International Journal of Research in Computing

Volume 05 Issue 02
July 2026

ISSN 2820-2139

International Journal of Research in Computing (IJRC)

Volume 05 Issue 02
July 2026

ISSN 2820-2139

Published by

Faculty of Computing, General Sir John Kotelawala Defence University
Sri Lanka

+94-11-2635268 — editor@ijrcom.org, editorijrc@kdu.ac.lk — http://ijrcom.org/

EDITORIAL COMMITTEE

EDITOR IN CHIEF

CHIEF ADVISORS

ASSOCIATE EDITORS IN CHIEF

MEMBERS OF THE EDITORIAL BOARD

EDITORIAL ASSISTANTS

CONTENTS

ORIGINAL RESEARCH

ShieldLink: Retry-Aware Authenticated Encryption for Secure and Reliable Chiplet Interconnects.
Michél Nguyen ..... (1–16)

Differentiated Thyroid Cancer Recurrence Classification Using Machine Learning Models and Bayesian Neural Networks with Varying Priors: A SHAP-Based Interpretation of the Best Performing Model.
HMNS Kumari, HMLS Kumari and UMMPK Nawarathne ..... (17–42)

Prompt Optimization Strategies for Large Language Models: Insights from Random Prompting with Gemini API.
OAM Jayawardana ..... (43–65)

Machine Learning-Based Student Academic Performance Prediction in a Nigerian University Context: A Comparative Evaluation of Classification Algorithms.
FI Mamudu, TO Taiwo and RO Folaranmi ..... (67–81)

REVIEW ARTICLE

Learning-Rate Scheduling for Neural Network Training: A Scoping Review of ReduceLROnPlateau, Cosine Annealing with Warm Restarts, Cyclical Learning Rates, and Linear Warmup with Linear Decay.
SPGE Illukkumbura ..... (83–107)

International Journal of Research in Computing

ORIGINAL RESEARCH
Volume 5, Issue 2 (2026)

ShieldLink: Retry-Aware Authenticated Encryption for Secure and Reliable Chiplet Interconnects

Michél Nguyen[1]

[1]University of the People, 595 E. Colorado Blvd., Suite 623, Pasadena, CA, 91101, USA

[1] michel_ng@icloud.com | https://orcid.org/0000-0001-6834-4422

Abstract

Chiplet-based systems-in-package increasingly rely on high-speed die-to-die links such as UCIe and CXL, where link-layer retry mechanisms and authenticated encryption are often implemented as separate reliability and security functions. This separation can create a time-of-check/time-of-use risk when acknowledgments advance before authentication completes. This study introduces ShieldLink, a retry-aware authenticated-encryption protocol that integrates link-layer delivery, buffer retirement, and cryptographic verification to enable secure and reliable chiplet interconnects. ShieldLink formalizes a deliverability invariant requiring CRC, sequence, and AEAD verification before receiver advancement. It defines per-frame authentication (Mode A) and epoch authentication (Mode B), evaluates them with a discrete-event Gilbert–Elliott burst-error model, and includes bounded safety exploration plus RTL control-plane resource sizing. Mode A removes the validity-before-verification race while improving goodput by approximately 2.4–2.5 percentage points over the secure naive baseline at representative burst probabilities. Mode B improves wire efficiency, reaching about 0.926 at $M = 32$, but its epoch-retransmission cost causes a crossover around $\pi_B \approx 0.04$ under the default $\beta = 0.2$ stress regime. The results show that authenticated delivery must be treated as part of retry semantics rather than an afterthought. ShieldLink provides an auditable design invariant and practical guidance on mode selection for future secure chiplet interconnect adapters, with relevance to SDG 9: Industry, Innovation and Infrastructure.

KEYWORDS
Chiplet Interconnect, Authenticated Encryption, ARQ, Gilbert–Elliott model, UCIe.

ARTICLE HISTORY
Published: 16 July 2026

DATA/CODE AVAILABILITY
Data and code are publicly available at: `https://github.com/pinkysworld/ShieldLink`; Code, Results, RTL and Figures.

SDG ALIGNMENT
SDG 4
SDG 9
SDG 16

1 Introduction

Chiplet integration is rapidly becoming the default path to scale compute density and yield, pushing more system functionality onto high-speed die-to-die (D2D) links. Standards such as Universal Chiplet Interconnect Express (UCIe) and Compute Express Link (CXL) [1], [2] define interoperable PHY and transaction semantics. Still, they implicitly assume that link reliability mechanisms and cryptographic integrity checks are mutually safe. In practice, designers often deploy

(i) CRC + automatic repeat request (ARQ) to tolerate transient bit errors, and

(ii) authenticated encryption with associated data (AEAD) to protect confidentiality and integrity against an interposer adversary.

This paper focuses on a subtle but high-impact composition hazard: a receiver may acknowledge (ACK) a frame after it passes a fast CRC check and before the slower AEAD verification completes. That early ACK can free buffers and advance sequence windows, creating a validity-before-verification race (a link-layer TOCTOU [8]). An active attacker who can induce or inject corrupted ciphertext can exploit this window to desynchronize the reliability state from the cryptographic state, thereby causing silent drops, use-after-free of payload buffers, or head-of-line stalls that are invisible to the ARQ layer.

ShieldLink addresses this problem by enforcing a deliverability invariant (D-Inv): the receiver must not emit an ACK that advances the sender's base sequence beyond any frame whose payload might later be rejected by AEAD. Intuitively, ACK implies deliverable. Figure 1 contrasts naive stacking with ShieldLink's co-designed gate. Figure 2 sketches the race exploited by early ACK, and Figure 3 summarizes the receiver control states that separate reliability NAKs (recoverable) from security drops (non-recoverable without rekey/reset).

Contributions:

1. We formalize the deliverability invariant (D-Inv) for D2D links with retransmission and authenticated delivery.
2. We propose ShieldLink Mode A (per-frame authentication with ACK gating) and Mode B (epoch authentication) as two operating points.
3. We evaluate wire overhead, goodput, and tail commit latency under a Gilbert–Elliott burst-error model, and report confidence intervals across random seeds.
4. We provide supplementary scripts and RTL stubs to support reproducibility and independent validation.

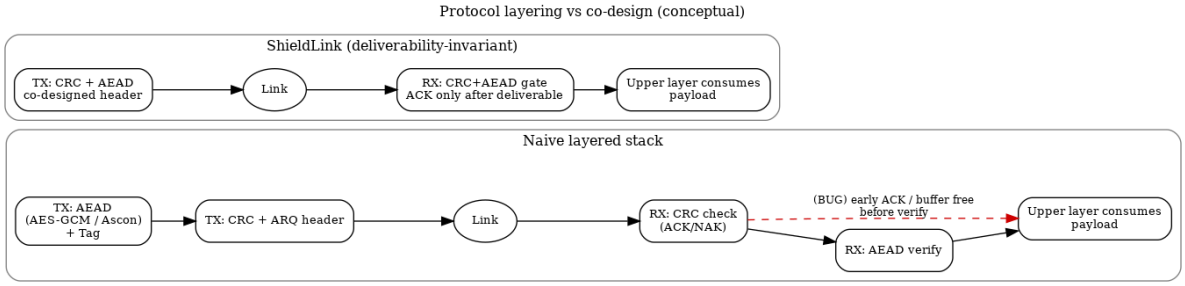


Figure 1: Naive layering can ACK on CRC before AEAD finishes, violating deliverability. ShieldLink couples verification to ACK.

