Available online at www.ijournalse.org

Emerging Science Journal

(ISSN: 2610-9182)

Vol. 10, No. 2, April, 2026

QLAF: Q-Learning Adaptive Forwarding for Disaster Emergency Communication in Named Data Networking

Ratna Mayasari [1, 2*], Galih N. Nurkahfi [1, 3], Nana R. Syambas [1], Eueung Mulyana [1]

[1] *School of Electrical Engineering and Informatics, Institut Teknologi Bandung, Bandung 40132, Indonesia.*

[2] *Center of Excellence for Intelligent Sensing-IoT, Research Institute of Sustainable Society, Telkom University, Bandung 40257, Indonesia.*

[3] *Research Organization of Electronics and Informatics, National Research and Innovation Agency (BRIN), Bandung, Indonesia.*

Abstract

Reliable communication is essential for effective disaster response; however, conventional IP-based networks often fail when the network infrastructure is damaged. Disaster communication networks need adaptive forwarding strategies that maintain reliability under rapid topology changes, various link qualities, and resource constraints. This research proposes a Q-Learning-based Adaptive Forwarding (QLAF) strategy designed to enhance reliability in heterogeneous disaster emergency communication networks. QLAF implements reinforcement learning into the NDN forwarding plane, enabling each router to autonomously learn optimal forwarding faces based on multiple performance metrics: Round-Trip Time (RTT), throughput, and link stability. The proposed strategy was implemented in the Named Data Networking Forwarding Daemon (NFD) and evaluated using the MiniNDN emulator over a BRITE-generated 25-node disaster topology that integrates terrestrial, cellular, and satellite links. We compared QLAF and Adaptive Smoothed RTT-based Forwarding (ASF), Access strategy, and Self-Learning. Experimental results show that QLAF achieves a Packet Delivery Ratio (PDR) of 99.91%. These results show that QLAF gives a robust solution for reliability-sensitive disaster communication, guaranteeing high data delivery performance under unstable network conditions. However, its latency overhead limits its applicability to real-time scenarios.

Keywords:

Named Data Networking (NDN);
Disaster Communication;
Adaptive Forwarding;
Q-Learning;
Reinforcement Learning;
Reliability Optimization.

Article History:

Received:	21	October	2025
Revised:	27	February	2026
Accepted:	05	March	2026
Published:	01	April	2026

1- Introduction

Disaster response often meets communication problems that undermine the effectiveness of rescue operations. Natural disasters can damage existing communication systems, causing conventional IP-based networks to malfunction when people need reliable information. For example, during the 2011 East Japan Earthquake, communication networks failed, with about 1.9 million fixed-line services disrupted and tens of thousands of mobile base stations out of service. The 2015 Nepal earthquake also disrupted the communication infrastructure, preventing coordination between rescue teams [1]. These incidents reveal an essential weakness, where traditional host-centric networking architectures struggle to maintain connectivity under the dynamic, infrastructure-damaged conditions characteristic of disaster scenarios.

Non-permanent emergency communication systems use multiple network technologies' integration, combining satellite links, cellular networks, wireless mesh topologies, and UAV-assisted relays to provide redundancy in the event of terrestrial infrastructure failure [2]. Space-air-ground integrated networks (SAGINs) combine satellites, UAVs, and ground systems to provide wide coverage and reliable communication in disaster areas where infrastructure is damaged. Each component can work together or independently, depending on conditions. However, traditional IP-based networks have four main limitations in disaster scenarios:

* **CONTACT**: ratnamayasari@telkomuniversity.ac.id

DOI: https://doi.org/10.28991/ESJ-2026-010-02-03

- ***Fixed routing breaks easily*:** IP routing requires stable paths and routing tables, which fail when nodes move or break during disasters [3].
- ***Poor mobility support*:** Moving nodes must restart TCP sessions, causing connection disruptions in mobile environments like MANETs.
- ***Location-dependent access*:** IP requires knowing the exact data source location. When a server fails, users cannot access the same content from alternative sources.
- ***Limited path usage*:** IP networks can only use a few paths simultaneously.

During data transmission, if a router has a problem, the session that is already running will be interrupted. This means that the network cannot use redundant or heterogeneous communication paths [4]. The purpose of NDN in establishing an alternative to IP is to overcome the limits of existing content distribution methods and to facilitate content-centric communications. According to Jacobson et al. [5, 6], DN aims to develop a more efficient architecture to accommodate users' increasing needs for creating and consuming content. This is achieved by prioritizing content names based on their purpose and source, allowing for independent content development from any location. By naming data rather than hosts, NDN enables location-independent retrieval where consumers request content by name, and any network node caching that content can respond, regardless of whether the original producer remains accessible. This feature becomes critical when disaster damage isolates content sources, as cached copies at intermediate routers maintain information availability through in-network caching [7]. NDN does not connect to data sources and does not track sessions, making it suitable for highly mobile disaster response scenarios. NDN's stateful forwarding supports multipath transmission, allowing routers to maintain packet states through Pending Interest Tables (PITs). This schema enables routers to forward interest packets via multiple interfaces simultaneously, increasing delivery probability in unreliable networks, content delivery efficiency, and reliability [7-9]. The latest implementation shows the dangers of NDN in a blackout scenario. According to Tran & Kim [10], who used an NDN-based routing system in an edge computing infrastructure, the time to network convergence was 1.8 seconds after node propagation, compared to 10.5 seconds for an IP-based system using EIGRP routing.

Even though NDN offers numerous benefits, its forwarding process is dependent on an intelligent path selection mechanism called adaptive forwarding. Adaptive forwarding dynamically selects the best next hop for each Interest packet based on real-time factors, including network conditions, data availability, and interface performance. There are several outgoing interfaces (faces) available, such as a high-reliability satellite link with a long delay, a low-delay cellular/mobile link that can get congested, or an ad-hoc wireless path with varying quality. The forwarding strategy must be able to choose the best interface for each interest. The goal of adaptive forwarding is to improve content delivery by continuously monitoring the flow of interest and data packets, responding to network failures or congestion, and distributing traffic evenly across multiple paths. This approach enables faster data retrieval, higher network efficiency, and resilience compared to conventional static IP forwarding mechanisms [11].

Current adaptive forwarding strategies utilize immediate metrics: Adaptive Smoothed RTT-based Forwarding (ASF) picks faces based on the most recent round-trip time measurements [12]. Stochastic Adaptive Forwarding (SAF) also finds the best path for sending data packets by using probabilistic throughput estimation [13]. These methods can respond to the current state of the network, but they cannot learn from past data. They do not realize that some faces always fail during peak traffic times or that some network paths always recover after short outages. In disaster settings with repeating patterns, such as infrastructure deteriorating over time, responders following established routes, and frequent handoffs, reactive strategies continue to pursue paths that past evidence indicates will fail, wasting resources and slowing down delivery.

Reinforcement learning, specifically Q-Learning, offers a shift from reactive to adaptive forwarding by accumulating knowledge about face performance over time [14]. Q-Learning-based forwarding strategies in NDN maintain Q-tables that map each state–action pair, typically representing a network prefix and a candidate forwarding face, to an expected reward value. This schema enables nodes to make forwarding decisions based on experience rather than real-time performance measurements. When an Interest packet succeeds or fails, the Q-value for that state–action pair updates, gradually learning which interfaces perform best under changing conditions. This learning ability is important in dynamic disaster environments where link quality and topology frequently change. Previous studies show Q-Learning improves NDN forwarding efficiency and adaptability; however, its application to heterogeneous disaster communication infrastructures remains limited [15].

However, implementing NDN-based disaster communication also presents several challenges. First, disaster environments often suffer from limited bandwidth and unstable connectivity, so repeated Interest retransmissions or inefficient routing can quickly consume network resources. Second, additional processing overhead, such as maintaining tables for content names or caching decisions, must be justified by measurable gains in data delivery and delay reduction, especially for resource-constrained IoT nodes. Third, the heterogeneous structure of disaster communication networks can increase system complexity and needs further validation to guarantee scalability and timely response. Reinforcement

learning methods, such as Q-Learning, can be proposed to enhance the adaptive forwarding mechanism in NDN. Q-Learning is expected to improve reliability and also minimize redundant transmissions. Previous studies have shown its potential for adaptive and context-aware forwarding in NDN as well as for data dissemination in NDN-based IoT architectures within disaster environments [16].

This research proposes Q-Learning-based Adaptive Forwarding (QLAF), a forwarding strategy created for dynamic and heterogeneous emergency communication networks operating over edge computing infrastructure. QLAF integrates Q-Learning with the NDN forwarding plane, enabling routers to autonomously learn optimal face selection from each observation while maintaining compatibility with existing NDN mobility mechanisms.

We try to answer three key research questions: (1) Under which network conditions does Q-Learning's adaptive path selection get performance gains that justify its computational cost compared to reactive forwarding schemes (2) How does QLAF scale as network complexity increases, in terms of the number of faces, mobile producers, and hierarchical edge depth (3) Can QLAF be integrated with the existing NDN-based disaster architectures by improving packet delivery reliability without compromising infrastructure resilience.

QLAF is evaluated through emulation experiments using Mini-NDN with BRITE-generated 25-node topologies, simulating disaster scenarios with varying network complexities (5–10 faces per router). In this research, we compare QLAF with three existing algorithms: ASF (Adaptive Smoothed RTT-based Forwarding), Access (simple, lowest-cost forwarding), and broadcast-based Self Learning, which use flood-and-learn mechanisms with fixed heuristics for prefix derivation. A grid search with 432 parameter combinations identified the optimal QLAF configuration ($\alpha = 0.6$, $\gamma = 0.001$, $\epsilon = 0.1$), showing the extreme sensitivity of reinforcement learning to parameters in networking contexts. The analytical model that we built indicates that QLAF has 1.11ms of computational overhead per 5-hop path, compared 3 to ASF's 30ms of per-router processing. However, experiment results show a latency overhead of 381ms, with 99.7% of this overhead caused by path selection decisions that prefer reliable but slower routes over algorithm overhead. This finding concludes that optimizing Q-table operations provides a benefit compared to designing a reward function that influences forwarding behaviour.

QLAF achieves 99.91% packet delivery, outperforming ASF's 99.49%, Access's 99.81%, and Self Learning's 73.47%. QLAF works best at moderate network complexity (7-9 faces), where ASF packet loses 1.3-1.5% and self-learning packet loses 5-25%, while QLAF loses below 0.1%. This shows Q-Learning learns effective paths and avoids problematic routes that ASF and self-learning cannot handle well. Self-Learning's poor performance (26.53% average loss, 67.04% at 10 faces) comes from its fixed k-shorter prefix rule, which cannot adapt to different naming patterns, demonstrating the need for adaptive parameter optimization.

However, QLAF's reliability costs substantial latency: 447.6 ms average RTT versus ASF's 66.4 ms and Access's 44.3 ms, though much better than Self Learning's 1950.9 ms. When accounting for retransmissions (RTT/PDR), ASF delivers 6.7 times faster than QLAF (66.7 ms vs. 448.0 ms), while QLAF delivers 5.9 times faster than Self Learning (448.0 ms vs. 2655.6 ms). This positions QLAF as a middle-ground solution: more reliable than heuristic approaches but slower than simple measurement-based strategies. QLAF's latency makes it unsuitable for applications requiring sub-100ms response (voice/video calls). Additionally, QLAF requires 30,000-40,000 interest packets to converge before achieving stable performance. It also performs poorly in static topologies, where frequent network changes lead to unnecessary exploration overhead.

