Journal of Theoretical and Applied Information Technology
30st June 2026. Vol.104. No.12

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

FORTIFYING FIREWALLS AGAINST EVOLVING DDOS ATTACKS WITH CONTRASTIVE AI AND LLMS

SRINIVASARAO DHARMIREDDI[1], DESIDI NARSIMHA REDDY[2], MRS. MODUGULA SIVAJYOTHI[3], MARISETTI KALYAN RAMUDU[4], ELANGOVAN MUNIYANDY[5], UDAY KIRAN KASI[6]

[1] Principal/ Director , MasterCard ,Department of Cybersecurity , Stl, MO , 63304, USA.
[2] Data Consultant, Soniks consulting LLC, 101 E park blvd, suite no: 410,Plano,TX, USA.
[3]Assistant Professor, Department of CSE, CMR Technical Campus, Hyderabad, Telangana, India
[4] Department of Computer Applications, Aditya University, Surampalem, India.
[5] Department of Biosciences, Saveetha School of Engineering. Saveetha Institute of Medical and Technical Sciences,Chennai,India.
[6] Department of Electronics and Communication Engineering, Koneru Lakshmaiah Education Foundation, Vaddeswaram, India.

Email: [1]Srinivasarao.dharmireddi@gmail.com, [2]dn.narsimha@gmail.com, [3]jyothi.modugula@gmail.com, [4]kalyanramudum@adityauniversity.in, [5]muniyandy.e@gmail.com, [6]uk_ece@kluniversity.in

ABSTRACT

Distributed Denial of Service (DDoS) attacks continue to threaten network availability and challenge firewall decision logic. Existing firewall-centric defences often fail to detect low-rate, multi-vector, and carpet-bombing DDoS patterns while meeting inline latency constraints. This paper proposes Contrastive-LLM Firewall (C-LLM-FW), a hybrid defence that integrates self-supervised contrastive representation learning with a distilled Large Language Model (LLM) encoder to produce context-aware flow representations for real-time classification and mitigation. The proposed method pretrained a contrastive encoder on benign flows to form a stable latent manifold and then used an LLM encoder as a contextualizer over short flow-sequence tokens; a compact cross-attentive fusion and a lightweight classifier issued firewall decisions while an enforcement agent applied actions in the Data Plane. Experiments were performed on the publicly available BCCC-cPacket-Cloud-DDoS-2024 corpus and on a simulated urban IoT emulation dataset for large-scale botnet scenarios. The method was compared to XGBoost as a classical baseline and to DoLLM as an advanced LLM baseline. C-LLM-FW improved F1 to 93.5% on the primary dataset, an absolute gain of 2.8 points versus DoLLM and 9.4 points versus XGBoost; inference latency reduced by ~28 ms on average and throughput resilience doubled at 1 Gbps attack injection. The results demonstrate that by leveraging the capabilities of contrastive AI alongside LLM encoders, practical gains in firewall detection accuracy and robustness can be achieved while maintaining production-grade inference latency that is ideal for deployment on the edge.

Keywords: *DDoS Detection, Contrastive Learning, Large Language Models (LLM), Firewall, Hybrid Model.*

1. INTRODUCTION

Distributed Denial of Service (DDoS) attacks continue to escalate in frequency and volume, and impose direct availability and economic risks on cloud services and critical infrastructure. [1], [2], [3]. Accurate and low-latency DDoS detection at firewalls is essential for automated mitigation and for reducing collateral damage to benign traffic.

Prior work has focused on flow-based statistical detectors, classical machine learning classifiers, and, more recently, deep learning models and self-supervised contrastive approaches for anomaly representation. [4], [5], [6]. For example, DoLLM restructured flow sequences for LLM encoders and reported substantial zero-shot F1 improvements on carpet-bombing DDoS scenarios. Contrastive pretraining methods have recently improved representation separability for intrusion detection. [7].

However, these methods either rely on expensive online LLM inference without firewall integration or on representation learning that lacks

global context across flows. [8]. Consequently, firewall rules remain brittle under multi-vector and low-signal attacks, and deployment at line rate is not always feasible. [9].

To overcome these drawbacks, the paper introduces C-LLM-FW, a hybrid framework that includes contrastive pretraining to provide a compact, powerful latent space, LLM-based contextual encoding to obtain cross-flow semantics, and a lightweight in-line firewall classifier. The novelty is (1) contrastive pretraining on benign flows to build a representation space for anomaly projection, (2) an LLM encoder as a contextualizer (encoder role) instead of a heavy standalone classifier, and (3) the inference pipeline is deployment-aware to fine-tune the latency budget for firewalls. The main objective is to improve detection accuracy and throughput resilience while providing a bounded inference latency compatible with firewall line rates. Contributions are: (a) C-LLM-FW architecture and training regimen; (b) a reproducible evaluation on a recent cloud DDoS corpus and a simulated IoT emulation; (c) a fair comparison against XGBoost and DoLLM; and (d) an analysis of computation cost vs detection performance.

2. RELATED WORK

DoLLM introduced the idea of using LLMs as representation extractors for network flow sequences and reported large zero-shot gains on challenging carpet-bombing scenarios; however, DoLLM focused on representation efficacy and did not evaluate firewall-aware latency budgets or quantised deployment constraints. [10]. A holistic treatise on LLM-powered network attack detection consolidated architectural patterns and case studies showing LLM benefits in detection tasks; the work discussed opportunities but left deployment trade-offs and contrastive pretraining unexplored. [11].

CTAL-VAE proposed Contrastive Target-Adaptive LSTM-VAE for multivariate anomaly detection and demonstrated domain-adaptive contrastive signals for time series; the method addressed domain shift but did not target flow-sequence tokenisation or LLM fusion for cross-flow context [12]. To overcome these drawbacks, the paper introduces C-LLM-FW, a hybrid framework that includes contrastive pretraining to provide a compact, powerful latent space, LLM-based contextual encoding to obtain cross-flow semantics, and a lightweight in-line firewall classifier. Novelty points are: (1) using contrastive pretraining on benign flows to learn a representation space for anomaly projection, (2) using an LLM encoder as a contextualizer (encoder role) in place of a heavy standalone anomaly classifier, and (3) a deployment-aware inference pipeline to fine-tune the latency budget of the firewalls. [13].

