Minimax Theory and its Applications

Volume 11 (2026), No. 3s, 483 to 500

Recurrent Neural Network-Based Intrusion Detection for IoT Networks: Host-Temporal Sequence Modelling, Attention-Based Bidirectional GRUs and a Statistically Rigorous Comparison

S.Maria Sarlina[1], Dr.M.Anand[2], Dr.A.Vinoth kumar[3], Dr.B.Raja[4], Dr.F.Jerald[5]

[1]Research Scholar, [2,3,4]Professor, [5]Associate Professor, Department of Electronics and Communication Engineering, Dr.M.G.R.Educational and Research Institute, Chennai, India
Email: maria.restu@drmgrdu.ac.in[1], anand.ece@drmgrdu.ac.in[2],vinothkumar.ece@drmgrdu.ac.in[3]
raja.ece@drmgrdu.ac.in[4], jerald.ece@drmgrdu.ac.in[5]

Abstract
Intrusion detection is a crucial need of the hour for Internet of Things (IoT) networks. Several Machine Learning models are applied for this task. Recurrent neural networks (RNNs) are also popular in this list to try and look for intrusion in IoT networks.This article proposes Att-BiGRU- Attention-based bidirectional gated recurrent unit based on the principles of Recurrent Neural Networks (RNN). IoTID20 dataset is used for analysis. Five bench mark baseline models were used for comparing the performance of this work, they are, ten recurrent, convolutional, feed-forward, tree-based and linear models. Extensive statistical analysis is also performed using Friedman, Nemenyi critical differences, Holm-corrected Wilcoxon signed-rank tests.
By gradually extending the flows, the macro-F1 of proposed system increases from 0.7445 to 0.9586.. When the flow is five, the score was 0.7445 and when the flows are increased to 20, the score is increased to 0.9586. Usually, the recurrent modelsaccuracy increases as the window grows, while the window size also needs to be optimized. Apart from F1(macro), the other metrics used for performance analysis are F1(weighted), MCC Cohen's K, Precision (macro), Recall (mcro), AUC (macro), DR and FAR.
This paper concludes with the host-temporal window formulation has worked effectively even at 10 flows, and the performance metrics shown better results than the base line models. Unlike the windowed Random Forest, this method improve the performance as the window size grows.

Keywords: Internet of Things; Intrusion Detection System; RNN (Recurrent Neural Networks); gated LSTM; IoTID20; Ablation Study, Statistical Testing; Cross-Validation.

1. Introduction

The Internet of Things (IoT) becomes one of the primary infrastructures in Factories, Hospitals, Cites and Homes etc.,,has moved from a niche of connected sensors to the default architecture of homes, hospitals, factories and cities. IoT networks are computationally weaker compared to the other communication networks. The 2016 Mirai botnet demonstrated this consequence of possible weak password-basedvulnerabilities. [1][2] Since the single device cannot protect the IoT from external threats. A network-based Intrusion Detection [3] (NIDA) introduced and tested under various attack models.

Random forest based models well suited for tabular data [4] while transformers, auto encoders, CNN works good for multi-layer deep representations of data with rich features. [5]. But, RNNs are quite natural fit for network based security, especially to handle spoofer, unforeseen attack by bots, port sweepers etc., [6],[7],[8].

This paper addresses the following research questions with the temporal formulation perspective, the key contribution of the paper are,

1. **Host-temporal sequence formulation:** IoT intrusion detection system is formed using features of past flows based on the principle of Host-Temporal sequence. By this approach RNNs learn from purely temporal structures and not a larger memory and processing capacity is required.
2. **Detection using Att-BiGRU Algorithm:** The core work of this paper is development of attention-based bidirectional GRU with a learned flow embedding, which is discussed in section 4. It also includes a

padding mask channel.The design makes it easy for gateway deployment, though it handles 71,301 parameters.

3. **Extensive Peer Comparison:**10 Models are compared with the proposed work. CNN variant, MLP, Regression based model, Random Forest based model and 6 different RNN models are compared with the proposed work. Metrics such as, macro-F1, MCC, Cohen's κ, macro-AUC, detection rate and false-alarm rate are assessed.
4. **Statistical validation.**Several statistical tests were applied to ensure the conformity with the experimentation results. Friedman test,Nemenyi analysis, Wilcoxon signed-rank tests, Alpaydin's cv F-test and McNemar's test were explored.

The rest of the article is organised as follows. Related works are reviewed in Section 2, Threat model is discussed in Section 3, Architecture proposed in Section 4, Experimentation results are discussed in Section 5, Statistical results are discussed in Section 6 and the sections 7 and 8 discusses the implications, conclusion and scope for future work.

2. Literature survey of related works

2.1 IoT threats

NIDS research and IoT threats are published in various literature based on public dataset results. General flow datasets UNSW-NB15 [9] and CICIDS2017 [10] were used predominantly in several works. With respect to the threat space, Mirai and BASHLITE were commonly analyzed [11]; Other familiar threats discussed are, BoT, DDoS, DoS, and Reconnaissance. [12][13][14]. The bi-directional flow IoTID20 also tested with threats.The issue with the flow datasets is, they contain artefacts, as it was reported in several articles. The mislabeled flows, temporal overlap between training and test dataset are such artefacts.[15]

2.2 Deep non-recurrent detectors

Yang et al. developed tree-based IDSs for mobile networks and developed the open-source IDS-ML toolkit [16], while, Shone et al. [17] used deep feed-forward and autoencoder for non-symmetric processes.Diro and Chilamkurti proposed a Random Forest based IDS [18], Shone et al. combined non-symmetric deep autoencoders with a a distributed deep-learning scheme [19].Vinayakumar et al. applied deep neural networks to detect anomalies using several open source datasets [20]. Meidan et al. used deep autoencoders for botnet detection [21]. Ullah and Mahmoud built the convolutional encoder architecture for intrusion detection in Edge IoT networks [22].

2.3 Recurrent neural networks for intrusion detection

Kim et al. applied LSTM classifier for intrusion detection based on KDD dataset [23]. Yin et al. studied NSL-KDD for classification threats for IoT networks [24]. Roy applied a bidirectional LSTM for high density Industrial IoT based on the dataset UNSW-NB15 [25], and Popoola et al proposed a BiLSTM based IDS, while Imran et al. applied LSTM autoencoder for botnet detection [26]. From these works, it is clearly visible that recurrent models match or exceed the performance of classical learners. But when the temporal context is added to the IDS, the performance may improve or not, that question needs to be examined.

2.4 Evaluation practice

Various metrics and practices used for evaluation of IDS for IoT are discussed in this section. Demšar [27] established comparative performance measure for classifiers. Dietterich [28] and Alpaydin [29] utilized the cross validation for analyzing the correlation between the threats. While MCC was advocated by Chicco and Jurman [32] for statistical validation of results.Sommer and Paxson [33] extensively calculated accuracy, F1 score and AUC-ROC of several datasets under spurious correllations.

Table 1. Summary of Literature Survey

Study	Model family	Dataset(s)	Unit classified	Temporal context across flows
Kim et al. [8]	LSTM	KDD Cup '99	Connection record	No
Yin et al. [7]	RNN	NSL-KDD	Connection record	No
Shone et al. [18]	Deep autoencoder + RF	KDD Cup '99, NSL-KDD	Connection record	No
Diro &	Deep NN	NSL-KDD	Connection record	No

Chilamkurti [16]	(distributed)			
Roy & Cheung [17]	BiLSTM	UNSW-NB15	Flow record	No
Meidan et al. [13]	Deep autoencoders	N-BaIoT	Aggregated window statistics	Implicit (statistics)
Vinayakumar et al. [19]	Deep NN	Multiple	Record	No
Imrana et al. [40]	BiLSTM	NSL-KDD	Connection record	No
Popoola et al. [41]	LSTM-AE + BiLSTM	BoT-IoT	Flow record	No
Ullah & Mahmoud [42]	CNN	IoT datasets incl. IoTID20	Flow record	No
Proposed work	Att-BiGRU + 11 baselines	IoTID20	Flow + last W host flows	Yes (explicit, ablated)

3. Threat Model and Problem Statement

3.1 Threat model

An Enterprise with heterogenous IoT devices is considered as a organization in this work. The devices are cameras, sensors, smart speakers, actuators, electronic controllers etc., All these devices are communicate through the gateway to the external world. The gateway shall be the vulnerable point. The different adversarial options are,

(i) Mirai-style botnet attack by recruiting the nodes.
(ii) Volumetric flooding of services
(iii) Denial-of-service (DoS)
(iv) Man in the Middle attacks host and port reconnaissance (Scan), or
(v) mount man-in-the-middle attacks.

The defender monitors the bidirectional flow passively but not checking the payloads.