These results show that QLAF is a good solution for reliability-critical disaster communication with 400–500ms latency tolerance, not a universal forwarding strategy. QLAF performs best in heterogeneous networks (7–9 interfaces) with varying path quality, where learned preferences justify its implementation complexity and latency costs. The four-strategy comparison (QLAF, ASF, Access, and Self Learning) provides insight: use ASF or Access for latency-sensitive applications; use QLAF for reliability-critical messages tolerating moderate latency; and do not use unoptimized learning approaches like Self Learning, which already failed at moderate network complexity.

The remainder of this paper is organized as follows: Section 2 reviews related research on emergency communication network architecture, Named Data Networking forwarding strategies, and Reinforcement Learning (RL) applications in networking, positioning QLAF within the broader research area. Section 3 describes the QLAF algorithm design, including the multi-metric reward function formulation and Q-Learning implementation details; presents the disaster network architecture used for evaluation with heterogeneous link types (terrestrial, cellular, and satellite); and formulates the research problem addressing adaptive forwarding in dynamic disaster scenarios. Section 4 details the experimental setup using MiniNDN with BRITE-generated 25-node topologies; shows the computational overhead model that differentiates algorithmic costs from path selection effects; provides comprehensive performance results comparing QLAF vs ASF, Access, and Self-Learning strategies across packet delivery ratio and latency metrics; creates systematic parameter sensitivity analysis through grid search optimization; and provides complete mathematical traceability from measured values to final performance metrics. Section 5 discusses the implications of findings for disaster communication deployment, acknowledges limitations of the MiniNDN-based evaluation approach with quantitative

predictions of expected real-world performance differences, and outlines directions for future research, including hardware testbed validation and federated testbed integration. Section 6 concludes the research with a summary of key contributions, practical recommendations for disaster communication network deployment, and a discussion on QLAF's position as a reliability-critical forwarding strategy with defined applicability boundaries.

2- Related Work

2-1- Emergency Communication Network Architectures

In the disaster management field, efficiency and reliability of communication are very important. However, it is known that conventional IP-based networks often show inefficiency in their functionality due to various infrastructure problems, including power outages and an excess of network traffic [17-19]. A conventional communication network is deficient in its ability to sustain connections and ensure reliable data transmission in dynamic and challenging scenarios [20, 21]. This weakness comes from the focus of the conventional network on device-centric solutions.

To solve this issue, researchers have studied various emergency communication technologies, including satellite, cellular networks, wireless mesh, ad-hoc networks, and Unmanned Aerial Vehicles (UAVs) [22]. Q-learning-based multi-hop routing algorithms have been utilized for UAV-assisted communication, with the purpose of optimizing energy usage, enhancing data transmission, and maintaining stable connections despite changes in the communication network infrastructure. Also, IP-based architectures have several structural limitations because of their dependency on existing infrastructure are unable to accommodate highly unstable connections [19]. It has been proven that these issues often result in communication failure during critical situations when information sharing is most important.

2-2- Named Data Networking for Disaster Response

NDN has emerged as a data-centric solution solving issues caused by emergency communication in disasters [19]. It has been demonstrated that these applications prove the importance of Q-learning in addressing communication challenges, from enhancing NDN forwarding to optimizing resource allocation in complex infrastructures [19]. NDN has advantages in its ability to identify and retrieve data by name, an ability that contrasts with the existing IP systems, which rely on device addresses. This shows that content access is differentiated from its physical location [23].

The adoption of this scheme in communication network architecture gives several benefits regarding disaster response: NDN facilitates communication without the need for full connectivity, uses data storage within the network, and automatically ensures data security [19, 24].

Additionally, NDN has been proven to enable the rapid restoration of network functionality after disasters due to its capacity for data retrieval from diverse copies or temporary storage locations, even in scenarios where the network has been split into multiple sections [23]. The resiliency of NDN is attributed to its ability to access content from any source and utilize multiple routes [25].

As mentioned in Aboud et al. [26], how NDN forwards packets affects network performance and reliability when conditions change rapidly. Traditional NDN forwarding strategies use fixed rules from routing protocols, which cannot easily adapt when the network changes. This lack of flexibility can cause several problems: network congestion, slower data delivery, and poor path selection. These issues become especially serious in disaster communication scenarios [27].

To solve these problems, researchers have proposed several adaptive forwarding strategies. These methods adjust forwarding decisions based on current network conditions to improve performance. Some schemas send interest packets randomly to avoid link failures and congestion. Adaptive Multi-Path Interest Forwarding distributes traffic across multiple paths, which improves data throughput and reduces delivery time [23]. In mobile networks, strategies like Content Connectivity Forwarding and Location-Aware Forwarding use information about content location and node position to make forwarding decisions automatically. These approaches work well in disaster scenarios where network infrastructure is limited or damaged [25].

Recent advances in RL have improved network adaptability and reliability. Q-learning, a model-free RL method, has become an important approach for developing intelligent forwarding and routing strategies in dynamic networks [28-30]. Q-learning allows network devices to learn autonomously by interacting with the environment and receiving feedback rewards. This autonomous learning makes Q-learning suitable for networks that frequently face unexpected failures or topology changes [31, 32]. Q-learning has been integrated into drone-assisted networks to improve multi-hop routing efficiency in terms of energy consumption and data transmission speed. Deep Q-Learning has also been studied in Software-Defined Networks (SDNs) and data centers to reduce delay and optimize link utilization [33, 34].

Despite these achievements, research gaps remain. Most Q-learning and RL-based forwarding algorithms focus on optimizing single performance metrics, such as throughput or latency. Research that explores multiple interrelated performance metrics simultaneously is limited. Therefore, there is a need to develop Q-learning-based adaptive forwarding strategies that can improve communication quality in disaster scenarios.

3- System Model and Problem Formulation

3-1- QLAF Algorithm Design

In this section, we explain the methodology for developing and evaluating QLAF. The methodology includes algorithm design, Q-Learning implementation, and simulation to validate performance. This process tries to identify key performance metrics: packet delivery ratio, latency, and adaptability to network topology changes. This methodology supports the structured development of QLAF, ensuring that each component is tested to achieve optimal performance in dynamic NDN environments.

Figure 1 shows the workflow of the proposed forwarding strategy, which integrates Q-Learning-based decision-making. The design consists of two main components: next-hop selection and state updating. Each forwarding decision uses Q-values that are continuously updated based on network feedback, including RTT, throughput, and failure events such as timeouts or NACKs. Using the epsilon-greedy mechanism with decay and adaptive reset, the algorithm strikes a balance between exploring new paths and exploiting the best-performing routes to achieve efficient and reliable content delivery. The algorithm adapts to changes in network quality, such as increased delay or packet loss on specific paths. Punishment is applied progressively when timeouts occur repeatedly, while a recovery bonus is given when previously failing paths regain stability. This adaptive mechanism is expected to improve data delivery reliability, accelerate convergence of the forwarding strategy, and reduce overhead from poor path selection [30].

Recent research on NDN forwarding techniques has shown the importance of using historical measurements and probabilistic decision-making for path selection. Adaptive Smoothed RTT Forwarding (ASF) [35] and Stochastic Adaptive Forwarding (SAF) [36]. Use the past measurements and probabilistic decisions to select paths. However, these methods are limited in balancing exploration and exploitation when network parameters change rapidly. Reinforcement learning-based solutions, such as Q-Learning Forwarding Strategy (QRF) [37] and its variants [32], have shown potential in overcoming this problem. However, the implementations often focus on static scenarios or simplified performance metrics.

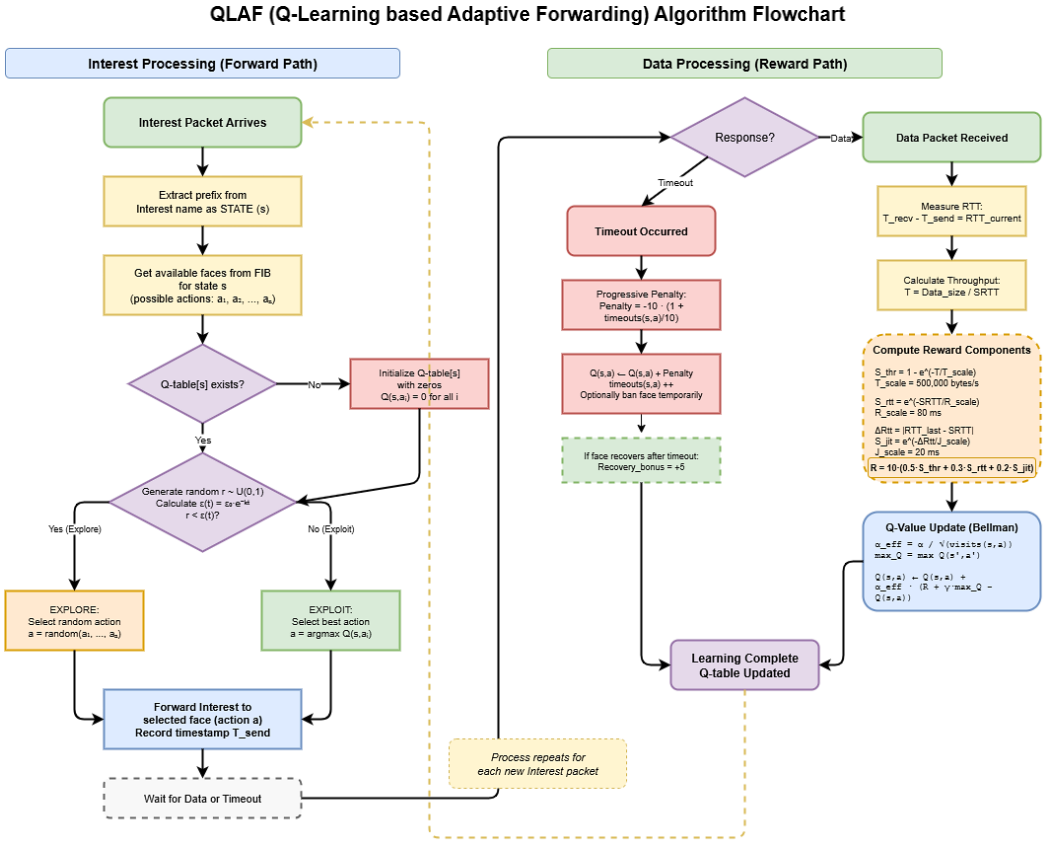

Figure 1. Disaster Communication Network Topology

Our approach improves existing methods by integrating Q-learning with adaptive reward and punishment mechanisms and an exploration-exploitation control policy. The main contribution of this research is its ability to self-adjust in dynamic environments, penalize persistent failures, and reward recovery. This aligns with the research direction of self-learning and autonomous networking [38]. Implementing Q-Learning in the forwarding strategy shows a significant advancement in addressing reliability, scalability, and adaptability challenges in next-generation NDN architectures.

Reward Convergence Analysis

Figure 2 shows QLAF's reward convergence under dynamic network conditions using moving average reward (300-second window) combined with EMA (α=0.02) for three configurations (6, 8, and 10 available faces). All configurations show rapid reward growth from around 1.0 to 4.0, indicating fast learning of effective forwarding decisions. The 10-face, 8-face, and 6-face configurations reach rewards of around 4.2, 4.3, and 4.0, respectively, demonstrating QLAF effectively exploits available paths under stable conditions.