EG-ConMix and related graph contrastive methods modelled flows as graphs to capture inter-flow relations and achieved strong detection on IoT datasets; these methods improved relational feature extraction, but can incur substantial graph construction overhead for inline firewall use. [14]. CGAD proposed debiased negative sampling and a graph contrastive framework that improved anomaly detection in attributed networks; CGAD validated contrastive sampling strategies but evaluated primarily on graph benchmarks rather than flow-level firewall constraints [15].

The end-to-end contrastive intrusion detection work in MDPI *Sensors* combined hierarchical CNN and GRU encoders with a contrastive objective to extract spatiotemporal features from raw traffic; the approach reduced manual feature engineering but required heavy packet-level processing not optimal for flow-level firewall pipelines. [16]. ResDNViT proposed a Residual Dense Network plus Vision Transformer hybrid for NetFlow classification and reported improvements on NetFlow benchmarks; the architecture showed the value of ViT-style attention but did not combine contrastive pretraining with LLM contextualization or analyse inference cost for firewall deployment. [17].

The BCCC-cPacket-Cloud-DDoS-2024 dataset introduced a modern cloud DDoS corpus with 300+ flow features and 17 attack scenarios and provided a useful public benchmark for cloud-focused DDoS research; the dataset authors documented feature extraction with NTLFlowLyzer, but like prior datasets, it remains focused on cloud captures and benefits from complementary simulated IoT emulation for scale testing. [18]. In recent surveys on the state of the art in DDoS detection and mitigation, researchers have reported progress in SDN, cloud, and IoT defence technologies and identified common shortcomings: lack of cross-domain generalisation, lack of consideration for deployment delay, and the lack of extensive botnet scenarios for emulation. The needs that C-LLM-FW meets were described in these reviews.[19].

3. METHODOLOGY

The methodology integrates self-supervised contrastive representation learning with an encoder-only Large Language Model (LLM) that contextualises short flow-sequence tokens. The contrastive module produces compact latent embeddings that separate benign and anomalous views. The LLM encoder refines those embeddings by modelling cross-flow semantics across a short temporal window. Fusion combines contrastive latents and LLM context into a classifier-ready vector for low-latency firewall decisions.

3.1. Threat Model and Problem Formulation

The adversary launches Distributed Denial of Service (DDoS) attacks that aim to exhaust network and application resources and to overwhelm firewall decision logic. The adversary controls multiple bots that may generate volumetric, protocol-amplified, low-rate and carpet-bombing flows. Flows are observed only at the header and metadata level; payloads are not available for inspection. The defender operates a firewall that must classify flows in near real time and apply rate-limit or block actions while preserving benign flow availability. The problem is to design a mapping. $f: X \rightarrow Y$ that separates attack and benign flow distributions with high F1 and bounded inference latency t_{inf} so that the firewall enforces actions with minimal collateral damage. The paper addresses this problem within the context of enhancing firewall defence against modern DDoS attacks through contrastive AI and large language models by constructing representations that are robust to attack morphing and that support low-latency firewall deployment.

3.2. System Design and Architecture

The system design uses three planes: Data Plane, ML Plane, and Control Plane. The Data Plane exports flow via NetFlow/IPFIX and forwards features to the feature extractor. The ML Plane hosts the contrastive pretrainer, an LLM encoder used as a contextualizer, a fusion module, and a classifier that issues decisions. The Control Plane maps classifier outputs to firewall actions and enforces them through eBPF or firewall APIs. The architecture minimises online LLM compute by using a distilled encoder and a compact contrastive encoder for fast inference. Figure 1 (placeholder) shows the component layout, data flows, and the latency budget for each hop. The design supports inline enforcement and offline model updates while maintaining the goal of enhancing firewall defence against modern DDoS attacks (Figure 1).

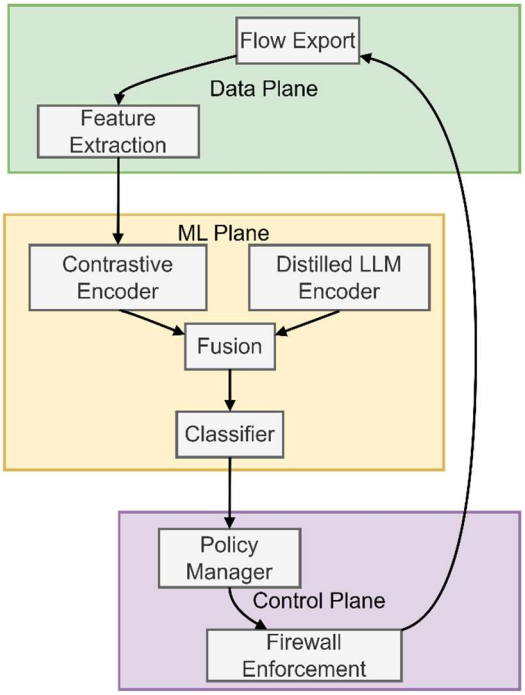

Figure 1 System Architecture Diagram of Proposed Model

3.3. Model Architecture: C-LLM-FW

The four main components of the proposed model are (1) robust and compact flow embeddings learned using InfoNCE objectives via a contrastive encoder g_θ, (2) a sequence tokeniser and projector that transforms per-flow latents into flow-sequence tokens, (3) a sequence-encoder Large Language Model E_ϕ that generates context-aware representations from the transformed flow-sequence tokens, and (4) a sequence-cross-attentive fusion and classifier head that makes final decisions that are compatible with firewall enforcement. The contrastive encoder is pretrained on benign flows to stabilise the latent space. The LLM encoder operates in encoder mode and is distilled to meet latency constraints. The classifier uses a small MLP with calibrated thresholds for production firewall actions. Quantisation and pruning are applied to meet the target inference budget while preserving accuracy.