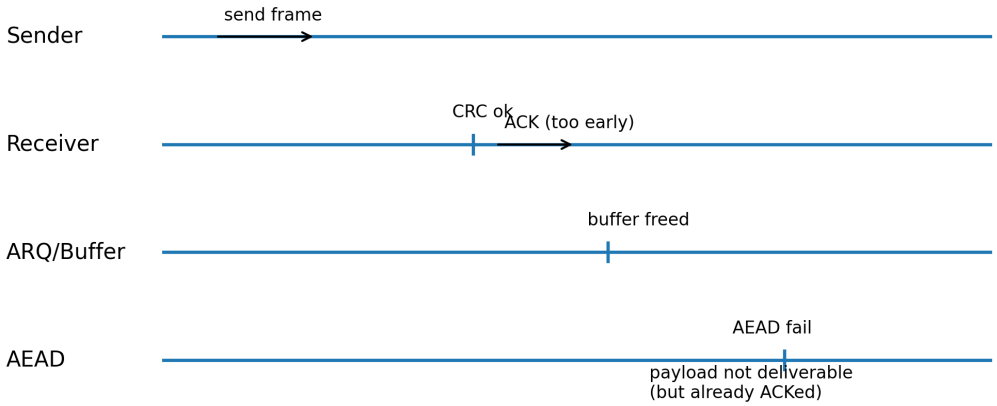


Figure 2: Validity-before-verification race: early ACK/buffer free precedes the AEAD verdict. The gap Δt between CRC_OK and AEAD_FAIL is the vulnerability window in naive stacking.

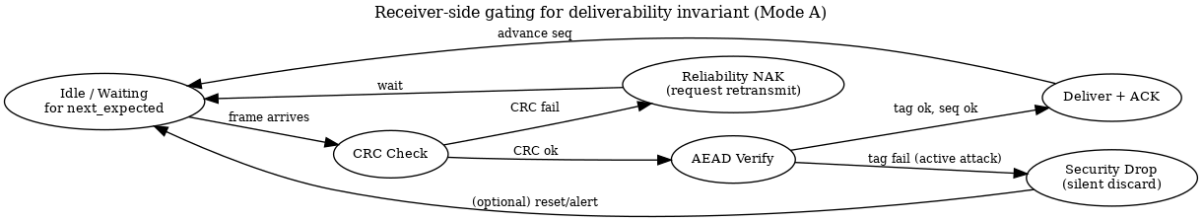


Figure 3: Receiver control separates reliability NAKs (CRC failures) from security drops (AEAD failures without an ACK), thereby avoiding cryptographic oracles and enabling attack detection via telemetry.

2 Literature Review / Background

Chiplet links commonly incorporate CRC, ARQ, and increasingly link security. PCIe and CXL specify Integrity and Data Encryption (IDE)—typically AES-GCM-based AEAD—to provide confidentiality, integrity, and replay protection. However, IDE engines still rely on proper interaction with link-layer sequencing and retransmission: the ShieldLink problem is not a cryptographic primitive but the composition of retransmission state machines with delayed verification.

Prior work motivates these assumptions. Contactless probing demonstrations show that chiplet interposer wiring and I/O drivers can be physically observed with lab equipment, making on-package link confidentiality and integrity relevant even when board-level links are protected [3]. Complementary defenses aim to prevent IP theft and reverse engineering (e.g., dynamic interposer obfuscation) but, by themselves, do not prevent on-path data manipulation [4]. At the system level, root-of-trust proposals for 2.5D integration address supply-chain trust and key provisioning across dies, and can be used to bootstrap ShieldLink security associations [5].

Our threat model overlaps with two well-studied classes: (i) active interposer adversaries that can observe and modify link traffic, and (ii) fault injection attacks (e.g., EMFI) that can transiently corrupt signals. In both cases, a layered CRC/ARQ + AEAD design can mistakenly treat "correct CRC" as "safe to consume". The general TOCTOU hazard is widely documented in software systems; ShieldLink shows an analogous pitfall at the D2D link layer.

2.1 Standards context: where reliability and IDE sit

UCIe defines a die-to-die adapter abstraction that can host reliability mechanisms (CRC, retry buffers, and ARQ) and optional security services. In contrast, CXL/PCIe defines Integrity and Data Encryption (IDE) for board-level links. In practice, these features are often implemented as distinct blocks: a link/transport controller that performs replay buffering and retries, and an IDE engine that performs AEAD-style protection (commonly AES-GCM) with monotonically increasing IV/counter state. The semantic mismatch addressed in this work arises precisely at this boundary: reliability seeks to retire and reuse buffers as early as possible, whereas IDE requires that retirement occur only after authentication completes. ShieldLink should therefore be understood as an adapter-internal contract that ensures CRC/ARQ and IDE are mutually consistent, rather than as a replacement for IDE itself. Our design is compatible with standardized management planes in UCIe (e.g., error counters and link reset hooks) and with IDE threat assumptions in CXL/PCIe. [1], [2], [6]

2.2 Physical and fault-injection threat reality

While cryptographic integrity is often motivated by a "man-in-the-middle" adversary, chiplet packages also admit adversaries that induce structured faults. Recent demonstrations show contactless probing and manipulation of chiplet interposer wiring and I/O die interfaces [3], and broader work emphasizes that 2.5D/3D integration expands the attack surface beyond on-die threats. Even without direct probing, electromagnetic fault injection (EMFI) and related perturbations can create correlated, burst-like errors rather than independent bit flips—an observation that motivates our use of the Gilbert–Elliott (GE) channel model [9], [10], and our discussion of active injection feasibility [12]. ChipletQuake further illustrates that attackers can exploit package-level properties (e.g., interposer impedance) to mount stealthy attacks and to complicate detection, reinforcing the need for link-visible telemetry and reset policies. [11]

2.3 TOCTOU lessons for chiplet links

The core failure mode—acting on a preliminary validity check before a stronger verification completes—is a time-of-check/time-of-use (TOCTOU) pattern. TOCTOU attacks are widely documented in software and firmware systems (e.g., CWE-367) [8] and have also been observed in hardware trust chains (e.g., TOCTOU attacks on device attestation flows) [13]. ShieldLink's "validity-before-verification" vulnerability is a link-layer instance of the same class: a CRC check is necessary for robustness but insufficient for security, and any state transition that depends solely on CRC can be exploitable if the AEAD result is still pending. Positioning the problem this way helps clarify why the proposed invariant is the appropriate abstraction: it elevates "ACK and buffer release" to a use that must be guarded by the strongest check.

2.4 Zero-trust chiplet integration and MPC-style authentication

Recent chiplet security work also explores boot-time and integration-time defenses, including roots of trust for 2.5D systems [5] and mechanisms that treat chiplets as mutually distrustful parties requiring explicit authentication and policy enforcement. Such approaches (including those based on multi-party computation or distributed authentication) primarily address who is allowed to participate and how keys are provisioned, whereas ShieldLink focuses on how the runtime data plane avoids semantic races once keys are in place. These themes are complementary: a zero-trust provisioning path can establish per-link keys and epochs, while ShieldLink ensures that retries and acknowledgments cannot subvert the resulting security guarantees.