Rewards drop sharply to around 0.8-1.0 at 120 seconds across all configurations due to simulated network degradation (link failure). This indicates QLAF's reward function accurately reflects degraded conditions through decreased throughput, increased RTT, and higher PER.

QLAF shows adaptive learning as rewards gradually increase. The algorithm explores alternative paths and updates Q-values based on new network state observations. Recovery shows oscillating patterns as ε-greedy exploration (ε=0.1) samples different faces while exploiting learned knowledge. The 8-face configuration shows strong recovery, indicating moderate path diversity balances exploration and decision complexity well (see Table 1).

Table 1. Notation Table Contents (organized by category)

Symbol	Definition	Unit	Value/Range
Q-Learning Parameters			
A	Learning rate controlling Q-value update magnitude	dimensionless	0.6
Γ	Discount factor for future reward weighting	dimensionless	0.001
E	Exploration probability in ε-greedy policy	dimensionless	0.1
Reward Function Weights			
w_tp	Throughput weight in multi-metric reward	dimensionless	0.5
w_rtt	RTT weight in multi-metric reward	dimensionless	0.3
w_jitter	Jitter weight in multi-metric reward	dimensionless	0.2
Reward Components			
R	Combined reward value	dimensionless	[-10, 10]
s_tp	Throughput score (normalized)	dimensionless	[0, 1]
s_rtt	RTT score (normalized)	dimensionless	[0, 1]
s_jitter	Jitter score (normalized)	dimensionless	[0, 1]
Normalization Thresholds			
Θ_tp	Throughput normalization constant	bytes/s	500,000
Θ_rtt	Target RTT threshold	ms	80
Θ_jitter	Jitter tolerance threshold	ms	20
Q-Learning State-Action			
Q(s,a)	Q-value for state-action pair	dimensionless	[-∞, +∞]
S	State (content prefix namespace)	string	-
A	Action (face index for forwarding)	integer	[0, n_faces-1]
Performance Metrics			
RTT	Round-trip time	ms	[10, 800]
σ_rtt	RTT jitter (instantaneous deviation)	ms	[0, 100]
Throughput	Data transmission rate	bytes/ms	[100, 10000]
PDR	Packet delivery ratio	percentage	[0, 100]
Computational Overhead			
T_base	Base NFD processing time	μs	15.0
T_comp	Computational overhead (algorithm only)	μs	[0, 300]
T_path	Path-induced latency	ms	[0, 500]
T_qlaf	Total QLAF per-packet overhead	μs	252.3
T_asf	ASF per-packet overhead	μs	30.1
n_faces	Number of available forwarding faces	count	[5, 10]
L_hop	Path length in network hops	count	[3, 7]

After about 400 seconds, all configurations reach stable reward levels (2.5-4.0) with fluctuations expected from continuous exploration (ε=0.1 = 10% random decisions). The 10-face configuration shows the highest variance due to a larger action space; the 6-face shows more stability but slightly lower rewards. Post-disruption rewards (3.0-4.0) don't fully recover to pre-disruption levels, showing persistent network degradation and QLAF's adaptation.

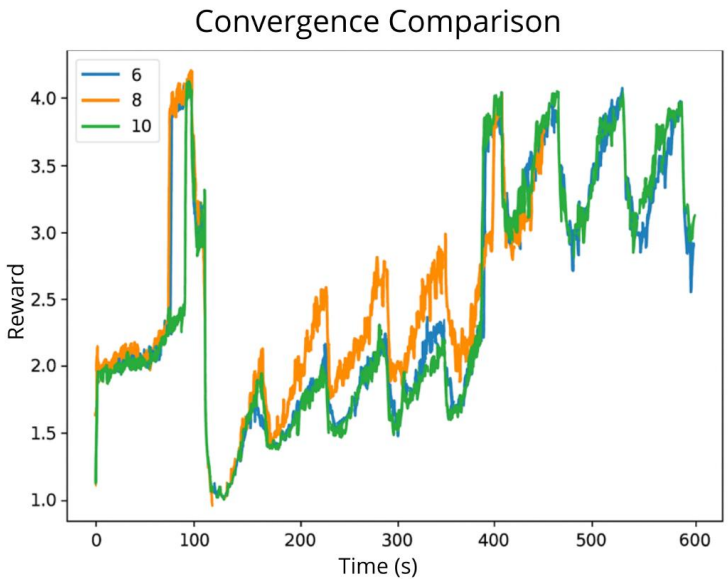

Figure 2. Reward Convergence Comparison

The pattern shows QLAF's adaptation ability in dynamic disaster scenarios. QLAF autonomously detects performance degradation through negative rewards and adjusts forwarding probabilities via its reinforcement learning approach. The reward function combining throughput, RTT, and jitter enables distinguishing reliable from unreliable paths. QLAF's multi-metric reward function addresses simultaneous optimization of conflicting performance objectives through explicit weight-based prioritization. The weighted reward function $r = w\_tp \cdot s\_tp + w\_rtt \cdot s\_rtt + w\_jitter \cdot s\_jitter$ with weights (w_tp=0.5, w_rtt=0.3, w_jitter=0.2) implements explicit prioritization. The 50% throughput weight prioritizes data delivery, while the combined 50% weight on latency metrics (30% RTT + 20% jitter) ensures balanced consideration. Moving from Configuration 3→2 trades 268ms for 1.11 percentage points PDR (0.0041 pp/ms efficiency). The low γ=0.001 appears appropriate, prioritizing immediate rewards over future values, potentially enabling faster adaptation than conventional RL (γ=0.9-0.99), though direct comparison needs additional experiments. Traditional RL involves sequential decisions where current actions significantly impact future states. NDN forwarding decisions are largely independent, selecting face for packet N minimally impacts optimal face for packet N+1 unless topology changes occur (rare at <1s timescales). With γ=0.9, Q-Learning would incorporate rewards from hypothetical future decisions weighted at 90% importance. However, network conditions change between packets, where RTT variance is measured in jitter, this condition making future reward predictions unreliable.

Disaster scenarios demand fast adaptation. Higher γ slows convergence by distributing reward credit across many time steps. With γ=0.001, Q-values primarily reflect immediate forwarding outcomes, enabling convergence within 180 seconds (Figure 1).

NDN forwarding involves a limited action space (5-10 faces) and relatively stable state transitions. This differs from large state spaces in gaming/robotics, where future planning is essential. Low γ suffices because in forwarding means considering next 1-2 hops, not extended sequences.

For forwarding, where immediate feedback such as: RTT, throughput, and jitter are directly observable and highly informative, immediate reward r should dominate updates. Then, γ=0.001 is optimal because NDN forwarding is a myopic decision problem where immediate feedback is maximally informative. All configurations converge to stable policies, confirming QLAF scales across network complexities. Similar final rewards (2.5-4.0) despite different face counts indicate the algorithm identifies best paths regardless of action space size. This supports the hyperparameter configuration (α=0.6, γ=0.001, ε=0.1) from grid search over 432 combinations, enabling consistent performance across varying conditions.

3-2- Disaster Network Architecture

The architecture proposed in this study reflects the condition of the disaster situation as explained in the previous part. It serves as the foundation for evaluating the Q-Learning-based Adaptive Forwarding (QLAF) strategy in heterogeneous network environments.

Figure 3 shows the disaster communication network architecture for implementing and evaluating QLAF. The network consists of seven nodes: five routers (R1–R5), one consumer node (C) representing the field responder, and one producer node (P) representing the command center.

Communication links are divided into three types with distinct latency characteristics: terrestrial (10–50 ms), cellular (100–200 ms), and satellite (500–800 ms). To simulate disaster degradation, a link failure between R2 and R4 represents lost infrastructure connectivity.

QLAF enables each router to learn optimal forwarding behavior based on experience rather than fixed metrics. By continuously observing network feedback, QLAF autonomously avoids failed or unstable links, prefers mid-latency reliable routes, and activates higher-latency paths only when necessary, maintaining data delivery reliability under severe and changing network conditions.

Figure 3 demonstrates how QLAF uses learning-based forwarding to dynamically balance latency, reliability, and availability across terrestrial, cellular, and satellite domains, ensuring continuous communication during emergency response operations.

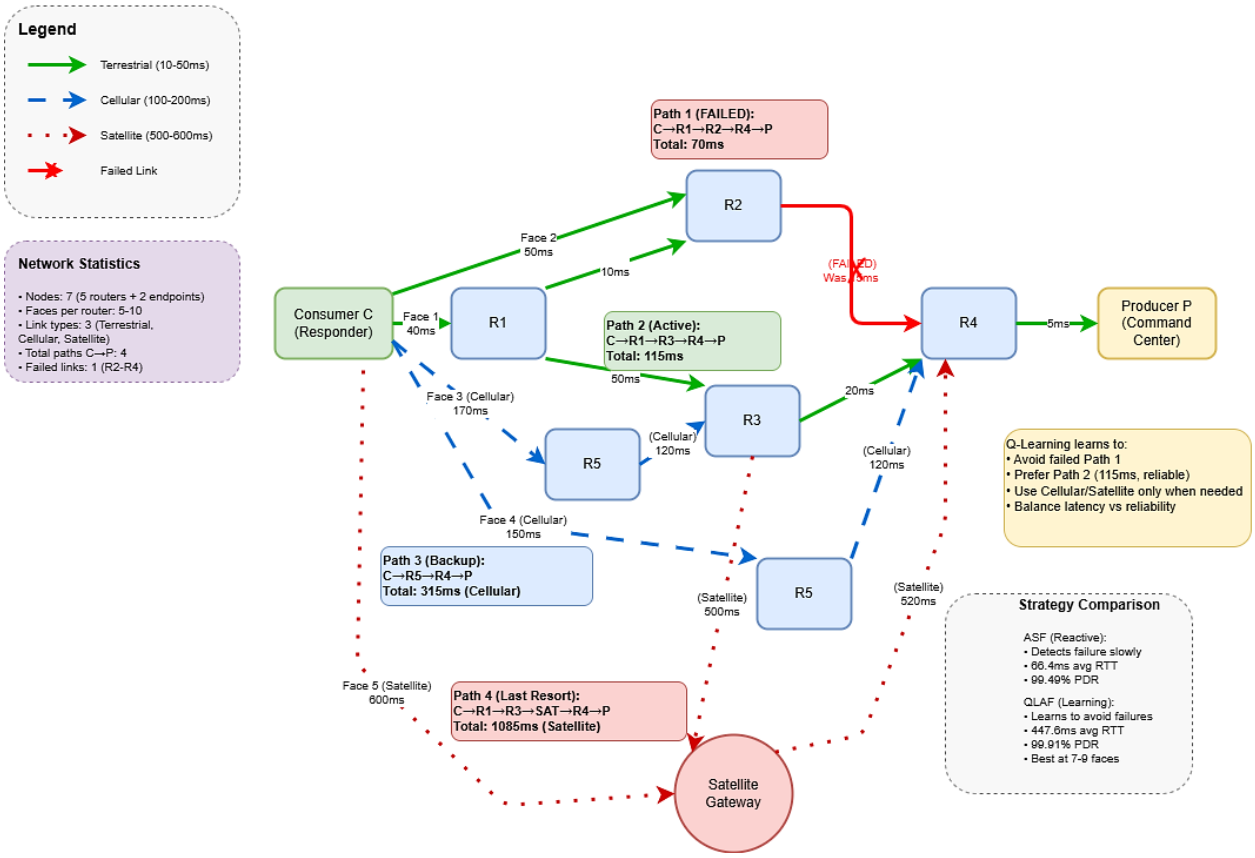

Figure 3. Disaster Communication Network Topology

3-3- Problem Statement

Although various technologies: satellite, cellular, mesh, and UAV networks, have been developed to improve communication availability, these approaches still rely on location-based routing and addressing mechanisms that cannot ensure content accessibility when data sources become isolated.