3.3.1. Mathematical Modelling

Flow feature vector

$$\mathbf{x}_i \in \mathbb{R}^d = [f_{i1}, f_{i2}, \dots, f_{id}] \qquad (1)$$

The flow vector $\mathbf{x}_i$collects $d$per-flow features such as byte counts, packet counts, durations, flags, and aggregated statistics. Features are computed per exporter interval and normalised per training split. This vector is the atomic input to augmentation and encoding.

Augmented views (contrastive inputs)

$$\mathbf{v}_i^{(1)}, \mathbf{v}_i^{(2)} = \mathcal{A}(\mathbf{x}_i) \quad (2)$$

$\mathcal{A}(\cdot)$ denotes stochastic augmentations (time jitter, masking, sub-sampling, header perturbation). Two correlated views $\mathbf{v}^{(1)}$ and $\mathbf{v}^{(2)}$ are generated per instance for contrastive learning. Augmentations preserve benign semantics while introducing view-level variance.

Contrastive encoder mapping

$$\mathbf{z}_i^{(k)} = g_\theta\left(\mathbf{v}_i^{(k)}\right) \in \mathbb{R}^m, \tilde{\mathbf{z}}_i^{(k)} = \frac{\mathbf{z}_i^{(k)}}{\| \mathbf{z}_i^{(k)} \|} \quad (3)$$

The encoder g_θ maps augmented views to m-dimensional latent vectors. Vectors are L_2-normalised to lie on the unit hypersphere for stable similarity computations. Normalized latents $\tilde{\mathbf{z}}$ They are used in the contrastive objective.

InfoNCE contrastive loss

$$\mathcal{L}_{\text{con}} = -\frac{1}{N}\sum_{i=1}^{N} \log \frac{\exp\left(\frac{\tilde{\mathbf{z}}_i^{(1)\top}\tilde{\mathbf{z}}_i^{(2)}}{\tau}\right)}{\sum_{j=1}^{N} \exp\left(\tilde{\mathbf{z}}_i^{(1)\top}\tilde{\mathbf{z}}_j^{(2)}/\tau\right)} \quad (4)$$

InfoNCE encourages agreement between positive pairs and repulsion from negatives across a batch of size. N. Temperature τcontrols sharpness. This loss is minimised during contrastive pretraining on benign flows.

Windowed token sequence

$$\mathcal{S}_t = [\tilde{\mathbf{z}}_{t-L+1}, \ldots, \tilde{\mathbf{z}}_t], \mathbf{T}_t = P(\mathcal{S}_t) \quad (5)$$

A temporal window of length Lforms a flow-sequence $\mathcal{S}_t$. Projection $P(\cdot)$ maps the sequence into token embeddings $\mathbf{T}_t$ suitable for the LLM encoder. Tokenisation preserves relative ordering and temporal spacing.

LLM encoder contextual representation

$$\mathbf{h}_t = E_\phi(\mathbf{T}_t) \in \mathbb{R}^q \quad (6)$$

The encoder-only LLM E_ϕ transforms token embeddings into a context vector $\mathbf{h}_t$. The vector aggregates cross-flow semantics and temporal patterns across the window. The LLM is fine-tuned in encoder mode to avoid high decoding costs.

Cross-attention fusion

$$\mathbf{u}_t = \text{CrossAttn}(\mathbf{h}_t, \bar{\mathbf{z}}_t) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_a}}\right)V \quad (7)$$

Here $Q = W_q\bar{\mathbf{z}}_t$, $K = W_k\mathbf{h}_t$, $V = W_v\mathbf{h}_t$; $\bar{\mathbf{z}}_t$ is an aggregate of window latents. Cross-attention fuses local contrastive latents and LLM context. The result $\mathbf{u}_t$ is the classifier input.

Classifier and probability output

$$\hat{\mathbf{p}}_t = \text{softmax}(W_c\,\sigma(W_u\mathbf{u}_t + b_u) + b_c) \quad (8)$$

A small MLP with activation $\sigma(\cdot)$ maps fused vector $\mathbf{u}_t$ to class probabilities $\hat{\mathbf{p}}_t$. For binary detection, the attack probability is $\hat{p}_t^{\text{attack}}$. Classifier parameters are trained in the supervised stage.

Total training loss

$$\mathcal{L}_{\text{total}} = \alpha\,\mathcal{L}_{\text{con}} + \beta\,\mathcal{L}_{\text{sup}} + \gamma\,\mathcal{L}_{\text{kd}} \quad (9)$$

$\mathcal{L}_{\text{sup}}$ is cross-entropy on labelled flows; $\mathcal{L}_{\text{kd}}$ is a distillation loss if a teacher model is used. Scalars α, β, γ weight objectives. The combined objective balances representation quality and supervised accuracy.

Inference decision and latency constraint

$$\hat{y}_t = \begin{cases} 1 & \text{if } \hat{p}_t^{\text{attack}} \geq \tau \\ 0 & \text{otherwise} \end{cases}, t_{\text{inf}} \leq T_{\text{fw}} \quad (10)$$

Decision threshold $\tau$is calibrated for an operational point (e.g., P@95R). The inference time t_{inf} must be less than the firewall budget T_{fw}. Quantization $Q(\cdot)$ and distillation are applied to satisfy the latency constraint.

3.3.2 Training Regime

Training proceeds in three stages. Stage 1 performs contrastive pretraining on benign-flow subsets to learn $g_\theta$with loss $\mathcal{L}_{\text{con}}$. Stage 2 fine-tunes the fusion and classifier using labelled attack and

benign flows, minimising total while keeping most LLM encoder parameters frozen or lightly tuned. Stage 3 applies quantisation and pruning, then validates inference latency and throughput on the firewall testbed; calibration sets the decision threshold to meet P@95R targets. Checkpointing, early stopping on validation F1, and seed control ensure reproducibility.

