NextGen Scientific Publication
World Forum on Cybersecurity, Machine Learning, and Intelligent Systems (WFCMLS-2026)

Pg. No. 49-60, WFCMLS26-106, 2026
https://doi.org/10.63282/3117-5481/WFCMLS26-106
Conference Date: 27th March 2026

Original Article

Embedded Intelligence for Network Switches and Routers: Architecture, Techniques, and Evaluation

***Sujay Kanungo**
Independent Researcher, Boston, USA.

Abstract:

The rapid evolution of network technologies has necessitated the integration of embedded intelligence within network switches and routers to enhance performance, efficiency, and adaptability. This paper explores the architectural frameworks that support embedded intelligence in network devices, focusing on various techniques employed to optimize data processing and routing functionalities. We examine the key components that constitute intelligent network architectures, including machine learning algorithms, real-time data analytics, and adaptive control mechanisms. Furthermore, we evaluate the effectiveness of these techniques through comparative analysis and performance metrics, highlighting their impact on throughput, latency, and energy consumption. By synthesizing current advancements and methodologies, this study aims to provide insights into future directions for designing smarter network infrastructures that can autonomously adapt to dynamic traffic conditions and evolving user demands.

Keywords:

AI, ML, Networking, Embedded Intelligence.

1. Introduction

Intelligent networking devices, such as switches and routers, are embedded with inference capabilities to extract useful information from traffic flows. This intelligently enables various applications such as monitoring, analysis, classification, and security enhancement. Such intelligence is substantial in meeting user demands on Quality of Service (QoS) assurance along with periodic service quality verification.

This work surveys architectural models, core techniques and algorithms, data management and telemetry, software frameworks and programmability, and evaluation methodologies for embedded intelligence in switches and routers. Current research gaps in embedded networking intelligence are identified. State-of-the-art architectural models that support intelligence on the data and control planes are summarized to provide insight into hardware-software co-design. Fundamental techniques are introduced to support real-time traffic analytics and security enhancement as an application of embedded intelligence. Data management and telemetry for intelligent monitoring and analysis are discussed in detail, along with edge computing and federated learning that enhance model efficiency on constrained devices [1] [2] [3].

2. Foundations of Embedded Networking Intelligence

Embedded intelligence refers to the inclusion of machine learning (ML) and artificial intelligence (AI) functionalities in routers and switches to facilitate local inferences on traffic, devices, or security without involving a central controller or management server. These functionalities often take the form of lightweight models that provide timely feedback on established flows. Compared to

control-plane intelligence, data-plane intelligence accommodates tighter latency requirements, is less susceptible to security threats, and allows for earlier integration into existing deployments. Recent trends such as AI-based traffic classification, telemetry gathering, anomaly detection, permission-control enhancement, and quality-of-service-aware routing illustrate the feasibility of embeddable tools that operate solely on net flow data and do not alter the packet structure. Policy-driven orchestration constitutes another important area, where high-level intents are translated to programmable data-plane policies for real-time enforcement across the network.

To achieve embedded intelligence, a combination of hardware and software must be jointly addressed. Pairing advanced-value applications with powerful but constrained embedded hardware instigates a co-design process that influences both the design and implementation of the runtime environment, operating system, application frameworks, and algorithms. Targeting energy-efficiency aspects may consequently opt for polling-based event-driven mechanisms and discrete-time or triggered telemetries while further ahead defining the amount of analytical information retaken to the controller [4].

2.1. Definitions and Scope

Embedded intelligence refers to the incorporation of AI models into network devices, such as switches and routers. By enabling these devices to perform inference at the edge, network operators gain valuable visibility into network health and security. Embedded intelligence addresses various tasks, including real-time telemetry, traffic classification, and intent-based networking. The goal is to provide lightweight yet effective solutions that can operate within the substantial resource constraints of network devices [2].

This work specifically focuses on the challenges and techniques for embedded intelligence on the data plane, which implements high-speed packet handling and consequently imposes stringent latency budgets. The complementary topic of control-plane intelligence, which encompasses all forms of analysis and inference sufficiently removed from the data path, is not considered in detail [4].

2.2. Hardware-Software Co-Design

In many networking scenarios, ranging from enterprise to datacenter networks, hardware and software come together in a tightly coupled manner involving a close interaction of complementary elements across different layers [3]. Hence, the design of embedded intelligence involves a significant degree of co-design, where partitioning decisions and the use of dedicated hardware accelerators have a crucial impact on performance. The different aspects of network intelligence can be allocated to various hardware components, thus optimizing a combination of speed, power, flexibility, and ease of deployment. Data-plane intelligence generally fits well on low-level hardware, determining per packet or per flow actions. In contrast to high-level routing and management protocols, which are mostly found in the control plane, control-plane intelligence has the potential to leverage higher information and standards, such as forwarding policies or service branches, allowing energy-efficient, direct hardware implementations. Dedicated edge unit NPUs or FPGAs are routinely available in next-generation routers, but benefit analysis should also consider additional processing equipment on the regular CPU, allowing parallel treatment of various intelligence aspects.

2.3. Data Plane Vs Control Plane Intelligence

The data plane and control plane perform different roles within a network device, which have implications for the integration of embedded intelligence.

The control plane manages routing or forwarding decisions based on topology and business policies, computes control policies, and communicates with components such as switches, routers, or firewalls. However, data-plane intelligence is often deployed in locked-down proprietary switches or routers where it is difficult to modify control-plane functionality. All data-plane-enforced policies must be specified in a language accepted by the data plane to enable compilation on a massive scale. Consequently, the automated extraction of an inter-domain routing policy and, consequently, an automation policy on a data-plane pipeline from a configuration file is possible. This has fueled new research in automating and securing BGP policy modelling, extraction, or control [5].