NDN introduces a data-centric paradigm designed to improve communication reliability under disaster conditions through in-network caching, multipath forwarding, and data-plane retransmission mechanisms. However, NDN effectiveness depends mainly on its forwarding strategy, which controls the selection of the optimal interface for forwarding Interest packets. Conventional strategies such as Best-Route and Adaptive Smoothed RTT-based Forwarding (ASF) rely on reactive local heuristics, while Self-Learning Forwarding often causes excessive flooding and significant overhead due to repeated path exploration.

The central research problem lies in designing a forwarding strategy that can**:** Adapt dynamically to changing network conditions without relying on control-plane routing mechanisms; Optimize the trade-off between reliability (measured by Packet Delivery Ratio) and latency in highly dynamic disaster environments; and use historical performance patterns to enable more efficient and convergent interface selection. To address these challenges, this study

proposes the Q-Learning-based Adaptive Forwarding (QLAF) strategy, a reinforcement learning-driven mechanism that operates entirely within the data plane. The algorithm enables routers to update Q-values adaptively based on RTT measurements and data delivery ratio, directing packets through the most reliable paths without requiring global topology information.

3-4- Comparison with Related Reinforcement Learning-Based Forwarding Strategies

To contextualize the contributions of QLAF within the broader landscape of reinforcement learning-based forwarding strategies in Named Data Networking, this section provides a comprehensive comparison with six representative approaches: SAF-DRL [39], AFSndn [40], IFS-QLSTM [41], DQN-AF [32], IFS-RL [30], and QRF [37]. These strategies represent the evolution of RL-based adaptive forwarding in NDN, ranging from traditional Q-learning implementations to deep reinforcement learning approaches. The comparison examines algorithmic characteristics, state-action representations, performance metrics, and practical implementation considerations.

3-4-1- Algorithmic and Architectural Comparison

Table 2 presents a comparative overview of the algorithmic foundations and design choices across the studied forwarding strategies. The approaches can be categorized into three main groups: traditional Q-learning methods (QLAF, AFSndn, QRF), hybrid learning approaches (IFS-QLSTM), and deep reinforcement learning methods (SAF-DRL, DQN-AF, IFS-RL).

Table 2. Comparison of RL-Based Forwarding Strategies: Algorithmic Characteristics.

Strategy	Algorithm	State Representation	Action Space	Reward Function	Simulator	NDN Modification
QLAF (Proposed)	Q-Learning	Interest name prefix	Available faces per FIB entry	Multi-metric: throughput (0.5), RTT (0.3), jitter (0.2)	MiniNDN	FIB modification
SAF-DRL [39]	TD3 (Twin Delayed DDPG)	Throughput, delay, error rate	Forwarding probability per face	Delivery rate optimization	ndnSIM	Probability tables
AFSndn [40]	Q-Learning + heuristic	Delay-based metrics	Interface selection	Delay minimization	ndnSIM	FIB modification
IFS-QLSTM [41]	Q-Learning + LSTM	PIT entry rate (predicted)	Congested path avoidance	Congestion-based penalty	ndnSIM	Data packet mod.
DQN-AF [32]	Deep Q-Network	Success rate, face availability	Face selection via DQN	Success/timeout based	ndnSIM	None
IFS-RL [30]	Actor-Critic	Historical RTTs, usage counts	Face + learning granularity	RTT-based	ndnSIM	Interest packet mod.
QRF [37]	Q-Learning	PIT fullness rate	Face selection	PIT-based congestion	ndnSIM	None

QLAF distinguishes itself through several design choices. First, unlike SAF-DRL and DQN-AF that employ deep neural networks, QLAF uses traditional Q-learning with Interest name prefixes as state identifiers. This design trades the representational power of deep learning for reduced computational overhead (252μs per router versus the multi-millisecond inference times typical of neural network-based approaches). Second, QLAF's multi-metric reward function integrating throughput, RTT, and jitter contrasts with single-metric approaches such as QRF (PIT fullness only) and AFSndn (delay only), enabling more nuanced path quality assessment. Third, QLAF preserves NDN architectural principles by avoiding packet modifications, unlike IFS-QLSTM and IFS-RL which require changes to Data and Interest packets respectively.

3-4-2- State-Action Representation Analysis

The effectiveness of reinforcement learning in NDN forwarding depends critically on state representation design. Table 3 compares the state-action characteristics and their implications for learning performance and scalability.

Table 3. State-Action Representation Comparison and Trade-offs

Strategy	State Variables	Advantages	Limitations	Q-Table Size
QLAF	Interest name prefix string (avg. 70 chars)	Direct content-name mapping; preserves NDN semantics	String hashing overhead (18μs); memory growth with content diversity	Dynamic
SAF-DRL	Throughput, delay, error rate vectors	Comprehensive network view; handles high-dimensional FIB entries	Higher computational overhead; requires a continuous state space	N/A (neural net)
AFSndn	Delay measurements per interface	Simple and fast; low overhead	Limited perspective; ignores throughput and loss metrics	Undetermined
IFS-QLSTM	PIT entry rate (LSTM-predicted)	Predictive congestion detection; proactive path switching	LSTM computation overhead; training data requirements	Undetermined
DQN-AF	Face success rate, availability	Captures reliability; moderate complexity	May miss latency variations; neural network inference cost	N/A (neural net)
IFS-RL	Historical RTT sequences, usage counts	Temporal awareness: canonical input for topology changes	Large state space with many interfaces	Bounded (max 48)
QRF	PIT fullness percentage	Congestion awareness; minimal overhead; determined size	Local information only; no path quality history	Determined

3-4-3- Performance Metrics Comparison

Direct performance comparison across studies is challenging due to varying experimental setups, topologies, and traffic patterns. Table 4 synthesizes reported performance improvements relative to baseline strategies in each study, providing contextual benchmarks for evaluating QLAF's contributions.

Table 4. Performance Comparison: Reported Results from Each Study

Strategy	Packet Delivery / Loss	Latency / RTT	Throughput	Key Findings
QLAF	99.91% PDR vs ASF 99.49%, Self-Learning 73.47%	447.6ms avg RTT; 381ms overhead vs. ASF	Not primary metric	Superior reliability at 7-9 faces; latency trade-off for reliability
SAF-DRL [42]	Lower loss than BR, RFA, SAF across LFR 10-70%	Lower delivery time than baselines at 1/3/5 Mbps	Improved vs baselines	TD3 handles overestimation; robust to link failures
AFSndn [40]	Lower loss than MFC algorithm	32% lower delay than MFC; comparable to RFA	Not reported	Heuristic accelerates convergence (cycle 13 vs 43)
IFS-QLSTM [43]	0.16% loss (high cong.) vs. ASF 14.7%, BR 18.8%	Stable under congestion	21.1% higher than BR in high congestion	LSTM predicts congestion; minimal loss under stress
DQN-AF [32]	Similar to ASF (~0.07%)	Faster adaptation to 4s link failures	89.69 pkt/s vs BR 77.56 (15.6% improvement)	RMSprop optimizer outperforms Adam by 0.49-6.09%
IFS-RL [41]	Best among tested strategies	Not primary focus	Highest among EPF and others	Suboptimal load balance due to minimal RTT focus
QRF [27]	Outperforms BR in congestion	Discovers time-moderate paths via exploration	Not reported	No NDN modifications; reasonable Q-table size

3-4-4- Convergence and Scalability Analysis

Convergence behavior and scalability characteristics vary significantly across the compared strategies. AFSndn demonstrates accelerated convergence through heuristic knowledge injection, reaching stable Q-values at cycle 13 compared to standard Q-learning's cycle 43 (70% reduction). The number of learning agents also affects convergence speed: configurations with N=1, N=2, and N=6 agents converge at 12, 9, and 5 cycles respectively [27]. QLAF requires approximately 30,000-40,000 Interest packets before achieving stable performance, which is longer than AFSndn but provides more robust adaptation to heterogeneous disaster scenarios.

SAF-DRL exhibits consistent convergence behavior where all agents converge to approximately the same stable value despite different initial learning speeds, demonstrating the stability of deep RL approaches in high-dimensional state spaces [42]. IFS-RL addresses topology change scalability through canonical input/output formats supporting up to 48 interfaces, with masking applied to the softmax output for available interfaces [30]. QRF employs dynamic switching between exploration and exploitation modes based on packet count and Q-value divergence thresholds, enabling real-time adaptation without predetermined convergence periods [37].

3-4-5- Implementation Complexity and Trade-offs

Table 5 summarizes the practical trade-offs between implementation complexity, performance, computational overhead, and scalability across the compared strategies.

Table 5. Implementation Trade-off Analysis

Strategy	Complexity	Performance	Overhead	Scalability
QLAF	Moderate (Q-table + multi-metric reward)	Excellent PDR (99.91%); High latency (447ms)	Moderate (252µs/router)	Good (stable across 5-10 faces)
SAF-DRL	High (TD3 neural network)	Excellent delivery; Low latency	High (neural network inference)	Good (handles large FIB)
AFSndn	Moderate (Q-learning + heuristic)	Good delay reduction	Moderate (FIB modification)	Moderate (undetermined Q-table)
IFS-QLSTM	High (Q-learning + LSTM)	Excellent under congestion	High (LSTM computation)	Moderate (LSTM training needs)
DQN-AF	Moderate (2-layer MLP)	Very good throughput	Low-Moderate	Good (no packet modifications)
IFS-RL	Moderate (Actor-Critic)	Very good throughput	Low	Good (bounded 48 interfaces)
QRF	Low (basic Q-learning)	Good congestion handling	Very Low	Excellent (determined Q-table)

3-4-6- Discussion and Positioning of QLAF

Comparative analysis reveals that QLAF occupies a distinct position in the landscape of RL-based NDN forwarding strategies. While deep RL approaches (SAF-DRL, IFS-QLSTM, DQN-AF) generally achieve superior latency performance through sophisticated neural network architectures, they incur higher computational overhead and implementation complexity. Lightweight Q-learning approaches (QRF, basic AFSndn) offer simpler implementation but may lack the adaptability needed for heterogeneous disaster scenarios.

QLAF's key differentiator lies in its explicit optimization for reliability over latency, making it particularly suitable for disaster communication scenarios where message delivery is more critical than real-time response. The multi-metric reward function (throughput 50%, RTT 30%, jitter 20%) enables QLAF to make nuanced path quality assessments that single-metric approaches cannot achieve. Furthermore, QLAF's preservation of NDN architectural principles through zero packet modifications enhances deployment feasibility compared to approaches requiring protocol changes.