2.5 Positioning against IDE/ARQ baselines

Table 1 summarizes the novelty boundary. Existing CRC/ARQ mechanisms provide robustness against noise, and IDE-class AEAD engines provide cryptographic protection, but neither alone specifies the receiver-side contract needed when retransmission state and authentication verdicts are separated by an implementation boundary. ShieldLink contributes to the contract through D-Inv and shows two ways to enforce it.

Table 1: Comparison matrix positioning ShieldLink against reliability-only, IDE-style, and layered baselines.

Approach	Reliability / retry behavior	Authentication behavior	ACK and buffer-retirement semantics
CRC+ARQ only	Retries frames that fail CRC or sequence checks.	No confidentiality or cryptographic integrity.	ACK can retire a CRC-valid frame, but the frame is not security-verified.
IDE / AEAD engine used as a separate block	Depends on surrounding link controller and replay-buffer policy.	Provides confidentiality, integrity, and replay protection after verification completes.	Public standards rarely expose whether ARQ retirement is gated on AEAD; safety depends on integration.
Naive CRC+ARQ + AEAD stacking	Reliability controller may accept and ACK before delayed AEAD verdict.	AEAD can reject after the reliability layer has already advanced.	Illustrates the TOCTOU class: ACK/buffer release is decoupled from authenticated deliverability.
ShieldLink Mode A	Per-frame retry with immediate NAK on CRC failure.	Per-frame AEAD verification.	ACK and NEXT_EXPECTED advance only when CRC_ok, AEAD_ok, and Seq_ok all hold.
ShieldLink Mode B	Epoch retry; all-or-nothing baseline flush on epoch failure.	One tag authenticates M-frame epoch.	ACK is delayed until epoch verification, trading lower overhead for higher buffering and tail latency.

3 Methodology

3.1 Study design and goals

We follow a co-design methodology: define an invariant that captures safe interaction between reliability and security, design protocol mechanisms that enforce it, and evaluate performance and tail behavior under burst errors. The primary correctness objective is D-Inv: ACKs must never outpace authentication.

Deliverability Invariant (D-Inv): For every ShieldLink frame (SLF) s,

$$\text{Deliverable}(s) \equiv \text{CRC\_ok}(s) \wedge \text{AEAD\_ok}(s) \wedge \text{Seq\_ok}(s).$$

Only deliverable SLFs may advance NEXT_EXPECTED and trigger an ACK (buffer retirement); CRC failures trigger a reliability NAK, while AEAD failures trigger a security drop and may contribute to a link-reset threshold.

3.2 Protocol overview

ShieldLink frames (SLFs) carry a fixed 256 B payload, a 16 B link header, and a 4 B CRC; authentication material depends on the operating mode. For comparison, the CRC+ARQ-only baseline in Table 2 assumes a minimum 8-B header (sequence/control) plus a CRC. Mode A appends a per-frame AEAD tag, verifies AEAD before emitting the ACK that advances the sender base, and uses NAKs only for CRC failures. If AEAD fails, the receiver treats it as a security drop (non-recoverable without rekeying or resetting) and requests a link reset/rekey via sideband. Mode B amortizes tags by authenticating M frames with a single tag; the receiver buffers the epoch and appends the tag on the final frame. A CRC failure anywhere in the epoch triggers go-back retransmission of that epoch.

3.3 Simulation model

We implement a discrete-event simulator that models (i) serialization of frames over a link of width $W_{link} = 8$ B/cycle, (ii) CRC delay (1 cycle), (iii) AEAD/epoch verification delay (8 cycles), (iv) propagation delay (2 cycles), and (v) go-back-N ARQ with window $N_{win} = 64$ frames. Errors follow a two-state Gilbert–Elliott channel [9], [10] with good-state corruption probability $p_G = 10^{-6}$, bad-state corruption probability $p_B = 0.1$, and bad-state exit probability $\beta = 0.2$ (mean bad-state duration of five frames). For each bad-state probability π_B, we derive α such that $\pi_B = \alpha/(\alpha+\beta)$. Each point represents the average of five random seeds and reports a 95% confidence interval.

Parameter rationale: p_B is treated as an effective frame-corruption probability in the Bad state rather than a raw bit-error rate. The value $p_B = 0.1$ intentionally stresses the retry/security interaction so that epoch retransmission effects are visible within a bounded simulation. $\beta = 0.2$ corresponds to a mean bad-state duration of $1/\beta = 5$ frames, matching the intended burst-error stress regime; in deployment, these parameters should be calibrated from link telemetry rather than treated as universal constants.

3.4 Metrics

We report normalized goodput (the delivered payload rate divided by the raw link capacity) and p99 latency to assess deliverability. For per-frame protocols (Mode A and Naive), latency is measured from the start of the first transmission to the time the frame becomes deliverable. For Mode B, the deliverable unit is an epoch; we report the p99 epoch-head latency (from the start of the first-frame transmission to epoch commit).

3.5 Bounded safety exploration

To avoid overclaiming formal verification, we treat our protocol proof effort as a bounded safety sanity check, not as a complete formal proof. In addition to the TLA+-style sketch summarized in Appendix D, we implemented a small, explicit-state explorer in Python that models a single stream with go-back-N retransmission, CRC/AEAD outcomes, and duplicate-delivery handling. For a bounded configuration (sequence numbers modulo 8, ARQ window $N_{win} = 4$, exploration depth 14 steps), the included `formal_sanity_check.py` explorer enumerates 845,625 reachable states and finds no counterexamples to the core safety properties: (P1) the receiver never emits ACK unless $\mathrm{CRC\_ok} \wedge \mathrm{AEAD\_ok} \wedge \mathrm{Seq\_ok}$ holds, and (P2) receiver state advances only on deliverable frames. These checks do not prove liveness under all adversarial schedules, but they increase confidence that the FSM and its ACK-gating logic are internally consistent. [14]

3.6 Preliminary RTL structural cost analysis (FPGA/synthesis-friendly control plane)

To address concerns about implementation realism without requiring a full PHY or end-to-end RTL prototype, we developed a small, synthesizable RTL skeleton of the ShieldLink control plane (Modes A and B). The goal is to make the hardware impact of the Deliverability Invariant explicit in terms of (i) state bits, (ii) buffer bits, and (iii) incremental combinational logic depth. We do not claim measured PPA in this section; instead, we report exact memory sizing and a conservative gate-equivalent estimate for the incremental D-Inv gating.

The RTL follows the paper's microarchitecture: CRC and AEAD blocks produce registered one-bit verdicts (crc_ok, aead_ok). The ACK/NAK FSM consumes these registered verdicts, so D-Inv adds only a constant-depth gate on the control path (ack_en = crc_ok ∧ aead_ok ∧ seq_eq). In this organization, the datapath critical path is dominated by the AEAD pipeline and the PHY boundary rather than by the ACK gate itself.

The synthesizable SystemVerilog RTL skeleton, resource-sizing script, and updated simulation generator are provided as supplementary material (Appendix C). They enable independent synthesis experiments, but any LUT/FF/BRAM, f_{max}, area, or power numbers for a specific FPGA or ASIC target should be interpreted as future implementation results rather than claims made by this paper.

4 Results

4.1 Wire efficiency and buffering

Table 2: Overhead and buffering by scheme (payload = 256 B, W_{link} = 8 B/cycle).