The integration of embedded intelligence at the data plane level, rather than at the control plane level, has emerged as a promising model. Data packets moving from input ports to output ports traverse the data plane; therefore, the data plane should be an optimal place to add intelligence. It has the potential to automatically categorize malicious advertisements and undertake actions throughout BGP convergence. The action taken can be indication messages, policy adjustment instructions, or topology alteration suggestions. Carrying out these actions during BGP convergence rather than waiting until all updates are received markedly shortens

the time window during which a global view is missing [6]. Furthermore, when a flow passes through middle-box locations such as intrusion detection systems or firewalls, the embedded intelligence can evaluate the condition of each middle box, ensuring that network requirements and quality of service constraints hold. Embedded intelligence can be integrated into a middle box directly or can monitor the middle boxes through reflections.

3. Architectural Models for Embedded Intelligence

Embedded Networking Intelligence comprises two core components: intelligence (knowledge and reasoning about behaviors and events) and networking (complex, dynamic, highly-connected systems supporting rich interactions). Emerging embedded networking domains—from vehicle-to-everything (V2X) communication to cyber-physical systems (CPS)—demand intelligence that monitors and directs traffic flows, predicts failures, and permits efficient configuration. These systems require guaranteed latency, but integrity, support for different levels of AI models, security, high-throughput policy translation, and data-driven abstractions remain challenging. Research is active at the confluence of networking and machine learning (ML), with limited exploration of the spectrum of inference and functionality and how to combine multiple models flexibly.

Embeddings are critical to articulating intelligence across the networking space, involving broad and deep modelling characterization and co-design spanning multiple layers, device types, and cross-technology interactions. Embedding network intelligence into switches and routers remains underexplored, despite the opportunity to resolve, train, or reconfigure topological flows that are richer when codifying and reasoning on flows, policies, and attacks through autoencoders. A hierarchical approach at the protocol level and a set-theoretic representation in the policy layer provide increased geo-spatial generalization, enabling geographically-sensitive policy derivation. Attacks can be captured and recovered via flow-to-flow spaces over joint topologies or the full-graph model extending set- and vector-field approaches, further articulated by static data decoupled from control and configuration signals [2].

3.1. Monolithic Vs Modular Architectures

The two principal architectural models currently considered for embedded intelligence in network devices today are monolithic and modular architectures [3]. Monolithic architectures integrate all functionalities of a device into a single, harmonious unit. This approach leads to tight coupling among functions and, in many cases, even leads to the elimination of operating systems. A monolithic architecture minimizes the number of components involved and leads to optimized performance, lower energy consumption, and simplified development. However, it can also pose scalability challenges, make maintenance and upgrade difficult, and often require redesign of the entire device [7].

Modular architectures, in contrast, provide flexibility for scaling, maintenance, and upgrades, since functions can be added, replaced, or enhanced independently. The overall device can span multiple form factors or dimensions, such as input/output bandwidth, parallel processing capability, and support for new features. At the same time, manufacturers can adopt generic components, reusing pre-existing implementations. For embedded intelligence, this modularity allows new intelligence features to be introduced without binding other portions of the system, while still supporting other critical constraints, such as real-time requirements and low power consumption.

3.2. System-On-Chip and Heterogeneous Computing

Switches, routers, and other network devices continue to rely heavily on near-software-only solutions hosted on multi-core CPUs due to their high flexibility at low cost. Some embedded systems integrate dedicated hardware networking accelerators, but few include general-purpose parallel processors. Further specialization often occurs through hardware/software co-design or heterogeneous computing, which retains the general-purpose capability of a single multi-core CPU while adding a second processor capable of executing GPU-like parallel data flow or neural network operations. Such designs for high-performance intelligent switches employ CPU and packet forwarding engines at 40-400 Gbps and various structured Abacaba or TPU-like applications featuring 4/8/16 128-bit data-words at lower 8/16/32 Gbps. Thirdly, Application-Specific Integrated Circuits (ASICs), Field-Programmable Gate Arrays (FPGAs), and architectures combining multiple types of accelerators excel in highly parallel edge inference and address power, size, and latency requirements prohibiting deployment of trained models on larger heterogeneous intelligent service switches still needing gradient training. [8]

3.3. Programmable Data Planes and Inference Engines

Data planes serve as ubiquitous building blocks in high-speed packet processing. They generally comprise physical interfaces for packet reception/transmission, schedulers/buffers for buffering packets, and various functions for restructuring packet headers/enriching packet metadata. A sand-boxed, high-level programming language and a corresponding compiler offer the flexibility to adapt packet-processing functions at run-time.

Programmable data planes, presently widely regarded as a promising avenue for future systems, provide protocol-independent primitives for packet processing, a powerful match-action computation model that performs operations on packet headers in parallel over multiple processing stages, and the capability of in-field, run-time reprogramming of data-plane specifications. Formal description languages help characterize packet-processing specifications. Besides P4—arguably the de-facto programming language for programmable data planes—data-plane specifications have also been expressed in NetKAT, ProVerif, and other formalisms. Also, a growing kraken of inference frameworks facilitates the training-based generation of data-plane programming/code for demanding programmable networking tasks such as deep packet inspection and traffic classification.

