scientific reports

OPEN

An intrusion detection model based on Convolutional Kolmogorov-Arnold Networks

Zhen Wang[1,2], Anazida Zainal[2], Maheyzah Md Siraj[2], Fuad A. Ghaleb[3], Xue Hao[2] & Shaoyong Han[4]✉

The application of artificial neural networks (ANNs) can be found in numerous fields, including image and speech recognition, natural language processing, and autonomous vehicles. As well, intrusion detection, the subject of this paper, relies heavily on it. Different intrusion detection models have been constructed using ANNs. While ANNs are relatively mature to construct intrusion detection models, some challenges remain. Among the most notorious of these are the bloated models caused by the large number of parameters, and the non-interpretability of the models. Our paper presents Convolutional Kolmogorov-Arnold Networks (CKANs), which are designed to overcome these difficulties and provide an interpretable and accurate intrusion detection model. Kolmogorov-Arnold Networks (KANs) are developed from the Kolmogorov-Arnold representation theorem. Meanwhile, CKAN incorporates a convolutional computational mechanism based on KAN. The model proposed in this paper is constructed by incorporating attention mechanisms into CKAN's computational logic. The datasets CICIoT2023 and CICIoMT2024 were used for model training and validation. From the results of evaluating the performance indicators of the experiments, the intrusion detection model constructed based on CKANs has an attractive application prospect. As compared with other methods, the model can predict a much higher level of accuracy with significantly fewer parameters. However, it is not superior in terms of memory usage, execution speed and energy consumption.

Keywords Kolmogorov-Arnold Networks, Convolutional neural network, Intrusion detection, Deep learning, Artificial intelligence

In modern society, internet technology has become an integral part of almost every aspect of daily life. With access to the Internet, you can video chat with loved ones and friends thousands of miles away, shop online, and access information at your fingertips. One of the high-profile application areas is the Internet of Things (IoT), where almost all electronic devices can be integrated through information technology[1]. IoT applications range from home and industrial automation[2,3]to smart cities[4]and connected cars[5]. IoT devices can be used to collect data, monitor and control physical assets, and make decisions[6]. IoT has the potential to revolutionize many industries and have a significant impact on the global economy[7]. However, while these technologies bring convenience and better experiences, they also have corresponding pitfalls. The sheer size of the population using these technologies has led some unscrupulous individuals to engage in profit-taking in defiance of legal and ethical constraints. Recently, Internet security incidents have become more frequent[8], and the trend is expected to continue. Consequently, individuals, corporations and even nations will face unpredictable challenges. As a result, it is extremely important to build an effective protection system.

When building protection mechanisms, the current mainstream solution is to combine deep learning algorithms to design intrusion detection system (IDS). An innovative system for detecting intrusions in IoT networks using convolutional neural networks (CNNs) is proposed by the authors[9]. Compared to other models, this model exhibits good performance. Attention mechanisms for model optimization are very popular. Using attention mechanisms on top of the CNN, the authors optimized the model for faster processing of detection samples without sacrificing accuracy[10]. Some authors have also combined the excellent properties of spiking neural networks and CNNs for intrusion detection[11]. From the experimental results, it can be seen that the model is much less resource-intensive than the other models in terms of computational resource usage and

[1]School of Data Science and Artificial Intelligence, Wenzhou University of Technology, Wenzhou 325035, Zhejiang, China. [2]Faculty of Computing, Universiti Teknologi Malaysia, Johor Bahru, Johor 81310, Malaysia. [3]College of Computing and Digital Technology, Birmingham City University, Birmingham B4 7XG, UK. [4]School of Mathematics and Computer Science, Tongling University, Tongling 244061, China. ✉email: hanshaoyong@zuaa.zju.edu.cn

energy consumption. While doing this, it maintains the same level of detection accuracy. Recurrent neural networks have also been used by some researchers to develop detection models[12]. Experimental results show that the model exhibits better results compared to existing methods. There are many similar intrusion detection models built on ANNs[13–15]. Despite the popularity of ANNs for intrusion detection model building, it is still undeniable that they suffer from deficiencies in interpretability. And these models are usually constructed with a large number of processing layers to achieve sufficient accuracy.

Using the Kolmogorov-Arnold representation theorem as inspiration, the authors proposed KANs[16]. A KAN with a much smaller size can provide similar or better accuracy relative to a multi-layer perceptron (MLP) of a much larger size for the purpose of fitting data and solving partial differential equations (PDEs). The KANs were designed with easy interpretability in mind. The visualizations of KANs were intuitive, and the interaction with humans was easy[16]. The choice of KANs for this paper was based on these excellent qualities. Convolutional computing is also a much-needed capability for detection models due to its powerful feature processing. It is these advantages that motivate this paper to propose an intrusion detection model based on CKANs. Here are the main contributions to the framework designed in this paper:

- A novel data preprocessing procedure is proposed, where sample balancing and data normalization can be performed more rationally. A method for organizing features is proposed that augments significant features at the data level.
- Using the Kolmogorov-Arnold representation theorem, a new intrusion detection model is developed for the first time. The model's interpretability and accuracy can be enhanced. An intrusion detection model based on CKANs is designed and implemented, and attention mechanisms are employed to enhance the model's performance.
- The models were evaluated on the datasets CICIoT2023 and CICIoMT2024 for various accuracy metrics. The experimental results show that the model proposed in this paper can do a better job than other models. The computational resource requirements and energy consumption of the model are evaluated. The occurrence of these situations is also analyzed.

Throughout the rest of this paper, the following sections will be discussed. The work related to this topic is described in Sect. 2. A description of our proposed solution can be found in Sect. 3. A discussion of the results of the adopted models is provided in Sect. 4. An overview of the results of the experiments is presented in Sect. 5 along with suggestions for future research directions.

Related work

In full swing, research and development of learning models based on the KAN framework are being carried out. Researchers expect this new learning framework to replace classical MLPs and improve performance. The learning framework has been demonstrated to be feasible and advanced in a number of studies.

The KAN framework has been optimized by combining it with other techniques in several studies. Introduced B-splines and radial basis functions KAN (BSRBF-KAN), a KAN that combines B-splines and radial basis functions (RBFs) to fit input vectors in data training[17]. The authors found that BSRBF-KAN showed stability in 5 training sessions with competitive average accuracy. Deep operator KAN (DeepOKAN) is a new variant of neural operators that uses KAN architectures instead of CNNs[18]. When compared to MLP-based deep operator networks (DeepOnets), DeepOKANs achieve comparable accuracy with fewer parameters. A novel architecture, the fractional KAN (fKAN), is presented in the paper, which combines the distinctive features of KANs with a trainable adaptation of a fractional-orthogonal Jacobi function[19]. Investigating rational functions as a new basis function. The authors proposed two different approaches based on Pad´e approximation and rational Jacobi functions as trainable basis functions, establishing the rational KAN (rKAN) [20]. According to that paper, smooth, structurally informed KANs can reach equivalence to MLPs in specific classes of functions by incorporating smoothness [21]. Authors introduced wavelet kolmogorov-arnold network (Wav-KAN), a novel neural network architecture that utilizes the wavelet KAN framework to improve performance and interpretability [22]. In addition to enhancing accuracy, Wav-KAN provides faster training speeds and increased robustness due to its ability to adapt to the data structure in the paper.

Several studies have made improvements based on graph computing. The Fourier KAN graph collaborative filtering (Fourier KAN-GCF) recommendation model is a simple and efficient graph-based recommendation model[23]. This model is based on a novel Fourier KAN that replaces the MLP in graph convolution networks (GCN) during feature transformations. As a result, graph collaborative filtering (GCF) represents better data and is more straightforward to train by using a Fourier KAN as part of feature transformation during message passing. The authors presented the Graph Kolmogorov-Arnold Networks (GKAN), a novel neural network architecture extending the theory of the recently proposed KAN to graph-structured data[24]. As opposed to classic GCNs which are based on static convolutional architectures, GKANs employ learnable spline-based functions between layers, transforming the way data is processed across graph layers. By using a real-world dataset (Cora), GKAN is experimentally evaluated using a semi-supervised graph learning problem. In general, architecture provides better performance. A comparison was made between KANs and MLPs when it came to graph learning tasks[25]. The authors conducted comprehensive experiments on node labeling, graph analysis, and graph regression studies. Based on the experimental results, KANs are comparable to MLPs in classification tasks, but they are clearly superior in graph regression tasks.

Another research topic is time-series data analysis. The Kolmogorov-Arnold representation theorem inspired KANs, which use spline-parametrized univariate functions instead of linear weights, enable them to learn activation patterns dynamically. Their study demonstrates that KANs provide more accurate results with fewer learning parameters than conventional MLPs in satellite traffic forecasting[26]. In addition, the authors present a study of the impact of KAN-specific parameters on performance. As part of their exploration of KAN for time series prediction, they proposed two methods: temporal KAN (T-KAN) and multivariate temporal KAN (MT-KAN)[27]. Through symbolic regression, T-KAN can explain the nonlinear relationship between predictions and previous time steps, allowing it to be highly interpretable in dynamically changing environments due to its ability to detect concept drift within time series. On the other hand, MT-KAN is effective in improving prediction performance through the discovery and utilization of complex relationships among variables in multivariate time series. These approaches are validated by experiments that demonstrate their effectiveness in time series forecasting tasks by significantly outperforming traditional methods. As a result of their research, they developed a temporal KANs (TKANs), a neural network architecture inspired by KANs and long short-term memory (LSTM)[28]. In TKANs, recurring KAN layers (RKANs) are embedded in memory management, combining the strengths of both networks. This innovation allows us to forecast multiple time series with improved accuracy and efficiency.

In addition to these applications, several other directions are also being investigated. KAN was integrated with several pre-trained CNN schemes for remote sensing (RS) image classification tasks using the EuroSAT dataset for the first time, demonstrating a high level of accuracy[29]. KAN has been used to predict the pressure and flow rate of flexible electrohydrodynamic pumps[30]. The authors evaluated the KAN model against random forest (RF), and MLP models using a dataset of flexible electrohydrodynamic pump parameters. In the experimental results, KAN has been demonstrated to be exceptionally accurate and interpretable, making it an appropriate alternative to electrohydrodynamic pumping predictive modeling. In their study, the authors propose a different PDE form using KAN rather than MLP, known as Kolmogorov-Arnold-Informed Neural Networks (KINNs)[31]. They compare MLP with KAN in several numerical PDE examples. For a number of PDEs in computational solid mechanics, KINN exceeds MLP in terms of accuracy and convergence speed. According to the enthusiasm of the researchers and KAN's excellent performance in different applications, more research results will be made available for learning and application in the near future.

In the field of intrusion detection, there are also many excellent traditional deep learning models available. An approach based on hybrid learning is proposed to identify malicious traffic utilizing a lightweight, two-stage scheme[32]. The domain name system (DNS) is well protected in this way. A high-performance machine learning-based monitoring system for detecting malicious uniform resource locators (URLs) is presented in the paper[33]. A two-layer detection system is proposed in the proposal. In both binary and multi-class classification, it is superior. In this article, they propose a novel method for designing a smart IDS using software-defined networking (SDN) and deep learning[34]. This approach considers the SDN framework as a promising option that enables reconfiguration of static network infrastructure and separates the control plane from the data plane in smart consumer electronics networks. They propose a method for detecting and classifying network activity in an IoT system using predictive machine learning[35]. An evaluation of five supervised learning models was conducted to analyze their impact on the detection and classification of network activities in IoT systems. The results of their experiments indicate that their model is highly accurate in detecting anomalies. The article proposes a new method for detecting intrusions in IoT by stacking ensembles of deep learning models[36]. The model is evaluated on three open-source datasets, including binary and multi-class classification, and its results are compared with those of other standard machine learning methods. In experimental studies, it has demonstrated a high level of accuracy and a low false positive rate (FPR). These deep learning models based on traditional architectures have played a profound role in advancing the field. Table 1 summarizes these relevant research efforts.

Proposed scheme

This section describes the main elements of this study, including the data pre-processing process, the model construction process, and the internal mechanisms specific to the proposed model. A diagram depicting the overall framework of this study is shown in Fig. 1.

Data pre-processing

The datasets used in this study are CICIoT2023[37]and CICIoMT2024[38]. Even the current state-of-the-art software and hardware can't guarantee that the data collected is completely correct, so the datasets need to be cleaned first to remove invalid data. Remove data records with null values, infinite values, and negative values (when they should be positive). Following the completion of this step, all data records left behind are valid. The balancing of the data samples was then carried out. There are often large differences in the number of samples of different types of data in the collected dataset. In order to avoid unfairness for data types with small samples, the sample set will be quantitatively balanced. In this paper, each type of data is clustered using the K-Means algorithm. The sample point at the center of the cluster is used as representative sample in the set of samples for that cluster. A total of 10,000 representative samples were collected in this manner for each type of data.

