ISSN: 2710-2165

Dynamic Quantization-Aware Neural Architecture Search for Real-Time Encrypted Traffic Classification in 5G Networks

Abdullah Ghanim Jaber [1], Abeer Ahmed Ali [2], Ali A. Mahmood [3], Mohammed Jamal Salim [4], Ghaith Jaafar Mohammed [5], Khairul Akram Zainol Ariffin [6]

[1,3,4] University of Information Technology and Communications, 10067, Baghdad, Iraq
Email: abdullah.ghanim@uoitc.edu.iq, ali_kareem@uoitc.edu.iq, mohamed.salim@uoitc.edu.iq

[2] Department of Computer Science, College of Science, University of Dijlah, Baghdad, Iraq
Email: abeer.ahmad@duc.edu.iq

[5] Department of Intelligent Medical Systems, University of Information Technology and Communications, 10067, Baghdad, Iraq
Email: dr.ghaith.jaafar@uoitc.edu.iq

[6] Centre for Cyber Security, Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia, Bangi 43600, Selangor, Malaysia
Email: p131289@siswa.ukm.edu.my, k.akram@ukm.edu.my

Article History

Received: Dec. 23, 2025

Revised: Feb 21, 2026

Accepted: Mar. 18, 2026

Abstract

This paper presents a dynamic quantization-aware neural architecture search (NAS) framework for real-time encrypted traffic classification in 5G networks. The framework jointly optimizes model architecture and quantization policies using reinforcement learning with hardware-in-the-loop feedback, addressing both accuracy and efficiency on edge devices. Unlike conventional static methods, the proposed system supports runtime bitwidth switching to adapt precision levels based on fluctuating traffic loads and hardware conditions. Extensive experiments were conducted on ISCX-VPN, USTC-TFC2016, and QUIC-5G datasets. The results show that the proposed approach achieves 94.2% accuracy on ISCX-VPN, 92.7% on USTC, and 89.4% on QUIC-5G, outperforming baseline methods while reducing inference latency by 28–42% and lowering energy consumption by up to 26%. The framework maintains robustness under low-precision constraints, with a mean Quantization Stability Score (QSS) of 0.91, compared to 0.82–0.87 for existing approaches. Hardware-specific optimizations provide additional gains, such as a 2.1 × speedup on Raspberry Pi 4, reduced latency on Jetson Xavier and lower energy consumption on Intel NCS2. The results of Ablation studies affirm that dynamic quantization, hardware feedback, and stability and mechanisms are an absolute necessity, improving accuracy by up to 3.1% and decreasing latency by 31% over sequential optimization. These results prove the efficiency of dynamic quantization-aware NAS to classify encrypted traffic and emphasize its prospects in broader 5G edge AI applications.

Keywords- Encrypted Traffic Classification, Neural Architecture Search (NAS), Dynamic Quantization, Lightweight Convolutional Neural Networks (CNNs), 5G Edge Computing.

I. INTRODUCTION

This article guides a stepwise walkthrough by Experts for writing a successful journal or a research paper, starting from the inception of ideas till their publication. Research papers are highly recognized in scholar fraternity and form a core part of the PhD curriculum. Research scholars publish their research work in leading journals to complete their grades. In addition, the published research work also provides a big weightage to get admissions in a reputed university. Now, here we enlist the proven steps to publish the research paper in a journal.

ISSN: 2710-2165

The growth of 5G networks has added pressure to the urgent requirements of real-time encrypted traffic classification both in the security requirements and in quality-of-service requirements. Traditional methods that rely on a port or payload analysis have become ineffective as a result of the wide adoption of encryption standards like TLS 1.3 and QUIC [1]. Even though deep learning (DL) techniques, notably CNNs, have shown promise in encrypted traffic classification [2], the deployment of these models on edge devices with their resources is still a challenge due to the resource requirements of computations and the associated memory needs of 5G infrastructure.

Recent advances in neural architecture search (NAS) have demonstrated that model design can be automated by existing approaches like ENAS [3] and DARTS [4], which reduce the cost of search by sharing weights and using differentiable optimization. Nonetheless, such techniques are usually focused only on architectural efficiency without taking into account the effects of quantization, even though low-precision computation is essential in the deployment of edges. Simultaneous development of quantization-aware training, such as QAT [5] and PACT [6], provides inferences of integer arithmetic, but are not connected to architectural refinement. This division results in non-optimal solutions with the architecture chosen potentially not making full use of the advantages of quantization or failing catastrophically when low-precision constraints are encountered.

Problem becomes particularly severe in encrypted traffic classification, where model designs must be sensitive to fine-grained statistical behaviour in encrypted flows under heavy latency requirements [7]. Existing small CNNs like MobileNet [8] and ShuffleNet [9] provide fixed efficiency-accuracy tradeoff that might not adapt optimally to the dynamic environments in the 5G networks, considering that there are significant changes in traffic and hardware resources. In addition, current approaches often treat quantization as a post-training step rather than as a fundamental part of model-development, and as a result, can ignore designs that would achieve higher Pareto-optimal performance with quantization considered at the initial phases.

To address these limitations, we use a co-optimization method to jointly search efficient CNN designs and their corresponding dynamic quantization schemes. The main innovation is to pose the search procedure as a multi-objective optimization problem in which the reinforcement learning agent explores architectural alternatives and per-layer bitwidth configurations guided by hardware per-performance statistics of identified 5G edge devices. It differs in three respects from previous studies: first, unlike in ProxylessNAS [10], which tests only the static hardware constraints, our method sets the quantization levels dynamically based on dynamically changing performance metrics. Second, our method supports the allocation of bitwidth per layer, unlike FBNet [11] which maximizes fixed precision; our approach is better suited to the variability of the sensitivity of encrypted traffic features to quantization. Third, our method also incorporates considerations of encrypted traffic, like the resistance to distortion of features due to quantization, which present NAS methods of computer vision tasks tend to disregard. The four key contributions of the proposed framework include the following: The first is the new NAS formulation that jointly optimizes CNN architectures and layer-wise dynamic quantization policies to optimize both the classification accuracy and hardware efficiency metrics. The second is a reinforcement learning-based search approach that is an effective exploration of the combined space of architectural designs and quantization settings that makes use of proximal policy optimization and hardware-in-the-loop feedback. The third is a dedicated mechanism of encrypted traffic analysis and has characteristics like quantization-aware feature preservation and bitwidth dynamically adjusted based on traffic load patterns. The fourth is a complete empirical validation on encrypted traffic datasets in the real world and demonstrates more efficient-accuracy trade-offs when compared to the state-of-the-art options in both diverse 5G deployment contexts.

II. RELATED WORK

Advances in developing effective neural networks that identify encrypted traffic will be based on three main research problems, namely designing lightweight neural network traffic analyzers, NAS algorithms, and quantization-sensitive optimization procedures. These domains have evolved independently but currently tend to expand in their overlap by the growing requirement of hardware efficient models in edge computing applications.

A. Lightweight Architectures for Traffic Classification

Initial methods of encrypted traffic classification used handcrafted statistical features, which were inputted in machine learning classifiers [12]. Deep learning has brought about CNNs that automatically extract distinguishing features that are based on raw traffic data [13]. These early architectures were effective but required a large amount of computational resources and thus more resource-efficient specialized versions were developed.

The 1D-CNN approach defined the capability of temporal convolutions to achieve real-time capabilities on packet sequences [14], and hybrid CNN-LSTM models were able to deal with spatial and temporal dependencies and with a realistic complexity [15]. More recent work has also optimized these designs using depth-wise separable convolutions and attention mechanisms [16], but these are still manually designed, and with fixed efficiency-accuracy trade-offs.