Some intriguing direction-addressing data-plane characteristics have emerged in the context of embedded intelligence. For instance, the availability of various programming languages/unique specifications for programmable-data-plane programming has motivated the exploration of core-inputs and pipeline-parameters differences among the assorted concepts to ease their coexistence, so that multiple programmable-data-plane programming may be deployed in a single device without incompatibility. The programmability and hierarchical structure of the Language Agnostic Network Computation framework allow a juxtaposition between data-plane programming and the widespread edge-inference programming—known by terms such as TinyML, Embedded Neural Network, and Model Compression—to alleviate environmental constraints.

4. Core Techniques and Algorithms

Telecommunication networks evolve rapidly, yielding ever-increasing demands and ever-complex challenges. Concern over the efficiency of both fixed and mobile networks, including crisis situations and exceptional conditions, has given rise to intense research efforts. Engineering research teams now explore as-yet uncharted areas of traffic classification techniques that classify and identify traffic even when a network is in crisis, and more generally aim to distinguish traffic types with great accuracy based on packet header information. Such innovative research has been applied to router line cards, where core marking functions are performed in embedded systems that occupy a modest share of line-card space [9]. Limited embedded intelligence performs on-device machine-learning (ML) traffic classification based on six flow features extracted from packet headers. Feature selection reduces computation loads without sacrificing performance. Multi-class Naïve Bayes models provide excellent accuracy across a variety of network conditions [10].

A rapidly expanding telecommunications infrastructure, coupled with ever-growing traffic volumes, has necessitated the development of intelligent equipment. Next-generation NMS (network management system) equipment provides various intelligent network functions that require substantial resources, and active research is underway on distributed NMS architectures that separate policy management, monitoring, and analysis. In parallel, efforts are now being made to secure network infrastructure. The importance of traffic classification for various network management and security tasks, such as policy enforcement and forensic analysis, is universally acknowledged. To accommodate expanding bandwidth and keep unit costs low, traffic classification capability has begun to migrate from NMS servers to the line cards of high-capacity routers.

4.1. Real-Time Telemetry and Monitoring

Telemetry and monitoring represent a first critical application of embedded intelligence. Data flow through a router or switch comprises streams of packets monitored by a variety of ongoing processes. These data streams serve as telemetry data sources, providing valuable insights into the operational state of the network. The packets themselves are one obvious source of telemetry data. They capture both the forwarding behavior of the device and the interactions it has with other components of the network, such as queuing and forwarding. To extract useful insights from packets and other relevant streams requires collecting, processing, and analyzing a subset of the data in real time [11].

Sampling along these streams is typically necessary to avoid exposing data volumes larger than the available bandwidth. The sampling process affects the amount of telemetry data, ensuring that it remains below that threshold. Nevertheless, raw telemetry data cannot be transmitted over the network, even when sampling reduces its volume to an acceptable level. Instead, data needs to be

processed, distilled, and aggregated to a format that the network can accommodate. Modern pipeline architectures assure that valuable telemetry data can be selected early in a stream while maintaining sufficient throughput. Interesting observations about the continuous operational state of a device can likewise be derived from an early-stage data stream that tracks measurements historically. Distribution-based querying frameworks applied along packet-processing streams enable extracting a diversified array of telemetry data from packet, flow, and event streams presented during online or streaming settings [12].

4.2. Anomaly Detection and Security Enrichment

Anomaly detection aims to identify unusual patterns in network traffic, distinguishing intrusions from normal data. Intrusive activity is a subset of anomalous activity. False negatives occur when intrusions are not detected, while false positives are normal activities flagged as intrusions. True positives and negatives represent correct detections. Existing incremental anomaly detection approaches often face high false alarm rates, scalability issues, and are unsuitable for high-speed networks. Techniques can be supervised, semi-supervised, or unsupervised. Capturing network traffic is essential for analysis, typically using tools like Libpcap to monitor link-layer frames across different systems.

The quality of sensor health monitoring is enhanced by security enrichment through the inclusion of threat intelligence feeds. Feeds such as VirusTotal, AbuseIPDB, Open Threat Exchange, and InfectedOrNot provide valuable information on the maliciousness of IP addresses or URLs associated with specific alarms. By cross-referencing metadata from alarms with these intelligence feeds, it is possible to gauge the level of urgency required for further investigation or response. To facilitate this process, threat intelligence can be integrated into existing security information and event management (SIEM) infrastructures, enabling easier consumption of the services provided. [13]

4.3. Traffic Classification and Qos-Aware Routing

Unclassified packets arrive at the device's ports in bursts. A macroscopic classifier attached to the incoming ports labels each packet with predefined rules. The incoming ports and classifier are encountered at the beginning of the processing pipeline. Slotting of packets into a classifier queue is thus performed at first. They are appropriately scheduled later, depending on input-output link pair selections.

High-performance routing engines operate under variable conditions. Aiming to meet the availability of the global context — namely, routing policies and line rates— a deterministic policy-routing approach is adopted. The routing policies depend on the packet count rate within the whole network; a remanent event data structure monitors per-flow input-output packets at the line-speed programming level. Multiple output port paths cannot be ignored as the available bandwidth of the links between input-output pairs — whose topological positions are physically connected— may significantly differ. Repeated dijkstra algorithms calculated over the modified link-state mechanism select the flow destination path.