3.2 Defining the problem statement

Let $\mathcal{D} = \{(\mathbf{x}_i, s_i, \tau_i, y_i)\}_{i=1}^{N}$ be a set of network flows, where $\mathbf{x}_i \in \mathbb{R}^F$ is the vector of F flow statistics, s_i the sourceidentifier, τ_i the flow start time and $y_i \in \mathcal{Y} = \{1, \dots, C\}$ the traffic category. Then for each flow i, the ordered set of flows shall be described as,

$$\mathcal{H}_i^{(W)} = (j_1, \dots, j_L), \quad j_\ell \in \{j : s_j = s_i,\ (\tau_j, j) \preccurlyeq (\tau_i, i)\}, \quad L = \min\big(W,\ \big|\{j : s_j = s_i, (\tau_j, j) \preccurlyeq (\tau_i, i)\}\big|\big), \qquad (1)$$

The learning problem shall be represented as,

$$\theta^\star = \operatorname{argmin}_\theta \frac{1}{N} \sum_{i=1}^{N} \omega_{y_i}\ \ell\big(y_i,\ f_\theta(\mathbf{S}_i)\big) + \lambda\, \Omega(\theta), \qquad (2)$$

where ℓ is the cross-entropy, ω_c is a class weight compensating for imbalanceand Ω denotes implicit regularisation (dropout and early stopping).

4. Methodology

A three-tier deployment IoT architecture is proposed. First layer is the perception layer, followed by gateway layer and SOC layer. Perception layer integrates the devices Smart camera, Smart speaker, Sensors, and Attacker also. In the second layer, three different processes such as packet capture, pre-processing and temporal window are defined. The data are analyzed using Att-BiGRU detector.

The sequence is numbered in the figure as per the processing order. The workflow is summarized in Fig.2.

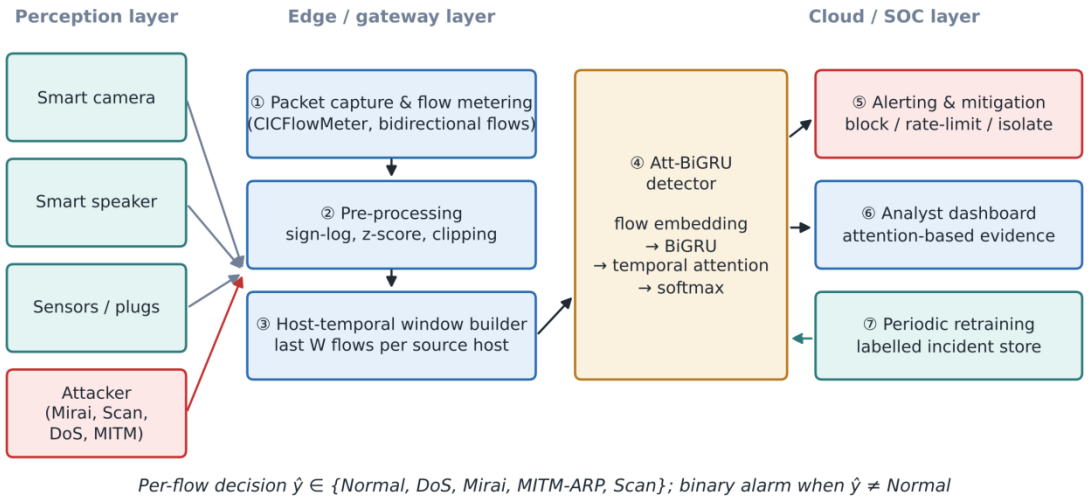


Figure 1. Deployment architecture of the host-temporal recurrent IDS at an IoT edge gateway.

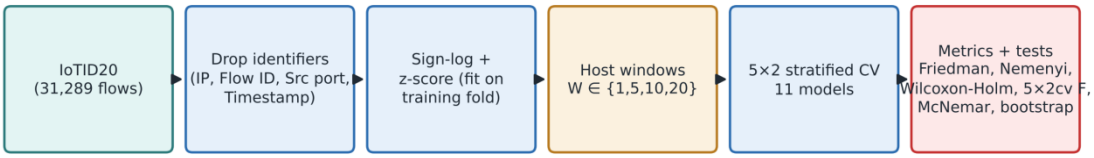


Figure 2. Experimental workflow.

4.1 Dataset

IoTID20 dataset generated from a smart-home testbed used in this research. In which pcap captureswere converted to 83 bidirectional flow features. They again redistributed 5 % stratified sample released with the open-source IDS-ML toolkit. The released file mapped the codes to category names by alphabetical label-encoding order The resulting class distribution is shown in Figure 3: Mirai accounts for 66.4 % of flows, while Man in Middle was the lowest.

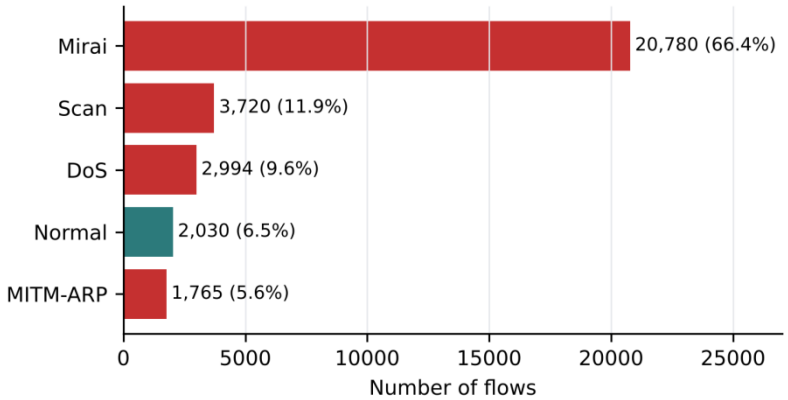


Figure 3. Class distribution of the IoTID20 sample.

4.2 Pre-processing

In the pre-processing, the flow ID, IP addresses and time stamp are removed. Constant columns were dropped. Finally, only 68 features left.Flow statistics are heavy-tailed with a signed logarithm, so hencestandardised with the training fold only:

$$\phi(x) = \operatorname{sgn}(x)\ln(1+|x|), \qquad (4)$$

$$z_k = \operatorname{clip}\left(\frac{\phi(x_k)-\mu_k^{\mathrm{tr}}}{\sigma_k^{\mathrm{tr}}}, -10, 10\right), \qquad \mu_k^{\mathrm{tr}} = \frac{1}{|\mathcal{T}|}\sum_{i\in\mathcal{T}}\phi(x_{ik}), \quad \sigma_k^{\mathrm{tr}} = \sqrt{\frac{1}{|\mathcal{T}|}\sum_{i\in\mathcal{T}}(\phi(x_{ik})-\mu_k^{\mathrm{tr}})^2}, \qquad (5)$$

where $\mathcal{T}$ is the training index set of the current fold. Fitting $\mu^{\mathrm{tr}}, \sigma^{\mathrm{tr}}$ inside each fold prevents information from the test partition leaking into the scaler.

4.3 Host-temporal window

Flows are sorted by (source host, timestamp, record index) and, for every flow, the indices of the W most recent flows of the same host are gathered (Eq. 1, Figure 4). In the evaluated data 90.4 % of flows have at least one predecessor from the same host and 89.5 % have a full window at $W = 10$, because a small number of hosts (including the compromised devices) generate most of the traffic. Hosts that emit a single flow yield a window containing only the current flow plus padding, so the model degrades gracefully to flow-level classification.

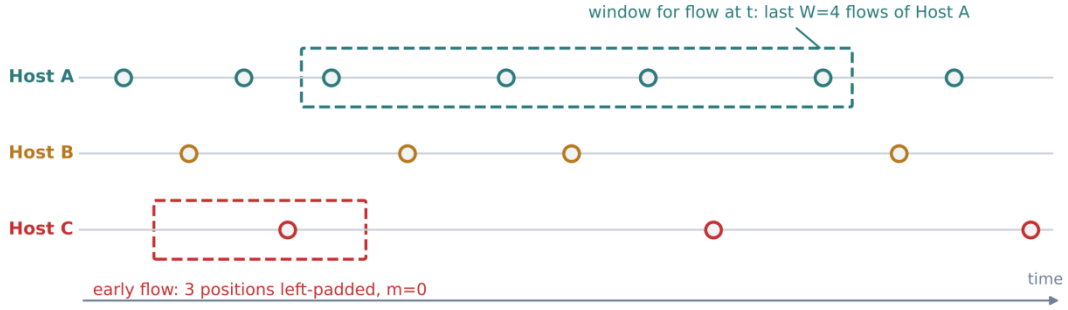


Figure 4. Host-temporal window construction: each flow is classified together with the preceding flows of the same source host; hosts with short histories are left-padded and masked.