Data normalization operations are then performed on these selected samples. Due to their uneven distribution when analyzing the features, these values must be further processed prior to normalization. Figure 2shows the result after one field was analyzed using Isolation Forest (iForest)[39]. As can be seen from the blue area on the far left, this represents the normal data points obtained from the analysis, which constitutes only a small part of the range of values. However, the entire range of values expands dramatically due to a small number of values. In the area of the anomaly, the background color indicates the density of outliers within that region. In the iForest analysis, the proportion of outliers was set to 0.1. As can be seen from the figure, the vast majority of the sample points are concentrated in a very small area; whereas a small percentage of the samples are spread over a large area. Normalizing these samples directly on a proportional basis would result in a decrease in the distinguishability of most sample points. Because their values will be squeezed together. Without these widely varying values, most sample points could be normalized to give more discriminatory results. Because this is when the original small scope becomes global, so that the differences between features can be brought out as

Paper	Year	Models	Datasets	Advantage
Hoang [17]	2024	BSRBF-KAN	MNIST, Fashion-MNIST	Stability, convergence, and accuracy
Diab [18]	2024	DeepOKAN	Simulated	Few parameters required, high accuracy
Alireza [19]	2024	fKAN	MNIST	Higher accuracy
Alireza [20]	2024	rKAN	MNIST	Efficacy and practicality of function approximation
Moein [21]	2024	KAN	None	Developing and improving computational biomedicine models
Zavareh [22]	2024	Wav-KAN	MNIST	Faster training, higher accuracy, and greater robustness
Jinfeng [23]	2024	FourierKAN-GCF	MOOC, Games	Lower training difficulty, higher performance
Mehrdad [24]	2024	GKAN	Cora	Higher accuracy
Roman [25]	2024	KAGNN	Cora, Citeseer, Ogbn-arxiv, Cornell, Texas, Wisconsin, Actor	A clear advantage in graph regression tasks
Cristian [26]	2024	KAN	Simulated	With fewer learning parameters, it produces more accurate results
Kunpeng [27]	2024	T-KAN, MT-KAN	A financial dataset	Improved predictability and interpretability
Rémi [28]	2024	TKANs	Self-collected	Accurate and efficient multi-step forecasting of time series
Minjong [29]	2024	KAN	EuroSAT	Satisfactory accuracy
Yanhong [30]	2024	KAN	A dataset of flexible EHD pump parameters	Excellent accuracy and interpretability
Yizheng [31]	2024	KINN	None	Significantly improved accuracy and convergence rate
Qasem [32]	2023	Ensemble ML models	CIRA-CIC- DoHBrw-2020	Lightweight and high performance
Qasem [33]	2023	Ensemble ML models	ISCX-URL2016	Better classification performance
Danish [34]	2023	Bidirectional Long Short-Term Memory	CICIDS-2018	Testing time and classification accuracy
Abdulaziz [35]	2022	Shallow neural networks, DT, NB, SVM, KNN	IoTID20	Higher accuracy
Riccardo [36]	2023	Stacked ML model	ToN_IoT, CICIDS2017 and SWaT	Accurate with very few false positives

Table 1. Related work.

much as possible under the existing conditions. Based on this analysis, the outlier value is set to the nearest inlier value and then normalized. The following equation illustrates the normalization operation:

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{1}$$

where X denotes the original value. X_{min} is the minimum value of the feature column and X_{max} is the maximum value of the feature column.

Next, feature selection is performed by a particle swarm algorithm. Features were selected using a regression XGBoost model fitted to the classification to which the samples belonged and evolved in the direction that minimized the root mean square error. The population size is 30 and the number of iterations is 100. Eventually the feature set with the highest score in the evaluation will be selected. In order to facilitate currently available deep learning models using the processed dataset, these features will be binary encoded and organized in the form of a 2D matrix. The selected features will be evaluated for information gain. Information gain can be calculated using the following formula:

$$IG\left(S,A\right) = H\left(S\right) - \sum\nolimits_{t\in T} \frac{|S_t|}{|S|} H\left(S_t\right) \tag{2}$$

where, $IG(S, A)$ denotes the information gain of dataset S with respect to feature A. t is the subset partitioned according to feature A. $|S_t|$ is the number of samples in the subset S_t. And $|S|$ is the total number of samples in the original dataset S. $H\left(S_t\right)$ is the information entropy of the subset S_t. Which is defined in Eq. 3.

$$H\left(X\right) = \sum\nolimits_i P\left(x_i\right) I\left(x_i\right) = -\sum\nolimits_i P\left(x_i\right) log_b P\left(x_i\right) \tag{3}$$

where b is a constant and x_i is a sample point in a finite sample set. The few features with the highest information gain will have two chances to appear in the final feature representation. Make key information in the feature set more prominent by replicating it in a way that increases its visibility. So, it ends up being 36 features, each assembled into a 4*4 matrix, the features are a 6*6 matrix, and the resulting sample is a 24*24 matrix.

Kolmogorov–Arnold Networks

In the work of Vladimir Arnold and Andrey Kolmogorov, they showed that a multivariate continuous function on a bounded domain can be described as a finite synthesis of continuous functions of a one variable plus the binary operation of addition. Specifically, this can be expressed in the following equation:

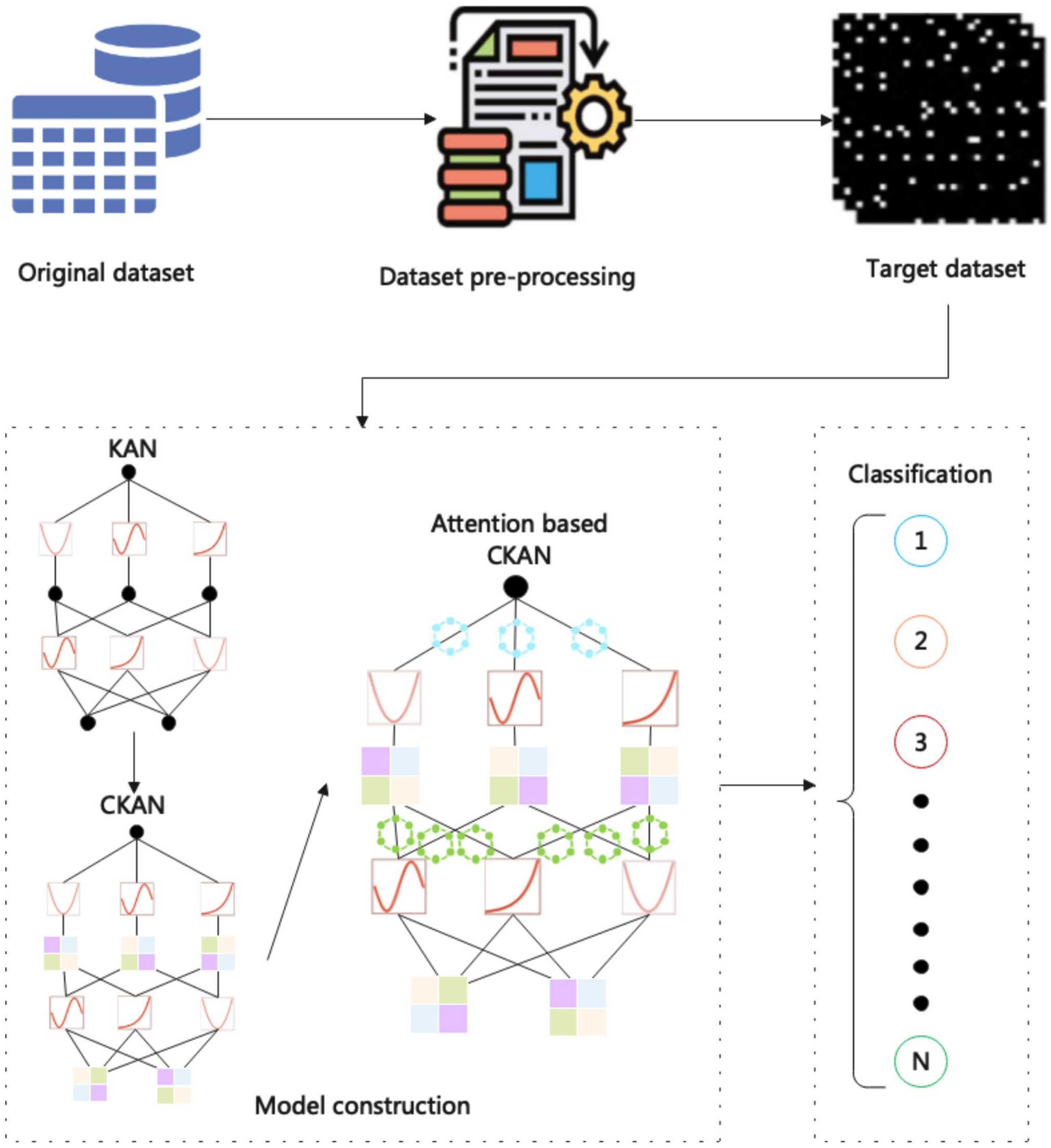

Fig. 1. Overall Framework.

$$f\left(X\right)=f\left(x_1,x_2,\ \ldots\ ,\ x_n\right)=\sum\nolimits_{q=1}^{2n+1}\varnothing_{\,q}\left(\sum\nolimits_{p=1}^{n}\phi_{\,q,p}\left(x_p\right)\right) \tag{4}$$

where $\phi_{\,q,p}:[0,1]\to\mathbb{R}$ and $\varnothing_{\,q}:\mathbb{R}\to\mathbb{R}$. Each layer of the KAN is composed of these learnable one-dimensional functions:

$$\varnothing\ =\left\{\phi_{\,q,p}\right\},\qquad p=1,2,\ldots\ ,\ n_{in},\qquad q=1,2,\ldots\ ,n_{out} \tag{5}$$

Each function $\phi_{\,q,p}$ is rendered as a B-spline. B-spline functions are segmented continuous polynomial functions that are continuous and finitely derivable over the whole curve. Through the use of node vectors and basis functions, the B-spline curve is a derivation of the Bézier Curve that allows finer control over the shape of

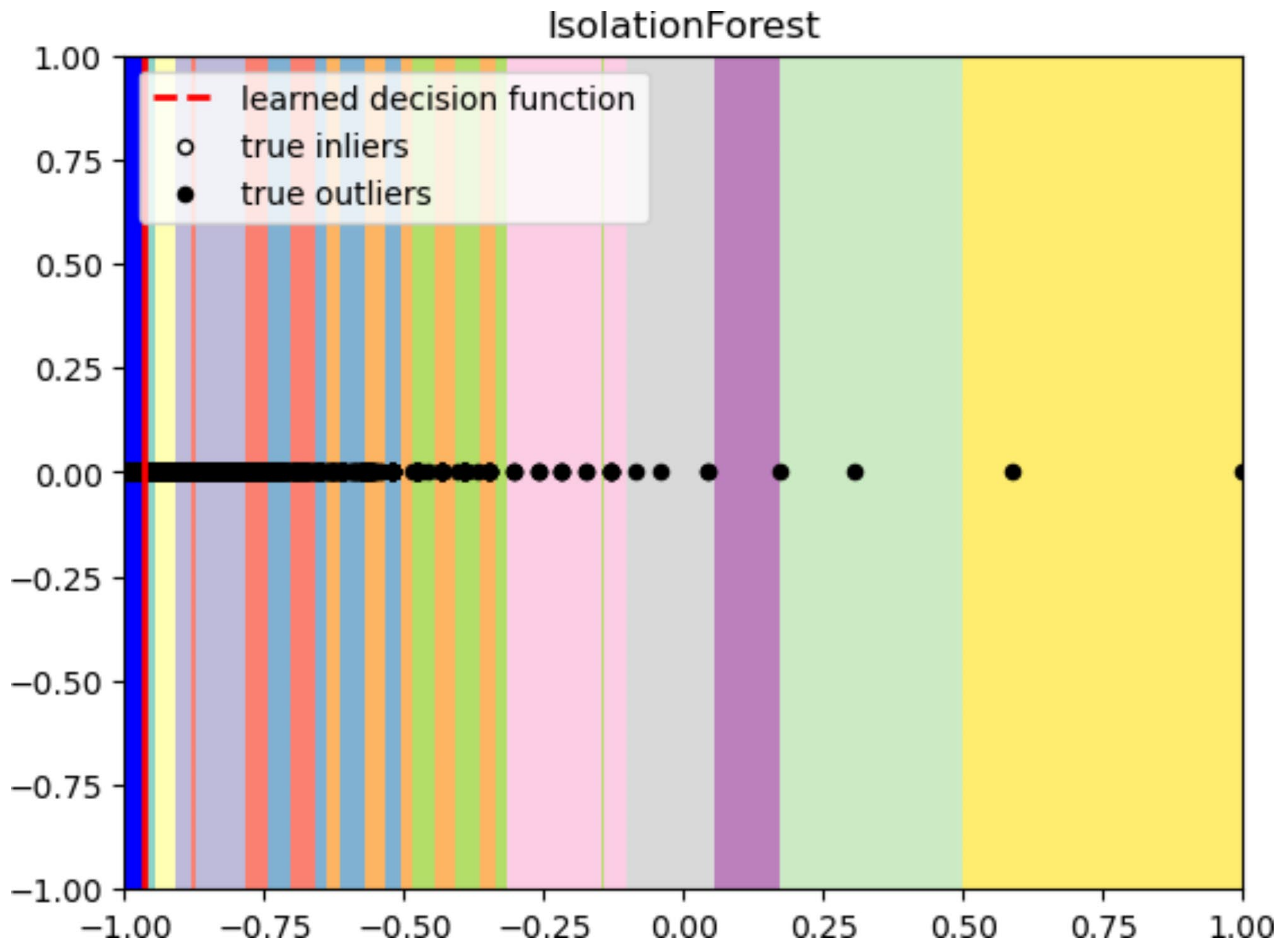

Fig. 2. Isolation Forest Analysis.