4.4. Policy-Driven Orchestration and Intent-Based Networking

Intent-based networking (IBN) is an emerging paradigm that allows network administrators to use simple high-level intents to express how the network should behave, abstracting away low-level configuration complexity. High-level intents can be translated into low-level network policies and protocols in various approaches. In policy-driven orchestration, software-defined networking (SDN) enables to define high-level intents that are automatically translated into policies for execution by stateful middleboxes. With the advancement of software-defined wide area networks (SD-WAN), traffic engineering, service chaining, and access control policies are still essential concerns. Furthermore, networks gradually change from pure forwarding to intelligent operation because of advanced services that require data analytics over network traffic. Policy-driven orchestration is integrated with intent-based networking to enrich intents specification with a flexible policy model and automate the implementation of complex network operation policies [14].

4.5. Edge AI and Model Efficiency for Constrained Devices

The primary objective of edge artificial intelligent (AI) models is to process information locally to lower privacy risk, control data movement, and raise responsiveness. Edge AI must therefore consider efficiency and support on-device machine learning (ML) and continuous learning. Multiple techniques can enhance model efficiency; their robotic applications demand constraints on memory and computational cost, energy consumption, and latency. Building on the performance and resource efficiency of deep learning models, transformational and attention quantization techniques can represent a greater model size while maintaining performance. Compression at the architecture level and neural architecture search (NAS) weight quantization can also downsize models

considerably. A distributed computation approach enhances smart traffic monitoring by dividing a model into several portions and storing them across distinct nodes in a vehicular network. Competitive edge AI models can be retrieved via federated learning with friendly cyber-physical control [15].

5. Data Management and Telemetry

Data management encompasses feature extraction, streaming analytics, privacy, security, and governance.

A fundamental step in any data analysis is feature extraction. This process captures relevant information from raw data. In the networking domain, raw telemetry data may exhibit high dimensionality, rendering it unsuitable for many online machine learning algorithms. During pipeline preprocessing, the network telemetry system identifies salient features from the raw data, reducing dimensionality and generating summaries suitable for streaming. Moreover, the presence of edge devices limits the potential for simplified processing [12]. The selected information varies significantly across applications; for instance, traffic classification may extract flow-level statistics, while anomaly detection might focus on time-based statistics of packet or byte counts. A manageable number of summary items, typically on the order of ten or fewer, is required for effective streaming processing on an edge device.

Data streams are inherently continuous, necessitating adaptation of traditional analytical approaches. A widely employed methodology involves temporal windowing, which divides the continuous stream into isolated segments. Streaming analytics defines various windowing schemes, associated temporal operators, and constraints on latency and backlog. The selection of suitable windows hinges on the specific dataset being processed.

Privacy, security, and governance are critical considerations for data collection and dissemination. When data is acquired from user devices, ownership remains with the user, necessitating strict access controls to prevent unauthorized data agencies. Anonymization techniques must also be applied when data is shared. Simultaneously, audit trails become essential: they record all data-related actions taken by organizations, enabling secondary-level verification of access controls and guaranteeing compliance with legal obligations.

5.1. Feature Extraction and Summarization

Latency-sensitive applications utilize different types of data. Network analysis, which performs monitoring, telemetry, or anomaly detection, is executed in real-time and generally uses statistical metrics [16]. Summarization of statistical metrics is often attained via broker services and a random set of flows from edge devices is sampled. Demand forecasting, either at the source, central, or cloud tiers necessitates both historical and current values. Various sampling or encoding strategies are designed to meet the space and efficiency requirements and dependent-dependent data is furthermore also valuable [17].

5.2. Streaming analytics and windowing

A data stream refers to a sequence of data items continuously generated over time. Streaming analytics, therefore, comprises techniques that perform various computations on such continuous data streams. Telemetry data collected from network devices forms an important class of streaming data. Consequently, the modeling of streaming analytics and the incorporation of windowing schemas is crucial for the deployment of data management and telemetry practices in embedded-intelligence driven network devices.

Continuous data streams, e.g., collected telemetry data, naturally have a concept of time associated with them. Therefore, the definition of time-related windows—also known as temporal windows—is important. Various windowing schemes exist, such as tumbling, sliding, and counting windows. Streaming analytics and windowing in general enable particular classes of computations on continuous data streams. For example, operators that can be executed on the most basic tumbling window include statistics computation (e.g., average, sum, count) and temporal-analysis operators (e.g., prediction, trend determination). The following windows can be defined for such operations:

- “per-second”, in which input data arrive at most once per second, which bounds the time between the arrival of records in a same window to at most one second;
- “per-millisecond”, in which at most one input datum arrives before the windowing period expires, which bounds the time between the arrival of records in a same window to at most one millisecond;
- “Per-minute”, which emits at most one record per minute produced by continuous analytics.

The incorporation of windowing schemes naturally poses additional latency requirements. Several scenarios concerning such latencies can be considered. When a window is not closed and one record arrives for the same window, a full copy of the record must still be retained. Once a window is closed, the accumulation of records for that window must cease. Additional constraints that streaming applications must consider when delivering materials to network devices concern the backlog of input records. A backlog arises in several situations, including:

- The preceding output stage has not consumed the records belonging to a certain time frame, that is, there are still unsent windows corresponding to that time frame;
- Restrictions imposed by the target devices must likewise be adhered to.
  [11]

5.3. Privacy, Security, and Data Governance

Many organizations benefit from data that traces usage patterns, typically in aggregate form retaining anonymity [18]. Ensuring data provenance is also essential to establish the exercising context and compliance–particularly if datasets are collected and aggregated before analysis. Regulatory compliance makes it vital to demonstrate the tracking of all changes applied to data used in network functions [19], and therefore preserving origin, sequence, and outcome. Data science algorithms can enrich telemetry data to improve detection of anomalies, attacks, and misconfigurations. Setting up such processes on already congested monitoring pipelines dedicated to logging and transient storage is impractical; in-device proper and therefore timely dataset curation is necessary for light, relevant, effective, and exhaustive processing.