4.4 Recurrent encoders

Each time step is first projected into a $d = 64$ dimensional embedding,

$$\mathbf{e}_t = \mathrm{ReLU}(\mathbf{W}_e\tilde{\mathbf{x}}_t + \mathbf{b}_e), \qquad \mathbf{W}_e \in \mathbb{R}^{d\times(F+1)}. \qquad (6)$$

The simplest recurrent unit updates a hidden state $\mathbf{h}_t \in \mathbb{R}^H$ as

$$\mathbf{h}_t = \tanh(\mathbf{W}_{xh}\mathbf{e}_t + \mathbf{W}_{hh}\mathbf{h}_{t-1} + \mathbf{b}_h). \qquad (7)$$

Back-propagating a loss $\mathscr{L}$ through time gives

$$\frac{\partial\mathscr{L}}{\partial\mathbf{h}_k} = \frac{\partial\mathscr{L}}{\partial\mathbf{h}_T}\prod_{t=k+1}^{T}\frac{\partial\mathbf{h}_t}{\partial\mathbf{h}_{t-1}} = \frac{\partial\mathscr{L}}{\partial\mathbf{h}_T}\prod_{t=k+1}^{T}\operatorname{diag}(1-\mathbf{h}_t^2)\mathbf{W}_{hh}, \qquad (8)$$

whose norm is bounded by $(\gamma \,\|\, \mathbf{W}_{hh} \,\|_2)^{T-k}$ with $\gamma \le 1$; the gradient therefore vanishes (or explodes) geometrically with the temporal distance $T-k$ [38], [39]. Gated units mitigate this.

Long Short-Term Memory (LSTM) maintains a cell state $\mathbf{c}_t$ regulated by input, forget and output gates:

$$\mathbf{i}_t = \sigma(\mathbf{W}_i\mathbf{e}_t + \mathbf{U}_i\mathbf{h}_{t-1} + \mathbf{b}_i), \quad \mathbf{f}_t = \sigma(\mathbf{W}_f\mathbf{e}_t + \mathbf{U}_f\mathbf{h}_{t-1} + \mathbf{b}_f), \quad \mathbf{o}_t = \sigma(\mathbf{W}_o\mathbf{e}_t + \mathbf{U}_o\mathbf{h}_{t-1} + \mathbf{b}_o), \qquad (9)$$

$$\tilde{\mathbf{c}}_t = \tanh(\mathbf{W}_c\mathbf{e}_t + \mathbf{U}_c\mathbf{h}_{t-1} + \mathbf{b}_c), \qquad \mathbf{c}_t = \mathbf{f}_t\odot\mathbf{c}_{t-1} + \mathbf{i}_t\odot\tilde{\mathbf{c}}_t, \qquad \mathbf{h}_t = \mathbf{o}_t\odot\tanh(\mathbf{c}_t). \qquad (10)$$

The additive cell update yields $\partial\mathbf{c}_t/\partial\mathbf{c}_{t-1} = \operatorname{diag}(\mathbf{f}_t) + \cdots$, so information can be carried across many steps when $\mathbf{f}_t \approx \mathbf{1}$.

4.5 Masking

The proposed model pools all states with additive attention as explained by Behansu [4], accordingly the masking done.

$$\varepsilon_t = \mathbf{v}^{\top}\tanh(\mathbf{W}_a\mathbf{h}_t + \mathbf{b}_a) - \kappa\,(1 - m_t), \qquad \kappa = 10^9, \qquad (14)$$

$$\alpha_t = \frac{\exp(\varepsilon_t)}{\sum_{t'=1}^{W}\exp(\varepsilon_{t'})}, \qquad \mathbf{c} = \sum_{t=1}^{W}\alpha_t\,\mathbf{h}_t, \qquad (15)$$

with $\mathbf{W}_a \in \mathbb{R}^{64\times 2H}$ and $\mathbf{v} \in \mathbb{R}^{64}$. The weights α_t satisfy $\alpha_t \geq 0$, $\sum_t \alpha_t = 1$ and $\alpha_t \approx 0$ for padding, and they expose which past flows of a host drove a decision.

The detailed architecture is given in Fig.5.

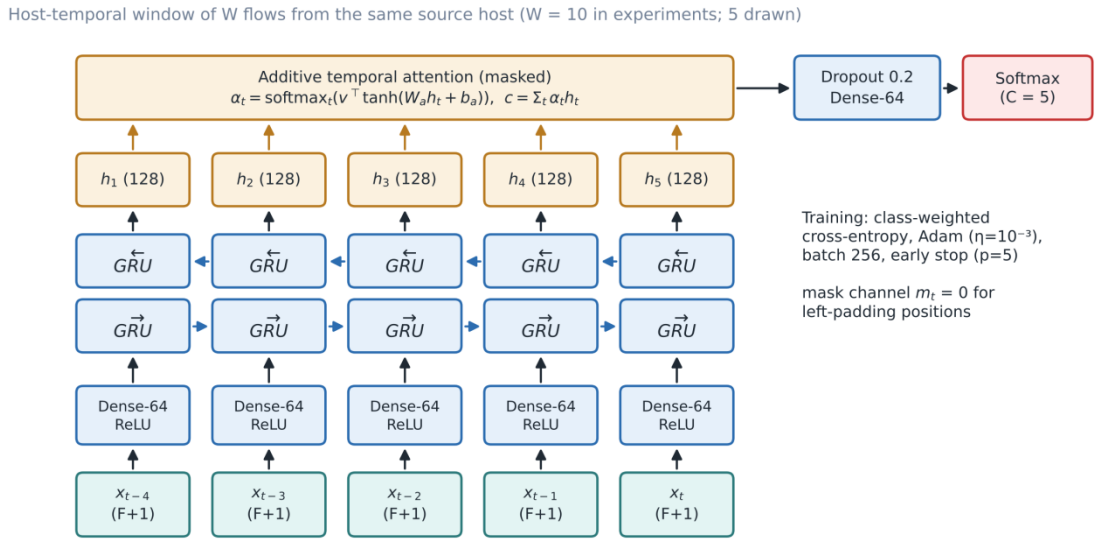


Figure 5. Architecture of the proposed Att-BiGRU detector (window of W flows; five positions drawn).

4.6 Classification head and loss

The context vector is regularised with dropout [30] (rate $p = 0.2$) and mapped to class probabilities,

$$\mathbf{g} = \mathrm{ReLU}\big(\mathbf{W}_g\,(\mathbf{c} \odot \boldsymbol{\xi}) + \mathbf{b}_g\big), \;\; \xi_k \sim \mathrm{Bernoulli}(1-p), \qquad \hat{\mathbf{p}} = \mathrm{softmax}(\mathbf{W}_o\mathbf{g} + \mathbf{b}_o), \quad \hat{p}_c = \frac{e^{o_c}}{\sum_{c'} e^{o_{c'}}}. \qquad (16)$$

Training minimises the class-weighted cross-entropy

$$\mathscr{L}(\theta) = -\frac{1}{|\mathscr{B}|}\sum_{i\in\mathscr{B}}\omega_{y_i}\log\hat{p}_{i,y_i}, \qquad (17)$$

over mini-batches $\mathscr{B}$ with Adam [29]:

$$\mathbf{m}_k = \beta_1\mathbf{m}_{k-1} + (1-\beta_1)\nabla_\theta\mathscr{L}, \quad \mathbf{v}_k = \beta_2\mathbf{v}_{k-1} + (1-\beta_2)(\nabla_\theta\mathscr{L})^2, \qquad (18)$$

$$\theta_k = \theta_{k-1} - \eta\,\frac{\mathbf{m}_k/(1-\beta_1^k)}{\sqrt{\mathbf{v}_k/(1-\beta_2^k)} + \epsilon}, \qquad (19)$$

with $\eta = 10^{-3}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-7}$. Training stops when the validation loss has not improved for five epochs (maximum 40), and the best weights are restored:

$$\theta^{\star} = \theta_{e^{\star}}, \qquad e^{\star} = \arg\min_{e \le E_{\text{stop}}} \mathscr{L}_{\text{val}}(\theta_e), \qquad E_{\text{stop}} = \min\{e:\ \mathscr{L}_{\text{val}}(\theta_{e-q}) \le \min_{e-q<e'\le e} \mathscr{L}_{\text{val}}(\theta_{e'}),\ q = 5\}. \tag{20}$$

The class weights are the "balanced" inverse-frequency weights

$$\omega_c = \frac{N_{\text{tr}}}{C\, N_c}, \qquad \sum_c \omega_c\, N_c = N_{\text{tr}}, \tag{21}$$

4.7 Baseline Models for comparisonand hyper-parameters

Ten baselines were taken into the consideration in this work. LSTM, GRU, BiLSTM and BiGRU are the recurrent baselines. A causal 1-D CNN with two 64-filter convolutions are used with multiple windows isolates the effect of *recurrence* from that of *context*. Two context-free deep and classical baselines see only the current flow: a multilayer perceptron (MLP, 128-64 units) and a Random Forest (RF, 300 trees) [31]; a Logistic Regression (LR) gives a linear reference. Finally, RF-window receives the flattened window ($W(F+1)$ features), giving a strong non-neural model the same information as the RNNs. All tree and linear models use balanced class weights.