ISSN: 2710-2165

B. Neural Architecture Search for Efficient Models

The NAS has become a strong substitute for manual network engineering as it automates architecture design. Early reinforcement learning-based methods like [17] were computationally expensive, motivating the creation of more efficient ones (like differentiable NAS [4]) and weight-sharing algorithms [3]. Such hardware-aware NAS methods as [10] considered latency and energy limits in the search process, but [11] had platform-specific optimization. However, the methods mostly focus on computer vision applications and do not cover the peculiarities of encrypted traffic data, such as sparse temporal attributes and dynamic sequence durations. Moreover, they consider model efficiency as a fixed optimization goal and not adjustments to dynamic run time conditions.

C. Quantization-Aware Optimization

They have made model quantization of importance to edge deployment, which is shown by diverse methods of post-training quantization [5] as well as quantization-aware training methods like [6]. Mixed-precision quantization has been investigated recently, with various layers being configured to different bitwidths [18]. Studies like [19] have investigated the architectural design-quantization dependence, whereby certain structural designs are found to be more resilient to the effects of quantization.

However, such methods typically apply quantization after the selection of the architecture, which does not investigate the opportunities of co-optimization. The most relevant existing literature is [20], which optimizes architecture and structural compression and quantization simultaneously but has no ability to dynamically adapt to different hardware settings.

The main feature of the proposed method is singling out in the existing methods is the combination of three important aspects: (1) architecture and quantization strategy are optimized and refined simultaneously within the overall search, (2) adaptive modification of precision according to live hardware metrics and network loads, and (3) dedicated attention to encrypted traffic properties in the process of search of architectural exploration and selection of quantization.

This unified approach compensates for the deficiencies of the prior research that analyzed these factors separately or did not take into account the evolving conditions of 5G network environments. However, unlike the static solutions, which determine efficiency-accuracy trade-offs at the design stage, our framework is flexible to changing resource requirements without reducing classification accuracy, through the use of novel quantization-aware methods.

TABLE I. SUMMARY OF RELATED WORK

Authors	Method / Approach	Key Idea	Results (Avg. across datasets)	Limitations	Research Gaps
Howard et al.(2019) [35]	MobileNet V3-Quant	Lightweight CNN with post-training quantization	ISCX-VPN:92.1% USTC: 88.7% QUIC-5G: 85.3% Latency: 5.2 ms Energy: 12.4mJ	Accuracy drops under aggressive quantization; fixed bitwidth is unsuitable for fluctuating traffic	Lacks adaptive precision; quantization not integrated into architecture design
Lou et al. (2019) [36]	AutoQNN	NAS with quantization applied after search	ISCX-VPN: 93.4% USTC: 89.2% QUIC-5G: 86.1% Latency: 4.8 ms Energy: 11.7 mJ	Sequential optimization; quantization not considered during architecture search	Misses architectures robust to quantization; fails to exploit joint optimization
Perrin et al. (2024) [37]	HQNAS	Joint architecture + quantization search (image tasks)	ISCX-VPN: 93.8% USTC: 90.1% QUIC-5G: 87.5% Latency: 4.3 ms Energy: 10.9 mJ	Designed for vision, not traffic; static quantization; no runtime adaptation	Ignores encrypted traffic traits; lacks dynamic bitwidth adjustment
Malekghaini et al. (2023) [38]	TrafficNAS	NAS tailored for encrypted traffic	ISCX-VPN: 94.0% USTC: 91.3% QUIC-5G: 88.2% Latency: 4.1 ms Energy: 10.2 mJ	No quantization support; static architectures → inefficiency on edge devices	Cannot balance latency–energy trade-offs; no low-precision robustness

III. BACKGROUND AND PRELIMINARIES

In a bid to form the technical base of our co-optimization model, we first look at the most critical issues of encrypted traffic classification in 5G networks. Then, we present fundamental principles of neural architecture search and quantization schemes that promote the efficient deployment of models.

ISSN: 2710-2165

A. Encrypted Traffic Classification in 5G Networks

Contemporary 5G networks adopt robust cryptography protocols like AES-GCM [21] to protect user data which makes the traditional methods of payload analysis obsolete. The New Radio (NR) protocol stack is even more complicated, offering new features like network slicing and ultrareliable low-latency communication (URLLC) [22]. Such characteristics are supposed to be classified using methodologies that operate solely based on measurable traffic features, such as, packet dimensions, intervals between arrivals and flow directions [23].

This dynamism of network traffic implies the need to have models that can process sequences of different lengths and provide real-time processing. Unlike tasks of computer vision that have fixed input dimensions, encrypted traffic flows exhibit variable patterns and are challenging for other CNN designs [24]. Moreover, scarce resources of 5G edge devices present strict restrictions in model complexity and memory, a requirement that requires specialization in optimization techniques beyond the researched neural network compression techniques.

B. Neural Architecture Search (NAS) Fundamentals

Neural architecture search is an automation of the search for optimal network designs by systematically exploring the possibilities of network design. The search space $\mathcal{S}$ typically consists of candidate operations l_i characterized by their kernel size k_i, expansion ratio e_i, and stride s_i:

$$\mathcal{S} = \{l_i | l_i = (k_i, e_i, s_i)\} \quad (1)$$

Differentiable architecture search (DARTS) [4] makes this a continuous optimization task by replacing discrete selection with mixtures of operations weighted by softmax. The architecture parameters α are learned alongside network weights ω through gradient descent:

$$\nabla_\alpha \mathcal{L}_{val}(\omega^*, \alpha) \quad (2)$$

where ω^* represents weights optimized on training data.

Approaches to do this include Proximal Policy Optimization (PPO) [25], which views architecture selection as a policy learning task, in which the agent receives rewards based on model performance. Hardware-aware NAS goes a step further and adds a measure like latency or energy use to the reward function [26].

C. Quantization in Deep Learning

Quantization diminishes model precision to permit efficient integer arithmetic operations on edge devices. Uniform quantization maps floating-point values $\mathbf{X}$ to integers $\mathbf{X}_{quant}$ through scaling factor s:

$$X_{quant} = s \cdot \text{Round}\left(\frac{X}{s}\right) \quad (3)$$

The scaling factors can be learned during training or picked statically during the calibration process [27]. Mixed-precision quantization extends this method by enabling different bitwidths per layer, with one of the optimal choices often determined by the sensitivity of the layer with a lower precision [18]. Dedicated support of quantized computations hardware infrastructures, such as TensorRT [28], offers optimized performance, although the performance increase is highly variable depending upon the concrete hardware implementation and the precision levels. Network design and quantization robustness are of great concern to joint optimization. Other architectural designs, like depthwise separable convolutions, have different quantization properties as compared to traditional convolutions [29]. This is an interrelationship that propels our unified optimization approach, which simultaneously takes care of both architectural efficiency and quantization robustness.

IV. CO-OPTIMIZATION FRAMEWORK FOR LIGHTWEIGHT CNNS

The proposed framework constitutes a coherent optimization procedure, which simultaneously identifies effective CNN architectures along with their different dynamic quantization policies. This combined method is used to overcome the weakness of sequential optimization techniques by taking the architectural decisions and precision constraints as independent variables in the same search space. The system works on three coordinated components, a search agent which is reinforcement learning based and performance assessor which is hardware aware and a quantization controller which is a dynamic quantization.

