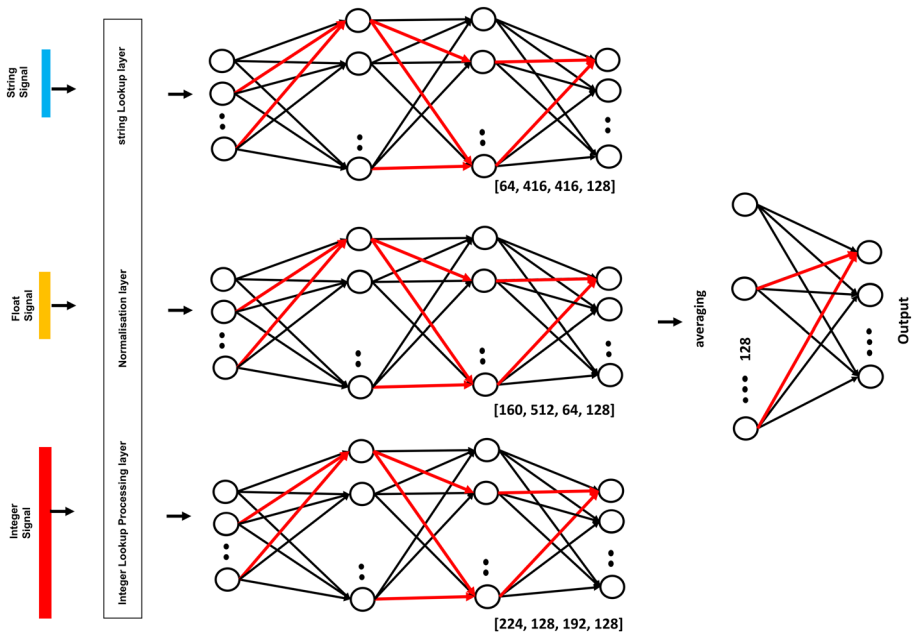

Fig. 5 The overall architecture of the late-fusion model, each data type was passed to a sub-model and each sub-model output has 128 neurons with ReLU non-linear activation function. Then, the average predictions of the sub-models define the output layer with the Softmax classifier