the curve. B-spline is a type of spline function created by a linear combination of base splines. That can effectively improve the network's ability to represent complex data. n_{in} refers to a layer's input features. A layer's n_{out}, on the other hand, indicates its output features, which reflect dimensional transformations.

Using an integer array, a KAN's shape can be represented as follows:

$$[n_0,\ n_1,\ \ldots\ ,\ n_L] \tag{6}$$

Computation graph nodes at layer i are represented by n_i. An ith neuron in the lth layer is represented by $(l,\ i)$. And its activation value is indicated by $x_{l,i}$. The activation functions between layers l and $l+1$ are $n_l n_{l+1}$. The activation function for $(l,\ i)$ and $(l+1,\ j)$ is as follows:

$$\phi_{\,l,j,i}, \quad l = 0, \ldots, L-1, \quad i = 1, \ldots, n_l, \quad j = 1, \ldots, n_{l+1} \tag{7}$$

KANs have an overall structure like MLPs, which stack layers. However, rather than relying on simple linear transformations and nonlinear activations, it makes use of complex functional mappings.

$$KAN\,(x) = (\varnothing_{\,L-1} \circ \varnothing_{\,L-2} \circ \ldots \circ \varnothing_{\,0})\,(x) \tag{8}$$

The computational logic of the KAN network with L layers is shown in Eq. 6. Transform each layer's input, x_l, to get the next layer's input, x_{l+1}, as follows:

$$x_{l+1} = \varnothing_{\,l}\,(x_l) = \begin{pmatrix} \phi_{\,l,1,1}(\bullet) & \cdots & \phi_{\,l,1,n_l}(\bullet) \\ \vdots & \ddots & \vdots \\ \phi_{\,l,n_{l+1},1}(\bullet) & \cdots & \phi_{\,l,n_{l+1},n_l}(\bullet) \end{pmatrix} x_l \tag{9}$$

It can be stated that the activation function $\phi\ (x)$ is the sum of the basis function $b\,(x)$ and the spline function:

$$\boldsymbol{\phi}\ (\boldsymbol{x}) = \boldsymbol{\omega}\ _1\boldsymbol{b}\,(\boldsymbol{x}) + \boldsymbol{\omega}\ _2\boldsymbol{spline}\,(\boldsymbol{x}) \tag{10}$$

$\boldsymbol{\omega}\ _1$ and $\boldsymbol{\omega}\ _2$ are the weight parameters of the corresponding parts. Here set:

$$\boldsymbol{b}\,(\boldsymbol{x}) = \boldsymbol{silu}\,(\boldsymbol{x}) = \boldsymbol{x}/(1+\boldsymbol{e}^{-\boldsymbol{x}}) \tag{11}$$

The $spline\,(x)$ is a linear combination of B-splines. The learnable spline functions are:

$$spline\,(x) = \sum\ _ic_iB_i\,(x)\,, \qquad c_is\ are\ trainable\ coefficients \tag{12}$$

Attention based Conversational Kolmogorov–Arnold Networks

The Convolutional Kolmogorov-Arnold Network is similar to the CNN. It successfully integrates the advantages of KAN and the computational mechanisms of CNN. In comparison with other architectures, the CKAN has the advantage of requiring a relatively small number of parameters[40]. In KAN Convolutions, there is a significant difference between the kernel and that of CNN Convolutions. CNNs utilize weights, whereas Convolutional KANs use nonlinear functions that utilize B-splines to construct the kernels. A convolution kernel consists of the same elements as Eq. 10. Set K as the KAN convolutional kernel $\in\ \mathbb{R}^{\boldsymbol{N}\times\ \boldsymbol{M}}$. The KAN Convolution can be defined as follows:

$$(\boldsymbol{Matrix}{*}\boldsymbol{K})_{\boldsymbol{i},\boldsymbol{j}} = \sum\ _{\boldsymbol{k}=1}^{\boldsymbol{N}}\sum\ _{\boldsymbol{l}=1}^{\boldsymbol{M}}\boldsymbol{\phi}\ _{\boldsymbol{kl}}\,(\boldsymbol{e}_{\boldsymbol{i}+\boldsymbol{k},\ \boldsymbol{j}+\boldsymbol{l}}) \tag{13}$$

Suppose there is the following input matrix for which KAN convolution calculation needs to be performed.

$$Matrix = \begin{bmatrix} e_{11} & e_{12} & \cdots & e_{1j} \\ e_{21} & e_{22} & \cdots & e_{2j} \\ \vdots & \vdots & \ddots & \vdots \\ e_{i1} & e_{i2} & \cdots & e_{ij} \end{bmatrix} \tag{14}$$

If the kernel of the KAN convolution is 3*3:

$$KAN\ Convolution\ Kernel = \begin{bmatrix} \phi\ _{11} & \phi\ _{12} & \phi\ _{13} \\ \phi\ _{21} & \phi\ _{22} & \phi\ _{23} \\ \phi\ _{31} & \phi\ _{32} & \phi\ _{33} \end{bmatrix} \tag{15}$$

The result is shown below:

$$Matrix * KAN\ Convolution\ Kernel =$$
$$\begin{bmatrix} \phi\ _{11}\,(e_{11}) + \phi\ _{12}\,(e_{12}) + \ldots + \phi\ _{33}\,(e_{33}) & \cdots & \phi\ _{11}\left(e_{1(j-2)}\right) + \phi\ _{12}\left(e_{1(j-1)}\right) + \ldots + \phi\ _{33}\,(e_{3j}) \\ \phi\ _{11}\,(e_{21}) + \phi\ _{12}\,(e_{22}) + \ldots + \phi\ _{33}\,(e_{43}) & \cdots & \phi\ _{11}\left(e_{2(j-2)}\right) + \phi\ _{12}\left(e_{2(j-1)}\right) + \ldots + \phi\ _{33}\,(e_{4j}) \\ \vdots & \ddots & \vdots \\ \phi\ _{11}\left(e_{(i-2)1}\right) + \phi\ _{12}\left(e_{(i-2)2}\right) + \ldots + \phi\ _{33}\,(e_{i3}) & \cdots & \phi\ _{11}\left(e_{(i-2)(j-2)}\right) + \phi\ _{12}\left(e_{(i-2)(j-1)}\right) + \ldots + \phi\ _{33}\,(e_{ij}) \end{bmatrix} \tag{16}$$

The overall structure of the proposed model is shown in Fig. 3. The core innovation of the KAN framework is to place learnable activation functions on the edges. In contrast, traditional frameworks place them in nodes and fix them. It is with this approach that the model will be able to learn more complex functional relationships between data. The weight parameters have been replaced by parametric spline functions, which enhances the model's expressive potential. Having done this, it will be able to gain a deeper understanding of more detailed and complex information through deep learning. Compared to traditional deep learning techniques, the KAN framework is also more interpretable. KAN is a structured, easy-to-understand system that facilitates human-computer interaction. Consequently, scientists can gain a solid understanding of the inner workings of the model, and even participate directly in its optimization and discovery. Models can be guided by scientists so that the laws of mathematics and physics can be discovered or verified, thus facilitating the collaboration with scientists and artificial intelligence.

The execution logic of the attention mechanism is shown in algorithm 1.

Algorithm 1: Attention mechanism
Input: Tensor
Output: Tensor with added attention mechanism 1 max_pool $\leftarrow$ use maximum pooling to obtain global features for each channel
2 avg_pool $\leftarrow$ obtaining global features for each channel using average pooling 3 # Define MLP, where channel_in and channel_out of Conv2d are equal.
4 mlp $\leftarrow$ Sequential (Conv2d (channel_in, channel_out, 1, bias = False),
5 ReLU (),
6 Conv2d (channel_in, channel_out, 1, bias = False))

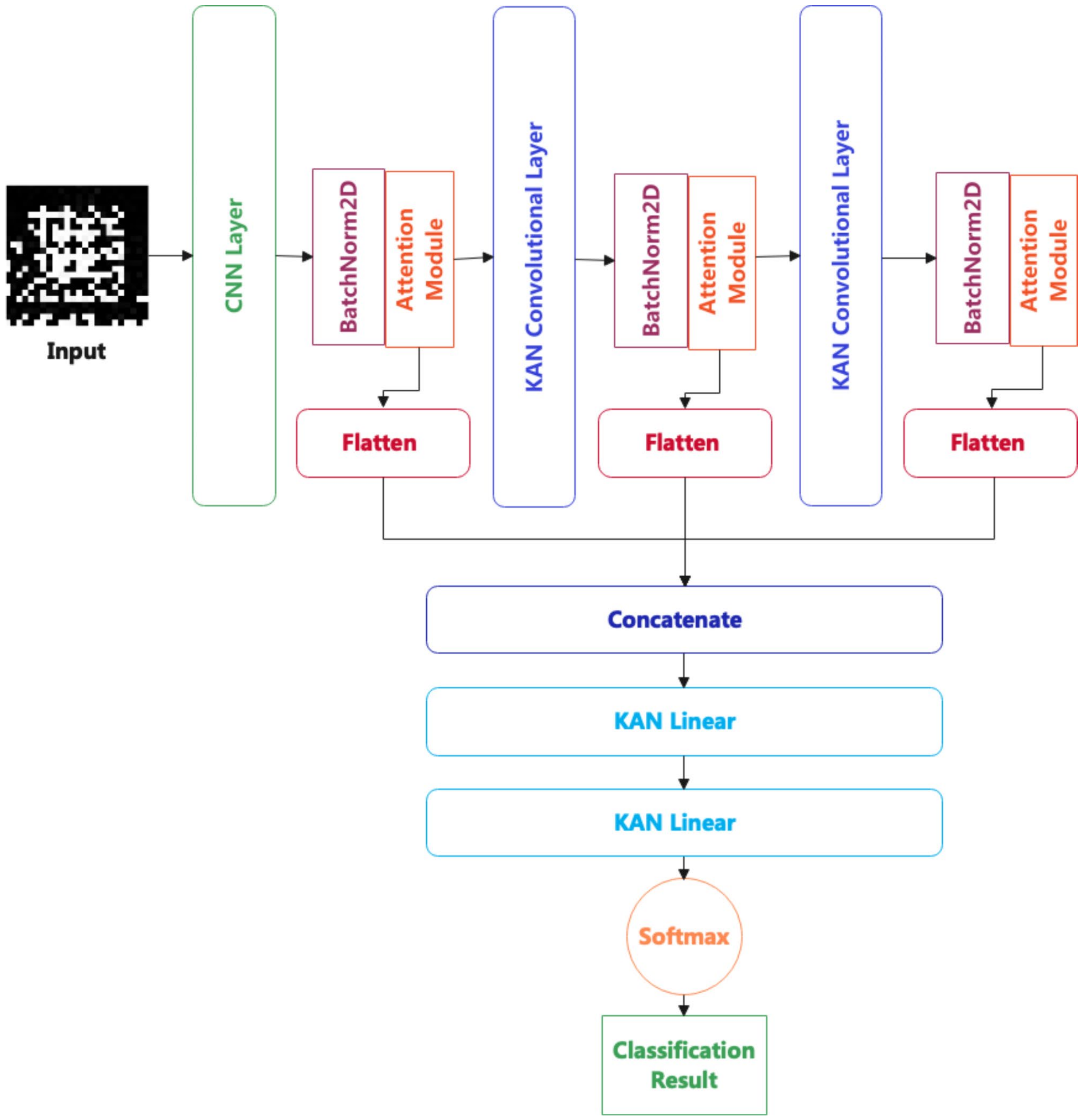

Fig. 3. The Architecture of Attention-Based Convolutional KAN.