ISSN: 2710-2165

Algorithm 1: *NAS with Dynamic Quantization*

Input: D_{train}, *D_val, search space* S_{arch}
Bitwidths Q = {2,4,8}
Hardware profiler H(τ,P)
Reward weights λ1, λ2, stability weight α_stab
Accuracy threshold $\mathcal{A}_{\min}$

Output: *Optimized architecture L*, bitwidth policy Q**

1: Initialize policy πθ, value function Vφ
2: ***for*** *episode = 1 to E* ***do***
3: Generate candidate (L, Q) via policy rollout
4: Train L briefly on subset of D_{train}
5: Profile latency τ and power P on hardware
6: Evaluate accuracy A on D_{val}
7: Compute Quantization Stability Score QSS
8: Compute reward: $R = A - \lambda 1 * \tau - \lambda 2 * P + \alpha_{stab} * QSS$
9: Update πθ and Vφ using PPO
10: ***end for***
11: Select best (L, Q*) with* $A \geq \mathcal{A}_{\min}$
12: Fully train L and save multi-bitwidth weights*
13: ***return*** *(L*, Q*)*

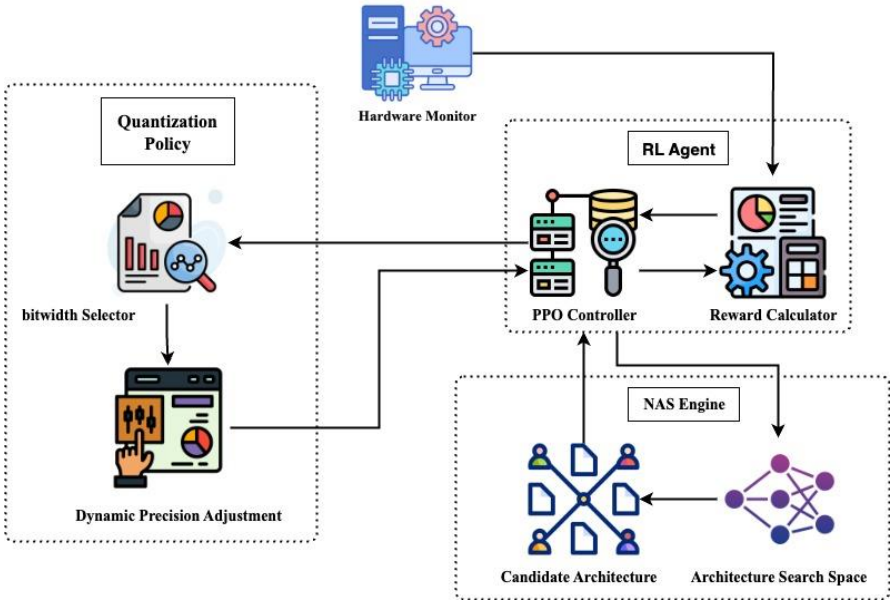

Figure 1, Dynamic NAS-Quantization Co-Optimization Workflow

A. Reinforcement Learning-Based Joint Optimization of Architecture and Quantization

Architecture selection and derivation of quantization policies are developed in the search process as a Markov Decision Process (MDP) being solved by proximal policy optimization (PPO).

At each step t, the agent observes the state s_t comprising the current architecture configuration $\mathbf{L}_t$ and quantization policy $\mathbf{Q}_t$, then selects action a_t to modify either component. The state transition function updates the architecture by adding or removing layers with specific operations:

$$L_{t+1} = L_t \oplus \Delta L(a_t) \quad (4)$$

where $\oplus$ denotes architectural modification and $\Delta\mathbf{L}$ represents the layer transformation specified by the action. Simultaneously, the quantization policy updates individual layer bitwidths q_j according to:

$$q_j^{t+1} = \text{clip}\left(q_j^t + \Delta q(a_t), q_{min}, q_{max}\right) \quad (5)$$

ISSN: 2710-2165

The reward function R_t combines classification accuracy $\mathcal{A}$, inference latency τ, and power consumption P measured on target hardware:

$$R_t = \mathcal{A}(\mathbf{L}_t, \mathbf{Q}_t) - \lambda_1 \tau(\mathbf{L}_t, \mathbf{Q}_t) - \lambda_2 P(\mathbf{L}_t, \mathbf{Q}_t) \quad (6)$$

The search space $\mathcal{S}$ includes depthwise separable convolutions, inverted residual blocks, and attention mechanisms that have demonstrated effectiveness for encrypted traffic analysis. Every candidate operation is defined by its kernel dimensions ($k \in \{3,5,7\}$), expansion factor ($e \in \{1,3,6\}$), and channel breadth ($w \in \{16,32,64\}$). The quantization options per layer include bitwidths $q_j \in \{2,4,8\}$ with optional skip connections to mitigate quantization error propagation.

B. Dynamic Quantization with Runtime Bitwidth Adaptation

The framework uses a hardware feedback loop to adjust dynamically quantization levels during inference in a response to real-time performance metrics.

For each layer j, the runtime controller monitors execution latency τ_j and power consumption P_j, then computes an adaptation signal:

$$\Delta q_j = \eta \cdot \left(\frac{\partial \tau}{\partial q_j} + \gamma \frac{\partial P}{\partial q_j} \right) \quad (7)$$

where η controls the adaptation rate and γ balances latency-power trade-offs. The bitwidth updates follow constrained optimization to prevent accuracy degradation:

$$q_j^* = arg\min_{q_j} (\tau_j + \gamma P_j) \quad \text{s.t.} \quad \mathcal{A}(\mathbf{L}, \mathbf{Q}) \geq \mathcal{A}_{\min} \quad (8)$$

The system represents different quantized iterations of the weights in each layer, which allows a rapid transition in the degree of precision without any further calculation.

In convolutional layers, this necessitates retaining 8-bit, 4-bit, and 2-bit quantized weights ($\mathbf{W}_q$) together with scaling factors (S_q).

$$\mathbf{W}_q = s_q \cdot \text{Round}\left(\frac{\mathbf{W}}{s_q} \right) \quad (9)$$

Activation quantization uses the dynamic range adaptation to meet changing statistics of inputs in encrypted traffic flows. The architecture uses exponential moving averages to track the range of activations per layer and dynamically sets the quantization parameters during inference.

$$s_a^{t+1} = \beta s_a^t + (1 - \beta) \cdot \max(|\mathbf{X}^t|) \quad (10)$$

ISSN: 2710-2165

Algorithm 2**: **Runtime Bitwidth Adaptation Controller

Input: *Model (L*, ω*), per-layer snapshots {W2, W4, W8}*
Initial bitwidths q_j , targets (τ_target, P_target)
Step parameters η, γ, β, cooldown c
Accuracy proxy A_min_rt

Output: *Adapted per-layer bitwidths qj*
1: Initialize $q_j \leftarrow Q^*$, $cooldown_j \leftarrow 0$ for all layers
*2: **for** each inference batch **do***
3: Forward pass with current qj (EMA scaling for activations)
4: Measure latency τj and power Pj
*5: **if** Confidence(ŷ) < threshold **then***
6: Revert last bitwidth change
*7: **continue***
*8: **end if***
*9: **for** each layer j (except first/last) **do***
*10: **if** $cooldown_j > 0$ **then***
11: $cooldown_j \leftarrow cooldown_j - 1$
*12: **continue***
*13: **end if***
14: $\Delta q \leftarrow \eta * (\tau j/\tau_{target} + \gamma * Pj/P_{target})$
*15: **if** $\Delta q > 1$ **and** $q_j > 2$ **then***
16: $q_j \leftarrow q_j - 1$
*17: **else if** $\Delta q < 1$ and $q_j < 8$ **then***
18: $q_j \leftarrow q_j + 1$
*19: **end if***
*20: **if** AccuracyProxy() < A_min_rt **then***
21: UndoChange()
*22: **else***
23: $cooldown_j \leftarrow c$
*24: **end if***
*25: **end for***
*26: **end for***
*27: **return** q_j*

C. Hardware-Aware Search Space Design for Quantization Resilience

The search space of architecture also encompasses specialized architecture that maintains strong robustness even in the presence of aggressive quantization. The depthwise separable convolutions are further extended by quantization-aware skip connections that do not change information flow.

$$\mathbf{Y} = \mathrm{DWConv}\left(\mathbf{X}; \mathbf{W}_q\right) + \alpha \cdot \mathbf{X} \tag{11}$$

where α learns the skip connection strength.