6. Software Frameworks and Programmability

Networking equipment manufacturers have adopted various strategies to facilitate rapid networking function development and deployment on heterogeneous embedded devices. Remarkable network data plane programmability frameworks include P4 and OpenFlow [6]. These frameworks provide reciprocal programming interfaces, allowing network services to be expressed in a high-level programming language, compiled into target hardware, and mapped onto existing programmable architectures. P4 prioritizes data plane programmability, while OpenFlow emphasizes the separation of data and control planes.

Network services can be modeled as a sequential replication of data and control operations over multiple devices. Network services implemented with embedded intelligence can be composed in combination with control plane service orchestration.

In networking equipment, firmware and application software can be seamlessly updated under various constraints, and heterogeneous embedded devices can be integrated at different time granularity.

6.1. Runtime Environments for Network Devices

Commodity OSs, such as GNU/Linux and Microsoft Windows, are not designed for deterministic environments and have limited powers of control over hardware resources. Majority of software architectures for embedded systems are either missing OS or adopt application specific OS from proprietary vendors. A few open-source embedded OSes are still immature or lack off-the-shelf examples for network devices [20]. Hence, an appropriately customized and extensible OS should be developed to cater specific needs of embedded software for packet processing system.

6.2. Programmable Data Planes and P4-Inspired Concepts

Programmable data planes reduce design, testing, and adoption time for new protocols while encouraging experimentation and innovation in packet processing. Programmable data planes allow the reprogramming of switches in the field. An extensive P4 programming ecosystem, high investment from research agencies, major support from equipment vendors, and active research into further programmable abstraction models drive the development of programmable networks [6].

6.3. Life Cycle Management and Firmware Updates

Seamless firmware updates are crucial in enabling efficient management of network devices. High availability of network equipment on one side, and continuous improvements and features on the other side have created a significant gulf that network operators struggle to bridge. Safely deploying new firmware relies on features like versioning, rollback facilities, continuous deployment capabilities, and advanced safety nets to preserve connectivity and ensure business continuity. Popular solutions like

"zero-touch provisioning" or "configlet-based provisioning" can also reduce burden on installation teams by avoiding or significantly simplifying time-consuming configuration setups [21].

In the context of network devices, automated zero-touch provisioning has accumulated substantial attention and deployment worl dwide. Zero-touch provisioning with configuration synchronization captures the initial configuration of deployed routers/switches and avoids manual configuration through secure Internet Protocol (IP) address-based access. Upgraded configurations are automatically synchronized with the configuration archive after changes, adapting to the modifications in network architecture [22].

7. Evaluation Methodologies

Intelligent Network Devices. Embedded intelligence in network devices has been gaining popularity in recent years because the need for a larger data volume is accelerating. The enforcement of enhanced security policies, the aging of the Internet and fast-changing network patterns necessitate the imperative monitoring of network traffic. The amount of traffic generated on the network increases continuously, and the techniques designed for internet traffic analysis need to be improved significantly as they may not cover the iterative, sophisticated and flexible aspects of the traffic data. Network devices such as routers, switches and firewalls are commonly deployed to help monitor, control and analyze the network traffic. Many contemporary networking specialists have shifted their examination focus from embedded devices at the edge of the network, such as host, sensor and endpoint, to core or backbone devices of the network. Network traffic monitoring and data inspection have received attention over the years, culminating in the emergence of the area of network traffic intelligence [2].

Measuring the effectiveness of embedded intelligence solutions on network devices requires access to large datasets collected from such devices to enable benchmarking of the deployed solutions. Current literature does not prescribe datasets or benchmark suites specifically designed for measuring or evaluating embedded intelligence solutions on network devices. The lack of generic datasets poses a challenge to experimenting with heterogeneous embedded artificial intelligence (AI) solutions. Moreover, measuring and quantifying various performance metrics associated with embedded intelligence solutions on network devices is not very clear and specific in the current literature. Although several works characterize the performance of the AI solutions in terms of accuracy, energy consumption, or memory footprint, they fail to encompass a more comprehensive performance description associated with embedded AI and Intelligence Network Devices. A more detailed overview of the performance metrics that can be employed for measuring embedded intelligence is provided to bring more clarity to this area of interest and serve the active community pursuing Intelligence Network Devices [23].

7.1. Performance Metrics for Embedded Intelligence

Embedded systems for networking intelligence are often subject to stringent performance metrics, given that fine-tuned components typically perform fully deployed preprocessing, sampling, analysis, or processing, independently of other elements in the original deployment. The major performance metrics for embedded intelligence in switches and routers are latency, throughput, accuracy, energy, and memory footprint. Such components can be separate, but the requirement for interchangeable operability on standard networking hardware motivates the most compact realization. Consequently, the individual metrics of embedded components are even more critical than overall performance figures [3].

7.2. Benchmarking Suites and Reproducibility

Most research contributions in embedded networking intelligence are challenging to reproduce and fairly compare, often depending on proprietary code and private datasets. Although some proposals routinely release code [24] , embedded-intelligence research faces unique obstacles such as the confidentiality of both model architectures and training datasets. Various infrastructures for sharing models exist, but many still require extensive setup and do not adequately address the need for reproducible, fair, and accessible evaluations.