The counterintuitive optimal discount factor (γ=0.001) discovered through QLAF's grid search contradicts standard RL practice (γ=0.9-0.99) but aligns with findings from IFS-RL and QRF that NDN forwarding benefits from myopic learning prioritizing immediate rewards. This reflects the fundamental difference between NDN forwarding (single-hop decisions with immediate observable outcomes) and traditional RL domains (long action sequences affecting distant future states).

Future research directions emerging from this comparison include: (1) hybrid approaches combining QLAF's reliability focus with the latency advantages of measurement-based strategies like ASF, (2) adaptive reward weight adjustment based on application requirements, and (3) integration of predictive mechanisms similar to IFS-QLSTM's LSTM component for proactive congestion avoidance while maintaining QLAF's architectural simplicity.

4- Performance Analysis and Parameter Sensitivity

4-1- Experimental Setup

We implemented QLAF in NFD and compared its performance with three other forwarding strategies: ASF (Adaptive Smoothed RTT-based Forwarding), Access, and broadcast-based Self Learning. The experiments used the MiniNDN emulator with a 25-node network topology generated by BRITE. We sent 120,000 Interest packets following a Zipf distribution, which simulates realistic patterns where some content is requested more frequently than others. Network complexity varied from 5 to 10 available interfaces per router, representing different levels of path diversity that might exist in disaster scenarios.

The self-learning strategy, proposed by Shi et al. [42], uses broadcast-based forwarding. When the first Interest packet is sent, it floods across the entire network. Routers observe where the Data packet comes from and create FIB entries for future forwarding. This allows the network to operate without routing protocols and adapt when producers move or network conditions change. However, Self Learning has two limitations. First, it uses a fixed rule (k-shorter prefix approach) that removes the last k components from the Data name to create FIB entries. Second, unlike QLAF, it does not use systematic parameter optimization through learning algorithms.

Before conducting the main experiments, we performed systematic hyperparameter optimization through grid search for QLAF. Learning rate α and discount factor γ were each tested across 12 candidate values: {0.001, 0.01, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0}, while exploration rate ϵ was tested at three values: {0.001, 0.01, 0.1}. This yielded 432 total parameter combinations. Based on evaluation metrics, including packet delivery ratio and convergence stability, gamma equals 0.001, and., $\gamma = 0.001, \varepsilon = 0.1$ demonstrated the most stable performance.

4-2- Computational Overhead Model

To analytically separate Q-Learning's computational cost from network propagation delays, we developed a processing latency model based on our NFD implementation. The total processing time at the router i for the forwarding strategy S Decomposes into base NFD operations plus strategy-specific overhead:

$$L_{proc}^{S}(i) = L_{base}(i) + L_{strat}^{S}(i) \quad (1)$$

where, $L_{base}(i)$ Represents baseline NFD processing, including FIB (Forwarding Information Base) lookup and PIT (Pending Interest Table) management operations, standard to all strategies. The term $L_{strat}^{S}(i)$ denotes strategy-specific decision-making overhead, which varies substantially across forwarding algorithms.

The access strategy requires minimal overhead, approximately 10–15 μs per router, consisting solely of a simple FIB lookup to identify the lowest-cost next hop. The ASF strategy introduces adaptive behavior by maintaining smoothed RTT measurements per face. Its overhead grows linearly with face count:

$$L_{strat}^{ASF} = (5N_f + 5)\mu s \quad (2)$$

where, N_f represents the number of available faces in the FIB entry, the coefficient 5 reflects per-face operations: accessing SRTT (Smoothed Round-Trip Time) values requires 3μs per face, comparing faces to select the minimum SRTT takes 2μs per face, and scheduling RTO (Retransmission TimeOut) adds a constant 5μs. For typical disaster scenarios with N_f = 5 faces, the ASF overhead approximates 30 μs per router.

QLAF involves substantially more complex operations, decomposed as:

$$L_{strat}^{QLAF} = \tau_{QL} + \tau_{\epsilon} + \tau_{dec} + \tau_{rew} + \tau_{upd} \quad (3)$$

where each component represents a distinct processing phase. Q-table lookup (τ_{QL} = 29 μs) uses Interest name strings, averaging 70 characters, as state identifiers. Hash computation for string-based keys requires 18 μs using std::hash. Unordered_map access costs 8 µs, including collision resolution, and retrieving the Q-value vector from the map entry adds 3 μs. Epsilon decay (τ_{ϵ} = 17μs) computes the exploration rate decreasing exponentially over time via $\epsilon(t) = \epsilon_0 \cdot e^{-\lambda t}$ where ϵ_0=0.1 is the initial exploration rate, λ is the decay constant, and t is elapsed time. Epsilon-greedy decision (τ_{dec} = 2.4μs) generates a random number $r \sim U(0,1)$ using Mersenne Twister (1.2μs), compares r with current ϵ (0.4μs), then either selects a random action or the maximum Q-value action, averaging 0.8μs.

Reward computation (τ_{rew} = 95 μs) dominates QLAF overhead due to the multi-metric evaluation. The reward function combines throughput, RTT, and jitter scores:

$$R = 10 \cdot \left(w_{thr} \cdot S_{thr} + w_{rtt} \cdot S_{rtt} + w_{jit} \cdot S_{jit}\right) \quad (4)$$

where, w_{thr}=0.5, w_{rtt}=0.3, w_{jit}=0.2 are weight parameters determining optimization priority. Each score component requires the evaluation of an exponential function. Throughput score:

$$S_{thr} = 1 - e^{-T/T_{scale}} \quad (5)$$

where, T is observed throughput (bytes/ms) computed as the Data packet size divided by the smoothed RTT, and T_{scale}=500,000 bytes/s is a normalization constant representing typical network capacity. This sigmoid-like function maps throughput to the [0,1] range, approaching 1 for high throughput. Exponential computation costs 28μs.

RTT score:

$$S_{rtt} = e^{-SRTT/R_{scale}} \quad (6)$$

where, $SRTT$ is the smoothed round-trip time measured in milliseconds, and R_{scale}=80ms represents the target latency threshold. Lower RTT yields higher scores approaching 1, while high RTT drives scores toward 0. Exponential evaluation: 32μs.

Jitter score:

$$S_{jit} = e^{-|\Delta RTT|/J_{scale}} \quad (7)$$

where, $\Delta RTT = |RTT_{last} - SRTT|$ measures instantaneous RTT deviation from the smoothed value (jitter), and J_{scale}=20ms is the jitter tolerance threshold. Stable paths with low jitter receive scores near 1. Computing absolute difference (3μs) plus exponential (32μs) totals 35μs. Summing weighted scores and scaling by 10 to produce a final reward in [0,10] range adds 2μs. Total reward computation: 28 + 32 + 35 + 2 = 97μs, approximated as 95μs.

Q-value update (τ_{upd} = 8.5μs) applies the standard Q-Learning Bellman equation:

$$Q(s,a) \leftarrow (1-\alpha)Q(s,a) + \alpha\left(R + \gamma \max_{a'} Q(s',a')\right) \quad (8)$$

where, $Q(s,a)$ is the current Q-value for the state-action pair (s,a), α=0.6 is the learning rate controlling the update magnitude, R is the observed reward from Equation 4, γ=0.001 is the discount factor determining future reward importance, and $\max_{a'} Q(s',a')$ is the maximum Q-value over all actions in the next state s'. Identifying the maximum Q-value requires iterating through Q-table entries (3.5μs for typical 5-10 actions), computing the temporal difference $R + \gamma \max_{a'} - Q(s,)$ which involves two multiplications and two additions (2μs), scaling by the learning rate α (0.5μs), and writing the updated value back to the Q-table (2.5μs). Total: 8.5μs.

Accounting for CPU cache misses and memory access patterns in realistic deployment scenarios, total QLAF overhead per router reaches approximately 252μs (29 + 17 + 2.4 + 95 + 8.5 ≈ 152μs base plus 100μs cache miss penalty).

For disaster scenarios with a typical path length H=5 hops, the cumulative computational overhead difference between QLAF and ASF becomes:

$$\Delta L_{oh} = H \times \left(L_{strat}^{QLAF} - L_{strat}^{ASF}\right) = 5 \times (252 - 30) = 1.11 \text{ ms} \quad (9)$$

This analytical model establishes a theoretical lower bound on QLAF's computational cost attributable solely to algorithm execution. However, as empirical results demonstrate, actual end-to-end latency differences can substantially exceed this prediction if Q-Learning's path selection decisions favor reliable but slower routes over fast but potentially lossy paths.

4-3- Experimental Results

Packet Delivery Performance: Table 6 presents measured packet loss percentages across all four strategies.

Table 6. Packet loss Comparison (%)

Faces	QLAF	ASF	Access	Self-Learning
5	0.06	0.00	0.00	–
6	0.24	0.01	0.01	24.57
7	0.07	1.37	0.00	5.23
8	0.06	0.05	0.00	11.31
9	0.06	1.53	1.06	24.51
10	0.06	0.10	1.04	67.04
Average	**0.09**	**0.51**	**0.19**	**26.53**

QLAF achieved a 99.91% average packet delivery ratio (0.09% loss), compared to ASF's 99.49% (0.51% loss), Access's 99.81% (0.19% loss), and Self Learning's 73.47% (26.53% loss). Self-Learning's poor performance comes from its k-shorter prefix approach, which is unable to adapt to varying application naming patterns. Without systematic parameter optimization, Self Learning's flood-and-learn mechanism has two main problems: first, unnecessary flooding when FIB prefixes are too specific, and second, incorrect forwarding when prefixes are too general.

When routers had seven interfaces, QLAF's 0.07% loss compared to ASF's 1.37% shows a 1.30 percentage point improvement, while Self Learning had 5.23% loss. With nine interfaces, QLAF's 0.06% loss compared to ASF's 1.53% shows a 1.47 percentage point improvement, while Self Learning's loss increased to 24.51%. Self-Learning's increasing losses at higher interface counts (reaching 67.04% at 10 interfaces) show the growing problems from using fixed prefix rules without learning-based adaptation.

QLAF's Q-Learning mechanism solves Self Learning's fundamental limitations by learning effective forwarding decisions from experience rather than using fixed rules. The reward function (Equation 4), which combines throughput, RTT, and jitter values, allows QLAF to distinguish between reliable and unreliable paths and to avoid problematic interfaces during changing network conditions through its learning-based schema. QLAF's Q-Learning mechanism addresses Self Learning's fundamental limitations by learning effective forwarding decisions from experience rather than relying on fixed heuristics. The reward function (4), which includes throughput, RTT, and jitter values, makes QLAF able to distinguish between reliable and unreliable paths, avoiding problematic faces during transient network conditions through its learning-based approach.

Round-Trip Time Performance*:* Table 7 presents measured latency across all strategies.

Table 7. Round-Trip Time (milliseconds)

Faces	QLAF	ASF	Access	Self-Learning
5	486.16	106.56	11.57	–
6	220.39	109.12	104.13	1414.25
7	491.77	40.75	16.31	1234.79
8	482.71	44.38	12.45	1461.47
9	508.22	47.48	52.55	2308.13
10	496.23	50.23	68.90	3335.76
Average	**447.6**	**66.4**	**44.3**	**1950.9**

QLAF's 447.6ms average RTT represents a fundamental reliability-responsiveness trade-off. The 381ms latency overhead (447.6ms - 66.4ms) represents a design trade-off. For disaster applications with 400-500ms tolerance, QLAF's 0.42pp PDR improvement justifies the delay. For sub-100-ms requirements, ASF or Access remains more appropriate despite lower reliability. However, Self Learning showed extremely high latency with 1950.9ms average RTT, reaching 3.3 seconds at 10 interfaces. Self-learning's high latency comes from repeated flooding caused by incorrect FIB prefix granularity. When the prefix is too specific, for example, /A/B/C, when the producer actually serves /A/B, subsequent Interest packets for other content under the same producer, such as /A/B/D, trigger unnecessary flooding across the entire network.