5. Evaluation Metrics

5.1 Classification metrics

From the multi-class confusion matrix $\mathbf{M} \in \mathbb{N}^{C\times C}$ (M_{ab} = flows of true class a predicted as b) we derive, for each class c, $TP_c = M_{cc}$, $FP_c = \sum_{a\neq c} M_{ac}$, $FN_c = \sum_{b\neq c} M_{cb}$ and

$$\text{Acc} = \frac{\sum_c M_{cc}}{\sum_{a,b} M_{ab}}, \qquad P_c = \frac{TP_c}{TP_c + FP_c}, \qquad R_c = \frac{TP_c}{TP_c + FN_c}, \qquad F1_c = \frac{2P_cR_c}{P_c + R_c}. \tag{26}$$

Macro-averages weight all classes equally, which is essential when the majority class (Mirai) would otherwise dominate; weighted-F1 weights by support N_c:

$$\text{F1}_{\text{macro}} = \frac{1}{C}\sum_{c=1}^{C} F\,1_c, \qquad \text{F1}_{\text{weighted}} = \sum_{c=1}^{C} \frac{N_c}{N} F1_c. \tag{27}$$

The multi-class Matthews correlation coefficient [32] and Cohen's kappa account for chance agreement:

$$\text{MCC} = \frac{s\,n - \sum_c p_c\, t_c}{\sqrt{(n^2 - \sum_c p_c^2)(n^2 - \sum_c t_c^2)}}, \qquad s = \sum_c M_{cc}\,,\ \ t_c = \sum_b M_{cb}\,,\ \ p_c = \sum_a M_{ac}\,,\ \ n = \sum_{a,b} M_{ab}\,, \tag{28}$$

$$\kappa = \frac{p_o - p_e}{1 - p_e}, \qquad p_o = \frac{s}{n}, \qquad p_e = \frac{1}{n^2}\sum_c p_c\, t_c. \tag{29}$$

Threshold-free ranking quality is measured by the one-vs-rest macro area under the ROC curve,

$$\text{AUC}_{\text{macro}} = \frac{1}{C}\sum_{c=1}^{C} \int_0^1 \text{TPR}_c\left(\text{FPR}_c^{-1}(u)\right) du. \tag{30}$$

For operational alarm quality we collapse the four attack classes into one and report the detection rate and false-alarm rate,

$$\text{DR} = \frac{TP_{\text{att}}}{TP_{\text{att}} + FN_{\text{att}}}, \qquad \text{FAR} = \frac{FP_{\text{att}}}{FP_{\text{att}} + TN_{\text{att}}}, \tag{31}$$

where a "positive" is any flow predicted as an attack class.

5.3 Statistical tests

Based on the recommendations of Demšar [24], the models are first compared jointly with the Friedman test on their average ranks $\bar{r}_j$

$$\chi_F^2 = \frac{12N}{k(k+1)}\left[\sum_{j=1}^{k}\bar{r}_j^2 - \frac{k(k+1)^2}{4}\right], \qquad F_F = \frac{(N-1)\chi_F^2}{N(k-1)-\chi_F^2} \sim F\big(k-1,(k-1)(N-1)\big), \qquad (32)$$

where F_F is the less conservative Iman–Davenport correction. If the null hypothesis of equal performance is rejected, the Nemenyi post-hoc test declares two models different when their average ranks differ by more than the critical difference

$$\mathrm{CD} = q_\alpha\sqrt{\frac{k(k+1)}{6N}}, \qquad (33)$$

with $q_{0.05} = 3.219$ for $k = 11$. Because Nemenyi is conservative, the proposed model is additionally compared with each competitor using the Wilcoxon signed-rank test on the paired fold differences d_i,

$$T^+ = \sum_{i:\,d_i>0}\mathrm{rank}\,(|d_i|), \qquad z = \frac{T^+ - N(N+1)/4}{\sqrt{N(N+1)(2N+1)/24}}, \qquad r = \frac{|z|}{\sqrt{N}}, \qquad (34)$$

with family-wise error controlled by Holm's step-down procedure [28]: ordering the mp-values $p_{(1)} \le \cdots \le p_{(m)}$, hypothesis $H_{(j)}$ is rejected if $p_{(l)} \le \alpha/(m-l+1)$ for all $l \le j$. Exact two-sided Wilcoxon p-values are computed for $N = 10$. We also apply Alpaydin's combined 5×2 cv F-test [26], which is designed for exactly this resampling scheme and has lower Type-I error than the resampled t-test:

$$F_{5\times2} = \frac{\sum_{i=1}^{5}\sum_{j=1}^{2}(p_i^{(j)})^2}{2\sum_{i=1}^{5}s_i^2} \sim F(10,5), \qquad s_i^2 = (p_i^{(1)}-\bar{p}_i)^2 + (p_i^{(2)}-\bar{p}_i)^2, \qquad (35)$$

where $p_i^{(j)}$ is the macro-F1 difference on fold j of replication i and $\bar{p}_i = \big(p_i^{(1)} + p_i^{(2)}\big)/2$. On a single held-out half (replication 1, fold 1; \$n=\$15,645 flows) error patterns are compared with McNemar's test [27] using continuity correction,

$$\chi^2_{\mathrm{McN}} = \frac{(|b-c|-1)^2}{b+c} \sim \chi_1^2, \qquad (36)$$

where b counts flows the proposed model classifies correctly but the competitor misclassifies and c the reverse. Uncertainty of macro-F1 on that partition is quantified by a non-parametric percentile bootstrap with $B = 1000$ resamples [45],

$$\mathrm{CI}_{95\%} = \big[\widehat{F1}^*_{(0.025B)},\ \widehat{F1}^*_{(0.975B)}\big]. \qquad (37)$$

Finally, effect sizes are reported with Cliff's delta, a non-parametric dominance statistic,

$$\delta = \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}\mathrm{sgn}\,\big(a_i - b_j\big) \in [-1,1], \qquad (38)$$

interpreted as negligible ($|\delta| < 0.147$), small (< 0.33), medium (< 0.474) or large otherwise. All tests use $\alpha = 0.05$.

6. Experimental Results

All the experimental results are listed in the table 2.The typical strategy of means ± standard deviations over multiple cross-validation of 5x2 folds applied.

Table 2. IoTID20 Dataset used in this study.

Category	Code	Flows	Share (%)	ω (Eq. 21)	Description
DoS	0	2,994	9.57	2.09	SYN flooding
MITM-ARP	1	1,765	5.64	3.545	ARP spoofing
Mirai	2	20,780	66.41	0.301	Botnet brute-force and floods
Normal	3	2,030	6.49	3.083	Benign traffic
Scan	4	3,720	11.89	1.682	Host/port scanning
Total		31,289	100.00		68 features; 3,005 hosts

Table 3. Experimental Results- Modelscomparison

Model	Configuration	Uses window	Trainable params
Att-BiGRU	Dense-64 → BiGRU-64 → masked attention → Dense-64 → softmax	Yes	71,301
Att-BiLSTM	As Att-BiGRU with LSTM cells	Yes	87,429
BiGRU	Dense-64 → BiGRU-64 (last state) → Dense-64	Yes	62,981
BiLSTM	Dense-64 → BiLSTM-64 (last state) → Dense-64	Yes	79,109
GRU	Dense-64 → GRU-64 → Dense-64	Yes	33,925
LSTM	Dense-64 → LSTM-64 → Dense-64	Yes	41,989
CNN1D	2 × causal Conv1D(64, k = 3) → global max-pool → Dense-64	Yes	30,149
MLP	Current flow → Dense-128 → Dense-64	No	21,701
RF-window	300 trees, balanced, flattened window	Yes	—
RF	300 trees, balanced, current flow	No	—
LR	L2, lbfgs, balanced, current flow	No	—

For the operational attack versus normal binary case, every windowed model detects more than 99.7 % of attack flows. Att-BiGRU raises alarms on 1.12 % of benign flows, which is lowest of all models. The FAR advantage of Att-BiGRU is significant, but not against the other recurrent variants. The gap between accuracy and macro-F1 (e.g. 0.8633 vs 0.8384 for Att-BiGRU) reflects the minority attack classes. MCC and κ rank the models almost identically to macro-F1, confirming that the conclusions are not an artefact of the chosen summary metric [32].

Table 4a. Multi-class performance at W = 10 (mean ± sd over 5×2 CV; best per column in bold).

Model	Accuracy	F1 (macro)	F1 (weighted)	MCC	Cohen's κ
Att-BiGRU	0.8633 ± 0.0208	0.8384 ± 0.0218	0.8762 ± 0.0172	0.7855 ± 0.0268	0.7695 ± 0.0310
Att-BiLSTM	0.8613 ± 0.0151	0.8369 ± 0.0179	0.8747 ± 0.0122	0.7828 ± 0.0200	0.7663 ± 0.0227
BiGRU	0.8747 ± 0.0300	0.8527 ± 0.0349	0.8861 ± 0.0252	0.8008 ± 0.0409	0.7870 ± 0.0462
BiLSTM	0.8618 ± 0.0144	0.8370 ± 0.0169	0.8750 ± 0.0120	0.7830 ± 0.0188	0.7667 ± 0.0217
GRU	0.8564 ± 0.0110	0.8316 ± 0.0119	0.8704 ± 0.0092	0.7766 ± 0.0144	0.7590 ± 0.0163
LSTM	0.8617 ± 0.0175	0.8379 ± 0.0189	0.8750 ± 0.0140	0.7835 ± 0.0219	0.7669 ± 0.0256
CNN1D	0.9300 ± 0.0105	0.9075 ± 0.0102	0.9319 ± 0.0092	0.8729 ± 0.0158	0.8712 ± 0.0172
MLP	0.7910 ± 0.0075	0.7431 ± 0.0089	0.8135 ± 0.0064	0.6895 ± 0.0099	0.6626 ± 0.0105
RF-window	0.8743 ± 0.0018	0.8359 ± 0.0022	0.8841 ± 0.0016	0.7938 ± 0.0027	0.7820 ± 0.0028
RF	0.8492 ± 0.0033	0.7942 ± 0.0039	0.8625 ± 0.0026	0.7492 ± 0.0051	0.7392 ± 0.0052
LR	0.7695 ± 0.0035	0.7131 ± 0.0046	0.7921 ± 0.0033	0.6535 ± 0.0045	0.6289 ± 0.0048

Table 4b. Macro precision/recall, ranking quality and binary attack-versus-normal alarm quality at W = 10.