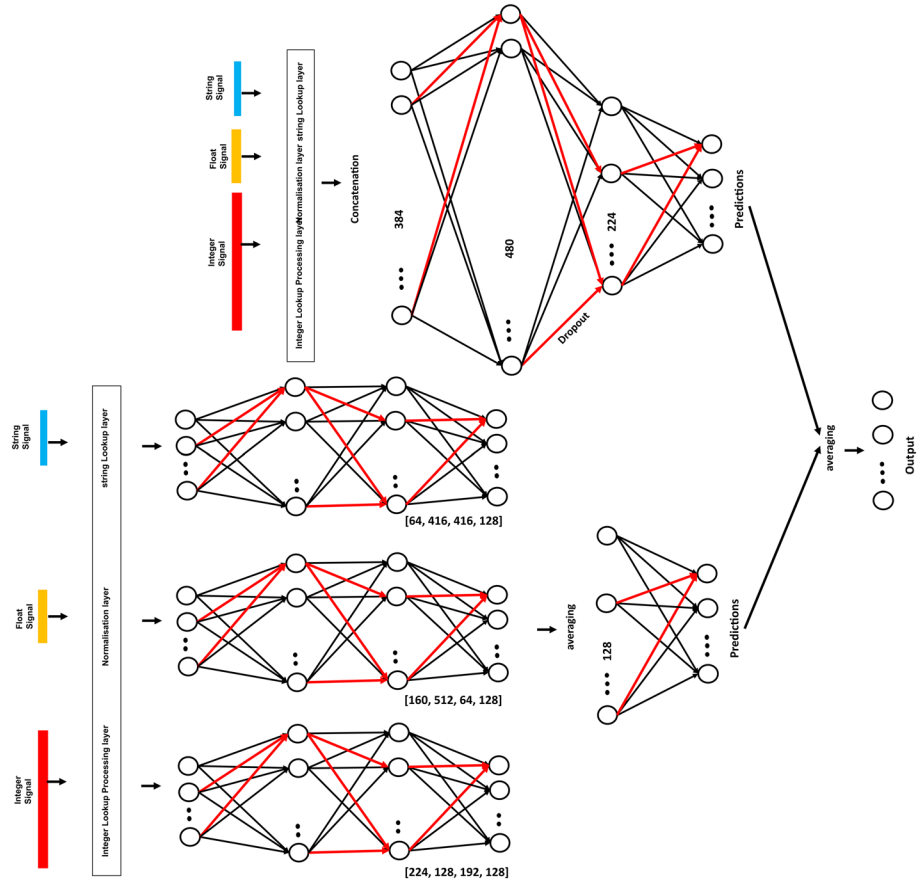

Fig. 6 The integration of early-fusion and late-fusion models as an ensemble learning mechanism

A multi-way classifier is in some sense intermediate between signature-based and anomaly-based mechanisms. Like signature-based mechanisms, it makes use of knowledge of existing attacks, albeit in a less explicit and fine-grained form, and is able to provide information that narrows down the nature of the attack. Like anomaly-based mechanisms, it is behaviour-based and uses machine learning. However, rather than recognising normal and treating everything else as 'other', it assigns examples to a finite set of categories, of which one is normal, and the others are malicious. The model should be much less sensitive to detailed changes in attack procedure than signature-based approaches, and hence more difficult for an attacker to evade. Nevertheless, the 'closed world' assumption could be problematic when faced with a novel attack that does not conform to one of the types in the training dataset. It is difficult to predict which category the input data will be assigned to in such circumstances. It is important, therefore, to be able to recognise when confidence in a prediction is low. An increase in the frequency of low confidence predictions might be an indication of need to introduce a new attack category and/or to retrain the classifier.

An important consideration in selecting a NIDS mechanism is the amount of training the software requires, prior to going live and during its active lifetime. Signature- and protocol analysis-based mechanisms are effective 'out of the box'. While