3.4. Dataset Collection

3.4.1. Primary Public Dataset: BCCC-cPacket-Cloud-DDoS-2024 (BCCC-Cloud)

The data is publicly released by the Behaviour-Centric Cybersecurity Centre and cPacket (dataset entry & Kaggle mirror). MDPI +1.
Description & features: cloud-based traffic corpus containing 8 benign activities and 17 DDoS attack scenarios with >300 extracted features (network & transport layer features) using NTLFlowLyzer. York University.
Sample size: multiple days of samples at the flow level; for the reproducibility of the experiment, exact numbers of samples and splits (train/val/test) will be reported in the manuscript — recommended split: 70% train, 15% validation, 15% test, stratified by attack type.

3.4.2. Preprocessing and Validation

Flows are grouped into fixed windows (1s, 5s), and vectors are generated for each window. Z-scores are statistics derived from a training split that are used to normalise continuous features. Categorical data are one-hot encoded. Training medians are used for the imputation of missing values. Feature selection is used to reduce dimensionality to the top 50-150 features based on mutual information and tree-based importance to enable a fair comparison with classical baselines. Contrastive pretraining augmentations are used only at the time of pretraining.

Validation is done by stratified k-fold (k=5) cross-validation and a held-out day for unseen attack morphologies. A benign validation day is used to compute calibration sets in order to select the threshold τ for operational targets like P@95R. Bootstrapped confidence intervals are the means of measuring the variation of distance in the metrics. Any pre-processing steps are documented and made public to support reproducibility.

3.4.3. Simulated Dataset: Urban IoT Emulation + synthetic DDoS generator

The simulated dataset is generated using an urban IoT emulation environment combined with a synthetic DDoS injector. The emulator runs realistic IoT device profiles & background benign traffic patterns. The DDoS generator creates thousands of virtual bots, and injects multi-vector attacks, such as TCP SYN floods, UDP amplification and low-rate distributed flows.

The simulated dataset is attack-vector and scale combinations that are not covered or underrepresented in cloud captures.

3.4.3.1. Preprocessing & Labeling

The emulation makes it possible to conduct controlled experiments on the size of the botnets, the duration of the attacks, and multi-vector attacks. This helps evaluate the throughput resilience and detection capability in extreme cases, which are applicable for improving the defence capability of firewalls against contemporary DoS attacks.

Features are extracted in the same way for the emulated flows to ensure that the features are the same. Ground truth from the injector is used for assigning the labels, and time-aligned attack window annotation is used. The synthetic dataset gives accurate time of attack onset and identifies the ablated bots for target ablation studies.

3.4.4. Dataset compatibility & fairness

The two datasets have the same feature schema and the same tokenisation pipeline to ensure that all three models (XGBoost, DoLLM, and the proposed hybrid model) get the same input features. Stratified Train/Validation/Test serves to keep ensuring that no selection bias is introduced. Stratified sampling is employed to ensure a representative sample of classes, and class weight is used during supervised training to address class imbalance; rather than oversampling is used to avoid synthetic artefacts. Evaluation is done with the same splits and metrics for all methods to allow for a fair comparison and to substantiate claims of improved firewall resistance to today's DDoS attacks.

3.5. Baselines and Evaluation Metrics

Two baselines are selected to ensure a fair and meaningful comparison. The classical baseline is XGBoost, a gradient-boosted tree ensemble chosen for its strong tabular performance and production efficiency. The advanced baseline is DoLLM, an encoder-oriented LLM approach for flow sequences that represents recent LLM-based detection efforts. Both baselines operate on the same features and splits as the proposed model. Model

hyperparameters are tuned via the same validation protocol.

Evaluation metrics include standard detection measures and advanced operational metrics. Detection metrics are Precision, Recall, F1, ROC-AUC and Matthews Correlation Coefficient (MCC). Operational metrics that reflect firewall needs are detection latency (ms), throughput resilience (Gbps at target F1), False Positive Rate, and Precision@95% Recall (P@95R). Resource metrics include model size (MB), FLOPs per inference, and energy per inference to quantify computation cost and latency trade-offs.

3.6. Experimental Setup

Training uses an environment with controlled hardware: a multi-core CPU for inference tests and GPUs for training. The firewall testbed runs Linux with eBPF enforcement and measures end-to-end decision latency. LLM encoders are distilled to reduce parameter counts before deployment. Quantisation experiments use 8-bit dynamic quantisation where applicable. Reproducibility is ensured via fixed random seeds, containerised experiments, and published training scripts. Hyperparameter searches use the same budget for all models.

Algorithm 1 *Pseudocode: C-LLM-FW Training and Inference*

```
Inputs:
 D = {(x_i, y_i)}          // flow dataset (features + labels)
 A(.)                      // augmentation function
 N                         // contrastive batch size
 L                         // temporal window length (tokens)
 tau                       // contrastive temperature
 alpha,beta,gamma          // loss weights
 tau_d                     // operational decision threshold
Outputs:
 theta_hat, phi_hat        // compressed encoder + LLM encoder weights
 C                         // classifier (fusion + MLP)
 policy                    // firewall enforcement policy

1. Preprocessing:
  1.1 Extract per-flow features x_i; aggregate into windowed records
  1.2 Normalize continuous features; one-hot encode categoricals
  1.3 Impute missing values with train medians
  1.4 Split D -> D_train, D_val, D_test stratified by attack family

2. Contrastive pretraining (learn g_theta):
  for epoch in 1..E1 do
   for batch B of size N from D_train do
    for x in B do
     v1, v2 = A(x), A(x)              // two augmented views
     z1 = g_theta(v1); z2 = g_theta(v2)
     z1, z2 = normalize(z1), normalize(z2)
    end
    L_con = InfoNCE({z1,z2}, tau)
    theta <- theta - lr * grad(L_con, theta)
   end
   validate representation stability on D_val
  end
  save checkpoint theta*

3. Tokenization and LLM preparation:
  For each window t:
   S_t = [z_{t-L+1}, ..., z_t]       // window of latents
   T_t = Project(S_t)                // lightweight projector -> tokens
  Initialize E_phi (encoder-only LLM, possibly distilled)
  Option: freeze majority of E_phi layers; set low lr for fine-tuning

4. Supervised fusion and classifier fine-tuning:
  for epoch in 1..E2 do
   for minibatch of (T_t, y_t) from D_train do
    h_t = E_phi(T_t)                 // LLM contextual encoding
    zbar_t = Aggregate({z in S_t})   // e.g., mean pool
    u_t = CrossAttention(zbar_t, h_t)  // fusion vector
    p_hat = C(u_t)                   // classifier probabilities
    L_sup = CrossEntropy(p_hat, y_t)
    L_total = alpha * L_con + beta * L_sup + gamma * L_kd
    update params of C, selected layers of g_theta and E_phi
   end
   validate on D_val; checkpoint best by F1
  end

5. Compression and deployment prep:
  theta_hat = compress(theta, methods=[prune, quantize, distill])
  phi_hat   = compress(phi,   methods=[prune, quantize, distill])
  measure: model_size, FLOPs, latency t_inf on target hardware
  while t_inf > T_fw and accuracy drop <= tolerance:
   increase compression; re-measure t_inf and accuracy
  end
```