Scheme	Bytes/frame	Cycles/frame	Wire efficiency	Verification	RX buffering
Reliability-only (CRC+ARQ)	268	34	0.955	per-frame CRC	≈1 frame
Naive Stacking (secure)	296	37	0.865	per-frame AEAD	≈1 frame
ShieldLink Mode A	288	36	0.889	per-frame AEAD	≈1 frame
ShieldLink Mode B ($M = 32$)	276+tag/32	≈35.0	0.926	per-epoch AEAD	≈16 KiB (2 epochs)

Mode B reduces per-frame overhead by amortizing tags, improving wire efficiency. Figure 4 moves the epoch-size sensitivity plot into the main results to make this amortization explicit. However, Mode B intentionally delays delivery until an epoch is verified, thereby increasing tail latency and making the design more sensitive to bursts.

4.2 Tail latency and goodput under burst errors

In this configuration, Mode B outperforms Mode A at low burst probability due to lower per-frame overhead. Still, once π_B reaches roughly 0.04 in the default $\beta = 0.2$ stress regime, epoch flushes dominate and goodput falls below Mode A. Mode A preserves low tail latency by allowing the ARQ machine to retry immediately on CRC failure while never advancing the ACK beyond verified data.

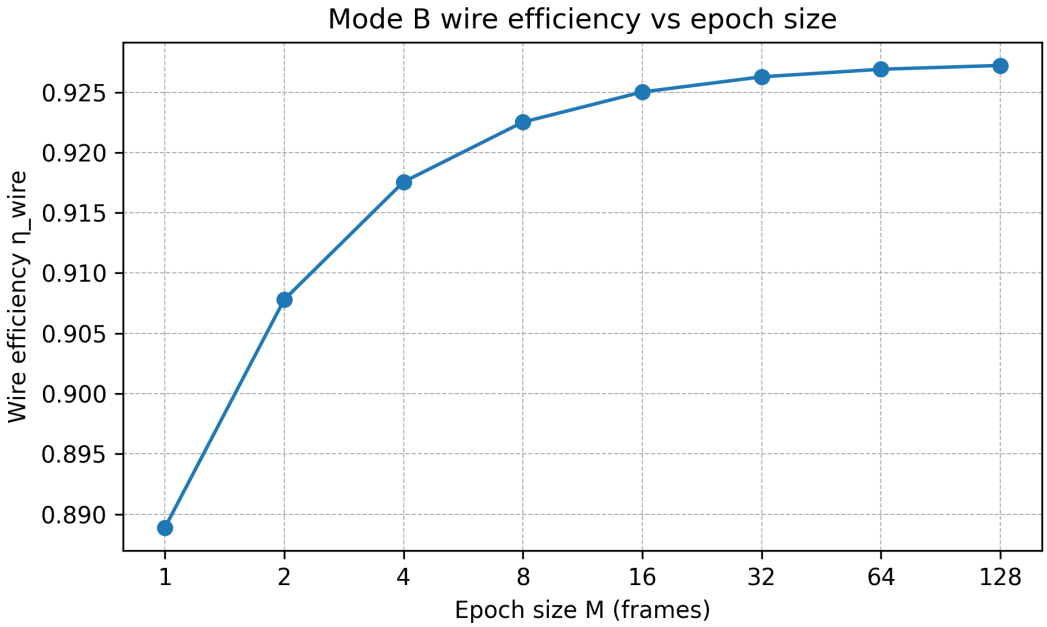


Figure 4: Mode B wire efficiency vs epoch size M (analytic). Larger epochs amortize the tag more effectively but increase buffering and commit granularity.

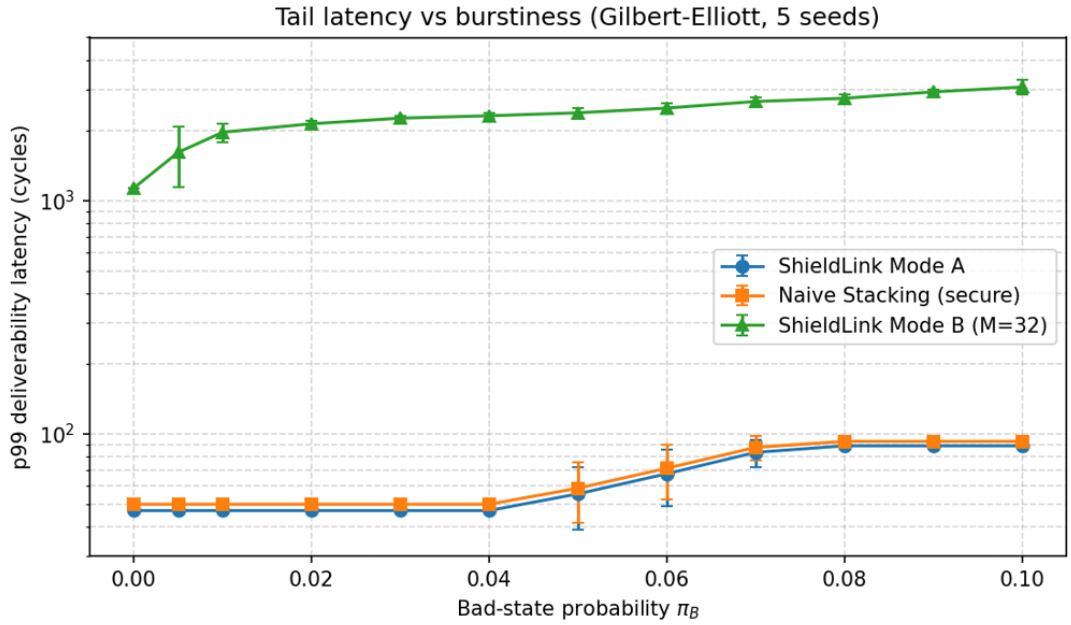


Figure 5: p99 deliverability latency vs bad-state probability π_B, updated with five random seeds. Mode B shows an abrupt tail increase due to epoch retransmissions.

4.3 Sensitivity and limitations

Appendix B reports a β sweep showing that the Mode B crossover depends on burst dynamics. Under a fixed, stationary bad-state probability π_B, larger β (shorter but more frequent bursts) tends to shift the crossover to smaller π_B because more bursts touch distinct epochs. Our simulator is intentionally minimal; it does not model higher-layer traffic patterns, adaptive PHY equalization, or selective repeat ARQ. Future work should include trace-driven evaluation.