A comprehensive benchmarking suite is under development to alleviate these issues. It comprises a straightforward framework with a wide range of included datasets, myriad support programs and scripts for adaptation to specific environments, and a collection of readily employed models reflecting the most common approaches proposed and applied in network devices [25]. With the goal of facilitating the design and evaluation of such intelligence, the suite will be publicly available.

Following model evaluation, a refinement procedure enables optimization for the targeted deployment platform without requiring access to the original training dataset, original model, or original training framework. The partitioning of end-to-end models into distinctive phases allows the systematic optimization of pertinent metrics, such as latency and energy consumption. Such compaction methods and corresponding hardware-software co-design approaches constitute a supplementary avenue for increasing the accessibility and reproductivity of embedded-network-intelligence research.

7.3. Validation in Real-World Deployments

The validation of machine-learning (ML) models for network traffic classification in real deployments ensures their correctness and performance. To achieve this, a prototype has been implemented that allows the flexible development of different models, including hybrid ones. The upstream training of hybrid models exploits scikit-learn for classical tree classifiers and XGBoost for gradient-boosted trees, run on an AWS EC2 instance with 36 vCPUs and 60GB RAM. The deployment targets Intel's Barefoot Tofino and NetFPGA-SUME platforms, with relevant data extracted from text files whose known feature lengths range from 1 to 15 ASCII characters. This enables features to be split across packets or embedded deep within them. The system automatically generates data-plane and control-plane code for both targets from configuration files that specify design constraints such as the maximum number of trees.

The Barefoot Tofino testbed comprises 64 100G ports. Traffic is sent via four servers in a snake topology that exercises full throughput and achieves 6.2Tbps with simple forwarding. Field trials measure the switch's resource consumption, classification performance, and end-to-end delay. These results, together with engineering feedback, are subsequently documented.

8. Applications and Use Cases

The applications of embedded intelligence in network switches and routers encompass multiple domains, each with different characteristics and requirements [16]. Intelligent fault detection and self-healing are fundamental capabilities, demanding rapid yet complex responses to diverse failures in critical infrastructure. Adaptive routing and traffic engineering aim to optimize performance by intelligently tuning the route of data flows, a task involving substantial real-time information where maintaining load equilibrium can be challenging. Security is a key concern, and analytics for data exfiltration detection, both qualitative and quantitative, has become an area of intense research activity. In the face of this growing threat, containment—initiating considerably earlier than conventional detection schemes—is high on the agenda of organizations worldwide. Finally, user sessions, Quality of Experience with communications packets, and Service Level Agreements are increasingly monitored during access to cloud-based services and applications, ensuring compliance even whilst mobile and roaming.

8.1. Intelligent Fault Detection and Self-Healing Networks

The deployment of intelligent fault detection and self-healing capabilities in networks strives to reduce downtime caused by the failure of critical network elements, leading to higher availability and improved Quality of Experience (QoE). The intelligent fault detection solution ensures precise fault localization which, in combination with a comprehensive catalog of recovery actions, accelerates the self-healing process.

Network monitoring contributes to the early detection of emerging faults, allowing the system to kick-start predictive intervention [26]. The availability of real-time telemetry data enables the use of machine-learning techniques to analyze the characteristics of failures and the measures undertaken to remedy them. Such information can be integrated into knowledge bases and decision-making models to optimize a network's recovery policy.

The intelligent fault detection and self-healing network relies on various monitoring sources that provide key features for detecting faults associated with critical equipment. The classification of faults and the corresponding fine-grained recovery actions are also within scope.

8.2. Adaptive Routing and Traffic Engineering

Intelligent routing mechanisms aim to achieve effective traffic engineering by dynamically adjusting paths used in routed flows, according to per-flow statistics. Adaptive routing sustains multiple alternative, pre-established paths among network nodes, and routes each flow along a subset of the paths such that a routing constraint (e.g., link utilization, path latency, etc.) remains satisfied [27]. There are two classes of adaptive routing functions, unstable and stable, which are determined according to their stability properties

with respect to a single flow´s round-trip time. The overall stability region of an unstable adaptive routing function can be unbounded, while a stable adaptive routing function can guarantee the convergence of the allocation scheme [28]. Traffic-engineering tools aim to guarantee that per-flow statistics, such as end-to-end delay and loss, stay bounded across the entire routing process. The notion of QoS maintenance can be extended to take other per-flow statistics into account. As flows do not transition from one traffic class to another and control methods can have different scales, the management of Quality-of-Service can follow the approach of per-flow traffic engineering [29].

8.3. Security Analytics and Threat Containment

While networks have become central to critical infrastructure, offering services for finance, power distribution, and transportation, attackers have begun to realize their potential and target these networks. For example, in 2020, it was reported that ransomware attackers exploited Vietnam's Ministry of Health email accounts and posted hundreds of personal data records from customers of the Vietcombank bank in an online forum. The COVID pandemic has worsened the situation due to the explosive increase in work-from-home arrangements and the emerging of new sophisticated malware. The Security-as-a-Service model is adopted here to refer to customers interested in improving the security level of the VNPT network operator and establishing decent economic collaborations with the network operator's clients. Under the model, threats move in and out of the predefined scope in a timely manner, and the VNPT operator needs to collect real-time threat intelligence from various sources and feed it into the detection system to prepare a rapid response to new attacks on the client networks. 3D Threat Intelligence from VNPT has been integrated into the detection and containment system to protect the VNPT network against unauthorized scans and other attacks and to provide Security As A Service to the VNPT clients. The VNPT Security-As-A-Service Detection and Containment System integrates embedded intelligence at routers from the edge of large client networks to the core of operator networks and then to the multi-domain control of network function virtualization. The spectrum analyzed ranges from key performance indicators through cyber-security-analysis features and content-type features to intent-based, policy-driven orchestrations for in-depth threat intelligence, extreme-event-based analytics and content-aware enforcement. [1]