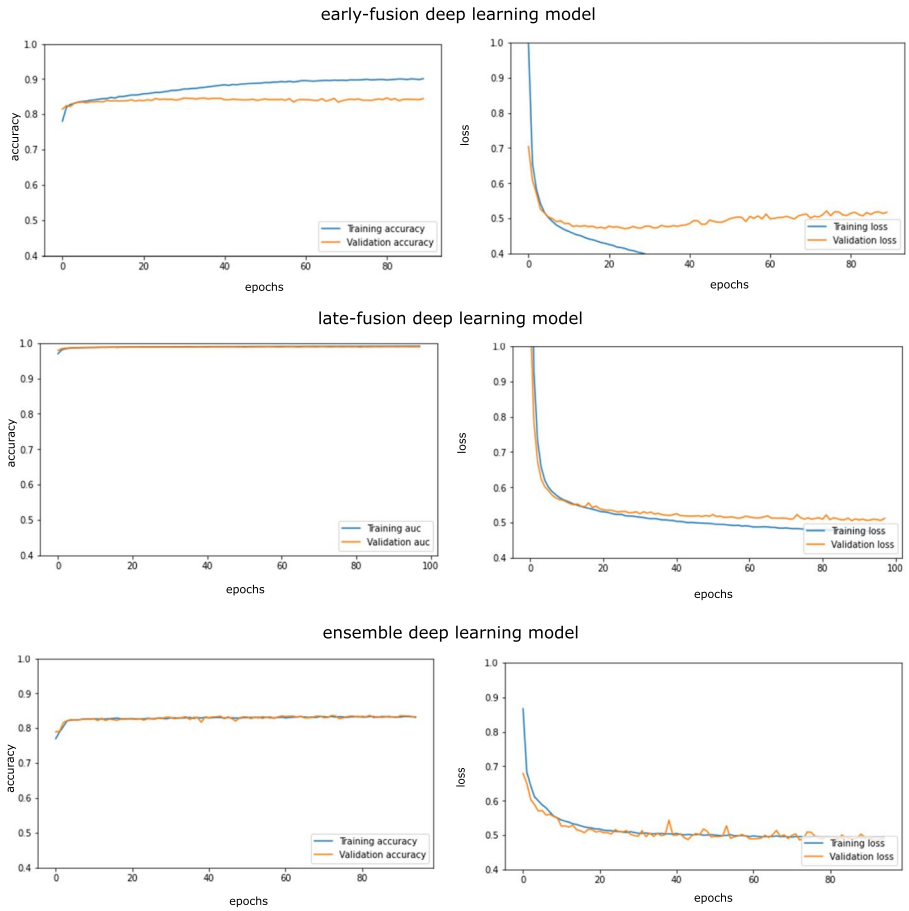

Fig. 7 Training and validation accuracy and loss curves of our deep learning models on the for the UNSW-NB15 dataset