Algorithm 1: Attention mechanism
7 conv $\leftarrow$ Conv2d (2, 1, kernel_size = 3, padding = 1, bias = False)
8 max_out $\leftarrow$ mlp(max_pool(input))
9 avg_out $\leftarrow$ mlp (avg_pool (input)) 10 channel_out $\leftarrow$ Sigmoid (max_out + avg_out) 11 out1 = channel_out * input 12 max_out $\leftarrow$ get the maximum value for each channel, along the channel dimension 13 avg_out $\leftarrow$ get the average value for each channel along the channel dimension 14 spatial_out $\leftarrow$ Sigmoid (conv (cat ([max_out, avg_out], dim = 1))) 15 out = spatial_out * out1

As follows is the definition of the ReLU function:

$$ReLU\,(x) = max(0,\ x) \tag{17}$$

And the sigmoid function:

$$Sigmoid\,(x) = \frac{1}{1+e^{-x}} \tag{18}$$

Algorithm 2 below demonstrates the computational logic of the loss function used in the proposal model training.

Algorithm 2: Loss function for the proposed model
Input: Number of classifications $\rightarrow$ n_class Sample prediction results $\rightarrow$ $predict$ Sample label $\rightarrow$ $target$
Output: Loss value between prediction and real label 1 correct $\leftarrow$ get the prediction result's probability of correct classification
2 predict $\leftarrow$ probability value of the current prediction classification 3 ids $\leftarrow$ rank the probability of the correct classification
4 $\alpha \leftarrow predict - correct$
5 $\beta \leftarrow 1 - correct$
6 loss $\leftarrow mean(n\_class * \alpha + (ids + 1) * \beta)$

Training a model uses the following strategy to adjust the learning rate:

$$\begin{cases} \zeta_1 = 0.001 \\ \zeta_i = 0.5 * \zeta_1 \left(1+\cos\left(\frac{i}{I_{\max}}\pi\right)\right), \quad 2 \le i \le I_{\max} \end{cases} \tag{19}$$

ζ_1 is the starting learning rate used for the first epoch of model training. In this situation, $I_{\max}$ represents the total number of epochs the model must undergo training and i represents the number of training rounds currently in progress.

Performance evaluation and discussion

This section focuses on experimenting with the proposal model. The experimental environment and evaluation metrics are presented, and the experimental results are compared and discussed with classical deep learning models widely used today.

Experimental environment

For model validation, the experimental environment is as follows:

- Operating system: Linux-5.15.120 + -x86_64-with-glibc2.31.
- CPU: Intel(R) Xeon(R) @ 2.20 GHz, 4 Core(s), 42.5 W.
- RAM: 32 GB, 11.76 W.

The GPU(s) used for model training:

- NVIDIA Tesla P100 (16 GB).

Evaluation metrics

A number of criteria were used for evaluating each model during the experiments, and the major evaluation criteria were as follows:

(1) The accuracy of detection, which is an indication of a model's basic capability. Evaluation indicators used in this study include:

$$\boldsymbol{Recall} = \frac{\boldsymbol{TP}}{\boldsymbol{TP}+\boldsymbol{FN}} \tag{20}$$

Dataset	Data types and their corresponding encodings							
	0	1	2	3	4	5	6	7
IoT-2023	Benign	Brute_Force	DDoS	DoS	Injection	Mirai	Recon	Spoofing
IoMT-2024	Benign	DDoS	DoS	MQTT	Recon	Spoofing	——	——

Table 2. Data types and encodings.

Model	Accuracy	
	IoT-2023	IoMT-2024
Alexnet	0.9486	0.9586
Resnet18	0.9491	0.9604
Resnet50	0.9490	0.9580
Mobilenet(V2)	0.9369	0.9499
Efficientnet	0.9431	0.9552
Densenet121	0.9495	0.9589
Googlenet	0.9407	0.9558
Shufflenet(V2)	0.9301	0.9490
Squeezenet	0.9451	0.9587
Spikformer	0.9312	0.9446
SpikingGCN	0.8044	0.9547
Proposed Model	**0.9529**	**0.9633**

Table 3. Classification accuracy.

$$Precision = \frac{TP}{TP + FP} \tag{21}$$

$$False\ Positive\ Rate = \frac{FP}{FP + TN} \tag{22}$$

$$F_1 - score = \frac{2 \times\ Precision \times\ True\ Positive\ Rate}{Precision + True\ Positive\ Rate} \tag{23}$$

TP stands for true positive. TN stands for true negative. FP and FN stand for false positives and false negatives, respectively.

(2) The complexity of a model is measured by three factors: the number of parameters included in it, the amount of computation required for each sample to be analyzed, and the amount of memory occupied by the model when training is complete.

(3) Execution speed, measured in terms of sample processing per second.

(4) Memory allocation of the model during sample processing.

(5) Energy consumption is determined by averaging the power consumption for 10,000 samples.

Experimental results and discussion

Different data types were encoded in the experimental session to facilitate the presentation of experimental results. The correspondence between specific data types and their encodings is shown in Table 2.

Table 3 shows the overall classification accuracy performance of all the models used in this study. Tables 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14 and 15 present detailed experimental results for each model, based on the classification of the data. Figures 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14 and 15 illustrate the confusion matrices corresponding to these models.

The comparative models in the experiments include nine commonly used classical models and two state-of-the-art models. These two models are Spikformer[41]and SpikingGCN[42], respectively. From the experimental results, it can be seen that the model proposed in this paper outperforms other models in terms of overall classification accuracy. This shows that the model has strong analysis capability of the data features. More detailed metrics also include recall, precision, f1-score and false positive rate. Recall means the proportion of samples predicted to be true out of all samples actually true. It is an effective way to assess the model's ability to find out the positive class of samples. Precision denotes the proportion of samples that are actually true out of all samples predicted to be true. It is a technique used to assess the quality of samples predicted to be positive by a model. The F1-score, on the other hand, is a reconciled average of the two, which is used for coordinated analysis. A false positive rate is calculated by dividing the number of false positive samples detected by the number of true negative samples. It can be used to assess the model's reliability and validity. The performance of these metrics, while slightly worse than other models in some classifications, remains dominant overall. The results indicate that the model proposed in the paper is superior across all accuracy metrics.

Category	Recall IoT-2023	Recall IoMT-2024	Precision IoT-2023	Precision IoMT-2024	F1-score IoT-2023	F1-score IoMT-2024	False Positive Rate IoT-2023	False Positive Rate IoMT-2024
0	0.970	0.960	0.960	0.953	0.965	0.956	0.006	0.009
1	0.865	0.993	0.895	0.987	0.880	0.990	0.015	0.003
2	0.975	0.983	0.965	0.994	0.970	0.988	0.005	0.001
3	0.960	0.953	0.984	0.973	0.972	0.963	0.002	0.005
4	0.912	0.935	0.879	0.957	0.895	0.946	0.018	0.008
5	0.987	0.926	0.985	0.891	0.986	0.908	0.002	0.023
6	0.954	——	0.964	——	0.959	——	0.005	——
7	0.967	——	0.961	——	0.964	——	0.005	——

Table 4. Alexnet verification.

Category	Recall IoT-2023	Recall IoMT-2024	Precision IoT-2023	Precision IoMT-2024	F1-score IoT-2023	F1-score IoMT-2024	False Positive Rate IoT-2023	False Positive Rate IoMT-2024
0	0.976	0.954	0.951	0.957	0.964	0.955	0.007	0.009
1	0.865	0.990	0.901	0.987	0.883	0.988	0.014	0.003
2	0.972	0.983	0.971	0.991	0.972	0.987	0.004	0.002
3	0.968	0.961	0.976	0.965	0.972	0.963	0.004	0.007
4	0.914	0.958	0.875	0.948	0.894	0.953	0.019	0.011
5	0.988	0.916	0.992	0.916	0.990	0.916	0.001	0.017
6	0.948	——	0.966	——	0.957	——	0.004	——
7	0.962	——	0.965	——	0.964	——	0.005	——

Table 5. Resnet18 verification.

Category	Recall IoT-2023	Recall IoMT-2024	Precision IoT-2023	Precision IoMT-2024	F1-score IoT-2023	F1-score IoMT-2024	False Positive Rate IoT-2023	False Positive Rate IoMT-2024
0	0.977	0.958	0.960	0.949	0.968	0.953	0.006	0.010
1	0.890	0.995	0.872	0.982	0.881	0.988	0.019	0.004
2	0.968	0.979	0.977	0.996	0.972	0.987	0.003	0.001
3	0.971	0.953	0.972	0.972	0.972	0.962	0.004	0.005
4	0.878	0.950	0.896	0.942	0.887	0.946	0.015	0.012
5	0.988	0.912	0.993	0.908	0.991	0.910	0.001	0.019
6	0.971	——	0.946	——	0.958	——	0.008	——
7	0.951	——	0.980	——	0.965	——	0.003	——

Table 6. Resnet50 verification.

The performance metrics exhibited by these models during execution are presented in Table 16. These include the number of parameters in the model, the number of floating-point operations to compute a single sample, and the size of the model when training is complete. In addition, the number and total amount of memory allocations required by the model during model validation were also counted. Data related to memory is derived from the tracemalloc Python library. These values are calculated from the system memory snapshot when the model processes a single sample. Energy consumption and sample processing speed were also quantified.

It can be seen from the experimental results that the proposed CKAN model can achieve classification accuracy superior to the other models. This is done while using only a much smaller number of parameters than the others. The size of the resulting model from the final training is also smaller than most models. However, in terms of memory allocation, both the number of allocations and the allocated memory space are greater than the other models. This also led to its poor performance in two subsequent metrics, power consumption and sample processing speed.

Category	Recall		Precision		F1-score		False Positive Rate	
	IoT-2023	IoMT-2024	IoT-2023	IoMT-2024	IoT-2023	IoMT-2024	IoT-2023	IoMT-2024
0	0.956	0.926	0.950	0.957	0.953	0.941	0.008	0.008
1	0.833	0.985	0.893	0.986	0.862	0.985	0.014	0.003
2	0.949	0.978	0.971	0.985	0.960	0.982	0.004	0.003
3	0.960	0.961	0.962	0.956	0.961	0.959	0.005	0.008
4	0.911	0.938	0.858	0.935	0.884	0.937	0.022	0.013
5	0.988	0.912	0.980	0.882	0.984	0.897	0.003	0.025
6	0.944	——	0.941	——	0.942	——	0.008	——
7	0.957	——	0.943	——	0.950	——	0.008	——

Table 7. Mobilenet(V2) verification.

Category	Recall		Precision		F1-score		False Positive Rate	
	IoT-2023	IoMT-2024	IoT-2023	IoMT-2024	IoT-2023	IoMT-2024	IoT-2023	IoMT-2024
0	0.969	0.946	0.957	0.953	0.963	0.950	0.006	0.009
1	0.904	0.989	0.838	0.984	0.870	0.987	0.025	0.003
2	0.962	0.981	0.975	0.989	0.969	0.985	0.004	0.002
3	0.969	0.951	0.973	0.968	0.971	0.960	0.004	0.006
4	0.833	0.946	0.907	0.939	0.868	0.942	0.012	0.012
5	0.990	0.917	0.989	0.899	0.989	0.908	0.002	0.021
6	0.953	——	0.952	——	0.952	——	0.007	——
7	0.966	——	0.961	——	0.964	——	0.005	——

Table 8. Efficientnet verification.

Category	Recall		Precision		F1-score		False Positive Rate	
	IoT-2023	IoMT-2024	IoT-2023	IoMT-2024	IoT-2023	IoMT-2024	IoT-2023	IoMT-2024
0	0.963	0.944	0.968	0.968	0.966	0.956	0.005	0.006
1	0.880	0.989	0.885	0.990	0.882	0.989	0.016	0.002
2	0.977	0.986	0.964	0.989	0.971	0.988	0.005	0.002
3	0.966	0.963	0.976	0.961	0.971	0.962	0.003	0.008
4	0.900	0.940	0.884	0.953	0.891	0.947	0.017	0.009
5	0.986	0.930	0.992	0.895	0.989	0.912	0.001	0.022
6	0.962	——	0.964	——	0.963	——	0.005	——
7	0.964	——	0.966	——	0.965	——	0.005	——

Table 9. Densenet121 verification.