Model	Precision (macro)	Recall (macro)	AUC (macro, OvR)	DR (%)	FAR (%)
Att-BiGRU	0.8038 ± 0.0214	0.9098 ± 0.0145	0.9795 ± 0.0051	99.76 ± 0.13	**1.12 ± 0.42**
Att-BiLSTM	0.8030 ± 0.0170	0.9088 ± 0.0135	0.9792 ± 0.0039	99.81 ± 0.11	1.16 ± 0.55
BiGRU	0.8212 ± 0.0379	0.9163 ± 0.0190	0.9819 ± 0.0061	99.84 ± 0.06	1.33 ± 0.37
BiLSTM	0.8041 ± 0.0175	0.9063 ± 0.0119	0.9795 ± 0.0037	99.78 ± 0.15	1.31 ± 0.60
GRU	0.7976 ± 0.0116	0.9056 ± 0.0112	0.9778 ± 0.0032	99.79 ± 0.15	1.31 ± 0.63
LSTM	0.8051 ± 0.0191	0.9074 ± 0.0105	0.9794 ± 0.0044	99.84 ± 0.08	1.36 ± 0.33
CNN1D	0.8932 ± 0.0184	0.9268 ± 0.0138	0.9917 ± 0.0016	99.86 ± 0.05	1.89 ± 0.79

MLP	0.7131 ± 0.0128	0.8233 ± 0.0062	0.9579 ± 0.0022	98.08 ± 0.68	13.53 ± 1.75
RF-window	0.8161 ± 0.0026	0.8782 ± 0.0038	0.9808 ± 0.0005	**99.91 ± 0.03**	3.32 ± 0.55
RF	0.7696 ± 0.0042	0.8395 ± 0.0036	0.9624 ± 0.0012	99.55 ± 0.08	5.20 ± 0.64
LR	0.6755 ± 0.0048	0.8001 ± 0.0043	0.9460 ± 0.0005	96.31 ± 0.31	12.39 ± 0.47

6.2 Statistical significance

The pairwise comparison test - Friedman test was performed on all models. Probability is better in our proposed system, BiLSTM is the next nearest contender. Distribution of Macro F1 over the datasets for ten 5x2 CV folds are plotted in Fig. 6. Shapiro–Wilk tests on the paired fold differences with N = 10, so that the non-parametric tests are quite optional.But, after Holm correction, the performance ofAtt-BiGRU is significantly better than the MLP, RF and LR. Alpaydin's 5×2 cv F-testconfirms the advantages over the MLP and LR . The critical difference is also plotted in Fig.7.

Table 5. Pairwise comparison of Att-BiGRU with each competitor on macro-F1

Model	**Rank**	**ΔF1**	**p (W)**	**p (Holm)**	**F_{5x2}**	**p (F)**	**δ**	**b/c**	**p (McN)**
Att-BiLSTM	4.80	+0.0015	1.000	1.000	0.98	0.545	+0.02	297/211	< 0.001
BiGRU	4.60	-0.0143	0.322	1.000	0.74	0.681	-0.20	433/117	< 0.001
BiLSTM	5.20	+0.0014	0.922	1.000	1.90	0.249	+0.04	403/160	< 0.001
GRU	5.50	+0.0068	0.432	1.000	0.79	0.652	+0.10	393/124	< 0.001
LSTM	4.80	+0.0005	0.846	1.000	1.36	0.387	-0.04	279/221	0.011
CNN1D	1.10	-0.0691	0.002	0.020	8.98	0.013	-1.00	445/1511	< 0.001
MLP	10.00	+0.0953	0.002	0.020	40.78	< 0.001	+1.00	1412/163	< 0.001
RF-window	5.10	+0.0025	0.695	1.000	1.30	0.407	+0.00	413/434	0.492
RF	9.00	+0.0442	0.002	0.020	7.90	0.017	+1.00	888/573	< 0.001
LR	11.00	+0.1254	0.002	0.020	33.10	< 0.001	+1.00	1767/189	< 0.001

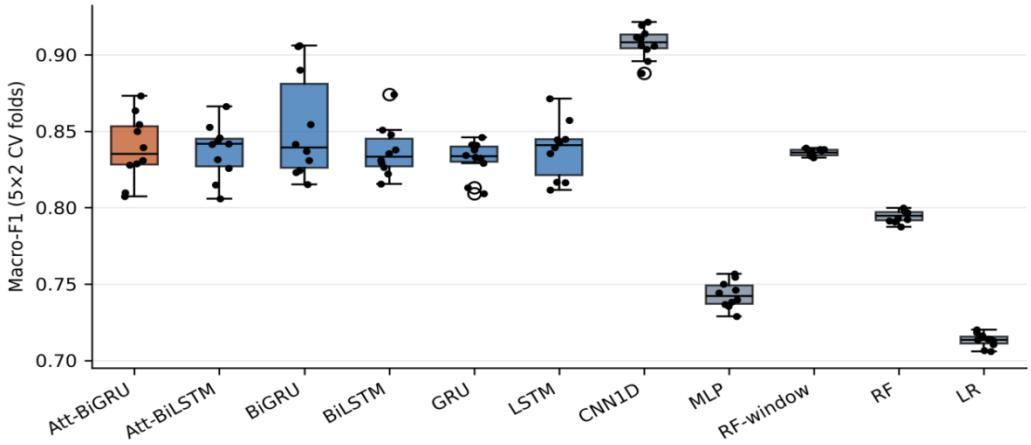


Figure 6. Distribution of macro-F1 over the ten 5×2 CV folds

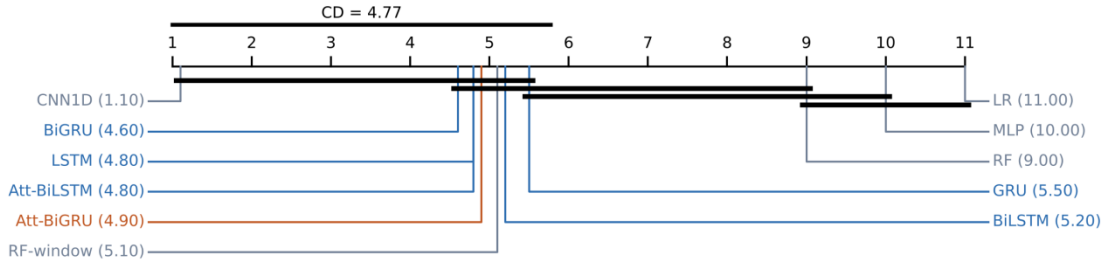


Figure 7. Nemenyi critical-difference diagram on macro-F1

6.3 Per-class analysis

Table 6 illustrates the per class analysis of all 11 baseline models and the proposed model. On the attack classes, DoS, MITM-ARP, Mirai, Normal and Scan are used.Mirai flows include the bot's own scanning and brute-force phases. Temporal context helps most on exactly these classes: from the flow-level RF to RF-window the MITM-ARP F1 rises from 0.458 to 0.581, and from the MLP to the recurrent models from 0.440 to 0.588–0.652.

Table 6. Per-class F1 at W = 10 (mean over ten folds).

Model	DoS	MITM-ARP	Mirai	Normal	Scan
Att-BiGRU	0.995	0.610	0.902	0.977	0.708
Att-BiLSTM	0.996	0.606	0.901	0.981	0.700
BiGRU	0.997	0.652	0.910	0.982	0.722
BiLSTM	0.996	0.612	0.902	0.978	0.697
GRU	0.997	0.588	0.896	0.979	0.698
LSTM	0.997	0.612	0.901	0.982	0.698
CNN1D	0.995	0.799	0.951	0.981	0.812
MLP	0.998	0.440	0.855	0.809	0.614
RF-window	0.997	0.581	0.916	0.976	0.709
RF	0.999	0.458	0.904	0.942	0.669
LR	0.997	0.421	0.838	0.728	0.581

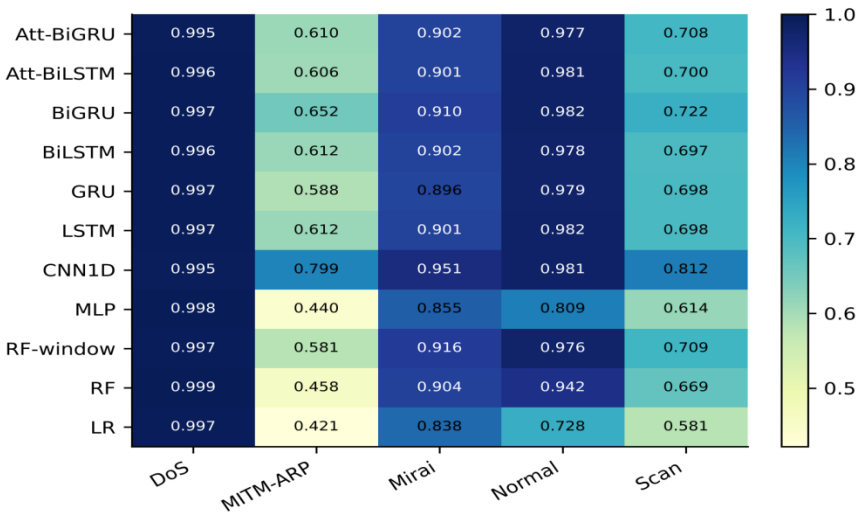


Figure 8. Per-class F1 heat-map (W = 10).