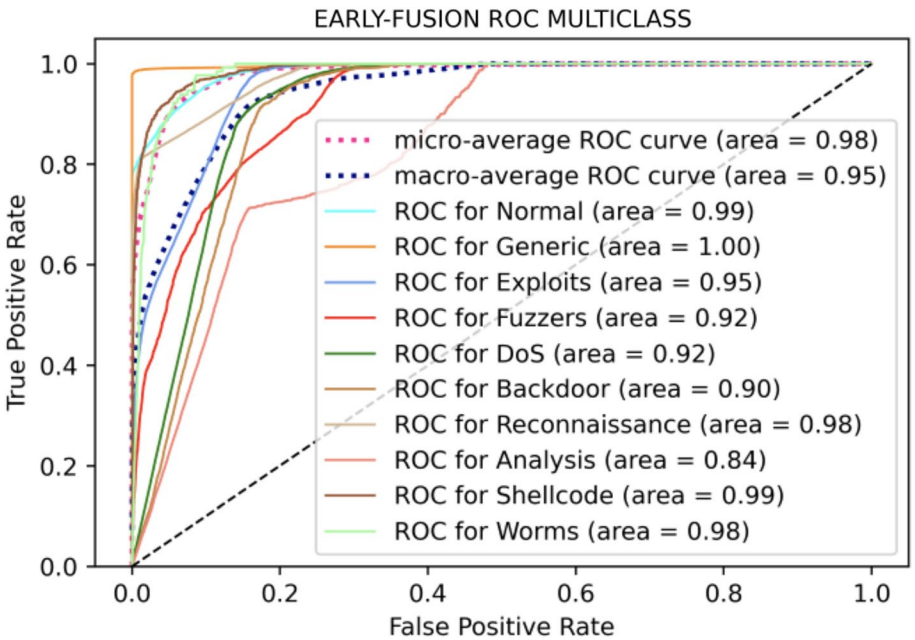

Fig. 8 The ROC curves of our early-fusion model in the UNSW-NB15 data set

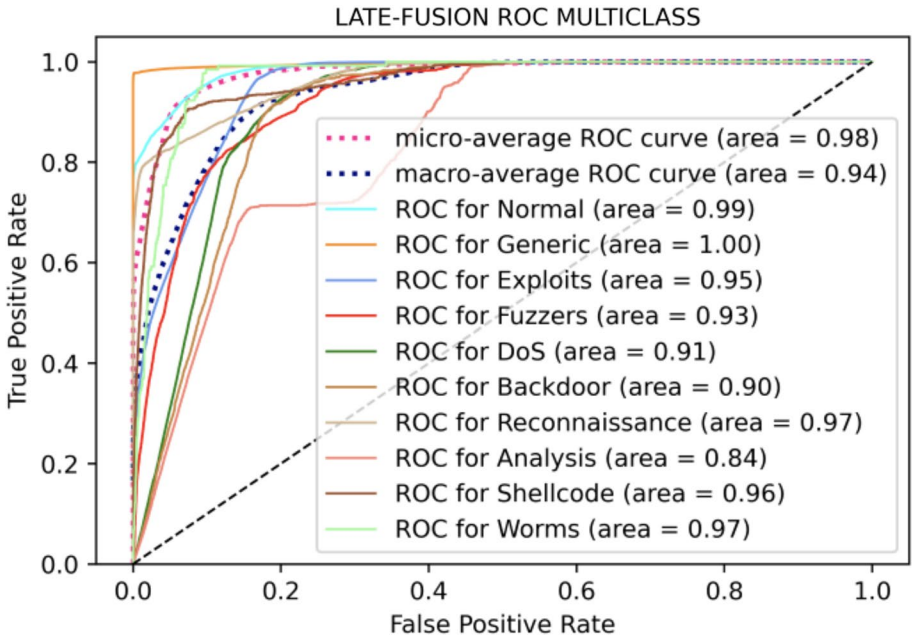

Fig. 9 The ROC curves of the late-fusion model in the UNSW-NB15 dataset

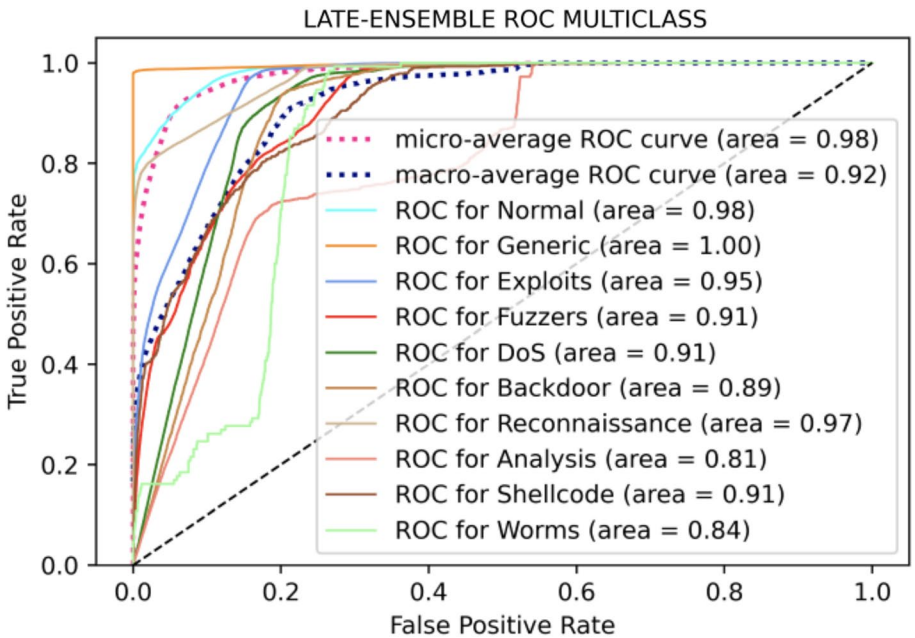

Fig. 10 The ROC curves of the late-ensemble model on the UNSW-NB15 dataset

signature databases need to be updated regularly to incorporate new vulnerabilities and exploits, this is well understood and easily automated. In contrast, anomaly-based schemes can take weeks of training to adapt them to a new deployment context, and require constant adjustment to limit false positives. It seems likely that the normal and attack behaviour profiles learned by a multi-way classifier will also be context-sensitive, and there is the added complication of an ever-changing threat