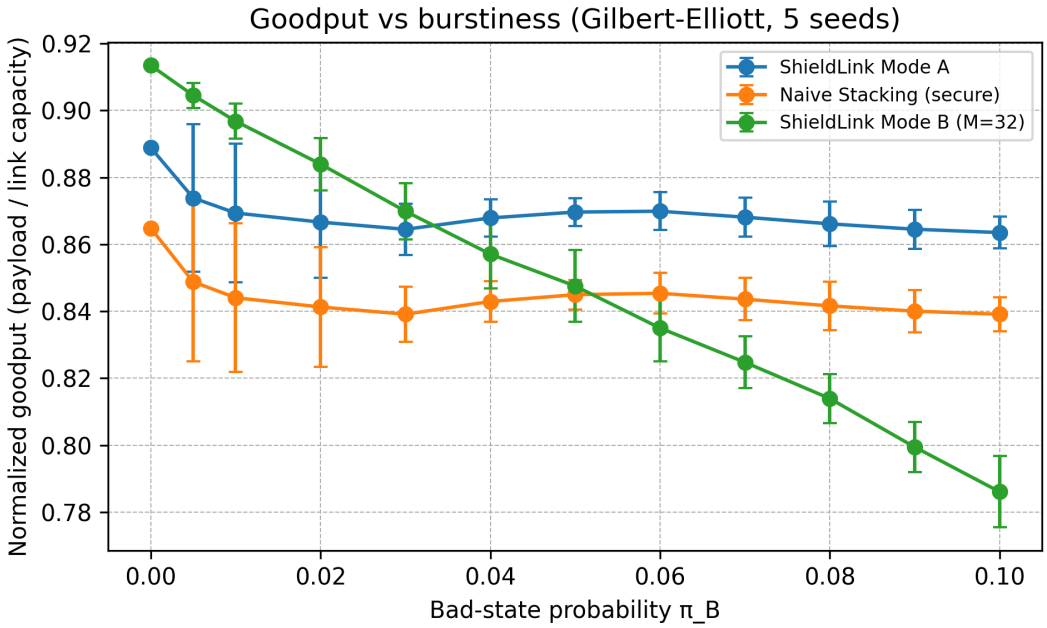


Figure 6: Normalized goodput vs bad-state probability π_B, updated with five random seeds. Mode B crosses below Mode A near $\pi_B \approx 0.04$ for this stress regime.

Table 3: Representative discrete-event simulation points ($\beta = 0.2$, $p_B = 0.1$), reported as mean ±95% CI across five random seeds.

π_B	Scheme	Goodput (mean ±95% CI)	p99 commit latency (cycles; mean ±95% CI)
0.01	Mode A	0.869 ± 0.021	47 ± 0
0.01	Naive stacking (secure)	0.844 ± 0.022	50 ± 0
0.01	Mode B ($M = 32$)	0.897 ± 0.005	1962 ± 182
0.05	Mode A	0.870 ± 0.004	55 ± 16
0.05	Naive stacking (secure)	0.845 ± 0.004	59 ± 17
0.05	Mode B ($M = 32$)	0.848 ± 0.011	2376 ± 112

4.4 Sender buffer pressure from ACK gating

A reviewer might ask whether delaying ACK until after AEAD verification increases sender replay-buffer pressure or requires a larger ARQ window to keep the link saturated. This effect is real but can be quantified with a bandwidth-delay product (BDP) argument: for a link that transmits one SLF in t_{tx} cycles and returns an ACK after RTT cycles, the minimum window to avoid transmitter idle time is $N_{win} \geq \lceil RTT/t_{tx} \rceil$. In Mode A, RTT includes serialization, two-way propagation, CRC check, AEAD verification, and a slight control delay. With our nominal parameters (256-byte payload, 8 B/cycle serialization, $d_{prop} = 2$ cycles, $d_{crc} = 1$ cycle, $d_{AEAD} = 8$ cycles), the required window remains small (Table 4).

This simple calculation explains why Mode A's ACK gating does not automatically imply large replay buffers: the AEAD verification delay is slight compared to serialization time at typical flit sizes. Mode B differs in construction—buffering is the price of epoch amortization—so it is best suited to traffic classes that can tolerate bulk commit latency.

Table 4: ACK-gating and minimum ARQ window size needed to sustain continuous transmission (nominal parameters).

Scheme	t_{tx} (cycles/SLF)	RTT to free first buffer (cycles)	$N_{win,min} \approx \lceil RTT/t_{tx} \rceil$	Notes
Mode A	36	50	2	ACK after per-SLF AEAD; window ≈2 frames is sufficient.
Naive stacking (secure)	37	52	2	ACK after per-SLF AEAD through an IP boundary; slightly larger wire time due to redundant header/metadata.
Mode B ($M = 32$)	36	1166	33	ACK amortized over an epoch; requires buffering $\gtrsim M$ frames to avoid mid-epoch stalls.

4.5 RTL/FPGA resource sizing (preliminary)

Table 5 reports the exact buffer sizes implied by the reference parameters used throughout this manuscript (payload $P = 256$ B, header=16 B, CRC=4 B, tag=12 B, $N_{win} = 64$, Mode B epoch $M = 32$). For FPGA intuition, we also map these memory bits to common block RAM granularities (18 KB and 36 KB). Table 6 complements this by summarizing control-plane FF/LUT structural estimates for the prototype RTL. These are resource-sizing estimates only, not vendor synthesis, post-place-and-route timing, power, or f_{max} results.

Table 5: RTL/FPGA buffer sizing and BRAM mapping (reference parameters).

Buffer	Total (KiB)	Total (bits)	18Kb BRAM blocks	36Kb BRAM blocks
Mode A TX retry buffer (N_{win})	18.000	147456	8	4
Mode B RX epoch buffer (M)	8.625	70656	4	2
Incremental storage for tags (Mode A vs. no-tag)	0.750	6144	1	1

Table 6: RTL control-plane resource estimate (logic only; excluding cipher core and buffers).

Block (prototype RTL)	FF (bits)	LUT-eq (rough)	Notes
Baseline CRC+ARQ control (unsafe early-ACK if paired with later AEAD)	195	≈200	ACK/advance on CRC+Seq only
ShieldLink Mode A control (D-Inv gated)	196	≈200	Δ ≈+1 FF, +1 AND vs baseline
ShieldLink Mode B control (epoch bookkeeping)	299	≈250–300	Adder+compare+bitmap reduction; flush-all baseline

LUT-eq counts are conservative RTL-structure estimates, not vendor-synthesis or post-place-and-route measurements. They are dominated by 64-bit add/compare logic that already exists in reliability-only ARQ; the incremental D-Inv gating is constant-depth and should not materially affect the control-plane critical path. Full timing closure and power validation remain future work for an integrated adapter/PHY implementation.

Figure 7 depicts the buffering cost for Modes A and B under the same reference parameters.

As a conservative logic-depth estimate, a 64-bit equality comparator (Seq==NextExpected)

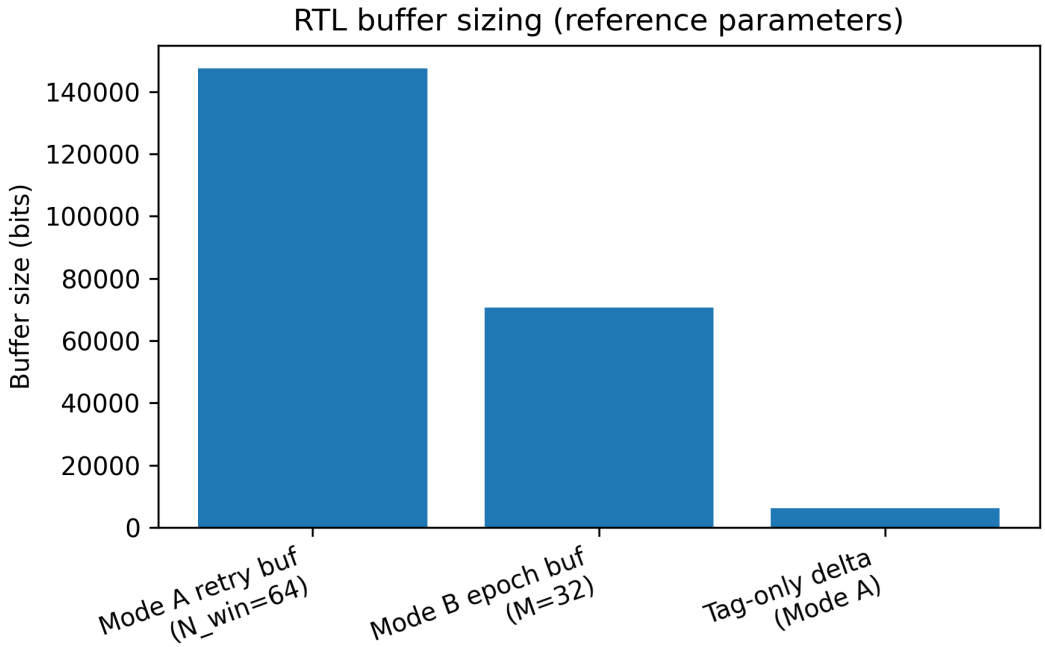


Figure 7: RTL buffer sizing for Mode A TX retry buffering and Mode B RX epoch buffering (reference parameters).