This problem becomes worse as network complexity increases. With six interfaces, Self Learning's 1414ms RTT is 6.4 times higher than QLAF's latency. With 10 interfaces, this difference grows to 6.7 times (3336ms versus 496ms), showing that Self Learning cannot handle complex networks. On the other hand, QLAF maintains relatively stable

latency across different interface counts, with an average of 447ms and a range of 288ms, because the learning mechanism prevents repeated flooding by adapting forwarding decisions based on observed network conditions.

4-4- Understanding QLAF's Latency Overhead

The actual latency overhead of 381ms is 345 times higher than the predicted 1.11ms from our calculation model. This huge gap cannot be explained by computational costs alone. The measured RTT consists of two main components:

$$RTT_{measured} = L_{comp} + L_{path} \tag{10}$$

where, L_{comp} represents computational latency from algorithm process (approximately 1.11ms according to our analytical model), and L_{path} represents path-induced latency resulting from forwarding decisions that select specific network paths over others. Solving for path-induced components:

$$L_{path} = RTT_{QLAF} - RTT_{ASF} - L_{comp} \approx 447.6 - 66.4 - 1.11 = 380.1 \text{ ms} \tag{11}$$

This calculation reveals that 99.7% of QLAF's measured overhead stems from the paths that Q-Learning selects, rather than the computational costs of the selection process itself.

Three mechanisms explain path-induced latency. First, exploration overhead persists even with a reduced ϵ value of 0.1. Random face selection still applies to approximately 10% of forwarding decisions, or around 12,000 out of 120,000 Interests in our experiments. Exploration necessarily samples diverse paths, including occasionally high-latency backup routes. In heterogeneous disaster networks that combine terrestrial links (with approximately 10–50ms RTT), cellular connections (approximately 100–200ms), and satellite backhaul (approximately 500–600ms), random exploration periodically selects the worst available paths. ASF's purely measurement-based approach, which exploits the lowest SRTT face, avoids this overhead.

Second, QLAF's multi-metric reward function (4) with weights w_{thr}=0.5, w_{rtt}=0.3, w_{jit}=0.2 explicitly prioritizes throughput score over RTT score. When Q-Learning encounters a choice between a fast-but-lossy path (for example, a congested terrestrial link dropping 5% packets with 40ms RTT) versus a slow-but-reliable path (for example, an uncongested satellite link with 0% loss and 500ms RTT), the reward function's throughput emphasis causes Q-values to favor the reliable path. Over 120,000 Interests, this systematic bias toward reliability accumulates a substantial latency penalty.

Third, QLAF maintains updated Q-values by sending some traffic to non-optimal interfaces, ensuring the algorithm can detect if those paths improve or if the primary path becomes worse. ASF concentrates all traffic on the interface with the lowest RTT at that moment. When that interface happens to be a fast path (for example, a 40ms terrestrial link), ASF achieves very low latency. QLAF uses more careful exploration by occasionally sending traffic to alternative paths (for example, a 500ms satellite backup), keeping Q-value estimates current for all available routes. This preparation for potential failures increases average latency during stable network periods.

4-5- Comparative Analysis: Learning-Based vs Heuristic Approaches

Self-Learning and QLAF both learn from experience, but they work very differently. Self-Learning uses a simple fixed rule: remove the last k parts from the content name to create forwarding entries. This same rule applies everywhere, regardless of network conditions or how different applications name their content. Shi and Newberry recognized this problem in their original paper. They proposed that content producers should announce their prefixes to the network, but this makes the system more complex and requires producers to participate actively.

QLAF's Q-Learning solves this problem by learning which interface works best for each content prefix through experience. The Bellman update formula (Equation 8) uses a very low discount factor ($\gamma = 0.001$), meaning it focuses on immediate results rather than trying to predict future outcomes. This approach worked well in our experiments with different naming patterns and does not require producers to do anything special; it works with standard NDN operations.

However, QLAF's better reliability comes with a higher delay compared to simpler strategies. To fairly compare strategies, we calculate expected delivery time, including packet retransmissions:

$$T_{expected} = \frac{RTT}{PDR} \tag{12}$$

Computing expected delivery times: QLAF achieves $\frac{447.6}{0.9991} = 448.0$ ms, ASF achieves $\frac{66.4}{0.9949} = 66.7$ ms, Access achieves $\frac{44.3}{0.9981} = 44.4$ ms, and Self Learning achieves $\frac{1950.9}{0.7347} = 2655.6$ ms. ASF delivers 6.7 times faster than QLAF, accounting for retries, while QLAF delivers 5.9 times faster than Self Learning. This positions QLAF in the middle ground, sacrificing latency for simpler measurement-based strategies but achieving substantially better reliability and latency compared to naive learning approaches without systematic parameter optimization.

4-6- Parameter Sensitivity and Design Implications

Grid search results show that reinforcement learning parameters are extremely sensitive in networking applications. The optimal configuration (α=0.6, γ=0.001, ε=0.1) achieved 99.91% PDR, while Self Learning's built-in parameters (no learning rate adjustment, no discount factor, fixed exploration through flooding, k=1 for prefix creation) produced only 73.47% PDR.

The optimal γ=0.001 is surprisingly low and goes against standard RL practice, which typically recommends γ=0.9–0.99 for balancing immediate and future rewards. This reflects a fundamental difference between NDN forwarding and traditional RL domains. In game playing or robotics, agents control long action sequences where current decisions affect distant future states, justifying high discount factors for the accumulation of long-term rewards. NDN routers make single-hop decisions where the next state, specifically whether the Data packet returns successfully, depends primarily on the immediate next-hop quality rather than distant downstream routers beyond control. Network volatility resulting from topology changes, link failures, and congestion fluctuations renders future reward estimation beyond immediate hops unreliable, necessitating myopic learning that prioritizes observed immediate rewards.

Self-learning failure at higher face counts (67% loss at 10 faces) versus QLAF's stability (0.06% loss at 10 faces) demonstrates the benefit of adaptive parameter optimization in this scenario. Fixed heuristics, as tested in Shi et al. [42] original work using 12-node networks fails to scale to moderate complexity (25 nodes, 10 faces) without learning-based adaptation that incorporates systematic parameter search.

4-7- Mathematical Model Summary and Traceability

This subsection explains the origin of all mathematical formulas and their calculation in the performance analysis. This makes it easy to verify calculations and understand how variables depend on each other. Figure 4 and Table 8 list each formula with definitions, data sources, and calculation steps.

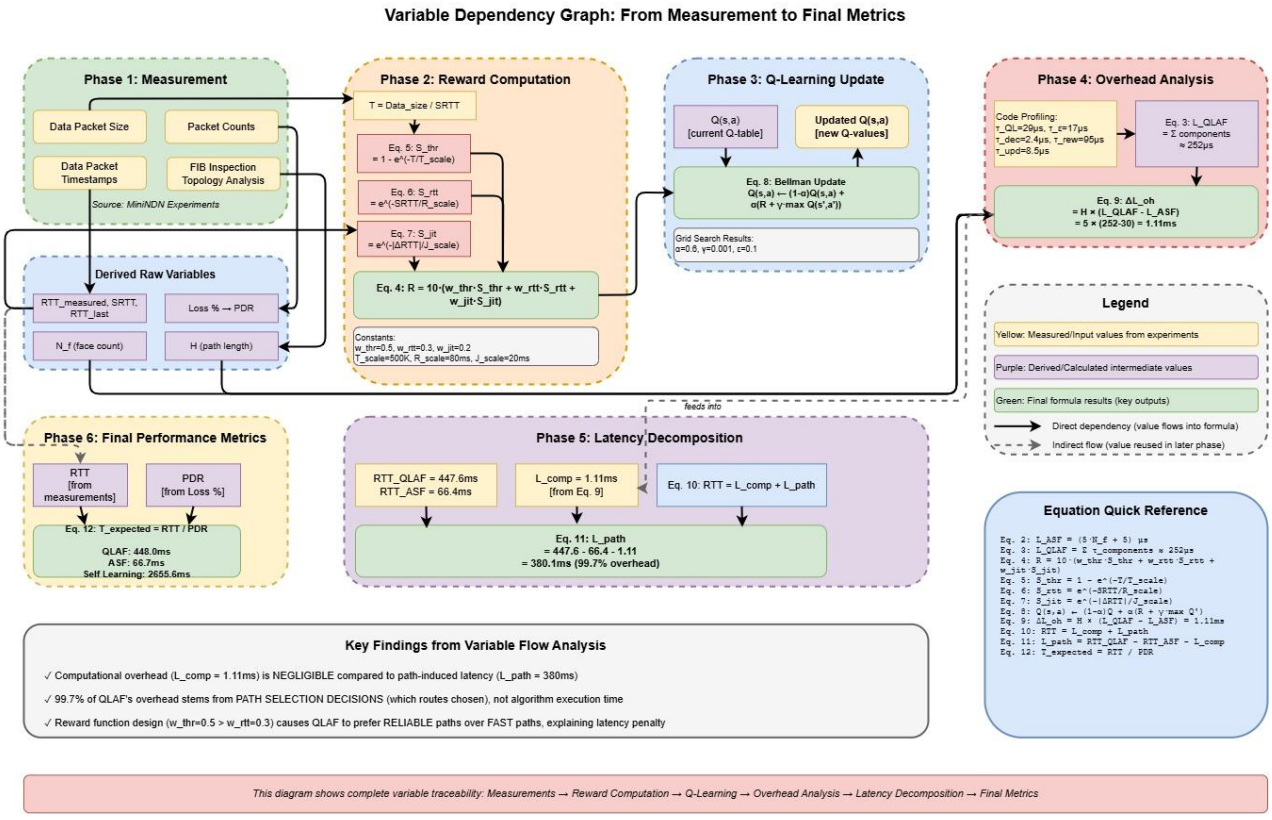

Figure 4. Variable Dependency Graph

Table 8 shows how we calculate the results step by step. There are six steps: (1) collect data from MiniNDN experiments, (2) calculate rewards using Equations 4 to 7, (3) update Q-values using Equation 8, (4) measure processing time using Equations 2 and 3, (5) separate computation delay from path delay using Equation 11, and (6) calculate final performance using Equation 12. The results show that 99.7% of QLAF's delay comes from path selection, not from the algorithm's processing time.

Table 8. Step by step experiment

Phase	Input	Process/Equation	Output
Measurement	Data packet timestamps	MiniNDN experiments	RTT_measured, SRTT, RTT_last
	Packet counts	Loss calculation	PDR
	Data packet size + SRTT	Throughput calculation	T
	FIB inspection	Face enumeration	N_f
	Topology analysis	Path measurement	H
Reward Computation	T	Equation 5	S_thr
	SRTT	Equation 6	S_rtt
	RTT_last, SRTT	Equation 7	S_jit
	S_thr, S_rtt, S_jit	Equation 4	R
Learning	R, Q(s,a), α, γ	Equation 8	Updated Q(s,a)
Overhead Analysis	Code profiling	Component timing	τ_QL, τ_ε, τ_dec, τ_rew, τ_upd
	Components	Equation 3	L_strat^QLAF
	N_f	Equation 2	L_strat^ASF
	H, L_strat^QLAF, L_strat^ASF	Equation 9	ΔL_oh
Performance Decomposition	RTT_QLAF, RTT_ASF, L_comp	Equation 11	L_path
Final Metrics	RTT, PDR	Equation 12	T_expected

1) ***Parameter Sources Summary**:* Table 9 categorizes all parameters by their origin, distinguishing measured values, design choices, and derived calculations.