8.4. Quality of Experience and Service Assurance

Quality of experience (QoE) refers to users' overall assessment of an interaction with a service or system. In contrast, service assurance (SA)—a distinction adopted in the telecommunications domain—focuses on the technical aspects of the network providing specific service- and protocol-level quality parameters. Users perceive QoE through one or more suitable independent metrics, termed QoX, such that high QoX leads to a high QoE, while low QoX leads to a low QoE.

QoE and SA objectives are measured from the user's perspective and therefore require telemetry to be taken at the user terminus of the service rather than from the aggregated traffic that characterizes the network. Granular QoE and SA objectives can be extracted from configuration files and from real-time monitoring applications available in the switch. The extracted metrics can be mapped to a wide variety of intent specifications.

9. Challenges, Open Problems, and Future Directions

Embedded networks increasingly deploy machine learning models onto network devices to streamline operations and improve service quality. Research remains needed to develop efficient methods for resource-constrained devices and to address obstacles arising from limited explainability, interoperability, and cyber-physical resilience.

Many hardware and software products already tackle these issues on large network devices; custom silicon is also emerging in commercial network routers [30]; [31]; [3]. Yet future work remains wide open for deployment of inference engines onto small embedded network switches and routers without sacrificing performance.

9.1. Resource Constraints and Energy Efficiency

Network switches and routers suffer from strict hardware and middleware resource constraints that limit the expressive power of networking intelligence applications and systems. Consequently, crucial network-functionality settings such as telemetry rates, routing-table sizes, policy numbers, and model sizes must be minimized [32]. Moreover, energy efficiency is particularly important in widely deployed devices whose marginal cost, thermal dissipation, and performance targets are already tightly constrained. Exploring multi-criteria optimization to maximize functionality under multiple constraints is not only practical, but also rich and challenging [33]. Therefore, switching/routing-capable hardware platforms that are chosen in early stages of design provide a vast design space

for an additional dimension of optimization [2]. These hard constraints, along with strict timing requirements and reliability concerns, arise in a host of designs, ranging from extremely compact and low-power Internet of Things edge-fog routers to ultra-high-speed and undetectably-edge-perturbed datacenter core switches. Hardware–software co-design remains a primary strategy for meeting evolving requirements, expanding the achievable capabilities of embedded-intelligence devices, prolonging lifetimes of ever-decreasing-budget network-deployment projects, and enabling the extraction of previously unexpected value from data.

9.2. Explainability and Auditability

AI-powered decision-makers, be they models or systems, generate recommendations based on certain parameters. However, a human user or operator finds it hard to understand these recommendations in many domains and interfaces. Consequently, the process would benefit from additional description and explanation to support and inform the user before any implementation begins. In a distributed architecture readable information appears on another networked device, where it can be available for audit purposes.

Artificial Intelligence (AI) embedded in cyber defence tools, like switches or routers, plays a vital role in these systems. Cyber-attacks occur in many forms and techniques to monitor, execute trace, and treat these incidents remain heavy in demand due to the evolution of security procedures. Anomaly detection models have been researched to offer automated risk assessment. The embedded use of machine learning enables capture of various features through telemetry streams and classification of monitored activity, thus avoiding the need for cloud. Monitoring via AI models request this additional transparency.

9.3. Interoperability and Standardization

Interoperability and standardization are two critical aspects for developing embedded intelligence for devices from different vendors. Providing seamless interoperability and compatibility with switched and routed devices requires the formal specification of corresponding data formats, protocols, programming languages, workflows, and data structures. A variety of concepts and standards have been proposed to facilitate such interoperability, including the Generalized Framework for Network Intelligence [34] , Wanda (Li, 2019), the Network Intelligence Framework (NIF) (Holberg et al., 2020), and the Interoperable Network Intelligence (INI) Framework (Cañete et al., 2022).

9.4. Cyber-Physical Resilience and Reliability

Approaches that enhance cyber-physical resilience and reliability are of utmost significance, especially for control plane operations that demand delicate coordination between network devices. Fault tolerance, assurance of recovery following disruption, and resilience-adaptive recovery have been widely investigated in the context of classical resilience frameworks to evaluate existing activation paths, failure scenarios, and recovery timelines [35]. Cyber-physical resilience improvement requires a full set of definitions, metrics, and design frameworks for communication networks, which presently remain largely absent.

10. Conclusion

Management and control plane functions embedded in data plane devices enable real-time adaptation, automatic recovery from faults, and detection of unauthorized actions. Detailed techniques and algorithms for these tasks have been identified and analysed. Programming interfaces and environments that map high-level workflow models to the underlying system have also been reviewed and compared.

The need for embedded intelligence in network devices has been articulated and demonstrated. It has been shown that data flows in and between devices can be of high volume, high frequency, and high variability. Many of the same characteristics apply to the management and control information exchanged between network devices. As a result, reliance on centralized and cloud-based services introduces performance issues, becomes impractical for some operational scenarios, and raises security concerns related to system access and data ownership. State-of-the-art hardware and data plane capabilities on widely-deployed devices today, such as carrier-grade switches and routers, can accommodate embedded intelligence involving real-time telemetry, monitoring, and security without incurring significant equipment replacement costs or operational disruption.