can be decomposed into 64 XNORs plus an AND-reduction tree (63 ANDs), i.e., ≈ 319 two-input gate-equivalents (assuming XNOR$\approx$ 4 2GE). The incremental D-Inv gating adds ≈ 2 2GE (gating ACK/state advance on aead_ok), which is $< 1\%$ of the comparator's cost. This supports the claim that D-Inv logic is negligible relative to the cipher core and buffering.

5 Discussion

ShieldLink's central design choice is to treat authentication as part of reliability: ACK is a promise that the corresponding payload is safe to deliver. This shifts security failures into an explicit state (a security drop) rather than allowing them to masquerade as progress in reliability. In deployment, a security drop would typically trigger a link reset and a key refresh, as many link encryption engines respond to integrity failures.

Mode selection can be adaptive: Mode B is appropriate for bulk transfers over benign channels, whereas Mode A is preferable when bursts are common or when latency is critical. Future work includes selective retries within epochs (bitmap NAK), dynamic epoch sizing, and combining ShieldLink with lightweight FEC to reduce CRC failures before ARQ is engaged.

5.1 Mitigation costs: AEAD delay, buffering, and timing closure

A recurring concern is whether enforcing the Deliverability Invariant (D-Inv) meaningfully complicates timing closure. ShieldLink's datapath does not place CRC and AEAD on the same critical combinational arc; CRC checking and AEAD verification are naturally pipelined and are already treated as multi-cycle operations in many link designs. The primary architectural cost of D-Inv is not additional cryptographic logic but the requirement to retain replay-buffer state until the AEAD result is known. For per-SLF authentication (Mode A and the secure naive baseline), Table 4 shows that the additional holding time yields only a slight increase in the bandwidth-delay product under our nominal parameters; consequently, a modest ARQ window

continues to saturate the link. The Mode B case is qualitatively different because it intentionally defers commitment until epoch authentication completes; the required buffering is therefore proportional to the epoch size M and should be considered an explicit design knob rather than a hidden overhead.

In addition to motivating the issue with a concrete race, the work offers a link-layer invariant that is easy to audit: an implementation is correct if and only if the receiver emits an ACK exactly when an SLF is deliverable (CRC_ok $\wedge$ AEAD_ok $\wedge$ sequence_valid). The two operating modes indicate that this invariant does not constrain a single point in the latency–efficiency design space: Mode A prioritizes tail latency for coherence-like traffic, whereas Mode B amortizes per-frame overhead for bulk transfers but exhibits a crossover region under burst errors. Future work should (i) implement selective retry within Mode B epochs, (ii) prototype the integrated adapter on FPGA/RTL to quantify f_{max}, area, and power under realistic PHY timing, and (iii) extend the evaluation to trace-driven CXL/UCIe traffic mixes and multi-hop topologies. These steps would turn the present design into a deployable template for secure, reliable chiplet interconnects.

Implementation note: Table 5 and Figure 7 show that buffering dominates the incremental cost, not the D-Inv gate. For the reference parameters, Mode B's additional epoch buffer is ≈ 8.6 KiB per protected stream, while Mode A's tag storage adds only 0.75 KiB over a no-tag retry buffer at $N_{win} = 64$. Because D-Inv consumes registered CRC/AEAD verdicts, it can be placed off the datapath critical path; this is an architectural expectation, not a measured f_{max} claim.

5.2 Is "naive stacking" a strawman?

It is fair to ask whether "naive stacking" actually occurs in practice. We do not claim that any particular commercial CXL/PCIe controller releases replay buffers on a CRC pass before IDE verification; public documentation rarely specifies such microarchitectural details. Instead, the baseline represents a class of unsafe integration contracts: reliability and IDE are implemented as separable blocks with an interface boundary that decouples "frame accepted by the link" from "frame authenticated." In this sense, ShieldLink is both a concrete design and an audit checklist. If an implementation cannot demonstrate that ACK implies CRC_ok $\wedge$ AEAD_ok $\wedge$ sequence_valid, then it is vulnerable to the semantic mismatch regardless of the underlying cipher. [8], [13]

5.3 Security drops, link reset, and higher-layer recovery

Silent drops on AEAD failure are controversial because they trade integrity-oracle resistance for availability: an active attacker can force timeouts and retries. ShieldLink adopts a conservative stance: an explicit "authentication failed" NAK would create a decryption oracle, whereas a silent drop preserves D-Inv. To make the resulting availability behavior operationally acceptable, the receiver must treat repeated authentication failures as a detectable anomaly and escalate. One practical policy is a link-reset threshold: after N_{auth} consecutive AEAD failures (or after an AEAD_fail-to-CRC_fail ratio exceeds a threshold), the receiver tears down the link, rekeys, and reports an error via sideband telemetry. At the system level, higher layers can then apply standard recovery actions—reroute to a redundant path, fail over to a spare chiplet, or mark the traffic class as degraded—analogous in spirit to the "poison/viral" reliability signaling used in CXL ecosystems. [2]

5.4 Mode B rigidity and a selective-retry path

The evaluation intentionally models Mode B with an all-or-nothing epoch-containment policy: if any SLF within an epoch fails CRC or AEAD, the receiver discards the buffered epoch, and the sender retransmits it. This flush policy is not presented as the only engineering choice;

rather, it is a clean baseline that exposes the head-of-line (HoL) mechanism behind Mode B's "goodput cliff." A practical optimization is selective retry within an epoch, for example, by attaching a compact bitmap to a NAK that identifies which SLFs in the epoch failed CRC checks, while maintaining a single epoch-level authentication tag. Such a scheme would preserve most of Mode B's header amortization while reducing retransmission volume during bursts, at the cost of additional receiver bookkeeping and sender-scheduling complexity. We treat this as low-hanging-fruit work because it can be layered onto the same D-Inv foundation: ACK remains gated on deliverability, but the retransmission granularity becomes finer.