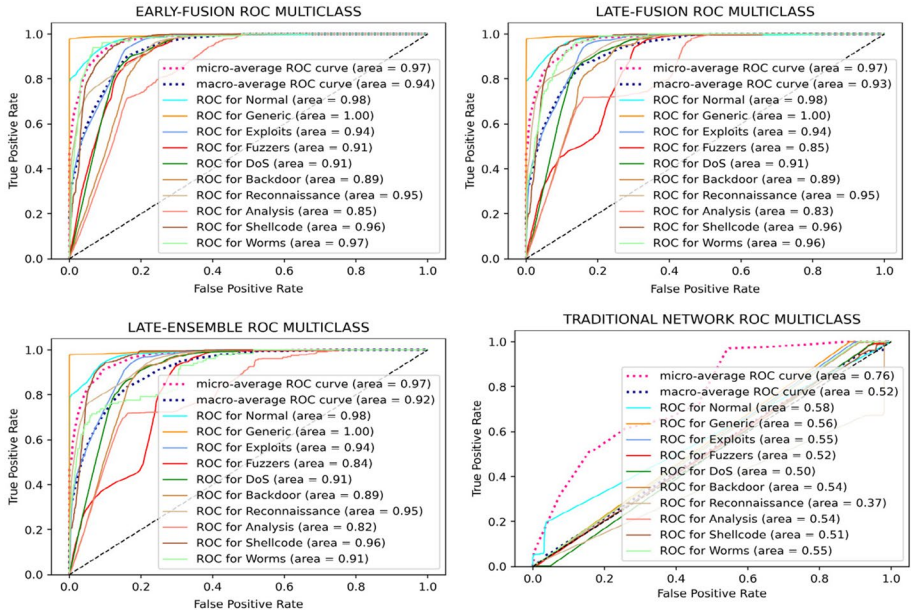

Fig. 11 The ROC curves of the four special versions of our deep learning models as a part of our ablation study

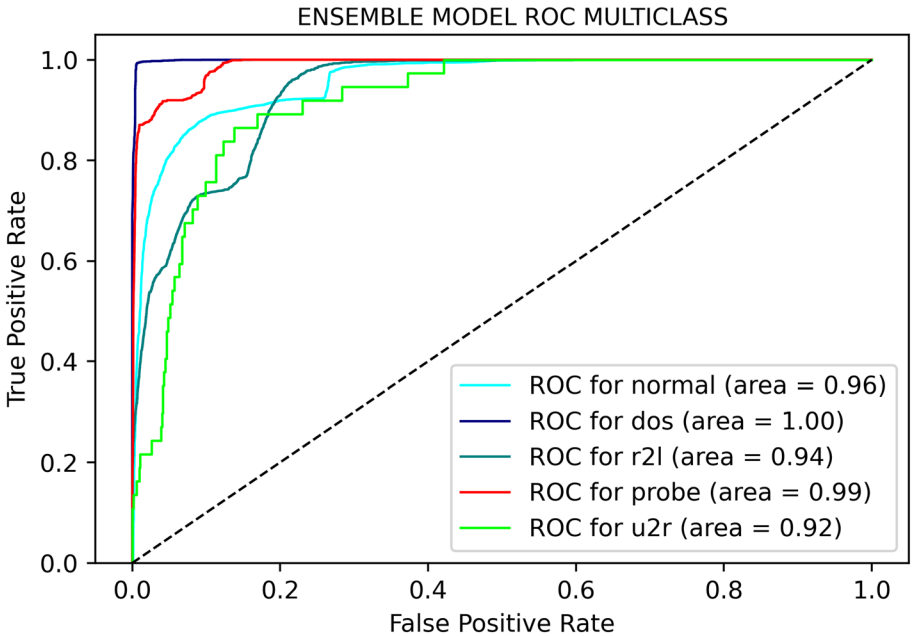

Fig. 12 The ROC curves of the late-ensemble model on the NSL-KDD dataset

environment. Generating or capturing and labelling training data for the attack types is therefore an issue. Further investigation is required to establish the degree of generalisation that is occurring, e.g. whether the classifier is merely learning attack signatures, is successfully generalising to discover more abstract characteristics of the various attack types used to create a training set, or at a higher level is discovering

some inherent difference between normal and malicious traffic. This final eventuality would greatly reduce the need for context/environment-specific training.