The proposed model is significantly smaller than the other models both in terms of network depth and number of parameters. The calculation of the samples, however, requires more memory than other models. It follows that the availability of memory in the runtime environment will play a very significant role in the execution of the model. An efficient memory allocation strategy will enhance the efficiency of model execution and vice versa, it will become an execution bottleneck for the model. It can also be observed from the experimental results that very little memory used by the model is reused during the inference process. Therefore, it needs to perform memory allocation and memory write operations more frequently. Consequently, the model detects samples at a slower rate than other models. While other models have a larger number of parameters, they can usually manage their own parameters in memory more conveniently. This is mainly due to the fact that KAN's connection weights are computed from B-spline functions, rather than simple linear weights. There is no doubt that it will be more computationally complex. Currently, the proposed model is only suitable for use in scenarios with sufficient computational resources so that it can prove its own advantages in detection. If used in resource-

Category	Recall IoT-2023	Recall IoMT-2024	Precision IoT-2023	Precision IoMT-2024	F1-score IoT-2023	F1-score IoMT-2024	False Positive Rate IoT-2023	False Positive Rate IoMT-2024
0	0.962	0.954	0.962	0.954	0.962	0.954	0.006	0.009
1	0.838	0.994	0.889	0.985	0.863	0.989	0.015	0.003
2	0.977	0.981	0.947	0.992	0.961	0.987	0.008	0.002
3	0.948	0.967	0.976	0.945	0.962	0.955	0.003	0.011
4	0.895	0.945	0.873	0.944	0.884	0.945	0.019	0.011
5	0.980	0.894	0.997	0.913	0.988	0.903	0.000	0.017
6	0.969	——	0.919	——	0.944	——	0.012	——
7	0.960	——	0.963	——	0.962	——	0.005	——

Table 10. Googlenet verification.

Category	Recall IoT-2023	Recall IoMT-2024	Precision IoT-2023	Precision IoMT-2024	F1-score IoT-2023	F1-score IoMT-2024	False Positive Rate IoT-2023	False Positive Rate IoMT-2024
0	0.950	0.925	0.928	0.950	0.939	0.937	0.011	0.010
1	0.837	0.982	0.899	0.985	0.867	0.983	0.014	0.003
2	0.959	0.978	0.941	0.983	0.950	0.981	0.009	0.003
3	0.940	0.954	0.952	0.960	0.946	0.957	0.007	0.008
4	0.916	0.948	0.862	0.921	0.889	0.935	0.021	0.016
5	0.972	0.906	0.990	0.896	0.981	0.901	0.001	0.021
6	0.931	——	0.932	——	0.932	——	0.009	——
7	0.937	——	0.942	——	0.939	——	0.008	——

Table 11. Shufflenet(V2) verification.

Category	Recall IoT-2023	Recall IoMT-2024	Precision IoT-2023	Precision IoMT-2024	F1-score IoT-2023	F1-score IoMT-2024	False Positive Rate IoT-2023	False Positive Rate IoMT-2024
0	0.970	0.958	0.952	0.957	0.961	0.958	0.007	0.008
1	0.882	0.987	0.870	0.987	0.876	0.987	0.019	0.003
2	0.970	0.984	0.963	0.986	0.966	0.985	0.005	0.003
3	0.970	0.960	0.971	0.968	0.971	0.964	0.004	0.006
4	0.878	0.959	0.889	0.934	0.884	0.947	0.016	0.013
5	0.983	0.903	0.995	0.920	0.989	0.911	0.001	0.016
6	0.957	——	0.961	——	0.959	——	0.005	——
7	0.953	——	0.963	——	0.958	——	0.005	——

Table 12. Squeezenet verification.

constrained settings, its computational performance can seem slow and less efficient. Furthermore, in terms of energy consumption, this model is not suitable for use in situations with a lack of adequate energy resources.

Although the model statistical results show the KAN framework requires far fewer parameters and floating-point computations. The performance, however, is poor when it comes to memory usage. It has more frequent memory operations and requires more memory space. Therefore, to improve the execution efficiency of deep learning models based on the KAN framework, it is necessary to start with memory and computational strategies. It is necessary to optimize the way calculations are performed, to maximize computational efficiency and reduce memory allocation requirements. Another idea is to design hardware architectures that are more suitable for composite computing. Current hardware designs are more favorable to traditional deep learning frameworks, highlighting the disadvantages of the KAN framework. As soon as the KAN framework is able to make significant progress in terms of execution efficiency, it will become one of the most desirable deep learning frameworks.

Category	Recall		Precision		F1-score		False Positive Rate	
	IoT2023	IoMT 2024	IoT2023	IoMT 2024	IoT2023	IoMT 2024	IoT2023	IoMT 2024
0	0.886	0.927	0.985	0.953	0.933	0.940	0.002	0.009
1	0.848	0.990	0.887	0.983	0.867	0.987	0.016	0.003
2	0.978	0.986	0.935	0.969	0.956	0.977	0.010	0.006
3	0.968	0.958	0.966	0.929	0.967	0.943	0.005	0.014
4	0.906	0.927	0.865	0.928	0.885	0.928	0.021	0.014
5	0.977	0.878	0.988	0.903	0.982	0.890	0.002	0.019
6	0.950	——	0.892	——	0.920	——	0.016	——
7	0.938	——	0.941	——	0.939	——	0.008	——

Table 13. Spikformer verification.

Category	Recall		Precision		F1-score		False Positive Rate	
	IoT2023	IoMT 2024	IoT2023	IoMT 2024	IoT2023	IoMT 2024	IoT2023	IoMT 2024
0	0.969	0.957	0.944	0.963	0.956	0.960	0.008	0.007
1	0.980	0.989	0.714	0.973	0.826	0.981	0.057	0.006
2	0.989	0.970	0.495	0.988	0.660	0.979	0.145	0.003
3	0.001	0.968	1.000	0.955	0.003	0.962	0.000	0.009
4	0.623	0.965	0.981	0.919	0.762	0.942	0.002	0.017
5	0.976	0.879	0.991	0.930	0.984	0.904	0.001	0.013
6	0.967	——	0.953	——	0.960	——	0.007	——
7	0.942	——	0.970	——	0.956	——	0.004	——

Table 14. SpikingGCN verification.

Category	Recall		Precision		F1-score		False Positive Rate	
	IoT-2023	IoMT-2024	IoT-2023	IoMT-2024	IoT-2023	IoMT-2024	IoT-2023	IoMT-2024
0	0.970	0.961	0.967	0.969	0.969	0.965	0.005	0.006
1	0.877	0.994	0.891	0.985	0.884	0.989	0.015	0.003
2	0.972	0.982	0.981	0.994	0.976	0.988	0.003	0.001
3	0.978	0.969	0.973	0.972	0.976	0.970	0.004	0.005
4	0.902	0.952	0.888	0.949	0.895	0.951	0.017	0.010
5	0.991	0.922	0.995	0.912	0.993	0.917	0.001	0.018
6	0.969	——	0.958	——	0.964	——	0.006	——
7	0.967	——	0.970	——	0.968	——	0.004	——

Table 15. Proposed model verification.

Conclusions and future work

During the experiments, the model proposed in this paper is compared with nine currently popular classical models and two state-of-the-art models. A comprehensive set of indicators is used in the comparison. In the overall picture, the CKAN model leads all the other models in classification accuracy. In terms of computational efficiency and energy consumption, however, it presents a limitation. The current model is more suitable for use in scenarios with sufficient computational resources and may not perform as well when computational resources are limited.

Deep learning models based on the KAN architecture replace the original connection weights with a form fitted by a finite number of spline functions compared to traditional deep learning models. There is a significant increase in computational effort associated with this substitution operation. Trainable parameters have changed from linear objects to non-linear objects. On the one hand, this results in a marked increase in the model's training duration. On the other hand, the model's sample processing speed during validation is lower than other

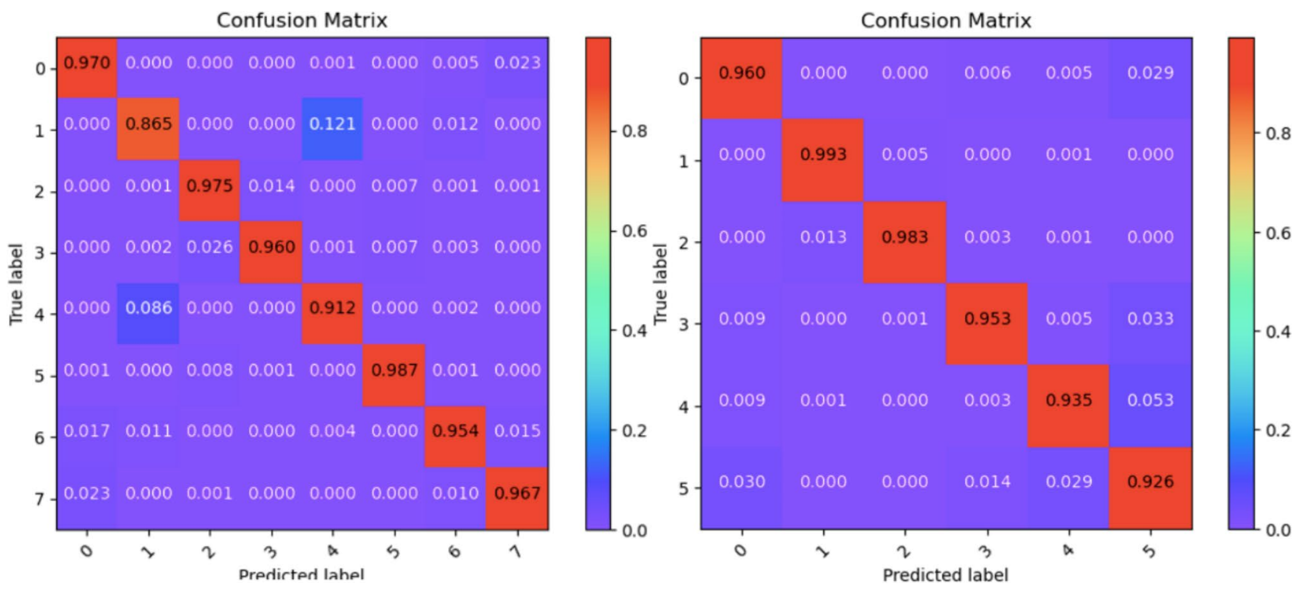

Fig. 4. Alexnet Verification.

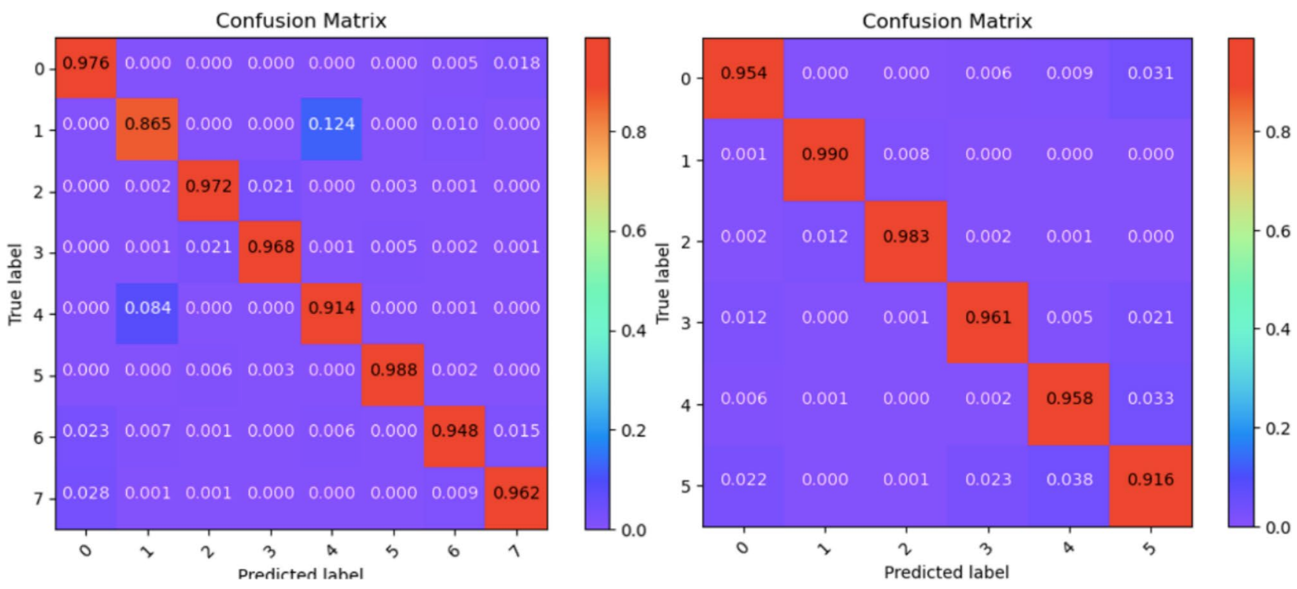

Fig. 5. Resnet18 Verification.

models. It can be seen that the model requires frequent memory manipulation during inference calculations. The advantage is that the model fits the data more precisely, leading to improved accuracy.