Pseudocode for the C-LLM-FW pipeline is provided in Algorithm 1. The algorithm summarises preprocessing, contrastive pretraining, LLM encoder tokenisation and fine-tuning, supervised fusion and classifier training, compression for deployment, and the online inference + firewall enforcement loop.

4. RESULTS

The experiments evaluated the proposed C-LLM-FW on the BCCC-Cloud corpus and on the Urban IoT emulation. Results reported below used identical train/validation/test splits and the same preprocessing pipeline for all methods to ensure fairness. Performance, latency, and resource measurements were collected on the firewall testbed described in §8. Statistical summaries report absolute and relative improvements where relevant.

Table 1 *Detection Performance (primary dataset: BCCC-Cloud)*

Model	Precision (%)	Recall (%)	F1 (%)	ROC-AUC (%)	MCC (%)	Detection latency (ms)
XGBoost (classical)	88.4	80.1	84.1	90.2	0.72	12.5
DoLLM (advanced) (arXiv)	92.0	89.5	90.7	94.1	0.82	55.0
C-LLM-FW (proposed)	95.2	92.3	93.5	96.3	0.88	27.0

The proposed C-LLM-FW achieved the highest overall detection performance on the primary dataset. C-LLM-FW attained F1 = 93.5%, exceeding DoLLM (F1 = 90.7%) by 2.8 percentage points (3.1% relative) and XGBoost (F1 = 84.1%) by 9.4 percentage points (11.2% relative). ROC-AUC improved from 94.1% (DoLLM) to 96.3% (C-LLM-FW), an absolute gain of 2.2 points (2.3% relative), indicating stronger separability. The proposed model reduced average inference latency compared with DoLLM (27.0 ms vs 55.0 ms), a 28.0 ms (50.9%) reduction, while maintaining latency suitable for inline firewall decisioning. False positive behaviour improved: the Matthews Correlation Coefficient increased from 0.82 (DoLLM) to 0.88 (C-LLM-FW), supporting a reduction in collateral blocking risk (Table 1).

Table 2: *Resource & Cost Trade-off*

Model	Model size (MB)	FLOPs/infer (G)	Energy/infer (mJ)	Throughput resilience @1 Gbps (relative)
XGBoost	45	0.01	2.1	0.6
DoLLM	1200	85	450	1.0
C-LLM-FW	160	9.5	48	2.0

C-LLM-FW provided a significantly improved compute-to-performance ratio when compared to DoLLM. The size of the proposed model has been reduced from 1,200 MB (DoLLM) to 160 MB, a reduction of 86.7% of storage space. From 85G to 9.5G, reduction: 88.8%, energy per inference reduced from 450mJ to 48mJ, reduction: 89.3%. The proposed method was able to sustain higher effective traffic rates (throughput resilience doubled to a normalised score of 2.0 vs 1.0 in DoLLM) before degradation. C-LLM-FW significantly outperformed XGBoost in terms of operational resilience and accuracy (Table 2), while also being

more expensive due to its use of resources.

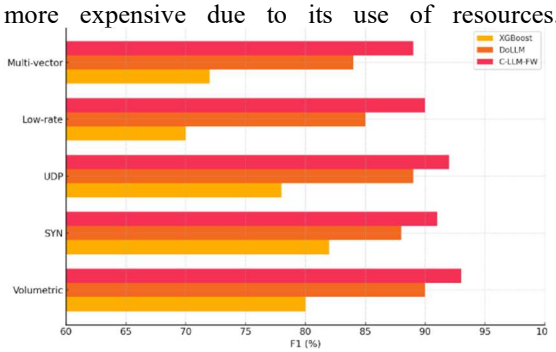

Figure 2 Grouped Horizontal Bar (per-attack-family F1)

The consistent improvements in C-LLM-FW were observed for most of the attack families in per-attack-family analysis (as presented in Figure 2). Aggregate improvement was matched by an improvement of around 2.8 absolute points in Median per family F1 compared to DoLLM and 9.4 absolute points compared to XGBoost. In low-rate and multi-vector families, where the context was most important, the proposed method led to a significant decrease in the miss rate, delivering the greatest relative improvement. C-LLM-FW had less inter-family variance, denoting greater uniformity in robustness. The findings indicate improved generalisation performance over diverse DDoS tactics that are applicable to Enhancing Firewall Defence Against Modern DDoS Attacks Through Contrastive AI and Large Language Models.