Our view is that a multi-way classifier is complementary to the existing mechanisms, and is best seen as a fourth element in a hybrid system. Hybrid schemes need to combine the outputs of their constituent mechanisms in some way, taking into account their strengths and weaknesses, in order to decide whether to issue an alert. As far as possible, both false positives and false negatives should be minimised, but a degree of trade-off is inevitable. In a NIDS context, a false positive is an irritation and waste of time, whereas a false negative is potentially catastrophic. Therefore, one needs to be very confident before rejecting a positive signal from any mechanism. On the other hand, a naïve conservative approach whereby an alert is generated if any single mechanism reports a positive is likely to overwhelm SOC analysts. It is difficult to say what levels are acceptable, but the vast majority of attacks should be detected, (recall ≈ 1), and sources suggest that a real alarm rate (precision) of between 60% and 90% is achievable.

Conclusions

This paper proposes novel deep learning architectures using deep fusion mechanisms called early-fusion, late-fusion, and late-ensemble deep learning models. Our feature fusion mechanisms have been designed to encourage the deep learning model to capture the relationships between the features. The late-fusion and late-ensemble models have shown better performance than the early-fusion model and other state-of-the-art models due to their ability to learn the relationships between more specialised features. Our deep learning architectures based on feature fusion provide a generic solution that can be extended to other deep learning models such as LSTM or CNN. As a future development, one can study the effect of feature fusion with recurrent units to encode the long dependencies between the features. Another research direction is to employ post-hoc explainable AI models to better understand the behaviour of the proposed models in distinguishing between the different classes by highlighting the most significant attributes that are contributing to the final prediction.

Acknowledgements
At the time the work was performed, both MMA and PK were with Birmingham City University.

Author contributions
Conceptualisation, MMA.; investigation, MMA and AA; methodology, MMA; administration, MMA; software, AA, Ako. and XS; supervision, MMA; validation, XS, AK, AKo; visualisation, AA, AKo; analysis: XS and PK; writing—original draft, MMA; writing—review and editing, XS, PK, IV, FS and MMA. All authors have read and agreed to the published version of the manuscript.

Funding
This work was supported by funding from Innovate UK under the Knowledge Transfer Partnership 12328.

Availability of data and materials
In this work, we used two publicly available datasets (UNSW-NB15 and NSL-KDD datasets) to validate our proposed models. The UNSW-NB15 dataset is available at https://www.unsw.adfa.edu.au/unsw-canberracyber/cybersecurity/ADFA-NB15-Datasets/. The NSL-KDD is available at https://www.unb.ca/cic/datasets/nsl.html. The code developed for our deep learning models is available at https://github.com/deen-g/NIDS-MODELS.

Declarations

Competing interests
The authors declare that they have no competing interests.

Received: 26 November 2022 Accepted: 3 October 2023
Published online: 08 November 2023