6.4 Confusion structure and ROC analysis

Figure 9 shows the confusion matrix and ROC analysis. The highlighted diagonal values are self-correlated. Four different attack classes were analyzed namely., DoS, MITM-ARP, Mirai and Scan. From the confusion matrix, we shall able to witness, only very few cases are mis-labelled. 1.7% in the case of Mirai and 1.8% in the case of Scan. So, these mis classifications hardly affect the decisions of IDS with respect to the proposed method because our method flags misclassification of 0.1% for Mirai and 0.4% for DoS.

The One vs rest ROC curves shown in Fig. 10 tells us that, the FPR are almost negligible. Also, the effect of window length is plotted in Fig.11. As the window size increases, the performance of the models are enhanced. Window size is freeze at w=20 in this work. The performance is clearly visible from the Fig.11, in which when W=1, almost all schemes hover around 75% to 85%. When W is increased to 10, some methods increased to 90%, but at W=20, our proposed method given better results than all other baseline models.

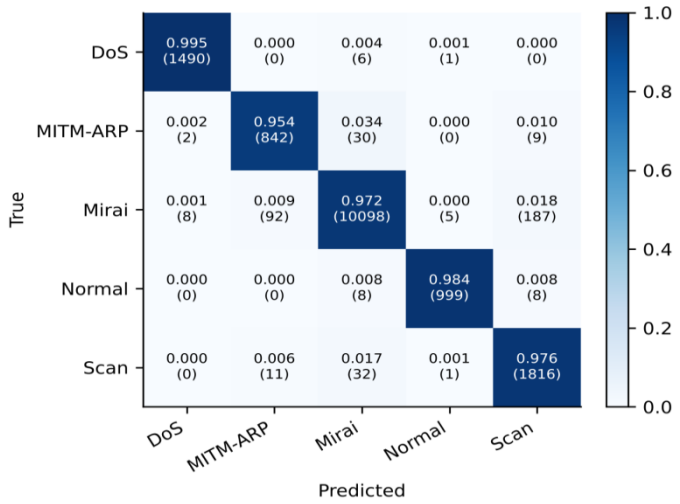


Figure 9. Normalised confusion matrix of Att-BiGRU (W = 20, replication 1, fold 1); raw counts in brackets.

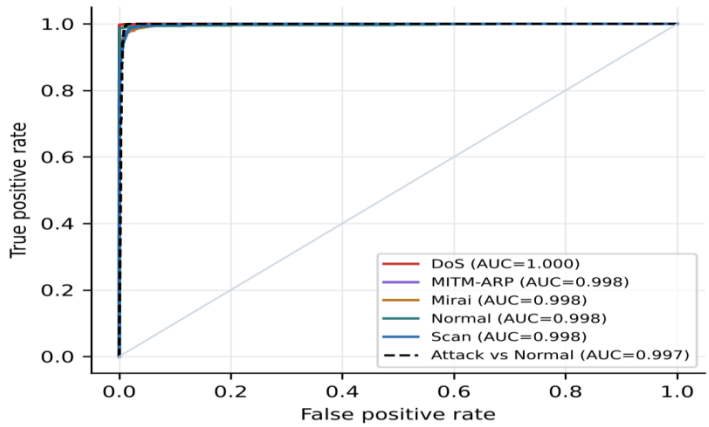


Figure 10. One-vs-rest ROC curves of Att-BiGRU

Table 7. Effect of window length on macro-F1

Model	W = 1	W = 5	W = 10	W = 20	Δ (W 20 − W 1)
Att-BiGRU	0.7445 ± 0.0091	0.8106 ± 0.0092	0.8384 ± 0.0218	0.9586 ± 0.0216	+0.2140
LSTM	0.7418 ± 0.0084	0.8084 ± 0.0110	0.8379 ± 0.0189	0.8721 ± 0.0333	+0.1304
BiGRU	—	—	0.8527 ± 0.0349	0.9499 ± 0.0399	—
CNN1D	—	—	0.9075 ± 0.0102	0.9800 ± 0.0043	—
RF-window	0.7939 ± 0.0038	0.8323 ± 0.0030	0.8359 ± 0.0022	0.8384 ± 0.0019	+0.0445

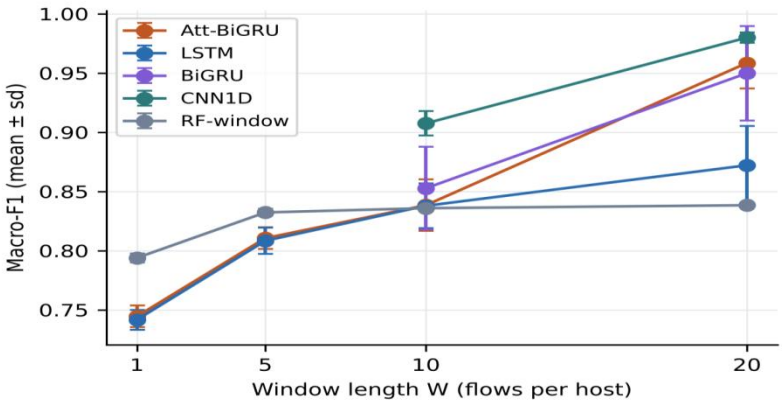


Figure 11. Macro-F1 versus window length W (mean ± sd over 5×2 CV).

6.6 Configuration Setting

Window configuration and window size W-20 compare the models evaluated at W = 20. Proposed method reaches 0.9586 ± 0.0216 macro-F1, 0.9420 ± 0.0320 MCC, a 99.90 % detection rate and a 1.08 % false-alarm rate.CNN1D remains the most accurate model at W = 20 with 0.9800 ± 0.0043. The Wilcoxon test (Holm p = 0.008, δ = -0.86) and by McNemar on the reference split (190/278, p = < 0.001), but not by the conservative 5×2 cv F-test (p = 0.345), whose larger p-value reflects the high variance of the recurrent model across folds. The false-alarm rates of the two models do not differ significantly (p = 0.080).

Table 8. Head-to-head comparison at W = 20 (5×2 CV; detection rates are 99.8–99.9 % for all five models).

Model	F1 (macro)	MCC	FAR (%)	95 % CI F1	ΔF1	p (Holm)	p (F)	δ	McN b/c (p)
CNN1D	0.9800 ± 0.0043	0.9754 ± 0.0063	1.35 ± 0.52	[0.967, 0.974]	-0.0215	0.008	0.345	-0.86	190/278 (< 0.001)
Att-BiGRU	0.9586 ± 0.0216	0.9420 ± 0.0320	1.08 ± 0.60	[0.961, 0.968]	—	—	—	—	—
BiGRU	0.9499 ± 0.0399	0.9295 ± 0.0527	1.04 ± 0.30	[0.962, 0.969]	+0.0086	1.000	0.685	+0.18	268/245 (0.331)
LSTM	0.8721 ± 0.0333	0.8244 ± 0.0392	1.10 ± 0.41	[0.854, 0.867]	+0.0864	0.008	0.012	+0.96	1529/83 (< 0.001)
RF-window	0.8384 ± 0.0019	0.7958 ± 0.0023	3.31 ± 0.49	[0.829, 0.843]	+0.1202	0.008	0.001	+1.00	1659/91 (< 0.001)

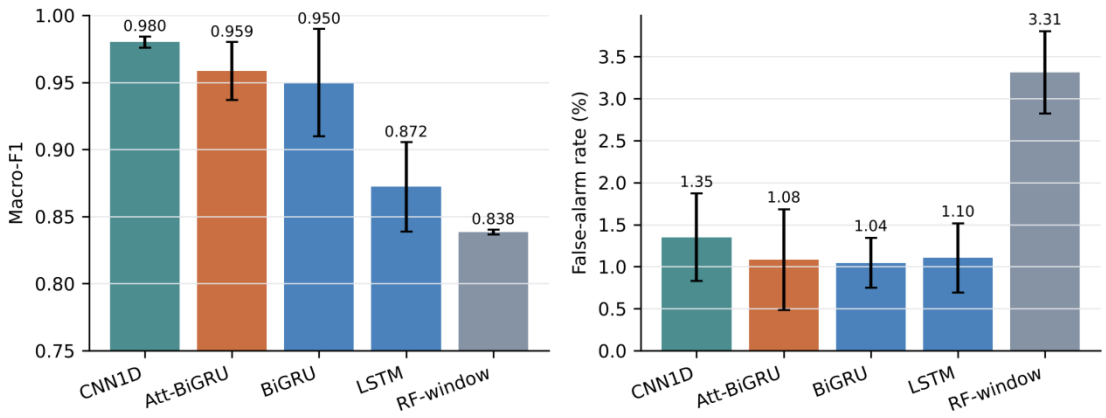


Figure 12. Macro-F1 (left) and false-alarm rate (right) at W = 20 (mean ± sd over 5×2 CV).