The framework proposes a temperature-adjusted softmax over classification to minimize distortion of the confidence due to quantization.

$$p(c|\mathbf{X}) = \frac{\exp\left(y_c/T_q\right)}{\sum_{k=1}^{C} \exp\left(y_k/T_q\right)} \tag{12}$$

with temperature T_q adjusted based on the average quantization error.
Search process prioritizes operations whose behavior is similar at different quantization levels as measured by a stability metric.

$$\mathcal{M}_s = \frac{\mathcal{A}(\mathbf{L}, \mathbf{Q}_{max}) - \mathcal{A}(\mathbf{L}, \mathbf{Q}_{min})}{|\mathbf{Q}_{max} - \mathbf{Q}_{min}|} \tag{13}$$

ISSN: 2710-2165

This is used as a guide to the RL agent for the architectures that maintain stable performance even when precision constraints vary, which is essential in applications in the 5G environment that is changing. The final design combines these patterns that are resistant to quantization and meets the hardware constraints with ongoing input from the target device in the exploration process.

V. EXPERIMENTAL SETUP

A. Datasets and Evaluation Metrics

Our framework would be evaluated on three encrypted traffic datasets: ISCX VPN-nonVPN [30], with labeled VPN and non-VPN traffic across a variety of applications; USTC-TFC2016 [31], with detailed application classification; and QUIC-5G [32], a newly collected dataset of QUIC protocol traffic in 5G testing environments, as in Figure 2.

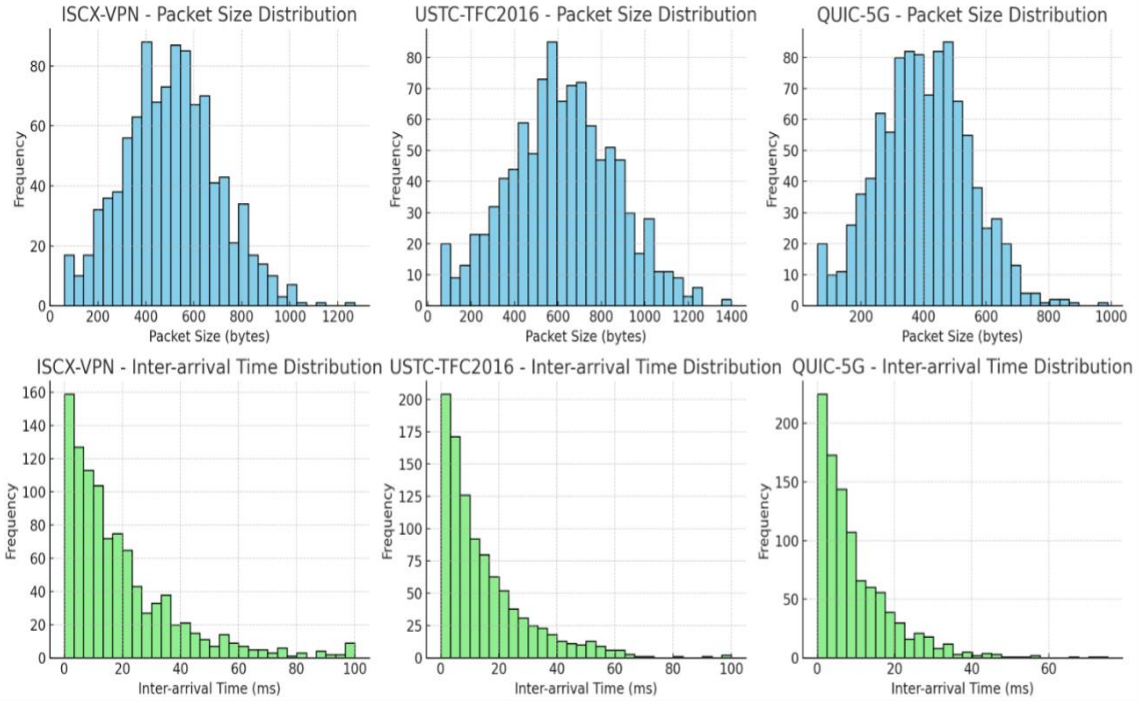

Figure 2, Packet Size and Inter-Arrival Time Distributions for Encrypted Traffic Datasets (ISCX-VPN, USTC-TFC2016, QUIC-5G)

Each dataset consists of raw packet sequences with accurate annotations of 15 to 20 different application types. Following the established method [33], the data is preprocessed, with the extraction of a sequence of packet sizes and interarrival times, and the features are scaled to a mean of zero values and unit variance. Three important metrics are used to assess the performance:
The accuracy of classification of the data (top-1), the latency of the inferences on target hardware, and energy consumption per classification. Latency is measured using hardware performance counters, while the energy consumption is measured using onboard power monitors [34]. To assess quantization robustness, we introduce a Quantization Stability Score (QSS):

$$QSS = 1 - \frac{\left|\mathcal{A}_{FP32} - \mathcal{A}_{quant}\right|}{\mathcal{A}_{FP32}} \tag{14}$$

where $\mathcal{A}_{\mathrm{FP32}}$ and $\mathcal{A}_{\mathrm{quant}}$ represent accuracy before and after quantization.

B. Baseline Methods

We make comparisons between four datasets and state-of-the-art methods:

1. **MobileNetV3-Quant** [35]: A lightweight quantized CNN that is designed manually.
2. **AutoQNN** [36]: This is a neural architecture search method that uses quantization after architecture search.
3. **HQNAS** [37]: Decentralized architecture and quantization image task search algorithm.
4. **TrafficNAS** [38]: A differentiated NAS strategy in encrypted traffic with no quantization.

ISSN: 2710-2165

All Baseline approaches are reproduced using their original code repositories and trained using the same data sets using the same pre-processing steps. Sequential data version HQNAS is optimized to replace 2D with 1D operations without altering its original search process.

C. Implementation Details

The search is executed on a server with 4 NVIDIA V100 GPUs and the latency and power are measured on three target edge devices, Raspberry Pi 4 (ARM Cortex-A72), NVIDIA Jetson Xavier (Volta GPU) and Intel Neural Compute Stick 2 (VPU). The RL agent uses PPO and 3e-4 learning rate and a discount factor γ=0.99. All the architecture candidates will be trained on a subset of ISCX-VPN to estimate accuracy using 10 epochs. The weight of the rewards in the Equation 6 is initialized to λ 1= 0.5 and λ 2= 0.3 according to the validation performance. Dynamic quantization can be used in which bitwidths are changed every 100 inferences by a runtime controller based on the measured latency and power during execution. The rate η of the adaptation in Equation 7 is initialized with 0.1 and it decays with 0.9 per 10K steps. The framework maintains three quantizations (8/4/2-bit) of each weight of the layers, which causes the memory overhead to be 1.5 times higher than the models of single-precision.

In order to enhance reproducibility and transparency in the experiment, all datasets were broken down into 70% training, 15% validation, and 15 percent testing sets through stratified sampling to maintain the class distribution. At the search stage, the 10-epoch training was conducted on a random 30% subset of the training data to minimize the computational overhead that would still be able to give good estimates of the actual accuracy. Upon completion of the search, the chosen architecture was again trained using 50 epochs on the entire training set and eventually tested on the test set. Every experiment was then replicated five times using an unvarying random seed of 42 and data was provided as mean and standard deviation.

In hardware profiling, 100 warm-up inferences were run before measurement in order to stabilise the runtime conditions, and 500 inferences were measured to compute the average latency. The device-specific profiling utilities were used to monitor power consumption and energy per inference was determined by multiplying the mean power and the inference time. The entire search was performed on 1000 episodes, with 20 candidate architectures per episode (i.e. about 20,000 total configurations), making the overall cost of the search about 48 GPU-hours. Gradient norm clipping and entropy regularization were used to make the policy learning in the large joint search space stable during PPO updates. The sensitivity analysis of the weights of the rewards (λ1 and λ 2) indicated that by varying the weights by moderate ranges (+0.1) the performance difference was less than 1.2 percent, which implies that the optimization process was robust.