Table 9. Parameter Source Categories

Category	Parameters
Measured from Experiments	$RTT_{measured}$, $SRTT$, RTT_{last}, Packet Loss %, Data packet size, N_f
Grid Search Results	α=0.6, γ=0.001, ϵ=0.1
Design Choices	w_{thr}=0.5, w_{rtt}=0.3, w_{jit}=0.2, Reward scale factor = 10
Literature/Standards	T_{scale}=500,000 bytes/s, R_{scale}=80ms, J_{scale}=20ms
Code Profiling	τ_{QL}=29µs, τ_{ϵ}=17µs, τ_{dec}=2.4µs, τ_{rew}=95µs, τ_{upd}=8.5µs
Topology Analysis	H=5 hops (average path length)
Calculated/Derived	S_{thr}, S_{rtt}, S_{jit}, R, $Q(s,a)$, L_{comp}, L_{path}, PDR, $T_{expected}$

2) ***Calculation Example**:* To illustrate the complete computational flow, consider a specific example from experimental data. To show how the calculations work, we use sample values from our experiments in Table 10.

Table 10. Sample Input Values

Parameter	Symbol	Value
Data packet size	-	1024 bytes
Smoothed RTT	SRTT	100 ms
Last RTT	RTT_last	110 ms
Number of faces	N_f	5 faces
Path length	H	5 hops

Table 11 shows an example that demonstrates complete variable propagation from raw measurements through reward computation, Q-Learning update, and overhead analysis, matching values reported in experimental results.

Table 11. Step-by-Step Calculation Results

Step	Calculation	Equation	Result
1	Throughput	T = 1024 bytes / 0.1 s	10,240 bytes/s
2a	Throughput score	S_thr = 1 - e^(-10240/500000)	0.0203
2b	RTT score	S_rtt = e^(-100/80)	0.2865
2c	Jitter	ΔRTT = |110 - 100|	10 ms
2d	Jitter score	S_jit = e^(-10/20)	0.6065
3	Reward	R = 10 × (0.5×0.0203 + 0.3×0.2865 + 0.2×0.6065)	2.18
4	Q-value update	Q(s,a) = 0.4×5.0 + 0.6×(2.18 + 0.001×6.0)	3.31
5a	ASF overhead	L_strat^ASF = (5×5 + 5)	30 µs
5b	QLAF overhead	L_strat^QLAF = 29+17+2.4+95+8.5	252 µs
5c	Total overhead difference	ΔL_oh = 5 × (252 - 30)	1.11 ms

The results show that QLAF delivers 99.91% of packets but takes longer 447.6ms compared to ASF's 66.4ms. This slower speed is acceptable when losing messages is worse than delayed delivery, such as sending survivor locations or resource requests. However, QLAF is not suitable for voice calls or videos that need fast response under 100ms. QLAF works best with 7-9 network interfaces per router. With fewer interfaces, simpler strategies work just as well. With more interfaces (10 faces), QLAF stays stable while Self-Learning fails badly 67.04% packet loss. The optimal discount factor γ=0.001 is much lower than typical RL settings γ=0.9-0.99 because NDN routers make decisions one hop at a time and cannot predict future network conditions. Before deployment, operators should prepare for the learning period (30,000-40,000 packets) by pre-training Q-tables or using ASF initially. A practical approach is to route critical messages through QLAF for reliability and routine messages through ASF for speed. Overall, QLAF is a specialized solution for disaster networks needing high delivery rates, not a replacement for all forwarding strategies.

5- Conclusion and Future Research

This research proposes Q-Learning-based Adaptive Forwarding (QLAF) for NDN in disaster communication. QLAF performs better than fixed rule-based methods by using adaptive Q-Learning and overcomes the scalability problems of broadcast-based Self Learning. We developed a model to measure QLAF's computational overhead at 252 µs per router (1.11ms for 5-hop paths) compared to ASF's 30 µs. However, experiments show a total latency overhead of 381ms, with 99.7% caused by path selection rather than computation. We tested 432 parameter combinations and found the best configuration: α=0.6, γ=0.001, ε=0.1. The unusually low γ=0.001 (compared to standard RL practice of γ=0.9-0.99) reflects how NDN forwarding works, network conditions beyond the immediate next hop are highly uncertain.

We tested QLAF using MiniNDN with a 25-node network and compared it against ASF (measurement-based), Access, and Self Learning (heuristic-based). QLAF achieved 99.91% of packet delivery versus ASF's 99.49%, Access's 99.81%, and Self Learning's 73.47%. However, QLAF's average delay of 447.6ms versus ASF's 66.4ms (6.7 times slower) makes it specialized for situations where reliability is critical. Self-Learning's k-shorter prefix rule cannot adapt to different naming patterns, causing major failures (67.04% loss at 10 interfaces). QLAF's learning-based adaptation maintains stable performance (0.06% loss at 10 interfaces), showing that adaptive parameters work better than fixed rules. QLAF improves packet delivery by 0.42 percentage points over ASF at 7-9 interfaces, where ASF loses 1.3-1.5%, though with 381ms delay overhead (99.7% from path choices). QLAF's multi-metric reward function is specifically designed for heterogeneous environments combining diverse link technologies. 5G that fast and steady, a Satellite that can send more data but is slow to respond, and Mesh is reliable but has limited speed. QLAF learns optimal policies for each link type through experience accumulation. For example, Q-Learning would converge to policies preferring 5G links for latency-sensitive coordination messages while utilizing satellite links for bulk data transfers requiring high throughput and reliability over responsiveness. This positions QLAF as a middle option: much more reliable than Self Learning but slower than measurement-based strategies.

However, important limitations exist. MiniNDN simulation cannot capture all disaster communication characteristics, including radio interference, infrastructure damage, and responder movement. QLAF needs much more implementation complexity than measurement-based strategies. Q-table management, multi-metric reward calculation (95 µs), epsilon decay scheduling, and Q-value updates are challenging for teams without machine learning experience, especially when making parameters work across different network types and traffic patterns. The 381ms delay makes QLAF unsuitable for delay-sensitive applications like voice and video calls that need under 100ms response times. Finding optimal parameters requires testing hundreds of combinations, and implementation is more complex than simpler approaches. Also, this study only tested a 25-node network. Real disaster networks can be much larger, with hundreds of nodes spread across multiple clusters. Future work should test QLAF on larger networks and may need methods to reduce Q-table memory usage as the network grows.

QLAF's per-packet overhead consists of two operations: Q-value lookup and reward calculation. Q-value lookup uses hash table indexing with O(1) time complexity. Reward calculation iterates over available faces with $O(n_{faces})$ complexity. For typical disaster network routers with n_{faces}=5-15, this represents constant-time operation.

Future research must address fundamental limitations. We found optimal parameters for one scenario, but whether these work for different network sizes and traffic patterns is unknown and needs testing across multiple scenarios. The 381ms delay is the biggest problem that needs solving. Changing the reward function to favor delay over throughput may reduce path-induced latency while keeping reliability benefits. Hybrid approaches using measurement-based forwarding normally and switching to QLAF when detecting high packet loss could provide speed during regular operation and reliability during disruptions. Each approach needs experimental validation, measuring actual delay-reliability trade-offs rather than theoretical predictions. This work validates QLAF for moderate-scale disaster networks. We recognize that real-world deployments often form natural clusters rather than large-unified topologies and extending QLAF to heterogeneous 5G-satellite environments remains a non-trivial challenge for future exploration.

6- Declarations

6-1- Author Contributions

Conceptualization, R.M. and N.R.S.; methodology, R.M.; software, R.M. and G.N.N.; validation, R.M., N.R.S., and E.M.; formal analysis, G.N.N.; investigation, G.N.N.; resources, R.M.; data curation, E.M.; writing—original draft preparation, R.M. and G.N.N.; writing—review and editing, R.M. and G.N.N.; visualization, G.N.N.; supervision, N.R.S. and E.M.; project administration, R.M.; funding acquisition, G.N.N. All authors have read and agreed to the published version of the manuscript.

6-2- Data Availability Statement

Due to the ongoing nature of this research, the experimental data and source code are not publicly available currently. The authors plan to make the dataset and QLAF implementation accessible upon completion of the broader research project. Reasonable inquiries regarding this study may be addressed to the corresponding author.

6-3- Funding and Acknowledgments

The authors gratefully acknowledge the financial support provided by Telkom University and Bandung Institute of Technology (ITB), Indonesia. This research is funded by Hibah PPMI ITB, which also covers the publication costs of this work.

6-4- Institutional Review Board Statement

Not applicable.

6-5- Informed Consent Statement

Not applicable.

6-6- Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this manuscript. In addition, the ethical issues, including plagiarism, informed consent, misconduct, data fabrication and/or falsification, double publication and/or submission, and redundancies have been completely observed by the authors.

7- References

[1] Parajuli, J., & Haynes, K. E. (2016). The earthquake impact on telecommunications infrastructure in Nepal: a preliminary spatial assessment. Regional Science Policy and Practice, 8(3), 95–109. doi:10.1111/rsp3.12075.

[2] Liu, J., Shi, Y., Fadlullah, Z. M., & Kato, N. (2018). Space-air-ground integrated network: A survey. IEEE Communications Surveys and Tutorials, 20(4), 2714–2741. doi:10.1109/COMST.2018.2841996.

[3] Anjum, M. J., Anees, T., Tariq, F., Shaheen, M., Amjad, S., Iftikhar, F., & Ahmad, F. (2023). Space-Air-Ground Integrated Network for Disaster Management: Systematic Literature Review. Applied Computational Intelligence and Soft Computing, 1–20. doi:10.1155/2023/6037882.

[4] Kumbhar, A., Koohifar, F., Güvenç, I., & Mueller, B. (2017). A Survey on Legacy and Emerging Technologies for Public Safety Communications. IEEE Communications Surveys and Tutorials, 19(1), 97–124. doi:10.1109/COMST.2016.2612223.

[5] Jacobson, V., Smetters, D. K., Thornton, J. D., Plass, M., Briggs, N., & Braynard, R. (2012). Networking named content. Communications of the ACM, 55(1), 117–124. doi:10.1145/2063176.2063204.

[6] Jacobson, V., Smetters, D. K., Thornton, J. D., Plass, M. F., Briggs, N. H., & Braynard, R. L. (2009). Networking named content. CoNEXT'09 - Proceedings of the 2009 ACM Conference on Emerging Networking Experiments and Technologies, 1–12. doi:10.1145/1658939.1658941.

[7] Chen, J., Arumaithurai, M., Fu, X., & Ramakrishnan, K. K. (2016). CNS: Content-oriented notification service for managing disasters. ACM-ICN 2016 - Proceedings of the 2016 3rd ACM Conference on Information-Centric Networking, 122–131. doi:10.1145/2984356.2984368.