5.5 Beyond single links: calibration and multi-hop extensions

Our simulation models a single D2D link; multi-link fabrics and multi-hop chiplet topologies introduce additional failure modes. First, burst parameters (α, β, and p_B) should be calibrated to the physical environment. Rather than guessing, a deployed system can estimate these parameters from CRC/NAK counters and recovery events observed during runtime, and then select Mode A or Mode B (or adjust M) to remain on the favorable side of the crossover region. Second, active fault-injection techniques, such as EMFI, can produce spatially and temporally correlated errors across adjacent links, thereby increasing the value of link-local telemetry when multiple links share an interposer or I/O die. Finally, extending D-Inv across multi-hop paths suggests a compositional view: each hop enforces its own deliverability invariant, while higher layers maintain end-to-end ordering and key epochs. Trace-driven CXL/UCIe traffic evaluation is an important next step for generalizability. [9]–[12]

6 Conclusion

Naively composing link-layer ARQ with delayed AEAD verification can create a validity-before-verification race: the reliability layer can commit data before cryptographic integrity is established. ShieldLink enforces D-Inv by gating acknowledgments and buffer retirement on AEAD verification (Mode A) or epoch verification (Mode B), cleanly separating recoverable link noise from non-recoverable security violations.

More broadly, ShieldLink makes an implicit contract explicit: on retrying links, delivery semantics and security semantics must align. Our analytical and simulation results quantify the wire overhead/latency trade-off and guide decisions about whether per-frame authentication (Mode A) or amortized authentication (Mode B) is preferable. Future work includes end-to-end FPGA validation for timing closure and power, and the exploration of tighter integration with standard IDE-class crypto backends.

References

[1] UCIe Consortium, "UCIe Consortium Introduces 3.0 Specification With 64 GT/s Performance and Enhanced Manageability," *Business Wire*, Aug. 5, 2025. [Online]. Available: `https://www.businesswire.com/news/home/20250805909613/en/UCIe-Consortium-Introduces-3.0-Specification-With-64-GTs-Performance-and-Enhanced-Manageability`. Accessed: Dec. 14, 2025.

[2] CXL Consortium, "CXL Consortium Releases the Compute Express Link 4.0 Specification Increasing Speed and Bandwidth," Nov. 18, 2025. [Online]. Available: `https://computeexpresslink.org/wp-content/uploads/2025/11/CXL_4.0-Specification-Release_FINAL_Website-Copy.pdf`. Accessed: Dec. 14, 2025.

[3] A. Deric, K. Mitard, S. Tajik, and D. Holcomb, "Evaluating Vulnerability of Chiplet-Based Systems to Contactless Probing Techniques," arXiv:2405.14821, 2024.

doi:10.48550/arXiv.2405.14821.

[4] J. Talukdar, A. Chaudhuri, J. Kim, S. K. Lim, and K. Chakrabarty, "Securing Heterogeneous 2.5D ICs Against IP Theft Through Dynamic Interposer Obfuscation," in Proc. DATE, 2023. doi:10.23919/DATE56975.2023.10137145.

[5] M. Nabeel, M. Ashraf, S. Patnaik, V. Soteriou, O. Sinanoglu, and J. Knechtel, "2.5D Root of Trust: Secure System-Level Integration of Untrusted Chiplets," *IEEE Trans. Comput.*, vol. 69, no. 11, pp. 1611–1625, Nov. 2020, doi: 10.1109/TC.2020.3020777.

[6] M. Dworkin, "Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC," NIST SP 800-38D, Nov. 2007. [Online]. Available: `https://csrc.nist.gov/pubs/sp/800/38/d/final`. Accessed: Dec. 14, 2025.

[7] M. S. Turan et al., "Ascon-Based Lightweight Cryptography Standards for Constrained Devices: Authenticated Encryption, Hash, and Extendable Output Functions," NIST SP 800-232, Aug. 2025. doi:10.6028/NIST.SP.800-232. [Online]. Available: `https://csrc.nist.gov/pubs/sp/800/232/final`. Accessed: Dec. 14, 2025.

[8] MITRE, "CWE-367: Time-of-check Time-of-use (TOCTOU) Race Condition," 2024. [Online]. Available: `https://cwe.mitre.org/data/definitions/367.html`. Accessed: Dec. 14, 2025.

[9] E. N. Gilbert, "Capacity of a Burst-Noise Channel," *Bell Syst. Tech. J.*, vol. 39, no. 5, pp. 1253–1265, 1960.

[10] E. O. Elliott, "Estimates of Error Rates for Codes on Burst-Noise Channels," *Bell Syst. Tech. J.*, vol. 42, no. 5, pp. 1977–1997, 1963.

[11] S. Khalaj Monfared, M. Saadat Safa, and S. Tajik, "ChipletQuake: On-Die Digital Impedance Sensing for Chiplet and Interposer Verification," *Sensors*, vol. 25, no. 15, Art. no. 4861, Aug. 2025, doi: 10.3390/s25154861.

[12] N. Mishra, A. Chakraborty, and D. Mukhopadhyay, "Faults in Our Bus: Novel Bus Fault Attack to Break ARM TrustZone," in Proc. Network and Distributed System Security Symposium (NDSS), 2024, doi: 10.14722/ndss.2024.24499.

[13] S. Hristozov, M. Wettermann, and M. Huber, "A TOCTOU Attack on DICE Attestation," in Proc. ACM Conference on Data and Application Security and Privacy (CODASPY), 2022, doi: 10.1145/3508398.3511507.

[14] L. Lamport, *Specifying Systems: The TLA+ Language and Tools for Hardware and Software Engineers*. Addison-Wesley, 2002.

Appendix

A Frame format and nonce construction

Mode A SLF = [Header (16 B) | Payload (256 B) | CRC (4 B) | Tag (12 B)]. We use a 96-bit truncated tag to reduce bandwidth overhead; the same framing also supports a full 128-bit tag. Mode B SLF = [Header | Payload | CRC] for frames 0 to $M-2$, with a 12 B epoch tag appended to frame $M-1$. Nonces are derived from (link_id, direction, sequence_number). Retransmissions send identical encrypted bits without re-encryption, so the nonce remains unchanged across retries.

B Additional plots and sensitivity

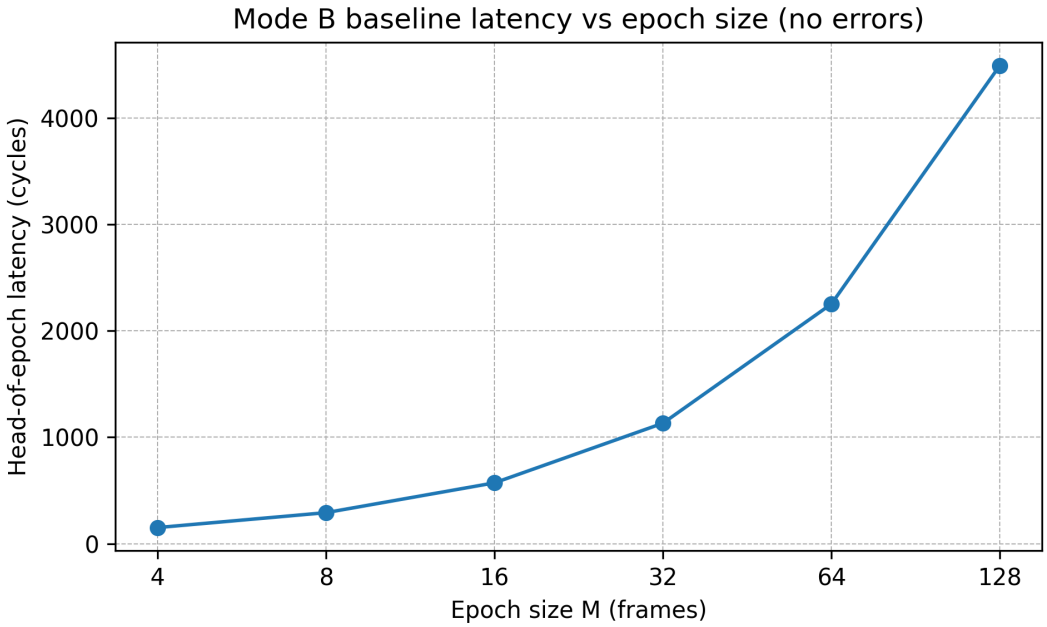


Figure 8: Mode B baseline head-of-epoch latency grows roughly linearly with M (no errors).