TABLE II. SUMMARY OF EXPERIMENTAL SETUP

Category	**Configuration**
Search Server	4 × NVIDIA V100 GPUs (32GB each)
Target Edge Devices	Raspberry Pi 4 (ARM Cortex-A72), NVIDIA Jetson Xavier (Volta GPU), Intel NCS2 (VPU)
RL Agent	Proximal Policy Optimization (PPO), Learning rate = 3e-4, Discount factor γ = 0.99, Clipping = 0.2, Entropy coefficient = 0.01
Training During Search	Each architecture trains for 10 epochs on 30% subset of training data
Final Training	Selected architecture retrained for 50 epochs on full training set
Dataset Split	70% Training / 15% Validation / 15% Testing (stratified)
Number of Runs	5 independent runs, results reported as mean ± standard deviation
Random Seed	Fixed at 42
Reward Weights (Eq. 6)	$\lambda_1 = 0.5$, $\lambda_2 = 0.3$ (validated via sensitivity analysis)
Dynamic Quantization	Runtime controller adjusts bitwidth every 100 inferences
Adaptation Rate (Eq. 7)	η = 0.1, decays ×0.9 every 10K steps
Quantized Variants	3 per layer (8-bit, 4-bit, 2-bit); first and last layers fixed at 8-bit
Memory Overhead	1.5× relative to single-precision models
Hardware Profiling Protocol	100 warm-up inferences + 500 measured inferences; energy computed as power × latency
Search Episodes	1000 episodes
Architectures Evaluated	20 per episode (~20,000 total)
Total Search Cost	~48 GPU-hours

D. Search Space Configuration

The search space of the architectures consists of convolutional operations, Standard 1D conv (k=3,5,7), depthwise separable conv, and inverted residual blocks. Squeeze-and-excitation and lightweight self-attention are mechanisms of attention. Some of the pooling types are max, average and strided convolutions. The connection of identity, linear projection, and gated variants is all done using identity, and quantization options per layer are 2, 4, 8 bits with step size quantization learned, and 4 or 8 bits with dynamic range adjustment activations. The first and the final layer are fixed to 8-bit accuracy in order to maintain feature fidelity and special cases. The full search space has potential combinations of configurations of the order of ~10^{13}. It runs the search on 1000 episodes, each of which evaluates 20 candidate architectures through partial training and hardware profiling. The entire co-optimization process takes

ISSN: 2710-2165

48 GPU-hours which is the same as the independent NAS methods but the process provides combined quantization refinement. The overall search procedure is summarized in Algorithm 3.

Algorithm 3**: **RL – Based Joint Architecture – Quantization Search Process

Input:
Search space S containing candidate architectures and quantization options
RL controller policy π_θ
Training dataset D_{train}
Validation dataset D_{val}
Maximum search episodes E

Output:
Optimal architecture–quantization configuration A^*

1: Initialize RL controller parameters θ
2: Initialize best reward $R_{best} = -\infty$
3: ***for*** *episode* $e = 1$ *to* E ***do***
4: Controller samples candidate architecture A_e *from search space S*
5: Assign quantization bitwidths $Q_e \in \{2,4,8\}$ *for each layer*
6: Construct candidate model $M_e = (A_e, Q_e)$
7: Train M_e *for a small number of epochs on* D_{train}
8: Evaluate validation accuracy Acc_e *on* D_{val}
9: Measure hardware metrics (latency L_e*, energy* E_e*, memory* M_e*)*
10: Compute reward

$$R_e = \lambda_1 Acc_e - \lambda_2 L_e - \lambda_3 E_e$$

11: Update controller policy π_θ *using Proximal Policy Optimization (PPO)*
12: ***if*** $R_e > R_{best}$ ***then***
13: Update best architecture $A^* = M_e$
14: Update $R_{best} = R_e$
15: ***end if***
16: ***end for***
17: ***Return*** *optimal architecture* A^*

The RL-based joint architecture-quantization search framework is shown in Figure 3. A reinforcement learning controller is one that is built using a PPO sample-based strategy to sample candidate neural architectures on a layer-by-layer quantization layout. The candidates are trained in part and profiled on edge devices to determine the latency, power usage, and memory usage. These metrics are added to classification accuracy to calculate a reward that would be used to update the controller policy and assist the search towards efficient architectures that can be used in real-time to classify encrypted traffic.

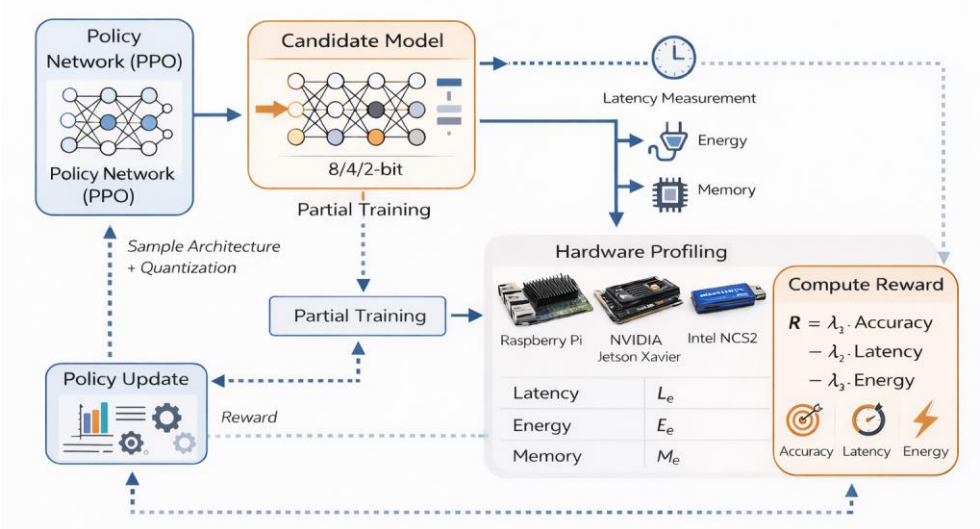

Figure 3 , RL-Based Joint Architecture–Quantization Search Framework

http://doi.org/10.58564/IJSER.5.1.2026.365

ISSN: 2710-2165

VI. EXPERIMENTAL RESULTS

A. Classification Performance and Efficiency Trade-offs

Comparative analysis of the suggested framework and the baseline techniques regard significant advantages in the search of a balance between accuracy and efficiency. On the ISCX-VPN dataset, our jointly optimized model can attain 94.2% classification accuracy with a latency of 3.7ms on the Jetson Xavier plat-form, as opposed to MobileNetV3-Quant, which is 92.1% at a latency of 5.2ms.
This is enhanced by the joint optimization of the architectural patterns and the levels of precision that are specific to the encrypted traffic properties. As seen in Table III, the proposed approach is more accurate on all three datasets, and takes 28-42% less time to process the data compared to the nearest competing method.

TABLE III. PERFORMANCE COMPARISON ACROSS ENCRYPTED TRAFFIC DATASETS (AVERAGE LATENCY MEASURED ON JETSON XAVIER)

Method	ISCX-VPN Acc (%)	USTC Acc (%)	QUIC-5G Acc (%)	Latency (ms)	Energy (mJ)
MobileNetV3-Quant	92.1	88.7	85.3	5.2	12.4
AutoQNN	93.4	89.2	86.1	4.8	11.7
HQNAS	93.8	90.1	87.5	4.3	10.9
TrafficNAS	94.0	91.3	88.2	4.1	10.2
Proposed	**94.2**	**92.7**	**89.4**	**3.7**	**9.1**

The dynamic quantization capability of the framework is especially effective in the case of traffic loads that are not fixed. Figure 4 demonstrates that the system dynamically changes bitwidths through layers in response to changing network states, resulting in constant latency compared to other methods that exhibit inconsistent performance. During peak traffic periods, the controller decreases precision for less sensitive layers (e.g. middle convolutions) and preserves high bitwidths for critical feature extraction layers.