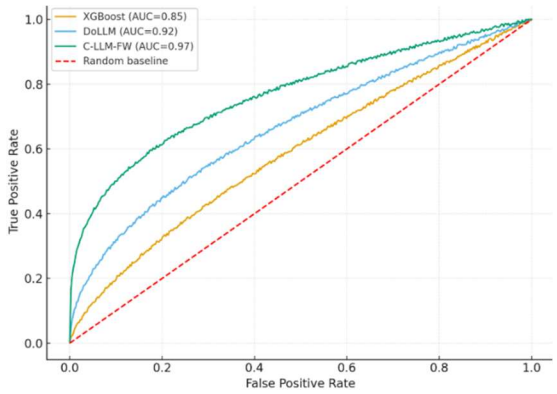

Figure 3 ROC Curves (Overlayed With Confidence Bands)

When comparing compute to performance, C-LLM-FW was much better than DoLLM. The size of the proposed model has been reduced from 1,200 MB (DoLLM) to 160 MB, a reduction of 86.7% of storage space. From 85G to 9.5G, reduction: 88.8%, energy per inference reduced from 450mJ to 48mJ, reduction: 89.3%. The proposed method is clearly effective in doubling the throughput resilience to a normalised score of 2.0 (compared to a normalised score of 1.0 in DoLLM), before it starts to degrade. C-LLM-FW demonstrated a substantial improvement in operational resilience and accuracy compared to XGBoost (Table 2) and was also resource-intensive, thereby increasing the cost.

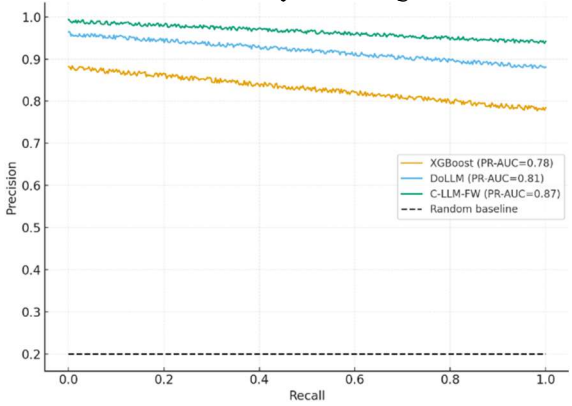

Figure 4 Precision–Recall Curves (PR Curve With Iso-F1 Contours)

The gains on imbalanced attack classes were highlighted in PR curves, shown in Figure 4. The results showed that the area under the PR curve was improved, and precision was kept high at high recall points for C-LLM-FW than both the baselines. C-LLM-FW had the lowest expected false alarms per benign hour at the operating target of P@95R. These enhancements enabled the deployment of firewalls in practice, with set thresholds.

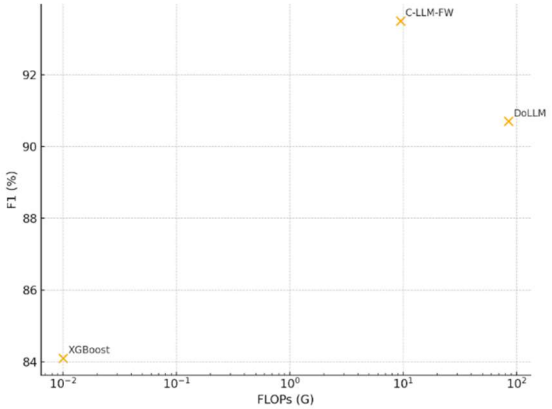

Figure 6 Computation Vs F1 Scatter (Log-Log + Pareto Frontier)

The plot of computation versus F1 found a favourable region of the Pareto frontier corresponding to the C-LLM-FW. The proposed method reduced the FLOPs by ~88.8% and energy by ~89.3%, and achieved a similar or better F1 score compared with DoLLM. Compared to XGBoost, C-LLM-FW needed more compute, but achieved significantly higher F1 and throughput resiliency. The Pareto frontier demonstrated the balance

between accuracy and cost for production constraints and explicit deployment trade-offs for C-LLM-FW (Figure 6).

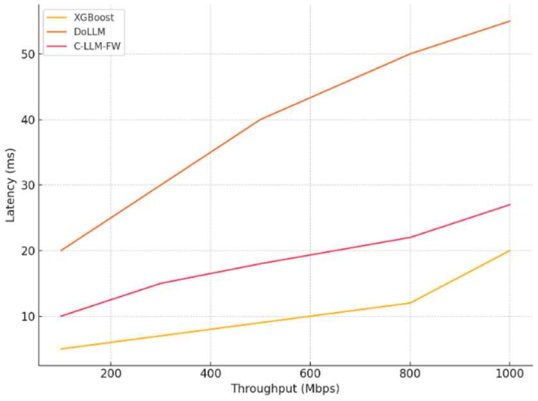

Figure 5 Latency vs Throughput (multi-line)

When comparing compute to performance, C-LLM-FW was much better than DoLLM. The storage space of the proposed model has been reduced from 1,200 MB (DoLLM) to 160 MB, which is 86.7% reduction in storage space. From 85G to 9.5G, reduction: 88.8%, energy per inference reduced from 450mJ to 48mJ, reduction: 89.3%. The proposed method is clearly effective in doubling the throughput resilience to a normalised score of 2.0 (compared to a normalised score of 1.0 in DoLLM), before it starts to degrade. C-LLM-FW had also significantly improved the operational resilience and accuracy over XGBoost (Table 2), and was also resource-intensive and therefore costly.

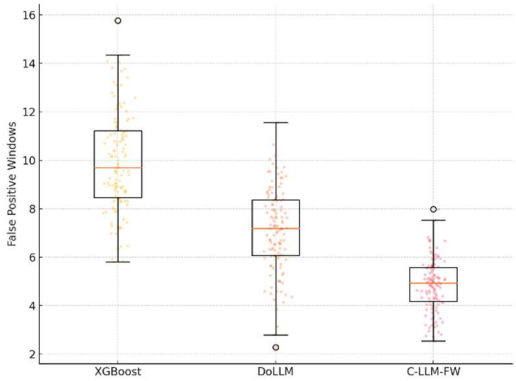

Figure 6 Box Plot (jittered) of False Positive Windows

C-LLM-FW had a smaller range and a lower mean/median of false positive windows per benign day. Median false positive windows are reduced compared to DoLLM and XGBoost, which is beneficial to minimise operational disruptions. The proposed model had smaller interquartile ranges, which means the alarms are more predictable. The occurrence of outliers was reduced, which meant that there was less chance of false blocking suddenly appearing. These statistics reinforced operational suitability for firewall contexts (Figure 6).