The spline functions fitting calculations in the model will be examined in greater depth in future work. As things stand, this part is where the bottleneck in the model's computational efficiency lies. If this part of the computational mechanism can be improved in an efficient way, it will lead to a significant enhancement in the execution efficiency of models based on the KAN architecture. If this is successfully achieved, it is expected that KAN-based deep learning models will grow significantly and shine in more and wider fields.

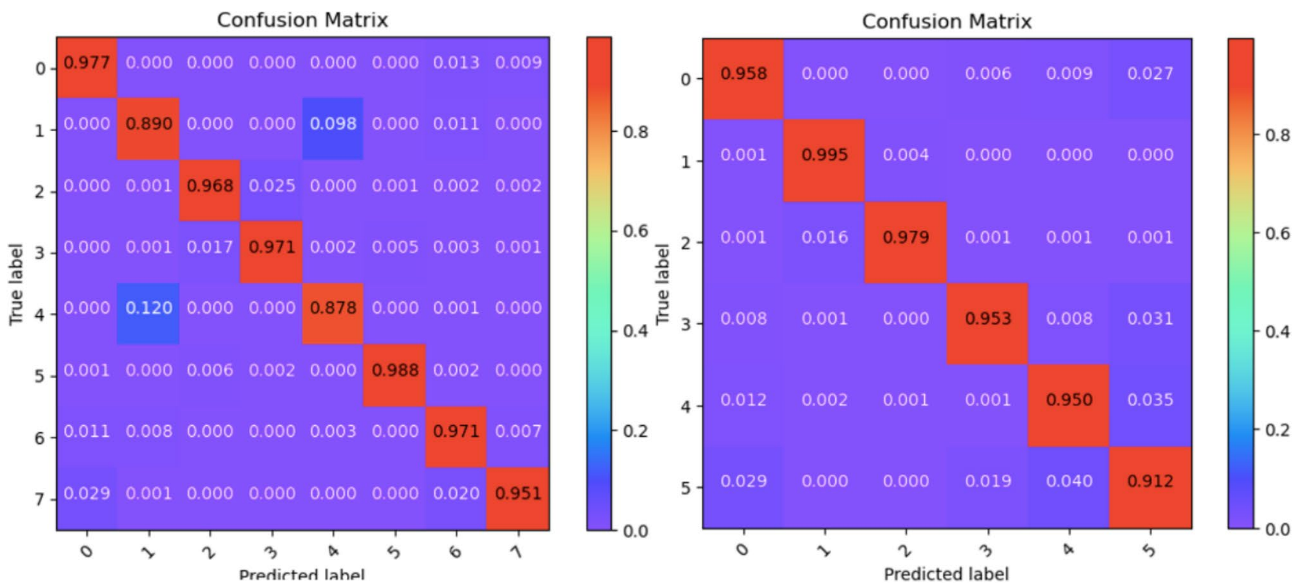

Fig. 6. Resnet50 Verification.

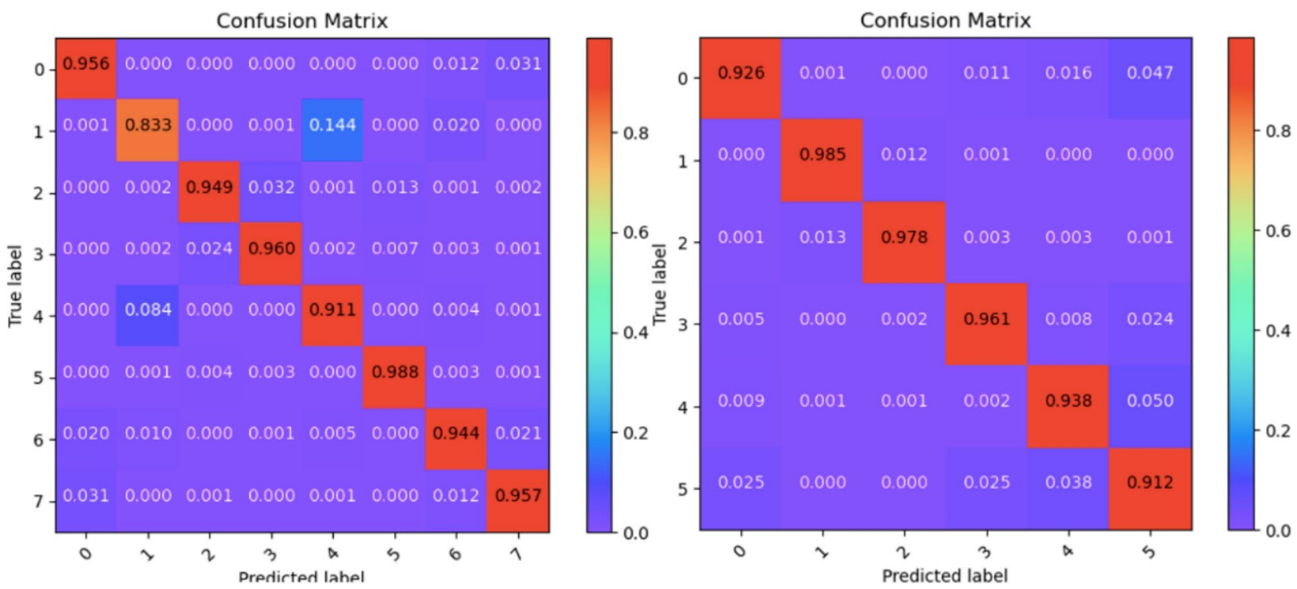

Fig. 7. Mobilenet(V2) Verification.

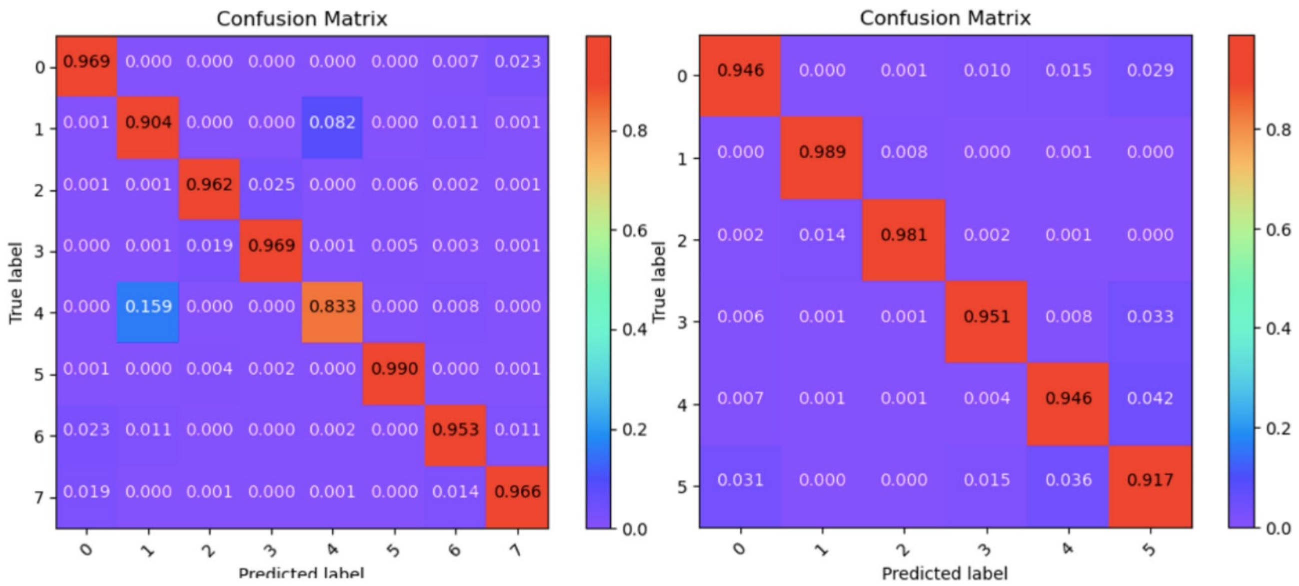

Fig. 8. Efficientnet Verification.

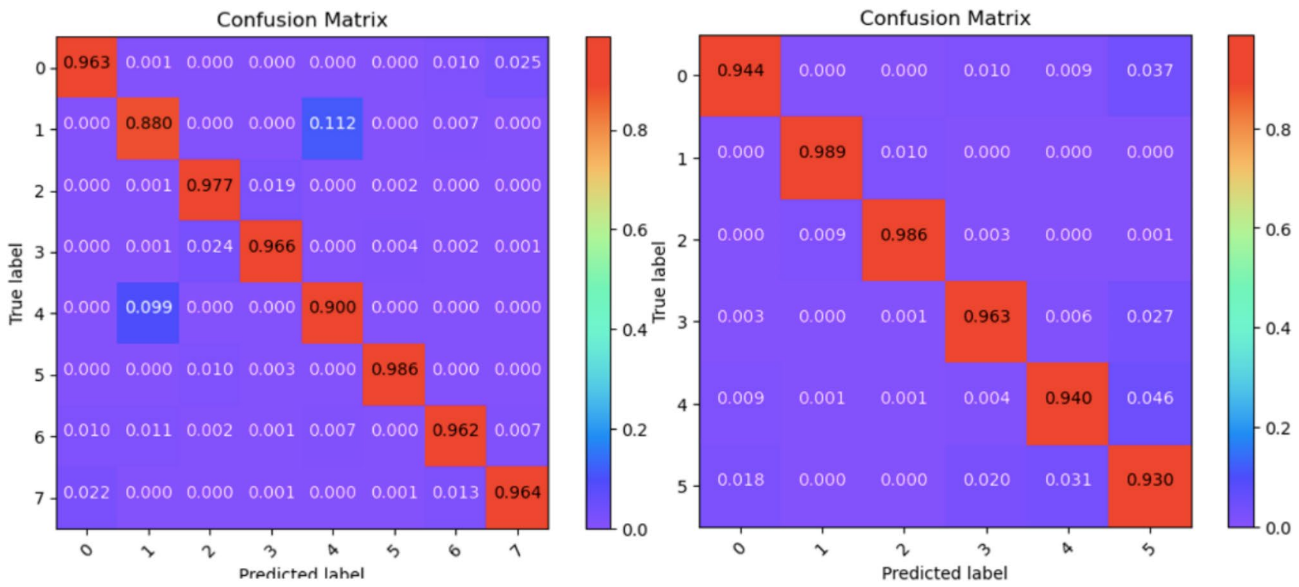

Fig. 9. Densenet121 Verification.