[8] Afanasyev, A., Shi, J., Zhang, B., Zhang, L., Moiseenko, I., Yu, Y., Shang, W., Li, Y., Mastorakis, S., Huang, Y., Abraham, J. P., DiBenedetto, S., Fan, C., Papadopoulos, C., Pesavento, D., Grassi, G., Pau, G., Zhang, H., Song, T., Yuan, H., Ben Abraham, H., Crowley, P., Amin, S. O., Lehman, V., Chowdhury, M., & Wang, L. (2014). NFD developer's guide (NDN Technical Report NDN-0021). Named Data Networking Project, 1-52. Available online: https://named-data.net/techreport/ndn-0021-1-nfd-developer-guide.pdf (accessed on March 2026).

[9] Zheng, W. E. N., Xin, Q. I., Keping, Y. U., & Takuro, S. A. T. O. (2019). Content-oriented common IoT platform for emergency management scenarios. 2019 22nd International Symposium on Wireless Personal Multimedia Communications (WPMC), 1-6. doi:10.1109/WPMC48795.2019.9096208.

[10] Tran, M. N., & Kim, Y. (2021). Named data networking-based disaster response support system over edge computing infrastructure. Electronics (Switzerland), 10(3), 1–19. doi:10.3390/electronics10030335.

[11] Yi, C., Afanasyev, A., Wang, L., Zhang, B., & Zhang, L. (2012). Adaptive forwarding in named data networking. Computer Communication Review, 42(3), 62–67. doi:10.1145/2317307.2317319.

[12] Zakariyya Gambetta Muhammad, K., Ahmad, B. I., Zafirah, F. D., Larasati, N. D., Riesani, N. T., Butsaina, F. H., Ahdan, S., Hamidie, E. A. Z., & Syambas, N. R. (2024). Performance Analysis of Forwarding Strategy in NDN-Based Vehicle Networks. Proceeding of 2024 the 10th International Conference on Wireless and Telematics, ICWT 2024, 1–6. doi:10.1109/ICWT62080.2024.10674671.

[13] Pu, C., Ahmed, I., Allen, E., & Choo, K. K. R. (2021). A stochastic packet forwarding algorithm in flying ad hoc networks: Design, analysis, and evaluation. IEEE Access, 9, 162614-162632. doi:10.1109/ACCESS.2021.3133850.

[14] Sutton, R. S., & Barto, A. G. (2005). Reinforcement Learning: An Introduction. IEEE Transactions on Neural Networks, 9(5), 712192. doi:10.1109/tnn.1998.712192.

[15] Delvadia, K., & Dutta, N. (2024). Reinforcement learning inspired forwarding strategy for information centric networks using Q-learning algorithm. International Journal of Communication Systems, 37(6), e5707. doi:10.1002/dac.5707.

[16] Hannan, A., Arshad, S., Azam, M. A., Loo, J., Ahmed, S. H., Majeed, M. F., & Shah, S. C. (2018). Disaster management system aided by named data network of things: Architecture, design, and analysis. Sensors (Switzerland), 18(8), 2431. doi:10.3390/s18082431.

[17] Gomes, T., Tapolcai, J., Esposito, C., Hutchison, D., Kuipers, F., Rak, J., De Sousa, A., Iossifides, A., Travanca, R., Andre, J., Jorge, L., Martins, L., Ugalde, P. O., Pasic, A., Pezaros, D., Jouet, S., Secci, S., & Tornatore, M. (2016). A survey of strategies for communication networks to protect against large-scale natural disasters. Proceedings of 2016 8th International Workshop on Resilient Networks Design and Modeling, RNDM 2016, 11–22. doi:10.1109/RNDM.2016.7608263.

[18] Karaman, B., Basturk, I., Taskin, S., Zeydan, E., Kara, F., Beyazit, E. A., Camelo, M., Bjornson, E., & Yanikomeroglu, H. (2026). Solutions for Sustainable and Resilient Communication Infrastructure in Disaster Relief and Management Scenarios. IEEE Communications Surveys and Tutorials, 8, 716–760. doi:10.1109/COMST.2025.3610793.

[19] Rezaeifar, Z., Wang, J., Oh, H., Lee, S. B., & Hur, J. (2019). A reliable adaptive forwarding approach in named data networking. Future Generation Computer Systems, 96, 538–551. doi:10.1016/j.future.2018.12.049.

[20] Awoyemi, B. S., Alfa, A. S., & Maharaj, B. T. (2018). Network restoration for next-generation communication and computing networks. Journal of Computer Networks and Communications, 4134878. doi:10.1155/2018/4134878.

[21] Khaloopour, L., Su, Y., Raskob, F., Meuser, T., Bless, R., Janzen, L., Abedi, K., Andjelkovic, M., Chaari, H., Chakraborty, P., Kreutzer, M., Hollick, M., Strufe, T., Franchi, N., & Jamali, V. (2024). Resilience-by-Design in 6G Networks: Literature Review and Novel Enabling Concepts. IEEE Access, 12(September), 155666–155695. doi:10.1109/ACCESS.2024.3480275.

[22] Sharvari, N. P., Das, D., Bapat, J., & Das, D. (2025). Improved Q-Learning-Based Multi-Hop Routing for UAV-Assisted Communication. IEEE Transactions on Network and Service Management, 22(2), 1330–1344. doi:10.1109/TNSM.2024.3522153.

[23] Abdi, F., Ahmadi, M., & Ghanem, M. (2023). AM-IF: Adaptive Multi-Path Interest Forwarding in named data networking. Future Generation Computer Systems, 148, 564–583. doi:10.1016/j.future.2023.06.021.

[24] Ayadi, M. I., Maizate, A., Ouzzif, M., & Mahmoudi, C. (2019). Deep Learning in Building Management Systems over NDN: Use Case of Forwarding and HVAC Control. 2019 International Conference on Internet of Things (IThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), 1192–1198. doi:10.1109/ithings/greencom/cpscom/smartdata.2019.00200.

[25] Chowdhury, M., Khan, J. A., & Wang, L. (2020). Leveraging Content Connectivity and Location Awareness for Adaptive Forwarding in NDN-based Mobile Ad Hoc Networks. ICN 2020 - Proceedings of the 7th ACM Conference on Information-Centric Networking, 59–69. doi:10.1145/3405656.3418713.

[26] Aboud, A., Touati, H., & Hnich, B. (2019). Efficient forwarding strategy in a NDN-based internet of things. Cluster Computing, 22(3), 805–818. doi:10.1007/s10586-018-2859-7.

[27] Hnaien, H., & Touati, H. (2020). Q-learning based forwarding strategy in named data networks. In International Conference on Computational Science and Its Applications, 434-444. doi:10.1007/978-3-030-58799-4_32.

[28] Akinwande, O. (2018). Interest forwarding in named data networking using reinforcement learning. Sensors (Switzerland), 18(10), 3354. doi:10.3390/s18103354.

[29] Gong, L., Wang, J., Zhang, X., & Lei, K. (2016, August). Intelligent forwarding strategy based on online machine learning in named data networking. 2016 IEEE Trustcom/BigDataSE/ISPA, 1288-1294. doi:10.1109/TrustCom.2016.0206.

[30] Zhang, Y., Xu, K., Bai, B., & Lei, K. (2018). IFS-RL: An intelligent forwarding strategy based on reinforcement learning in named-data networking. NetAI 2018 - Proceedings of the 2018 Workshop on Network Meets AI and ML, Part of SIGCOMM 2018, 54–59. doi:10.1145/3229543.3229547.

[31] Akinwande, O., & Gelenbe, E. (2018). A reinforcement learning approach to adaptive forwarding in named data networking. In Communications in Computer and Information Science (Vol. 935, pp. 211–219). doi:10.1007/978-3-030-00840-6_23.

[32] De Sena, Y. A. B. L., Dias, K. L., & Zanchettin, C. (2020). DQN-AF: Deep Q-Network based Adaptive Forwarding Strategy for Named Data Networking. In Proceedings - 2020 IEEE Latin-American Conference on Communications, LATINCOM 2020, 9282301. doi:10.1109/LATINCOM50620.2020.9282301.

[33] Fu, B., Qian, L., Zhu, Y., & Wang, L. (2017). Reinforcement learning-based algorithm for efficient and adaptive forwarding in named data networking. 2017 IEEE/CIC International Conference on Communications in China, ICCC 2017, 1–6. doi:10.1109/ICCChina.2017.8330354.

[34] Bouzidi, E. H., Outtagarts, A., Langar, R., & Boutaba, R. (2021). Deep Q-Network and Traffic Prediction based Routing Optimization in Software Defined Networks. Journal of Network and Computer Applications, 192. doi:10.1016/j.jnca.2021.103181.

[35] Lehman, V., Gawande, A., Zhang, B., Zhang, L., Aldecoa, R., Krioukov, D., & Wang, L. (2016). An experimental investigation of hyperbolic routing with a smart forwarding plane in NDN. 2016 IEEE/ACM 24th International Symposium on Quality of Service, IWQoS 2016, 7590394. doi:10.1109/IWQoS.2016.7590394.

[36] Posch, D., Rainer, B., & Hellwagner, H. (2017). SAF: Stochastic Adaptive Forwarding in Named Data Networking. In IEEE/ACM Transactions on Networking, 25(2), 1089–1102. doi:10.1109/TNET.2016.2614710.

[37] Mordjana, Y., Djamaa, B., & Senouci, M. R. (2021). A Q-learning based Forwarding Strategy for Named Data Networking. 5th International Conference on Networking and Advanced Systems, ICNAS 2021, 1–6. doi:10.1109/ICNAS53565.2021.9628982.

[38] Zhang, L., Afanasyev, A., Burke, J., Jacobson, V., Claffy, K. C., Crowley, P., Papadopoulos, C., Wang, L., & Zhang, B. (2014). Named data networking. Computer Communication Review, 44(3), 66–73. doi:10.1145/2656877.2656887.

[39] Hao, B., Wang, G., Zhang, M., Zhu, J., Xing, L., & Wu, Q. (2021). Stochastic Adaptive Forwarding Strategy Based on Deep Reinforcement Learning for Secure Mobile Video Communications in NDN. Security and Communication Networks, 2021, 1–13. doi:10.1155/2021/6630717.

[40] Zhang, M., Wang, X., Liu, T., Zhu, J., & Wu, Q. (2020). AFSndn: A novel adaptive forwarding strategy in named data networking based on Q-learning. Peer-to-Peer Networking and Applications, 13(4), 1176–1184. doi:10.1007/s12083-019-00845-w.

[41] Ryu, S., Joe, I., & Kim, W. T. (2021). Intelligent forwarding strategy for congestion control using Q-learning and LSTM in named data networking. Mobile Information Systems, 2021, 1–10. doi:10.1155/2021/5595260.

[42] Shi, J., Newberry, E., & Zhang, B. (2017). On broadcast-based self-learning in named data networking. 2017 IFIP Networking Conference (IFIP Networking) and Workshops: IEEE, 1-9. doi:10.23919/IFIPNetworking.2017.8264832.

[43] Krishnamurthi, R., & Kumar, R. (2025). AF-QLSTM: Acoustic Feature Prediction using Quantum LSTM for Smart City. Proceedings of the 15th International Conference on the Internet of Things, 271–275. doi:10.1145/3770501.3771305.