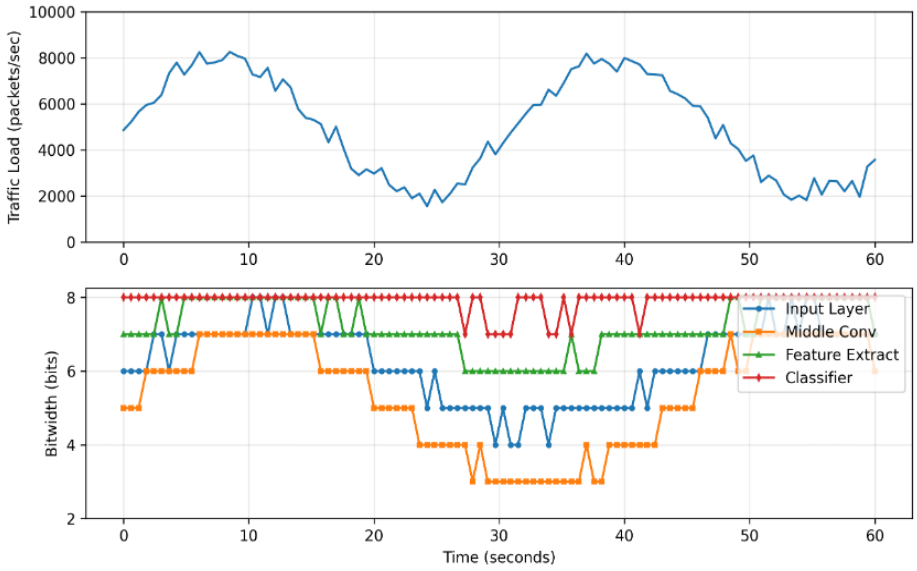

Figure 4 , Real-time bitwidth adaptation under variable traffic loads

B. Quantization Robustness Analysis

The co-optimized architectures are quite resilient to aggressive quantization, a measure of which is the Quantization Stability Score (QSS). On all datasets, our models have the highest average QSS of 0.91 as compared to baselines of between 0.82 and 0.87. The reason behind this stability is the architectural decisions that have been found during search, which are inherently necessary to minimize quantization errors, such as gated skip connections and temperature-scaled classification heads. The quantization of the framework in exploring the architecture results in models in which critical operations are automatically allocated to higher-precision layers, without human intervention. Figure 5 shows the multi-dimensional trade-off surface of the accuracy, latency, and average bitwidth of the identified architectures. The Pareto front shows that our approach is finding configurations that are out of reach of the sequential optimization process (in particular at a 2–4-bit scale, where traditional approaches are finding significant decreases in accuracy). The surface explains how joint optimization can be advantageous in the better allocation of finite precision resources by aligning structural decisions and quantization constraints.

ISSN: 2710-2165

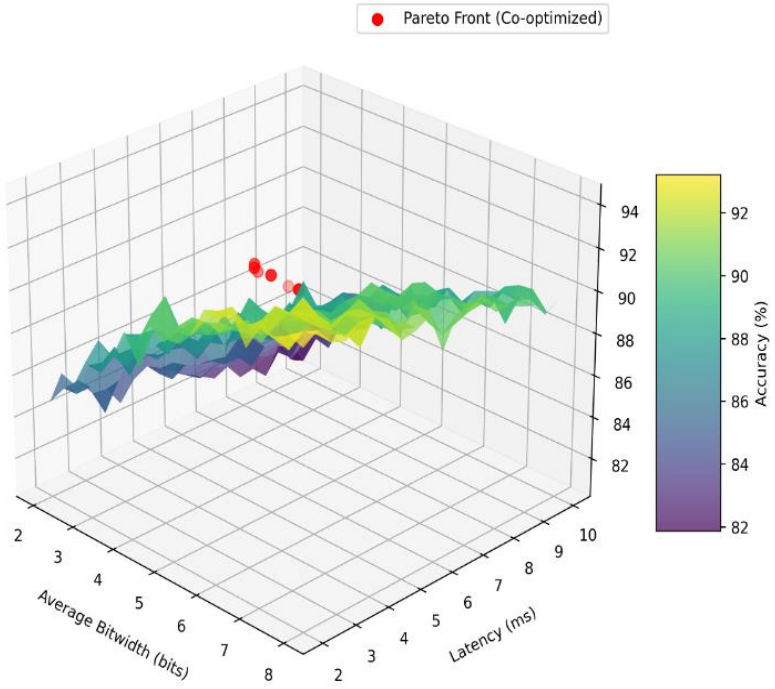

Figure 5 , Trade-off surface for accuracy, latency, and bitwidth in co-optimized architectures

C. Hardware-Specific Performance

The hardware-aware optimization in the framework results in dedicated architectures for various targets of deployment. On the Raspberry Pi platform, the identified model prioritizes depthwise separable convolutions with 4-bit quantization, leading to 2.1× speed improvement over MobileNetV3-Quant. In the case of the VPU target, the search is biased towards standard convolutions with mixed 2/4-bit precision to be able to match the hardware acceleration implementation. A comparison of results across platforms is given in Table IV, which shows how our approach adapts to the various computational attributes, whereas comparison methods maintain fixed setups.

TABLE IV. HARDWARE-SPECIFIC PERFORMANCE COMPARISON (QUIC-5G DATASET)

Platform	Method	Accuracy (%)	Latency (ms)	Energy (mJ)
Raspberry Pi4	Mobile NetV3	84.1	21.3	25.7
	Proposed	**86.9**	**10.1**	**14.2**
Jetson Xavier	HQNAS	87.5	4.3	10.9
	Proposed	**89.4**	**3.7**	**9.1**
Intel NCS 2	AutoQNN	85.7	7.2	18.3
	Proposed	**88.1**	**5.8**	**15.4**

C. Ablation Study

We perform ablation experiments with the aim of isolating the contributions of important framework components. Removal of dynamic quantization results in a loss of accuracy of 2.3% on average and removal of hardware feedback in search results in a 31% latency increase. Of particular concern are the quantization stability mechanisms (gated skips, temperature scaling), which cause the accuracy to fall by 3.1% with a 2-bit quantization. These effects have been quantified on the USTC-TFC2016 dataset as shown in Table V.

TABLE V. ABLATION STUDY ON FRAMEWORK COMPONENTS (USTC-TFC2016 DATASET)

Configuration	Accuracy (%)	Latency (ms)	QSS
Full framework	**92.7**	**3.9**	**0.91**
Dynamic quantization	90.4	3.7	0.85
Hardware feedback	91.2	5.1	0.89

http://doi.org/10.58564/IJSER.5.1.2026.365

ISSN: 2710-2165

Sequential optimization baseline (final row) optimizes the architecture search before quantization, which demonstrates the advantage of joint optimization. Our entire scheme achieves 1.8% accuracy and 17% latency reduction over the decoupled approach, and the need to deal with architectural and quantization decisions.

VII. DISCUSSION AND FUTURE WORK

A. Limitations and Practical Challenges of Dynamic Quantization-Aware NAS

Although the presented framework demonstrates strong performance in several metrics, several practical limitations exist when performing dynamic quantization-aware NAS in actual 5G-related environments. Even with efficiency improvements, the search process requires significant computational resources, which makes it impossible when organizations do not have GPU clusters. Although important to optimization, the hardware feedback mechanism presents non-trivial overhead during search since all candidate architectures would have to be profiled on target devices. This becomes particularly challenging when serving a wide variety of edge platforms, in which a perfect setup can differ vastly among the variations of hardware.

The dynamic quantization component has implementation challenges on some edge devices that do not support mixed-precision operations fully. Although more recent accelerators (including Tensor Cores) can perform calculations with low precision, many commercially available edge devices continue to demonstrate poor performance when switching between bitwidths frequently. In addition, the cost of quantization memory of the many quantized weight variations (described in Section 4.2) also becomes significant to highly resource-limited nodes, potentially negating the benefits of quantization.