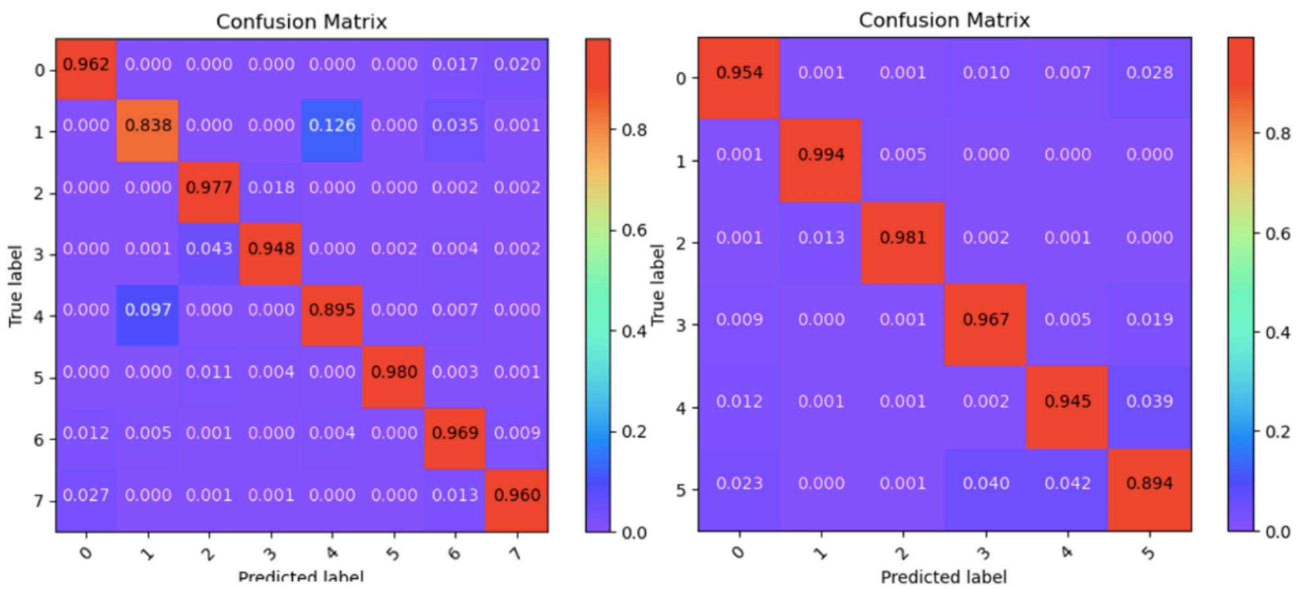

Fig. 10. Googlenet Verification.

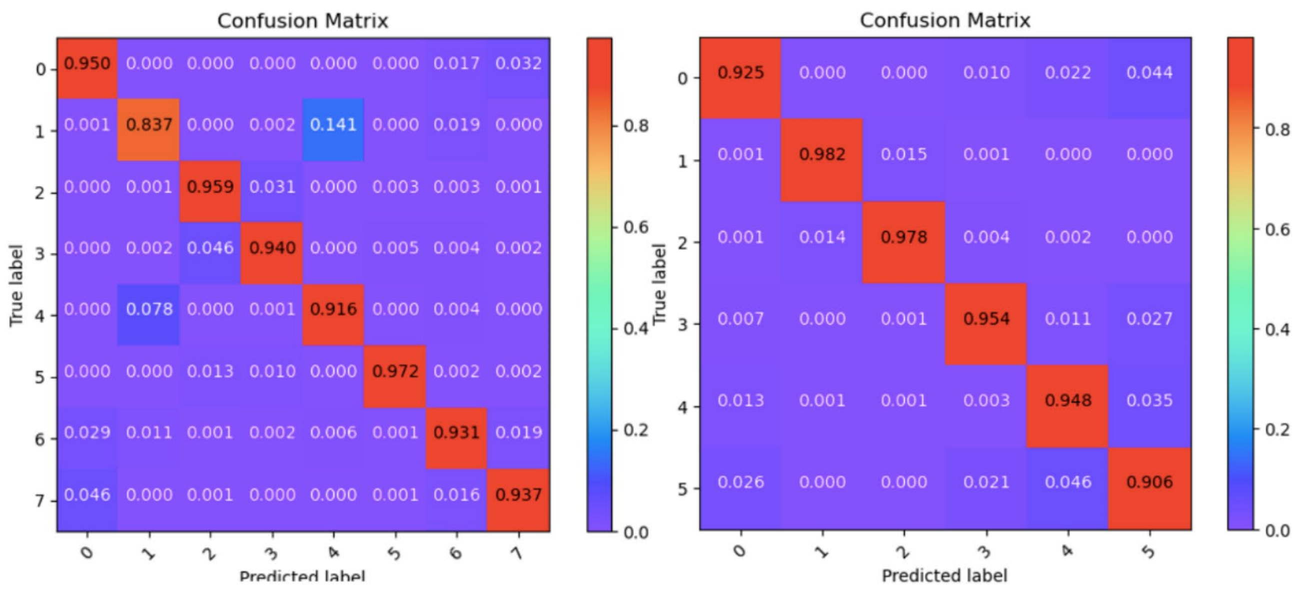

Fig. 11. Shufflenet(V2) Verification.

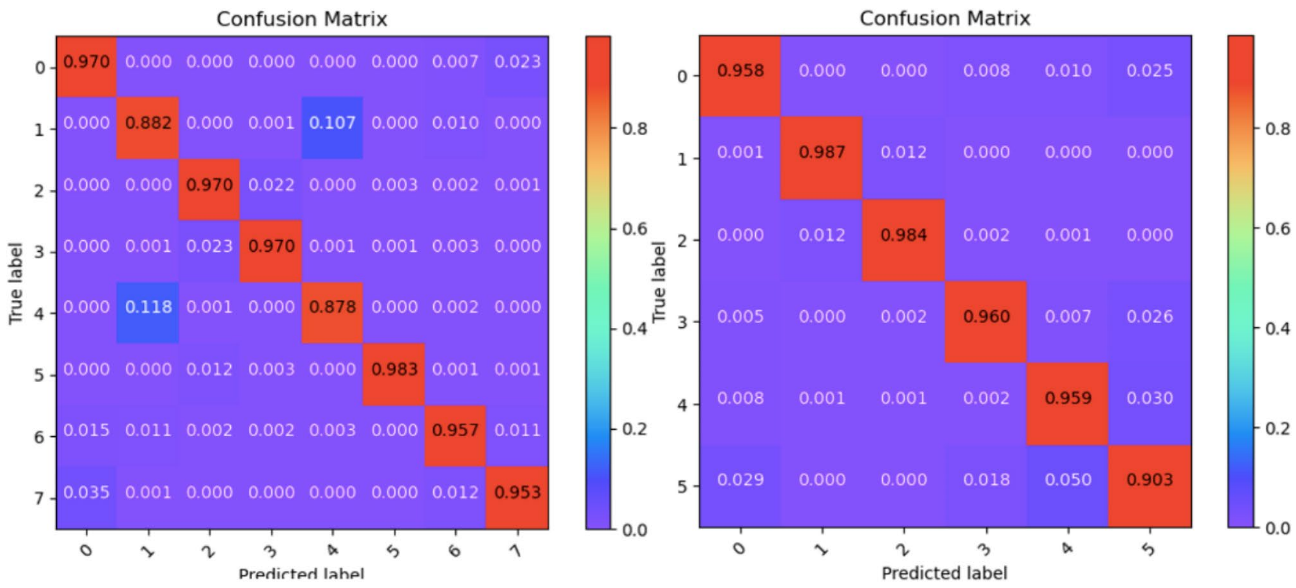

Fig. 12. Squeezenet Verification.

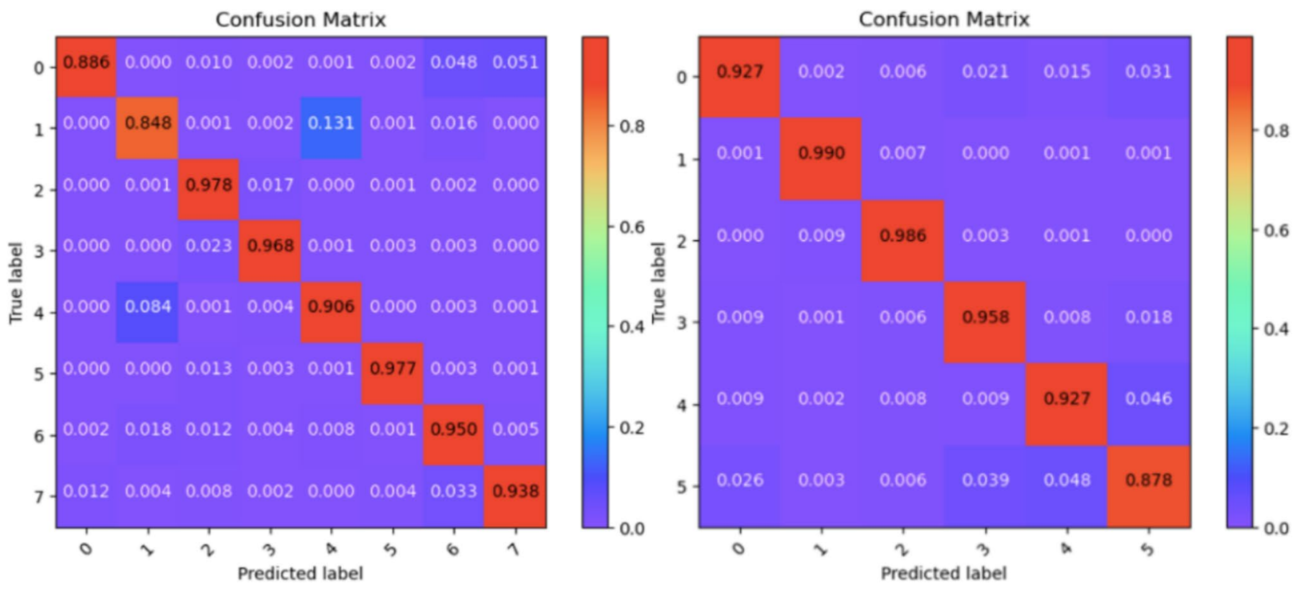

Fig. 13. Spikformer Verification.

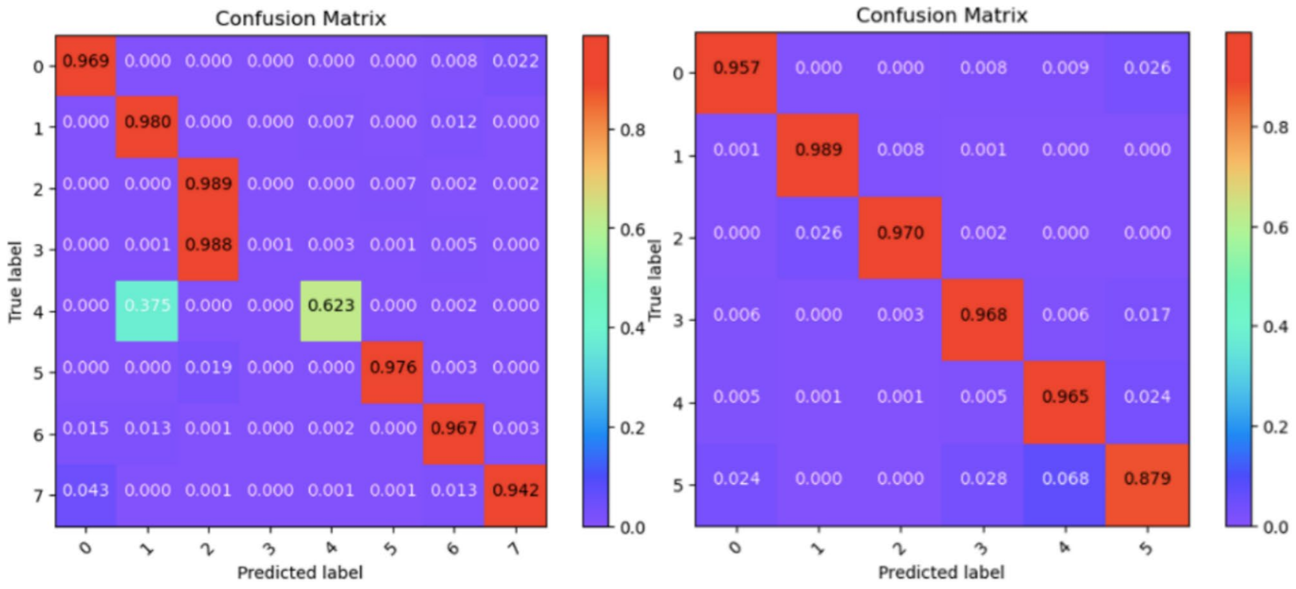

Fig. 14. SpikingGCN Verification.