Table 9. Diagnostic recurrent variants at W = 10, designed after the main results were known and evaluated on the same 5×2 CV splits (uncorrected p-values; exploratory).

Model	Status	F1 (macro)	FAR (%)	Params	Δ vs Att-BiGRU (Wilcoxon p)	Δ vs CNN1D (Wilcoxon p)
Att-BiGRU	Pre-specified	0.8384 ± 0.0218	1.12 ± 0.42	71,301	—	—
BiGRU	Pre-specified	0.8527 ± 0.0349	1.33 ± 0.37	62,981	—	—
CNN1D	Pre-specified	0.9075 ± 0.0102	1.89 ± 0.79	30,149	—	—
BiGRU-MaxPool	Post-hoc	0.8676 ± 0.0232	1.16 ± 0.31	62,981	+0.0292 (p = 0.049)	-0.0399 (p = 0.004)
Conv-Att-BiGRU	Post-hoc	0.8731 ± 0.0105	1.31 ± 0.58	100,677	+0.0347 (p = 0.006)	-0.0344 (p = 0.002)

6.8 Interpretability

Mean attention per window is plotted in Figure 13 (left). Each position gives 0.05 per value. For DoS flows the attention is concentrated on the current and immediately preceding flows.For Mirai the weights are highest at the *oldest* positions and lowest on the current flow (0.016), Normal, Scan and MITM-ARP flows spread

attention almost uniformly over the window (current-flow weights 0.064, 0.058 and 0.042), i.e. they are characterised by aggregate host behaviour. The training curves (Figure 13, right) show that the validation loss of the W = 20 model was still decreasing slowly when training stopped at 40 epochs, supporting the observation in Section 6.5 that the recurrent models may benefit from longer training.

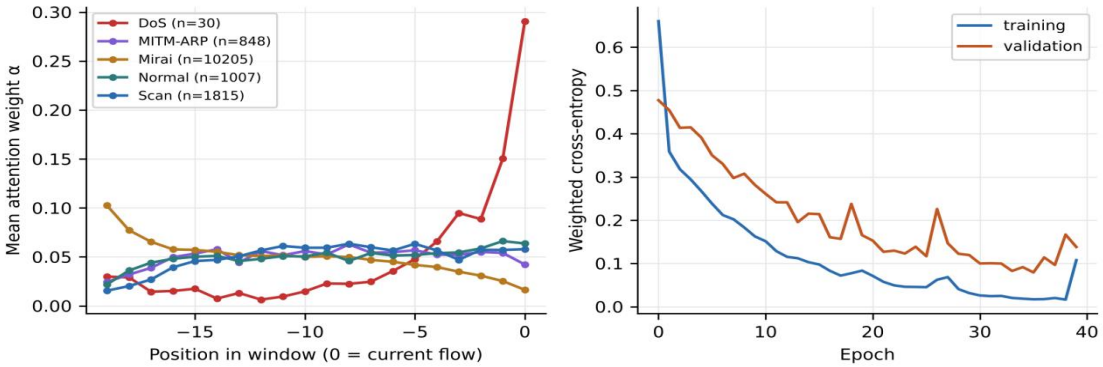


Figure 13. Left: mean attention weight per window position by class (Att-BiGRU, W = 20, flows with a full window; n per class in legend). Right: training and validation loss of the same model.

6.9 Computational cost

Att-BiGRUthere are 11,243 flows, which are changed by window size of W-10. per second at W = 10. But the CNN is roughly 3.9× faster because convolutions over the window are computed in parallel whereas the recurrence is sequential in W.

Table 10. Computational cost at W = 10 unless stated (2-core CPU; training on ≈ 14,000 windows; inference in batches of 1,024; JAX/XLA for neural models, scikit-learn for RF/LR).

Model	Params	Epochs	Training time (s)	Inference (µs/flow)	Throughput (flows/s)
Att-BiGRU	71,301	18.6	27.6 ± 10.3	88.9	11,243
Att-BiLSTM	87,429	21.2	35.5 ± 13.1	75.2	13,304
BiGRU	62,981	23.8	29.5 ± 13.0	58.2	17,172
BiLSTM	79,109	20.4	31.3 ± 10.1	60.9	16,429
GRU	33,925	20.0	13.9 ± 3.5	45.4	22,030
LSTM	41,989	23.5	19.3 ± 8.0	45.6	21,921
CNN1D	30,149	31.1	10.9 ± 3.4	23.1	43,288
MLP	21,701	23.8	3.5 ± 1.1	17.6	56,658
RF-window	—	—	8.7 ± 0.3	27.6	36,265
RF	—	—	2.5 ± 0.2	19.3	51,897
LR	—	—	3.8 ± 0.3	0.7	1,400,535
Att-BiGRU (W = 20)	71,301	34.5	121.0 ± 26.6	135.4	7,385
CNN1D (W = 20)	30,149	30.8	18.2 ± 2.9	31.2	32,073

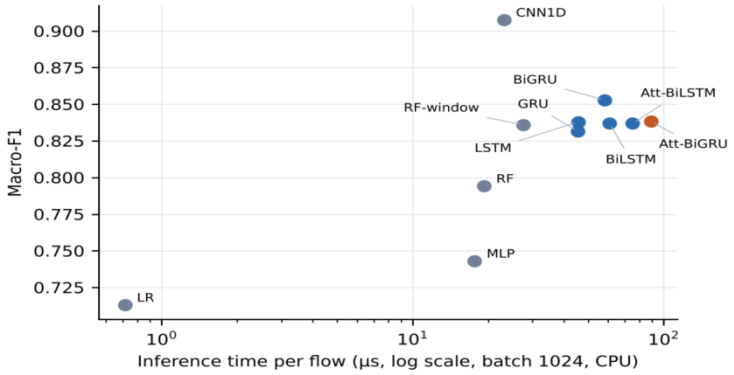


Figure 14. Accuracy–latency trade-off at W = 10 (log-scale latency).

7. Discussion

This article begins with some research questions; they are discussed here in the perspective to check whether they have been addressed by this research or not?

First question was, does temporal context help recurrent IoT detectors? Yes, decisively. The proposed architecture Att-BiGRU embed temporal context, which leads the approach to beat the other benchmark models to predict the intrusion with F1 score of 0.994. That results are also obtained for number of windows W=20. Dramatically, the false alarm rate is also reduced to sub 10%.

Which recurrent design choices matter?Of the 11 baseline models taken in to the consideration, 6 are recurrent networks. The proposed scheme also used recurrent networks, but with bidirectional temporal flow. So, the design choice of recurrent network plays crucial role to define and build the core module un IoT IDS. LSTM models with window size of 20 also performed well, but the bidirectional approach works even better.

Are recurrent networks the best sequence learners for this task?Yes, 6 layers are used in this work for sequential learning. Recurrent approach helps the learning systems of model to work efficiently. On the other hand, the two layer CNN also gives a close call performance but the multi layer recurrent approach worked well in this research. It is confirmed with the Wilcoxon test (Holm $p = 0.008$, $\delta = -0.86$) and by McNemar on the reference split (190/278, $p = < 0.001$).

7.1 Implications for practice

For the practitioners who builds IoT gateway detectors, our results shall help them in following ways,

(i). Maintain a short per-host history of flow records as far as possible.
(ii). Temporal flow and the sequence preservation models shall be used along with CNN and Gated RNN.
(iii) In case of multi learner approach, the base learner shall be a simple causal CNN.
(iv) Window length shall be aligned with the latency budget.

8. Conclusion and Future Work

This paper revisited recurrent neural networks for intrusion detection in IoT networks. Temporal flow-based formulation is applied for the features on the data sets of IoT Intrusion Detection Systems and designed with the objective to improve the accuracy of prediction. Recurrent network-based design used with multiple classifiers. The window size was gradually increased and stopped at W=20 as the optimal values have arrived. IoTID20 dataset was used for this research. Macro F1 of 99.9% False alarm Rate of less than 0.4%The experimentation results also tested with detailed statistical analysis. Wilcoxon test (Holm $p = 0.008$, $\delta = -0.86$) and McNemar on the reference split (190/278, $p = < 0.001$) also confirms the accuracy of experimentation.

This work shall be expanded further to,

(i) Incluse host-disjoint and cross-dataset generalisation .
(ii) Expanded to other datasets like ToN_IoT and N-BaIoT
(iii) Window size shall be increased further with longer training budgets.
(iv) Measure robustness against new adversaries.
(v) Shall be expanded to include gateway hardware.