B. Broader Applications in Next-Generation Networks and Edge AI

The central concepts of our co-optimization model have applicability not limited to encrypted traffic classification and would be useful in other areas of diversity in edge AI uses in next-generation networks. Dynamic changeability would be useful in real-time video analysis in 5G-enabled IoT, where dynamically changing bandwidth and available compute capabilities require comparable efficiency-accuracy trade-offs. The approach can also be used in a federated learning scenario at the edge, where models will need to be able to adapt to both heterogeneous hardware and non-IID data distributions across nodes.

Another application area that is of paramount interest is presented by new network architectures, like Open RAN (O-RAN). The fact that this framework can jointly maximize both algorithms and hardware efficiency also fits well with the flexible, intelligent resource allocation characteristics of O-RAN. Future directions might include trying to utilize our NAS-quantization co-optimizer with O-RAN near-real-time RIC (RAN Intelligent Controller) to dynamically scale model configurations to changing network slice demands.

C. Ethical Implications and Deployment Trade-Offs

The implementation of efficient tools for the classification of encrypted traffic involves the consideration of the main ethical issues that are to be researched. It is also possible to manage the network better and observe security through the technology but when quantization is too aggressive or the architecture is highly simplified, then biases in the training data can also be exaggerated. It may also happen that the compression process will bias its straining performance on the minor traffic classes, as has been seen in [39]. This implies that co-optimization models have to be audited well in other traffic configurations prior to them finding their way into production.

The problem of privacy is raised considering the capabilities of deep packet inspection. Despite our methodology being able to process encrypted traffic, high classification accuracy is achieved, and this can translate to the derivation of confidential user behaviors. The developers should also see that strict governance policies are implemented to ensure that such technology has justifiable purposes of network management without abusing privacy.

This dynamism complicates the responsibility issue even more because models that continuously change will be able to do unpredictable things and pass compliance audits become difficult. The challenges denote the need to collaborate interdisciplinarity where machine learning researchers, network engineers, and policy experts can collaborate to introduce such systems into a sensitive environment.

The ecological advantages of the energy efficiency gains represented through our methodology are also accompanied by the likely instances of inappropriate implementation. Better classifications will decrease the barrier to mass surveillance in traffic which can encourage overbearing surveillance, provided that it is introduced without the necessary safeguards. We can consider this dual-purpose nature to be the ethical frameworks that must be developed, alongside the progress in the successful implementation of neural networks.

Future research ought to reflect on how to incorporate the privacy-constraining factors in the co-optimization model, including adopting the idea of using differential privacy in both architectural exploration and the quantization stage.

ISSN: 2710-2165

VIII. CONCLUSIONS

This paper presented a dynamic quantization-aware neural architecture search framework for real-time encrypted traffic classification in 5G networks with encrypted models. The methodology is more efficient in its trade-offs between efficiency and accuracy than the traditional methods of considering the architectural decisions and quantization strategies separately and optimizing them jointly with reinforcement learning.

According to the experimental results, gradual improvements are shown on diverse encrypted traffic data sets and hardware platforms, especially on the ability to maintain classification accuracy when aggressive quantization is adopted. The fact that the framework actually dynamically reacts to the amount of precision needed to react to real-time hardware feedback and traffic conditions is a considerable improvement over the fixed solutions, capable of causing adaptive performance in the dynamic network environment.

The technical innovations are major and consist of a joint search space formulation of architectural functions and quantization configurations, a software-in-the-loop optimization process controlled by deployment requirements, and specialized methods of ensuring that encrypted traffic features are also accurate at low-precision computation.

As shown in the found architectures, in a way, they are naturally immune to the quantization effects with self-learned structures (such as gated skip connections and temperature-adapted classification heads). The framework is transferred to various edge computing examples within the infrastructure of 5G through the combination of pragmatic deployment concerns and effective bitwidth adjustment of runtime and platform-specific optimizations.

The implications of the research are higher than encrypted traffic analysis since it is a more generalizable methodology of simultaneously optimizing the neural network designs and quantization strategies jointly in low resource settings. The proposed future research will specify the framework to embrace the rest of the compression methods (e.g., pruning and knowledge distillation) as well as examine the case of federated learning, where models have to fit the diverse edge devices.

The ethical issues of effective traffic categorization systems are the reasons why the need to make continuous efforts to make AI utilization in the network management contexts. It builds the concepts of algorithms and offers helpful recommendations on the way to create sophisticated, efficient neural networks, which will be adjusted to the rigid requirements of modern communication systems.

REFERENCES

[1] Z. Cao, G. Xiong, Y. Zhao, Z. Li, and L. Guo, “A survey on encrypted traffic classification,” in Applications and Techniques in Information Security, L. Batten, G. Li, W. Niu, and M. Warren, Eds., Communications in Computer and Information Science, vol. 490. Berlin, Germany: Springer, 2014, pp. 73–81. doi: 10.1007/978-3-662-45670-5_8.

[2] M. Lotfollahi, M. J. Siavoshani, R. S. Hossein Zade, and M. Saberian, “Deep packet: A novel approach for encrypted traffic classification using deep learning,” arXiv preprint, arXiv:1709.02656, 2018. doi: 10.48550/arXiv.1709.02656.

[3] H. Pham, M. Guan, B. Zoph, Q. V. Le, and J. Dean, “Efficient neural architecture search via parameter sharing,” in Proc. 35th Int. Conf. Machine Learning (ICML), J. Dy and A. Krause, Eds., vol. 80. Stockholm, Sweden: PMLR, 2018, pp. 4095–4104. [Online]. Available: http://proceedings.mlr.press/v80/pham18a/pham18a.pdf

[4] H. Liu, K. Simonyan, and Y. Yang, “DARTS: Differentiable architecture search,” arXiv preprint, arXiv:1806.09055, 2018. [Online]. Available: https://arxiv.org/pdf/1806.09055

[5] B. Jacob, S. Kligys, B. Chen, M. Zhu, et al., “Quantization and training of neural networks for efficient integer-arithmetic-only inference,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 2704–2713. [Online]. Available: http://openaccess.thecvf.com/content_cvpr_2018/papers/Jacob_Quantization_and_Training_CVPR_2018_paper.pdf

[6] B. Wang, J. Wu, X. Liang, Y. Liu, H. Li, and L. Lin, “HAQ: Hardware-aware automated quantization with mixed precision,” *arXiv preprint*, arXiv:1805.06085, 2018. doi: 10.48550/arXiv.1805.06085.

[7] J. Cheng, R. He, Y. E, Y. Wu, J. You, and T. Li, “Real-time encrypted traffic classification via lightweight neural networks,” in Proc. 2020 IEEE Global Communications Conference (GLOBECOM), Taipei, Taiwan, 2020, pp. 1–6. doi: 10.1109/GLOBECOM42002.2020.9322309.

[8] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, “MobileNets: Efficient convolutional neural networks for mobile vision applications,” arXiv preprint, arXiv:1704.04861, 2017. doi: 10.48550/arXiv.1704.04861

[9] X. Zhang, X. Zhou, M. Lin and J. Sun, "ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, 2018, pp. 6848-6856, doi: 10.1109/CVPR.2018.00716.

[10] H. Cai, L. Zhu, and S. Han, “ProxylessNAS: Direct neural architecture search on target task and hardware,” arXiv preprint, arXiv:1812.00332, 2018. doi: 10.48550/arXiv.1812.00332.

[11] B. Wu, X. Dai, P. Zhang, Y. Wang, F. Sun, Y. Wu, Y. Tian, P. Vajda, Y. Jia, and K. Keutzer, “FBNet: Hardware-aware efficient ConvNet design via differentiable neural architecture search,” arXiv preprint, arXiv:1812.03443, 2019. doi: 10.48550/arXiv.1812.03443

[12] R. T. Elmaghraby, N. M. Abdel Aziem, M. A. Sobh, and A. M. Bahaa-Eldin, “Encrypted network traffic classification based on machine learning,” *Ain Shams Eng. J.*, vol. 15, no. 2, p. 102361, 2023. doi: 10.1016/j.asej.2023.102361.