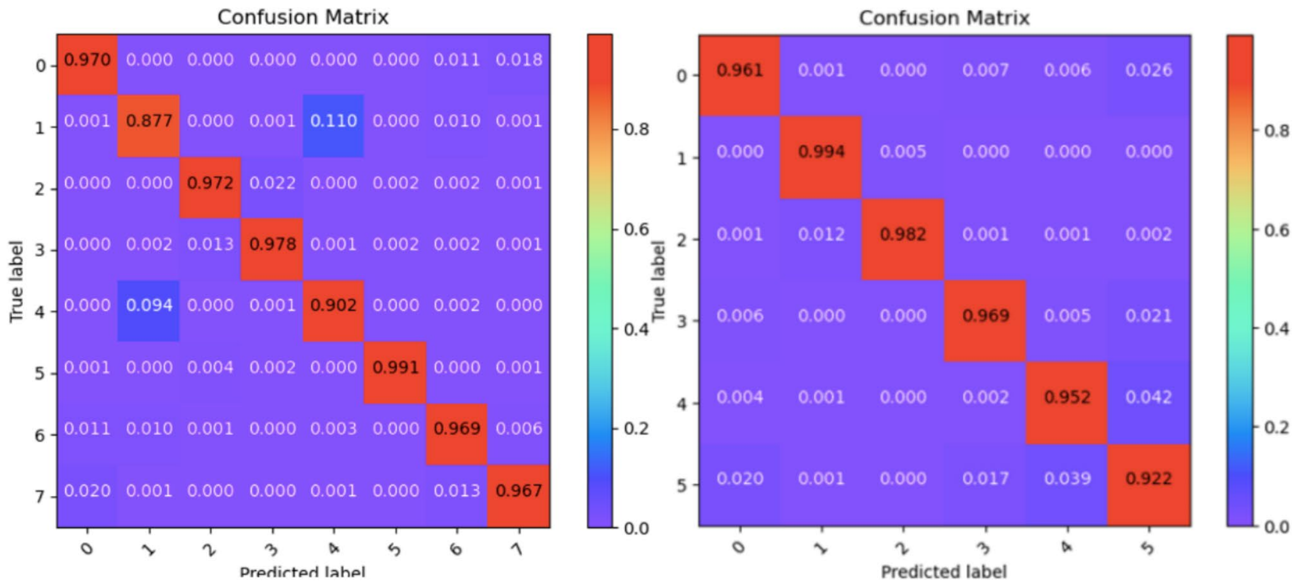

Fig. 15. Proposed Model Verification.

Model	Parameters	FLOPs	Model Size (MB)	RAM Allocation Count	RAM Allocation Size (KB)	Energy Consumption (kWh $\times 10^{-4}$)	Samples/s
Alexnet	57,005,766	145,516,544	228.03	29	4.129	7.400	204
Resnet18	11,170,822	93,348,352	44.78	60	8.879	9.138	165
Resnet50	23,511,558	209,957,888	94.39	125	13.957	12.210	123
Mobilenet(V2)	2,231,014	17,527,808	9.21	128	14.926	3.914	385
Efficientnet	4,014,690	23,881,312	16.42	188	21.293	5.334	282
Densenet121	6,951,238	125,159,168	28.46	287	26.203	8.515	177
Googlenet	11,979,702	38,258,816	48.14	129	13.351	6.038	250
Shufflenet(V2)	2,284,346	9,371,860	9.35	162	16.345	4.227	356
Squeezenet	725,254	28,542,012	2.94	50	6.591	0.904	361
Spikformer	9,328,470	2,093,287,680	37.53	1070	88.634	104.502	14
SpikingGCN	157,926	1,775,370,240	0.619	1156	91.894	13.103	115
Proposed Model	1536	63,532	8.00	324	27.622	17.075	88

Table 16. Complexity of the models and computational resource consumption.

Data availability

The datasets used in this paper are publicly available at https://www.unb.ca/cic/datasets/iotdataset-2023.html and https://www.unb.ca/cic/datasets/iomt-dataset-2024.html.

Received: 27 July 2024; Accepted: 31 December 2024
Published online: 14 January 2025

References

1. Chataut, R., Phoummalayvane, A. & Akl, R. Unleashing the power of IoT: A comprehensive review of IoT applications and future prospects in healthcare, agriculture, smart homes, smart cities, and industry 4.0. Sensors, 23(16), p.7194. (2023).
2. Huda, N. U., Ahmed, I., Adnan, M., Ali, M. & Naeem, F. Experts and intelligent systems for smart homes' Transformation to Sustainable Smart Cities: A comprehensive review. Expert Systems with Applications, 238, p.122380. (2024).
3. Kamm, S., Veekati, S. S., Müller, T., Jazdi, N. & Weyrich, M. A survey on machine learning based analysis of heterogeneous data in industrial automation. Computers in Industry, 149, p.103930. (2023).
4. Hui, C. X., Dan, G., Alamri, S. & Toghraie, D. Greening smart cities: An investigation of the integration of urban natural resources and smart city technologies for promoting environmental sustainability. Sustainable Cities and Society, 99, p.104985. (2023).
5. Choi, D. D. & Lowry, P. B. Balancing the commitment to the common good and the protection of personal privacy: Consumer adoption of sustainable, smart connected cars. Information & management, 61(1), p.103876. (2024).
6. Mary, D. S., Dhas, L. J. S., Deepa, A. R., Chaurasia, M. A. & Sheela, C. J. J. Network intrusion detection: An optimized deep learning approach using big data analytics. Expert Systems with Applications, 251, p.123919. (2024).
7. Mazhar, T. et al. Analysis of challenges and solutions of IoT in smart grids using AI and machine learning techniques: A review. Electronics, 12(1), p.242. (2023).
8. Shandler, R. & Gomez, M. A. The hidden threat of cyber-attacks–undermining public confidence in government. *J. Inform. Technol. Politics*. **20** (4), 359–374 (2023).
9. El-Ghamry, A., Darwish, A. & Hassanien, A. E. An optimized CNN-based intrusion detection system for reducing risks in smart farming. Internet of Things, 22, p.100709. (2023).
10. Wang, Z. & Ghaleb, F. A. An attention-based convolutional neural network for intrusion detection model. *IEEE Access*. **11**, 43116–43127 (2023).
11. Wang, Z., Ghaleb, F. A., Zainal, A., Siraj, M. M. & Lu, X. An efficient intrusion detection model based on convolutional spiking neural network. Scientific Reports, 14(1), p.7054. (2024).
12. Kasongo, S. M. A deep learning technique for intrusion detection system using a recurrent neural networks based framework. *Comput. Commun.* **199**, 113–125 (2023).
13. Kumar, G. S. C., Kumar, R. K., Kumar, K. P. V., Sai, N. R. & Brahmaiah, M. Deep residual convolutional neural network: an efficient technique for intrusion detection system. Expert Systems with Applications, 238, p.121912. (2024).
14. Bakhsh, S. A. et al. Enhancing IoT network security through deep learning-powered Intrusion Detection System. Internet of Things, 24, p.100936. (2023).
15. Hnamte, V., Najar, A. A., Nhung-Nguyen, H., Hussain, J. & Sugali, M. N. DDoS attack detection and mitigation using deep neural network in SDN environment. Computers & Security, 138, p.103661. (2024).
16. Liu, Z. et al. Kan: Kolmogorov-arnold networks. Preprint at https://doi.org/10.48550/arXiv.2404.19756 (2024).
17. Ta, H. T. BSRBF-KAN: a combination of B-splines and Radial Basic functions in Kolmogorov-Arnold Networks. Preprint at https://doi.org/10.48550/arXiv.2406.11173 (2024).
18. Abueidda, D. W., Pantidis, P. & Mobasher, M. E. Deepokan: deep operator network based on Kolmogorov Arnold networks for mechanics problems. Preprint at https://doi.org/10.48550/arXiv.2405.19143 (2024).
19. Aghaei, A. A. fKAN: Fractional Kolmogorov-Arnold Networks with trainable Jacobi basis functions. Preprint at https://doi.org/10.48550/arXiv.2406.07456 (2024).
20. Aghaei, A. A. rKAN: Rational Kolmogorov-Arnold Networks. Preprint at https://doi.org/10.48550/arXiv.2406.14495 (2024).
21. Samadi, M. E., Müller, Y. & Schuppert, A. Smooth Kolmogorov Arnold networks enabling structural knowledge representation. Preprint at https://doi.org/10.48550/arXiv.2405.11318 (2024).
22. Bozorgasl, Z. & Chen, H. Wav-kan: Wavelet Kolmogorov-Arnold networks. Preprint at https://doi.org/10.48550/arXiv.2405.12832 (2024).
23. Xu, J. et al. FourierKAN-GCF: Fourier Kolmogorov-Arnold Network–An effective and efficient feature Transformation for Graph Collaborative Filtering. Preprint at https://doi.org/10.48550/arXiv.2406.01034 (2024).
24. Kiamari, M., Kiamari, M. & Krishnamachari, B. GKAN: Graph Kolmogorov-Arnold Networks. Preprint at https://doi.org/10.48550/arXiv.2406.06470 (2024).

25. Bresson, R. et al. KAGNNs: Kolmogorov-Arnold Networks meet Graph Learning. Preprint at https://doi.org/10.48550/arXiv.2406.18380 (2024).
26. Vaca-Rubio, C. J., Blanco, L., Pereira, R. & Caus, M. Kolmogorov-Arnold networks (kans) for time series analysis. Preprint at https://doi.org/10.48550/arXiv.2405.08790 (2024).
27. Xu, K., Chen, L. & Wang, S. Kolmogorov-Arnold Networks for Time Series: Bridging Predictive Power and Interpretability. Preprint at https://doi.org/10.48550/arXiv.2406.02496 (2024).
28. Genet, R. & Inzirillo, H. Tkan: temporal kolmogorov-arnold networks. Preprint at https://doi.org/10.48550/arXiv.2405.07344 (2024).
29. Cheon, M. Kolmogorov-Arnold Network for Satellite Image classification in Remote Sensing. Preprint at https://doi.org/10.48550/arXiv.2406.00600 (2024).
30. Peng, Y. et al. Predictive modeling of flexible EHD pumps using Kolmogorov-Arnold Networks. Biomimetic Intelligence and Robotics, 4(4), p. 100184. (2024).
31. Wang, Y. et al. Kolmogorov Arnold Informed neural network: A physics-informed deep learning framework for solving PDEs based on Kolmogorov Arnold Networks. Comput. Methods Appl. Mech. Engrg. 433, p. 117518. (2024).
32. Abu Al-Haija, Q., Alohaly, M. & Odeh, A. A lightweight double-stage scheme to identify malicious DNS over HTTPS traffic using a hybrid learning approach. Sensors, 23(7), p.3489. (2023).
33. Abu Al-Haija, Q. & Al-Fayoumi, M. An intelligent identification and classification system for malicious uniform resource locators (URLs). *Neural Comput. Appl.* **35** (23), 16995–17011 (2023).
34. Javeed, D. et al. An intelligent intrusion detection system for smart consumer electronics network. *IEEE Trans. Consum. Electron.* **69** (4), 906–913 (2023).
35. Alsulami, A. A., Abu Al-Haija, Q., Tayeb, A. & Alqahtani, A. An intrusion detection and classification system for IoT traffic with improved data engineering. Applied Sciences, 12(23), p.12336. (2022).
36. Lazzarini, R., Tianfield, H. & Charissis, V. A stacking ensemble of deep learning models for IoT intrusion detection. *Knowl. Based Syst.* **279**, 110941 (2023).
37. Neto, E. C. P. et al. CICIoT2023: A real-time dataset and benchmark for large-scale attacks in IoT environment. Sensors, 23(13), p.5941. (2023).
38. Dadkhah, S. et al. *Raphael and Chukwuka Molokwu, Reginald and Sadeghi, Somayeh and Ghorbani, Ali, CiCIoMT2024* (Attack Vectors in Healthcare Devices-A Multi-Protocol Dataset for Assessing IoMT Device Security, 2024). CICIoMT2024: Attack Vectors in Healthcare devices-A Multi-Protocol Dataset for Assessing IoMT Device Security.
39. Liu, F. T., Ting, K. M. & Zhou, Z. H. December. Isolation forest. In 2008 eighth ieee international conference on data mining (pp. 413–422). IEEE. (2008).
40. Bodner, A. D., Tepsich, A. S., Spolski, J. N. & Pourteau, S. Convolutional Kolmogorov-Arnold Networks. Preprint at https://doi.org/10.48550/arXiv.2406.13155 (2024).
41. Zhou, Z. et al. Spikformer: when spiking neural network meets transformer. Preprint at http://arxiv.org/abs/2209.15425 (2022).
42. Zhu, Z. et al. Spiking graph convolutional networks. Preprint at http://arxiv.org/abs/2205. 02767 (2022).

Author contributions
Zhen Wang conducted most of the experiments and drafted the manuscript. Anazida Zainal, Maheyzah Md Siraj and Fuad A. Ghaleb organized discussions and provided many suggestions for this work. Xue Hao assisted with the experiments and collected and compiled the results. Shaoyong Han provided various experimental environments and computational resources, and offered insightful ideas during the research process.All authors reviewed the manuscript.

Funding
The Talents Subsidized Project (R23165) of Tongling University under Grant 2024tlxyrc020. The research project (K24061) of Tongling University under Grant 2024tlxyxdz280.

Declarations

Competing interests
The authors declare no competing interests.

Additional information
Correspondence and requests for materials should be addressed to S.H.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.