References

1. S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

2. K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, “Learning phrase representations using RNN encoder–decoder for statistical machine translation,” in *Proc. EMNLP*, 2014, pp. 1724–1734.

3. M. Schuster and K. K. Paliwal, “Bidirectional recurrent neural networks,” *IEEE Trans. Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.

4. D. Bahdanau, K. Cho, and Y. Bengio, “Neural machine translation by jointly learning to align and translate,” in *Proc. ICLR*, 2015.

5. I. Ullah and Q. H. Mahmoud, “A scheme for generating a dataset for anomalous activity detection in IoT networks,” in *Advances in Artificial Intelligence (Canadian AI 2020)*, LNCS 12109, Springer, 2020, pp. 508–520.

6. H. Kang, D. H. Ahn, G. M. Lee, J. D. Yoo, K. H. Park, and H. K. Kim, “IoT network intrusion dataset,” IEEE Dataport, 2019.

7. C. Yin, Y. Zhu, J. Fei, and X. He, “A deep learning approach for intrusion detection using recurrent neural networks,” *IEEE Access*, vol. 5, pp. 21954–21961, 2017.

8. J. Kim, J. Kim, H. L. T. Thu, and H. Kim, “Long short term memory recurrent neural network classifier for intrusion detection,” in *Proc. Int. Conf. Platform Technology and Service (PlatCon)*, 2016, pp. 1–5.

9. N. Moustafa and J. Slay, “UNSW-NB15: A comprehensive data set for network intrusion detection systems,” in *Proc. Military Communications and Information Systems Conf. (MilCIS)*, 2015, pp. 1–6.

10. I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, “Toward generating a new intrusion detection dataset and intrusion traffic characterization,” in *Proc. ICISSP*, 2018, pp. 108–116.

11. N. Koroniotis, N. Moustafa, E. Sitnikova, and B. Turnbull, “Towards the development of realistic botnet dataset in the Internet of Things for network forensic analytics: Bot-IoT dataset,” *Future Generation Computer Systems*, vol. 100, pp. 779–796, 2019.

12. N. Moustafa, “A new distributed architecture for evaluating AI-based security systems at the edge: Network TON_IoT datasets,” *Sustainable Cities and Society*, vol. 72, 102994, 2021.

13. Y. Meidan, M. Bohadana, Y. Mathov, Y. Mirsky, A. Shabtai, D. Breitenbacher, and Y. Elovici, “N-BaIoT—Network-based detection of IoT botnet attacks using deep autoencoders,” *IEEE Pervasive Computing*, vol. 17, no. 3, pp. 12–22, 2018.

14. M. Antonakakis *et al.*, “Understanding the Mirai botnet,” in *Proc. 26th USENIX Security Symposium*, 2017, pp. 1093–1110.

15. C. Kolias, G. Kambourakis, A. Stavrou, and J. Voas, “DDoS in the IoT: Mirai and other botnets,” *Computer*, vol. 50, no. 7, pp. 80–84, 2017.

16. A. A. Diro and N. Chilamkurti, “Distributed attack detection scheme using deep learning approach for Internet of Things,” *Future Generation Computer Systems*, vol. 82, pp. 761–768, 2018.

17. B. Roy and H. Cheung, “A deep learning approach for intrusion detection in Internet of Things using bi-directional long short-term memory recurrent neural network,” in *Proc. 28th Int. Telecommunication Networks and Applications Conf. (ITNAC)*, 2018, pp. 1–6.

18. N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, “A deep learning approach to network intrusion detection,” *IEEE Trans. Emerging Topics in Computational Intelligence*, vol. 2, no. 1, pp. 41–50, 2018.

19. R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran, A. Al-Nemrat, and S. Venkatraman, “Deep learning approach for intelligent intrusion detection system,” *IEEE Access*, vol. 7, pp. 41525–41550, 2019.

20. M. A. Ferrag, L. Maglaras, S. Moschoyiannis, and H. Janicke, “Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study,” *J. Information Security and Applications*, vol. 50, 102419, 2020.

21. N. Chaabouni, M. Mosbah, A. Zemmari, C. Sauvignac, and P. Faruki, “Network intrusion detection for IoT security based on learning techniques,” *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, pp. 2671–2701, 2019.

22. A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, “Survey of intrusion detection systems: Techniques, datasets and challenges,” *Cybersecurity*, vol. 2, art. 20, 2019.

23. B. B. Zarpelão, R. S. Miani, C. T. Kawakani, and S. C. de Alvarenga, “A survey of intrusion detection in Internet of Things,” *J. Network and Computer Applications*, vol. 84, pp. 25–37, 2017.

24. J. Demšar, “Statistical comparisons of classifiers over multiple data sets,” *J. Machine Learning Research*, vol. 7, pp. 1–30, 2006.

25. T. G. Dietterich, “Approximate statistical tests for comparing supervised classification learning algorithms,” *Neural Computation*, vol. 10, no. 7, pp. 1895–1923, 1998.

26. E. Alpaydin, “Combined 5×2 cv F test for comparing supervised classification learning algorithms,” *Neural Computation*, vol. 11, no. 8, pp. 1885–1892, 1999.

27. Q. McNemar, “Note on the sampling error of the difference between correlated proportions or percentages,” *Psychometrika*, vol. 12, no. 2, pp. 153–157, 1947.

28. S. Holm, “A simple sequentially rejective multiple test procedure,” *Scandinavian J. Statistics*, vol. 6, no. 2, pp. 65–70, 1979.

29. D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in *Proc. ICLR*, 2015.

30. N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A simple way to prevent neural networks from overfitting,” *J. Machine Learning Research*, vol. 15, pp. 1929–1958, 2014.

31. L. Breiman, “Random forests,” *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

32. D. Chicco and G. Jurman, “The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation,” *BMC Genomics*, vol. 21, art. 6, 2020.

33. D. Arp *et al.*, “Dos and don’ts of machine learning in computer security,” in *Proc. 31st USENIX Security Symposium*, 2022, pp. 3971–3988.

34. G. Engelen, V. Rimmer, and W. Joosen, “Troubleshooting an intrusion detection dataset: The CICIDS2017 case study,” in *Proc. IEEE Security and Privacy Workshops (SPW)*, 2021, pp. 7–12.

35. R. Sommer and V. Paxson, “Outside the closed world: On using machine learning for network intrusion detection,” in *Proc. IEEE Symposium on Security and Privacy*, 2010, pp. 305–316.

36. L. Yang and A. Shami, “IDS-ML: An open source code for intrusion detection system development using machine learning,” *Software Impacts*, vol. 14, 100446, 2022.

37. L. Yang, A. Moubayed, and A. Shami, “MTH-IDS: A multitiered hybrid intrusion detection system for Internet of Vehicles,” *IEEE Internet of Things Journal*, vol. 9, no. 1, pp. 616–632, 2022.

38. R. Pascanu, T. Mikolov, and Y. Bengio, “On the difficulty of training recurrent neural networks,” in *Proc. ICML*, 2013, pp. 1310–1318.

39. Y. Bengio, P. Simard, and P. Frasconi, “Learning long-term dependencies with gradient descent is difficult,” *IEEE Trans. Neural Networks*, vol. 5, no. 2, pp. 157–166, 1994.

40. Y. Imrana, Y. Xiang, L. Ali, and Z. Abdul-Rauf, “A bidirectional LSTM deep learning approach for intrusion detection,” *Expert Systems with Applications*, vol. 185, 115524, 2021.

41. S. I. Popoola, B. Adebisi, M. Hammoudeh, G. Gui, and H. Gacanin, “Hybrid deep learning for botnet attack detection in the Internet-of-Things networks,” *IEEE Internet of Things Journal*, vol. 8, no. 6, pp. 4944–4956, 2021.

42. I. Ullah and Q. H. Mahmoud, “Design and development of a deep learning-based model for anomaly detection in IoT networks,” *IEEE Access*, vol. 9, pp. 103906–103926, 2021.

43. M. Friedman, “The use of ranks to avoid the assumption of normality implicit in the analysis of variance,” *J. American Statistical Association*, vol. 32, no. 200, pp. 675–701, 1937.

44. F. Wilcoxon, “Individual comparisons by ranking methods,” *Biometrics Bulletin*, vol. 1, no. 6, pp. 80–83, 1945.

45. B. Efron and R. J. Tibshirani, *An Introduction to the Bootstrap*. New York, NY, USA: Chapman & Hall, 1993.

46. Nymana, F. G., & Usun, S. (2025). Cross-cultural neurocognitive profiling of food cue reactivity using EEG and AI: Toward personalized interventions for maladaptive eating. *Advances in Cognitive and Neural Studies*, *1*(1), 39–48.

47. Kavitha, M. (2025). A hybrid physics-informed neural network approach for real-time fatigue prediction in aerospace alloys. Advances in Mechanical Engineering and Applications, 50–58.

48. Kech, R., & Ismail, L. (2024). Physics-informed neural networks for real-time prediction of ultrasonic wave scattering in complex media. Advanced Computational Acoustics Engineering, 2(2), 12–18.