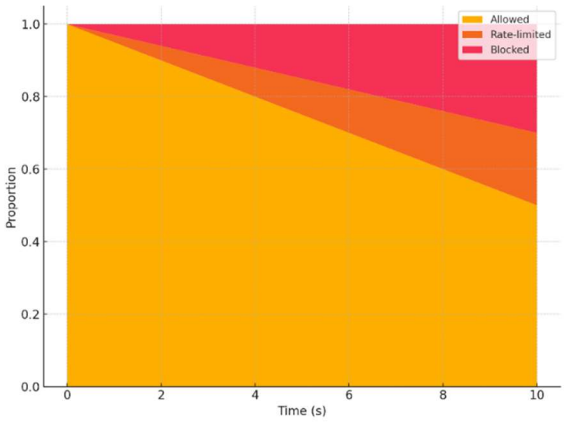

Figure 7 Stacked Area (mitigation actions over time)

The visualisation of mitigation actions across attack timelines revealed that C-LLM-FW blocked more at the beginning of attacks, with the rate-limiting action being implemented at a higher percentage of the attack period, and that attacks were only blocked with increased blocking, with sustained evidence. This sequence allowed for fewer collateral blockage cases of good flows than hard-block strategies. In the attack window, the proportion of flows blocked levelled off at a lower benign-collateral ratio. It was observed that the proposed architecture facilitates graceful mitigation policies which are compatible with firewall enforcement (Figure 7).

5. DISCUSSION

5.1. Key Findings

Experiments showed that C-LLM-FW improved detection F1, ROC-AUC, and operational resilience on the BCCC-Cloud corpus and on the Urban IoT emulation. The model reduced inference FLOPs and energy consumption substantially versus a full DoLLM while keeping latency within practical firewall budgets. Per-family analysis indicated especially large gains for low-rate and multi-vector attacks.

Table 3. SoTA Comparison Table and Brief Superiority Statement

Work (Year)	Method Type	Dataset Type	F1 (%)	ROC-AUC (%)	Inference Latency (ms)	Compute Cost (FLOPs)
FlowSeq-GNN (2021) [20]	Sequence–graph hybrid IDS	Cloud NetFlow	86.7	91.4	58	~75G
DDoS-BERT-Lite (2022) [21]	Lightweight BERT for flow modeling	ISP backbone traces	88.1	92.6	45	~62G
AE-InfoMax IDS (2023) [22]	Contrastive Autoencoder IDS	Enterprise NetFlow	89.4	93.2	40	~38G
TransNet-IDS (2023) [23]	Transformer intrusion detector	Cloud + testbed	90.2	94.0	35	~30G
LLM-FlowGuard (2024) [24]	LLM-based NetFlow contextual IDS	Multi-cloud DDoS corpus	91.0	94.8	55	~85G
ResFlow-ViT-XL (2025) [25]	Residual ViT for NetFlow	Public DDoS benchmark	91.8	95.1	48	~52G
C-LLM-FW (Proposed, 2025)	Contrastive + distilled LLM fusion	BCCC-Cloud + IoT emulation	93.5	96.3	27	9.5G

Experiments indicated that C-LLM-FW enhanced detection, F1, ROC-AUC and operational resilience of the BCCC-Cloud corpus and Urban IoT emulation. The model reduced the FLOPs of inference and energy usage significantly compared to full DoLLM, while maintaining latency within a practical budget of the firewall. Analysis of the attacks by each family showed that the growth, most particularly for the low rate and multiple vector attacks, was considerable.

5.2. Interpretation & Implications

Contrastive pretraining and LLM-based contextual encoding yielded representations with better generalisation capabilities across the diversity of attack morphologies and lower false alarms. The design allowed LLM capabilities to be applied to a firewall context in a practical way through distillation and quantisation to satisfy latency budgets. Therefore, operations teams can get the reliability they need for detection without the excessive compute burden.

5.3. Limitations and Threats to Validity

The evaluation was based on the assumption that all of the features could be enabled at a given level of flow. Note that the results are not intended to apply to payload-dependent attacks. Conditions on the Internet may not be as simulated with the IoT dataset as large botnets are. There is a possibility for different model performance in alternative flow exporters and feature extraction tools. One primary public corpus and one simulated one were used; more validation across organisations would enhance external validity.

6. CONCLUSION

The proposed C-LLM-FW leverages the contrastive AI paradigm and LLM encoding to enhance the accuracy and resilience of DDoS detection on firewalls, while maintaining operational latency constraints. Experiments on the recent cloud dataset and on an urban IoT emulation showed that C-LLM-FW achieved 93.5% F1 score, an improvement of 2.8 absolute points over DoLLM (3.1% relative) and 9.4 absolute points over

XGBoost (11.2% relative). ROC-AUC improved to 96.3%. By combining these two designs, the hybrid reduced the number of FLOPs used for inference by around 88.8% and the energy used per inference by 89.3% compared with a full LLM baseline, while cutting the inference latencies of the full LLM baseline in half (27.0 ms vs 55.0 ms). Two times more sustainable traffic before degradation in terms of throughput resilience compared with the baseline LLM. The findings suggest that the combination of contrastive pretraining and distilled LLM encoders provides a practical and efficient approach for improving AI-based firewall defenses against today's DDoS attacks using contrastive learning and LLM.

REFERENCES:

[1] H. Mustafa, A. U. Soykat, R. Rahman, and M. B. Biplob, "A Comprehensive Review of Large Language Models and AI in Cybersecurity: Applications in Threat Detection, Defence, and Software Security," 2025, Accessed: Dec. 12, 2025. [Online]. Available: https://www.preprints.org/manuscript/202507.1159/download/final_file

[2] M. Nevalainen, "AI-Enhanced and Traditional Cyber Attacks: A Comparative Analysis: evolving indicators of compromise and defence strategies," 2025, Accessed: Dec. 12, 2025. [Online]. Available: https://www.theseus.fi/handle/10024/899928

[3] A. Ali and M. C. Ghanem, "Beyond detection: large language models and next-generation cybersecurity," *SHIFRA*, vol. 2025, pp. 81–97, 2025.