References

1. Prasad P, Rich C. Market guide for AIOps platforms; 2018. https://tekwurx.com/wp-content/uploads/2019/05/Gartner-Market-Guide-for-AIOps-Platforms-Nov-18.pdf. Retrieved 12 Mar 2020.
2. Latha KM. Learn about intrusion detection and prevention. USA: Juniper Networks; 2016.
3. LeCun Y, Bengio Y, Hinton G. Deep learning. Nature. 2015;521(7553):436–44.
4. Ramachandram D, Taylor GW. Deep multimodal learning: a survey on recent advances and trends. IEEE Signal Process Mag. 2017;34(6):96–108.
5. Moustafa N, Slay J. Unsw-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In: Military communications and information systems (MilCIS). Canberra: IEEE; 2015. p. 6.
6. Moustafa N, Slay J. The evaluation of network anomaly detection systems: statistical analysis of the unsw-nb15 data set and the comparison with the kdd99 data set. Inform Sec J Glob Perspect. 2016;25(1–3):18–31.
7. Tavallaee M, Bagheri E, Lu W, Ghorbani AA. A detailed analysis of the kdd cup 99 data set. In: 2009 IEEE symposium on computational intelligence for security and defense applications. Ottawa: IEEE; 2009. p. 6.
8. Belavagi MC, Muniyal B. Performance evaluation of supervised machine learning algorithms for intrusion detection. Proc Comp Sci. 2016;89:117–23.
9. Sarker IH, Kayes A, Badsha S, Alqahtani H, Watters P, Ng A. Cybersecurity data science: an overview from machine learning perspective. J Big Data. 2020;7(1):1–29.
10. Shaukat K, Luo S, Chen S, Liu D. Cyber threat detection using machine learning techniques: A performance evaluation perspective. In: 2020 International conference on cyber warfare and security (ICCWS). Islamabad: IEEE; 2020. p. 6.
11. Soheily-Khah S, Marteau P-F, Béchet N. Intrusion detection in network systems through hybrid supervised and unsupervised machine learning process: a case study on the ISCX dataset. In: 2018 1st International conference on data intelligence and security (ICDIS). South Padre Island: IEEE; 2018. pp. 19–226.
12. Mok MS, Sohn SY, Ju YH. Random effects logistic regression model for anomaly detection. Exp Syst Appl. 2010;37(10):7162–6.
13. Bagui S, Kalaimannan E, Bagui S, Nandi D, Pinto A. Using machine learning techniques to identify rare cyber-attacks on the UNSW-NB15 dataset. Sec Priv. 2019;2(6):91.
14. Liao Y, Vemuri VR. Use of k-nearest neighbor classifier for intrusion detection. Comp Sec. 2002;21(5):439–48.
15. Al-Yaseen WL, Othman ZA, Nazri MZA. Multi-level hybrid support vector machine and extreme learning machine based on modified k-means for intrusion detection system. Exp Syst Appl. 2017;67:296–303.
16. Horng S-J, Su M-Y, Chen Y-H, Kao T-W, Chen R-J, Lai J-L, Perkasa CD. A novel intrusion detection system based on hierarchical clustering and support vector machines. Exp Syst Appl. 2011;38(1):306–13.
17. Janarthanan T, Zargari S. Feature selection in UNSW-NB15 and KDDCUP'99 datasets. In: 2017 IEEE 26th International symposium on industrial electronics (ISIE). Edinburgh: IEEE; 2017. pp. 1881–1886.
18. Kasongo SM, Sun Y. Performance analysis of intrusion detection systems using a feature selection method on the UNSW-NB15 dataset. J Big Data. 2020;7(1):1–20.
19. Kumar V, Sinha D, Das AK, Pandey SC, Goswami RT. An integrated rule based intrusion detection system: analysis on UNSW-NB15 data set and the real time online dataset. Clust Comp. 2020;23(2):1397–418.
20. Moustafa N, Slay J. The significant features of the unsw-nb15 and the kdd99 data sets for network intrusion detection systems. In: 2015 4th International workshop on building analysis datasets and gathering experience returns for security (BADGERS). Kyoto: IEEE; 2015. pp. 25–31.
21. Xuren W, Famei H, Rongsheng X. Modeling intrusion detection system by discovering association rule in rough set theory framework. In: 2006 International conference on computational inteligence for modelling control and automation and international conference on intelligent agents web technologies and international commerce (CIMCA'06). Sydney: IEEE; 2006. pp. 24–24.
22. Tsang C-H, Kwong S, Wang H. Genetic-fuzzy rule mining approach and evaluation of feature selection techniques for anomaly intrusion detection. Pattern Recognit. 2007;40(9):2373–91.
23. Yao H, Fu D, Zhang P, Li M, Liu Y. Msml: a novel multilevel semi-supervised machine learning framework for intrusion detection system. IEEE Internet Things J. 2018;6(2):1949–59.
24. Louk MHL, Tama BA. Dual-ids: a bagging-based gradient boosting decision tree model for network anomaly intrusion detection system. Expert Syst Appl. 2023;213: 119030. https://doi.org/10.1016/j.eswa.2022.119030.
25. Tama BA, Lim S. Ensemble learning for intrusion detection systems: a systematic mapping study and cross-benchmark evaluation. Comp Sci Rev. 2021;39: 100357. https://doi.org/10.1016/j.cosrev.2020.100357.
26. Ieracitano C, Adeel A, Morabito FC, Hussain A. A novel statistical analysis and autoencoder driven intelligent intrusion detection approach. Neurocomputing. 2020;387:51–62.
27. Vinayakumar R, Soman KP, Poornachandran Prabaharan, Akarsh S. Application of deep learning architectures for cyber security. In: Hassanien A, Elhoseny M, editors. Cybersecurity and secure information systems. Advanced sciences and technologies for security applications. Cham: Springer; 2019. p. 125–60.
28. Choi Y-H, Liu P, Shang Z, Wang H, Wang Z, Zhang L, Zhou J, Zou Q. Using deep learning to solve computer security challenges: a survey. Cybersecurity. 2020;3(1):1–32.
29. Javaid A, Niyaz Q, Sun W, Alam M. A deep learning approach for network intrusion detection system. EAI Endorsed Transact Sec Saf. 2016;3(9):2.
30. Alrawashdeh K, Purdy C. Toward an online anomaly intrusion detection system based on deep learning. In: 2016 15th IEEE International conference on machine learning and applications (ICMLA). Anaheim: IEEE; 2016. pp. 195–200.