References

[1] J. J. Repanshek, "A MULTI-GIGABIT NETWORK PACKET INSPECTION AND ANALYSIS ARCHITECTURE FOR INTRUSION DETECTION AND PREVENTION UTILIZING PIPELINING AND CONTENT-ADDRESSABLE MEMORY," 2005.

[2] T. Tao Ye, G. De Micheli, and L. Benini, "Analysis of power consumption on switch fabrics in network routers," 2011.

[3] A. N. D. R. E. A. BIANCO, M. MELLIA, F. NERI, F. I. N. O. C. H. I. E. T. T. O. J et al., "Scalable Layer-2/Layer-3 Multistage Switching Architectures for Software Routers," 2006.
[4] R. Eickhoff, J. C. Niemann, M. Porrmann, U. Rückert et al., "Adaptable Switch boxes as on-chip routing nodes for networks-on-chip," 2005.
[5] E. Kaljic, A. Maric, P. Njemcevic, and M. Hadzialic, "A Survey on Data Plane Flexibility and Programmability in Software-Defined Networking," 2019.
[6] E. F. Kfoury, J. Crichigno, and E. Bou-Harb, "An Exhaustive Survey on P4 Programmable Data Plane Switches: Taxonomy, Applications, Challenges, and Future Trends," 2021.
[7] S. 1973- Chatterjee, "Composable system resources as an architecture for networked systems," 2001.
[8] J. Moritz Joseph, L. Bamberg, D. Ermel, B. Razi Perjikolaei et al., "NoCs in Heterogeneous 3D SoCs: Co-Design of Routing Strategies and Microarchitectures," 2019.
[9] R. Ben Basat, X. Chen, G. Einziger, and O. Rottenstreich, "Efficient Measurement on Programmable Switches Using Probabilistic Recirculation," 2018.
[10] M. Collier, "Switching techniques for broadband ISDN," 1993.
[11] A. Fais, G. Lettieri, G. Procissi, S. Giordano et al., "Data Stream Processing for Packet-Level Analytics †," 2021.
[12] S. Landau Feibish, Z. Liu, and J. Rexford, "Compact Data Structures for Network Telemetry," 2023.
[13] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "Survey on Incremental Approaches for Network Anomaly Detection," 2012.
[14] B. Lewis, L. Fawcett, M. Broadbent, and N. Race, "Using P4 to Enable Scalable Intents in Software Defined Networks," 2018.
[15] Z. Wan, A. Sanjay Lele, and A. Raychowdhury, "Circuit and System Technologies for Energy-Efficient Edge Robotics," 2022.
[16] C. Zheng, Z. Xiong, T. T Bui, S. Kaupmees et al., "IIsy: Practical In-Network Classification," 2022.
[17] H. Tran, S. Nguyen, I. L. Yen, and F. Bastani, "Into Summarization Techniques for IoT Data Discovery Routing," 2021.
[18] R. Ahmed Shaikh, H. Jameel, B. J. d'Auriol, H. Lee et al., "Achieving Network Level Privacy in Wireless Sensor Networks," 2010.
[19] V. Wilder, "Security Device Roles," 2017.
[20] R. Jitendra Nayaka and R. C. Biradar, "Ethernet Packet Processor for SoC Application," 2012.
[21] L. Määttä, "Firmware Management in Wireless Sensor Networks," 2010.
[22] C. Gündoğan, C. Amsüss, T. C. Schmidt, and M. Wählisch, "Reliable Firmware Updates for the Information-Centric Internet of Things," 2021.
[23] T. Lukaseder, J. Fiedler, and F. Kargl, "Performance Evaluation in High-Speed Networks by the Example of Intrusion Detection," 2018.
[24] S. Gay, P. Schaus, and S. Vissicchio, "REPETITA: Repeatable Experiments for Performance Evaluation of Traffic-Engineering Algorithms," 2017.
[25] J. Tao, Z. Du, Q. Guo, H. Lan et al., "BENCHIP: Benchmarking Intelligence Processors," 2017.
[26] P. Ren, M. A. Kinsy, M. Zhu, S. Khadka et al., "FASHION: Fault-Aware Self-Healing Intelligent On-chip Network," 2017.
[27] A. Elwalid, C. Jin, S. Low, and I. Widjaja, "MATE: MPLS adaptive traffic engineering," 2001.
[28] Y. Kang, X. Wang, and Z. Lan, "Q-adaptive: A Multi-Agent Reinforcement Learning Based Routing on Dragonfly Network," 2024.
[29] J. Carlier, J. Lattmann, J. L. Lutton, D. Nace et al., "An automatic restoration scheme for switch-based networks," 2019.
[30] D. Kim, N. Lazarev, T. Tracy, F. Siddique et al., "A Roadmap for Enabling a Future-Proof In-Network Computing Data Plane Ecosystem," 2021.
[31] X. Huang, "Protocol and System Design for a Service-centric Network Architecture," 2010.
[32] C. Morfopoulou, "Queuing analysis and optimization techniques for energy efficiency in packet networks," 2013.
[33] N. A. N. F. A. N. G. LI, "Energy Saving and Virtualization Technologies in Switching," 2012.
[34] "Reconfigurable interconnection in optical switching fabrics with wavelength converters," 2015.
[35] A. Jabbar, "A Framework to Quantify Network Resilience and Survivability," 2010.