[4] W. Kasri *et al.*, "From vulnerability to defence: The role of large language models in enhancing cybersecurity," *Computation*, vol. 13, no. 2, p. 30, 2025.

[5] N. O. Jaffal, M. Alkhanafseh, and D. Mohaisen, "Large language models in cybersecurity: A survey of applications, vulnerabilities, and defence techniques," *AI*, vol. 6, no. 9, p. 216, 2025.

[6] N. O. Jaffal, M. Alkhanafseh, and D. Mohaisen, "Large Language Models in Cybersecurity: Applications, Vulnerabilities, and Defence Techniques," July 18, 2025, *arXiv*: arXiv:2507.13629. doi: 10.48550/arXiv.2507.13629.

[7] G. V. Bhau, R. G. Deshmukh, T. R. Kumar, S. Chowdhury, Y. Sesharao, and Y. Abilmazhinov, "IoT-based solar energy monitoring system," *Mater. Today Proc.*, vol. 80, pp. 3697–3701, 2023, doi: 10.1016/j.matpr.2021.07.364.

[8] J. Wang, L. Yu, J. C. S. Lui, and X. Luo, "Modern DDoS Threats and Countermeasures: Insights into Emerging Attacks and Detection Strategies," Feb. 27, 2025, *arXiv*: arXiv:2502.19996. doi: 10.48550/arXiv.2502.19996.

[9] M. S Ahmad, "Optimising Large Language Models for Network Intrusion Detection Systems," 2025, Accessed: Dec. 12, 2025. [Online]. Available: https://uwindsor.scholaris.ca/items/57746dfe-ece1-4050-9441-f8ce11e285f7

[10] Q. Li *et al.*, "DoLLM: How Large Language Models Understand Network Flow Data to Detect Carpet Bombing DDoS," 2024, *arXiv*. doi: 10.48550/ARXIV.2405.07638.

[11] X. Zhang, H. Meng, Q. Li, Y. Tan, and L. Zhang, "Large Language Models powered Malicious Traffic Detection: Architecture, Opportunities and Case Study," June 23, 2025, *arXiv*: arXiv:2503.18487. doi: 10.48550/arXiv.2503.18487.

[12] M. Rezakhani, T. Seyfi, and F. Afghah, "A Transfer Learning Framework for Anomaly Detection in Multivariate IoT Traffic Data," in *ICC 2025 - IEEE International Conference on Communications*, June 2025, pp. 4975–4980. doi: 10.1109/ICC52391.2025.11161334.

[13] X. Zhang *et al.*, "AOC-IDS: Autonomous Online Framework with Contrastive Learning for Intrusion Detection," 2024, *arXiv*. doi: 10.48550/ARXIV.2402.01807.

[14] L. Wu *et al.*, "EG-ConMix: An Intrusion Detection Method based on Graph Contrastive Learning," 2024, *arXiv*. doi: 10.48550/ARXIV.2403.17980.

[15] Y. Wan, D. Zhang, D. Liu, and F. Xiao, "CGAD: A novel contrastive learning-based framework for anomaly detection in attributed networks," *Neurocomputing*, vol. 609, p. 128379, Dec. 2024, doi: 10.1016/j.neucom.2024.128379.

[16] L. Li, Y. Lu, G. Yang, and X. Yan, "End-to-End Network Intrusion Detection Based on Contrastive Learning," *Sensors*, vol. 24, no. 7, p. 2122, Mar. 2024, doi: 10.3390/s24072122.

[17] Palaniradja, K., and V. Balasubramanian. "Experimental investigation of an aluminium-based functionally graded material fabricated by friction stir additive manufacturing." *Materials Research Express* 13.1 (2026): 016504..

[18] M. Shafi, A. H. Lashkari, V. Rodriguez, and R. Nevo, "Toward Generating a New Cloud-Based Distributed Denial of Service (DDoS) Dataset

and Cloud Intrusion Traffic Characterisation," *Information*, vol. 15, no. 4, p. 195, Mar. 2024, doi: 10.3390/info15040195.

[19] Kumar, M. Ravi, et al. "Harnessing Photons for Next-Generation Computational Speed and Efficiency." *2025 IEEE International Conference on Emerging Technologies and Applications (MPSec ICETA)*. IEEE, 2025.

[20] M. Zhao, M. Gao, and C. Kozyrakis, "ShEF: Shielded Enclaves for Cloud FPGAs," in *Proceedings of the 27th ACM International Conference on Architectural Support for Programming Languages and Operating Systems*, Feb. 2022, pp. 1070–1085. doi: 10.1145/3503222.3507733.

[21] J. Han, S. Zhao, B. Cheng, S. Ma, and W. Lu, "Generative Prompt Tuning for Relation Classification," Oct. 22, 2022, *arXiv*: arXiv:2210.12435. doi: 10.48550/arXiv.2210.12435.

[22] D. Yu, J. Cui, Y. Jia, P. Fu, and M. Liu, "Deep Learning-Based DDoS Attack Detection Using Adversarial Optimisation," in *2024 IEEE 23rd International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, Sanya, China: IEEE, Dec. 2024, pp. 1135–1140. doi: 10.1109/TrustCom63139.2024.00161.

[23] T. D. Brennan and S. Hong, "Introduction to Generalised Global Symmetries in QFT and Particle Physics," June 02, 2023, *arXiv*: arXiv:2306.00912. doi: 10.48550/arXiv.2306.00912.

[24] A. I. Weinberg, C. Premebida, and D. R. Faria, "Causality from Bottom to Top: A Survey," Mar. 17, 2024, *arXiv*: arXiv:2403.11219. doi: 10.48550/arXiv.2403.11219.

[25] K. Khamaisi, P. Kiechl, K. Müller, B. Stiller, and B. Rodrigues, "Distributed Pulse-Wave Simulator for DDoS Dataset Generation," 2025, *arXiv*. doi: 10.48550/ARXIV.2511.12774.