31. Potluri S, Ahmed S, Diedrich C. Convolutional neural networks for multi-class intrusion detection system. In: International conference on mining intelligence and knowledge exploration. Cham: Springer; 2018. pp. 225–238.
32. Shone N, Ngoc TN, Phai VD, Shi Q. A deep learning approach to network intrusion detection. IEEE Transact Emerg Top Comput Intell. 2018;2(1):41–50.
33. Vinayakumar R, Alazab M, Soman K, Poornachandran P, Al-Nemrat A, Venkatraman S. Deep learning approach for intelligent intrusion detection system. IEEE Access. 2019;7:41525–50.
34. Altwaijry N, ALQahtani A, AlTuraiki I. A deep learning approach for anomaly-based network intrusion detection. In: Big data and security: first international conference, ICBDS 2019, Nanjing, China, December 20–22, 2019, revised selected papers 1. Singapore: Springer; 2020. pp. 603–615 .
35. Al-Turaiki I, Altwaijry N. A convolutional neural network for improved anomaly-based network intrusion detection. Big Data. 2021;9(3):233–52.
36. Yin Y, Jang-Jaccard J, Xu W, Singh A, Zhu J, Sabrina F, Kwak J. IGRF-RFE: a hybrid feature selection method for MLP-based network intrusion detection on UNSW-NB15 dataset. J Big Data. 2023;10(1):1–26.
37. Salim S, Lahcen O. Accuracy improvement of network intrusion detection system using bidirectional long-short term memory (bi-lstm). In: Digital technologies and applications: proceedings of ICDTA'23, Fez, Morocco. Cham: Springer; 2023. pp. 143–152.
38. Sokolova M, Lapalme G. A systematic analysis of performance measures for classification tasks. Inform Proc Manag. 2009;45(4):427–37.
39. Papamartzivanos D, Mármol FG, Kambourakis G. Dendron: genetic trees driven rule induction for network intrusion detection systems. Future Gener Comp Syst. 2018;79:558–74.

Publisher's Note