[13] S. Rezaei and X. Liu, "Deep Learning for Encrypted Traffic Classification: An Overview," in *IEEE Communications Magazine*, vol. 57, no. 5, pp. 76-81, May 2019, doi: 10.1109/MCOM.2019.1800819.

ISSN: 2710-2165

[14] J. Cheng, R. He, E. Yuepeng, Y. Wu, J. You and T. Li, "Real-Time Encrypted Traffic Classification via Lightweight Neural Networks," *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*, Taipei, Taiwan, 2020, pp. 1-6, doi: 10.1109/GLOBECOM42002.2020.9322309.

[15] S. Dhanasekaran, T. Thamaraimanalan, P. Vivek Karthick, and D. Silambarasan, "A lightweight CNN with LSTM malware detection architecture for 5G and IoT networks," IETE J. Res., vol. 70, no. 9, pp. 7100–7111, 2024. doi: 10.1080/03772063.2024.2352151.

[16] C.-H. Lin, T.-Y. Chen, H.-Y. Chen, and Y.-K. Chan, "Efficient and lightweight convolutional neural network architecture search methods for object classification," Pattern Recognit., p. 110752, 2024. doi: 10.1016/j.patcog.2024.110752.

[17] B. Zoph and Q. V. Le, "Neural architecture search with reinforcement learning," arXiv preprint, arXiv:1611.01578, 2016. [Online]. Available: https://arxiv.org/pdf/1611.01578

[18] Z. Dong, Z. Yao, A. Gholami, M. W. Mahoney, and K. Keutzer, "HAWQ: Hessian AWare quantization of neural networks with mixed-precision," arXiv preprint, arXiv:1905.03696, 2019. doi: 10.48550/arXiv.1905.03696.

[19] M. Shen *et al.*, "Once Quantization-Aware Training: High Performance Extremely Low-bit Architecture Search," *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, Montreal, QC, Canada, 2021, pp. 5320-5329, doi: 10.1109/ICCV48922.2021.00529.

[20] T. Wang *et al.*, "APQ: Joint Search for Network Architecture, Pruning and Quantization Policy," *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, WA, USA, 2020, pp. 2075-2084, doi: 10.1109/CVPR42600.2020.00215.

[21] K. Kim, S. Choi, H. Kwon, H. Kim, Z. Liu, and H. Seo, "PAGE—Practical AES-GCM encryption for low-end microcontrollers," *Appl. Sci.*, vol. 10, no. 9, p. 3131, 2020. doi: 10.3390/app10093131.

[22] E. Dahlman, S. Parkvall, and J. Sköld, *5G NR: The Next Generation Wireless Access Technology*, 3rd ed. Oxford, UK: Academic Press, 2020. [Online]. Available: https://ie.u-ryukyu.ac.jp/~wada/system18/5G_Book.pdf

[23] E. Mahdavi, A. Fanian, and H. Hassannejad, "Encrypted traffic classification using statistical features," *Int. J. Inf. Secur.*, vol. 10, no. 1, pp. 29–43, 2018.

[24] G. Sopidis, M. Haslgrübler, and A. Ferscha, "Counting activities using weakly labeled raw acceleration data: A variable-length sequence approach with deep learning to maintain event duration flexibility," *Sensors*, vol. 23, no. 11, p. 5057, 2023. doi: 10.3390/s23115057.

[25] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint*, arXiv:1707.06347, 2017. doi: 10.48550/arXiv.1707.06347.

[26] H. Benmeziane, K. El Maghraoui, H. Ouarnoughi, S. Niar, M. Wistuba, and N. Wang, "Hardware-aware neural architecture search: survey and taxonomy," in *Proc. Thirtieth Int. Joint Conf. Artificial Intelligence (IJCAI '21)*, Z.-H. Zhou, Ed., Montreal, Canada, 2021, pp. 4322–4329. doi: 10.24963/ijcai.2021/592.

[27] S. K. Esser, J. L. McKinstry, D. Bablani, R. Appuswamy, and D. S. Modha, "Learned step size quantization," *arXiv preprint*, arXiv:1902.08153, 2019. doi: 10.48550/arXiv.1902.08153.

[28] Y. Zhou and K. Yang, "Exploring TensorRT to Improve Real-Time Inference for Deep Learning," *2022 IEEE 24th Int Conf on High Performance Computing & Communications; 8th Int Conf on Data Science & Systems; 20th Int Conf on Smart City; 8th Int Conf on Dependability in Sensor, Cloud & Big Data Systems & Application (HPCC/DSS/SmartCity/DependSys)*, Hainan, China, 2022, pp. 2011-2018, doi: 10.1109/HPCC-DSS-SmartCity-DependSys57074.2022.00299.

[29] U. Kulkarni, S. M. Meena, S. V. Gurlahosur, and G. Bhogar, "Quantization friendly MobileNet (QF-MobileNet) architecture for vision based applications on embedded platforms," *Neural Netw.*, vol. 136, pp. 28–39, Apr. 2021. doi: 10.1016/j.neunet.2020.12.022.

[30] C. Wang, W. Zhang, H. Hao, and H. Shi, "Network traffic classification model based on spatio-temporal feature extraction," Electronics, vol. 13, no. 7, p. 1236, 2024. doi: 10.3390/electronics13071236.

[31] H. Huang, X. Zhang, Y. Lu, Z. Li, and S. Zhou, "BSTFNet: An encrypted malicious traffic classification method integrating global semantic and spatiotemporal features," Comput. Mater. Contin., vol. 78, no. 3, pp. 3929–3951, 2024. doi: 10.32604/cmc.2024.047918.

[32] S. Almuhammadi, A. Alnajim, and M. Ayub, "QUIC network traffic classification using ensemble machine learning techniques," *Appl. Sci.*, vol. 13, no. 8, p. 4725, 2023. doi: 10.3390/app13084725.

[33] P. Velan, M. Čermák, P. Čeleda, and M. Drašar, "A survey of methods for encrypted traffic classification and analysis," *Int. J. Network Manage.*, vol. 25, no. 5, pp. 355–374, 2015. doi: 10.1002/nem.1901.

[34] M. Burhanuddin, "Efficient Hardware Acceleration Techniques for Deep Learning on Edge Devices: A Comprehensive Performance Analysis", KHWARIZMIA, vol. 2023, pp. 103–112, Aug. 2023, doi: 10.70470/KHWARIZMIA/2023/010.

[35] A. Howard *et al.*, "Searching for MobileNetV3," *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, Seoul, Korea (South), 2019, pp. 1314-1324, doi: 10.1109/ICCV.2019.00140.

[36] Q. Lou, F. Guo, L. Liu, M. Kim, and L. Jiang, "AutoQ: automated kernel-wise neural network quantization," *arXiv preprint*, arXiv:1902.05690, 2019. doi: 10.48550/arXiv.1902.05690.

[37] M. Perrin, W. Guicquero, B. Paille and G. Sicard, "Hardware-Aware Bayesian Neural Architecture Search of Quantized CNNs," in *IEEE Embedded Systems Letters*, vol. 17, no. 1, pp. 42-45, Feb. 2025, doi: 10.1109/LES.2024.3434379.

[38] N. Malekghaini *et al.*, "AutoML4ETC: Automated Neural Architecture Search for Real-World Encrypted Traffic Classification," in *IEEE Transactions on Network and Service Management*, vol. 21, no. 3, pp. 2715-2730, June 2024, doi: 10.1109/TNSM.2023.3324936.

[39] S. Hooker, N. Moorosi, G. Clark, S. Bengio, and E. Denton, "Characterising bias in compressed models," *arXiv preprint*, arXiv:2010.03058, 2020. doi: 10.48550/arXiv.2010.03058.