Table 7: Goodput crossover points where Mode B becomes worse than Mode A (targeted five-seed coarse sweep).

β	Mean bad burst length (frames)	Crossover π_B
0.05	20.0	0.055
0.10	10.0	0.050
0.20	5.0	0.040
0.50	2.0	0.025

C Reproducibility

Supplementary materials accompanying this revised submission include (i) `generate_assets.py`, which regenerates the Gilbert–Elliott simulation sweeps, Figures 4–6, Appendix B plots, and the coarse Mode B crossover table; (ii) the revised five-seed simulation CSV files; (iii) `resource_estimator.py` and RTL sizing data; (iv) `formal_sanity_check.py` for bounded safety exploration; and (v) the prototype SystemVerilog RTL skeleton of the ShieldLink control plane. The scripts are deterministic given their random seeds.

Supplementary RTL: The accompanying RTL package provides a synthesizable SystemVerilog skeleton for the sender/receiver controllers, the retry buffer, and baseline CRC+ARQ logic, along with the Mode A ACK-gating modifications that enforce D-Inv. The package is intended for independent reproduction and extension, not as a final PPA/timing sign-off artifact.

D Bounded protocol sanity check

This appendix summarizes the bounded explicit-state exploration described in Section 3.5. The goal is to sanity-check safety (no ACK without deliverability), not to prove liveness, timing

closure, or resistance to denial-of-service under all schedules.

D.1 Safety properties checked

We check two core properties over all explored interleavings:

- P1 (ACK gating): The receiver emits ACK only if CRC_ok ∧ AEAD_ok ∧ Seq_ok.
- P2 (monotonic delivery): The receiver advances next_expected only when it delivers a frame.

Here, Seq_ok means either (i) the frame's sequence equals next_expected (new in-order delivery) or (ii) the frame is a duplicate of an already-delivered sequence number (duplicate ACK without re-delivery).

D.2 Explorer model (high-level)

The explorer models a single stream with a go-back-N sender, a FIFO channel, and a receiver that implements ShieldLink's CRC/AEAD decision structure. At each step, it nondeterministically chooses among the following: the sender transmits, the receiver consumes a data frame, and the sender consumes an ACK/NAK. CRC_ok and AEAD_ok are treated as nondeterministic outcomes per transmission, capturing both noise and active manipulation. The included script reports 845,625 reachable states for the stated bounded configuration and no safety counterexample. The reference implementation is provided as `code/formal_sanity_check.py` in the supplementary bundle.

Funding Statement

No funding received.

Conflicts of Interest

The author declares that there are no conflicts of interest.

Ethical Considerations

The article does not contain any studies with human participants or animals performed by the author.

AI Usage Statement

AI tools were used in the following way(s): Grammarly for language editing; Figures: Anthropic Claude Opus 4.7; OpenAI Codex 5.5 Medium for GitHub README files and push & commit.

International Journal of Research in Computing

ORIGINAL RESEARCH
Volume 5, Issue 2 (2026)

Differentiated Thyroid Cancer Recurrence Classification Using Machine Learning Models and Bayesian Neural Networks with Varying Priors: A SHAP-Based Interpretation of the Best Performing Model

HMNS Kumari[1], HMLS Kumari[2], UMMPK Nawarathne[3]

[1]Department of Computer Science,Faculty of Science, University of Helsinki, Finland
[2]Computing Centre, Faculty of Engineering, University of Peradeniya, Sri Lanka
[3]Faculty of Computing, Sri Lanka Institute of Information Technology, Sri Lanka

[1] nadeeshashyamikumari@gmail.com | https://orcid.org/0000-0002-4708-5561

Abstract

Differentiated thyroid cancer (DTC) recurrence is a major public health concern, requiring classification and predictive models that are not only accurate but also interpretable and uncertainty-aware. This study introduces a comprehensive framework for DTC recurrence classification using a dataset containing 383 patients and 16 clinical and pathological variables. Initially, 11 machine learning (ML) models were employed using the complete dataset, where the Support Vector Machines (SVM) model achieved the highest accuracy of 0.9481. To reduce complexity and redundancy, feature selection was carried out using the Boruta algorithm, and the same ML models were applied to the reduced dataset, where it was observed that the Logistic Regression (LR) model obtained the maximum accuracy of 0.9611. However, these ML models often lack uncertainty quantification, which is critical in clinical decision making. Therefore, to address this limitation, the Bayesian Neural Networks (BNN) with six varying prior distributions, including Normal (0,1), Normal (0,10), Laplace (0,1), Cauchy (0,1), Cauchy (0,2.5), and Horseshoe (1), were implemented on both the complete and reduced datasets. The BNN model with Normal (0,10) prior distribution exhibited maximum accuracies of 0.9740 and 0.9870 before and after feature selection, respectively. As the BNN model with N(0,10) prior distribution employed after feature selection outperformed all the other models, it was chosen as the best-performing model for DTC recurrence classification. This model was further analyzed using epistemic and aleatoric uncertainty, reflecting the model's confidence in its prediction. In addition, to enhance this model's interpretability, SHapley Additive exPlanations (SHAP) values were calculated, providing valuable insights into the contribution of key variables to the model's output.

KEYWORDS
Bayesian Neural Network, Classification, Differentiated thyroid cancer recurrence, Machine learning, SHAP, Uncertainty quantification

ARTICLE HISTORY
In-Press:16 Nov 2025
Published:16 July 2026

DATA/CODE AVAILABILITY
Data are publicly available at Kaggle. Code is available from the corresponding author upon reasonable request.

SDG ALIGNMENT
SDG 3

1 Introduction

Thyroid cancer is a type of cancer that begins in the tissues of the thyroid gland, a small organ in the front, lower neck region that is essential for controlling metabolism through the hormone synthesis process [1,2]. Initially, when thyroid cancer is first developing, it may show no or few symptoms [3]. However, when symptoms appear, a clear lump or swelling in the neck, changes in voice, difficulty in swallowing, neck pain, or a recurrent cough that isn't related to a cold can be observed [4, 5]. Owing to its mild presentation, the disease may not be identified until it becomes severe or may be discovered by chance while imaging for another medical condition.

Thyroid cancer is the most common and widespread, making up over 95% of all endocrine-related cancers [4]. The prevalence of thyroid cancer has been rising significantly over the last few decades, partly due to improved imaging and diagnostic techniques. Although thyroid cancer is a rare cancer type compared to other cancers [6], it is still a significant public health issue, due to its increasing discovery rates and the fact that it frequently requires long-term medical care. However, nearly 90% of thyroid cancers are identified as differentiated thyroid cancers [7], making it the most common type of thyroid cancer, which comprises two histological subtypes: papillary thyroid carcinoma (PTC) and follicular thyroid carcinoma (FTC) [8].

DTC patients often record a high survival rate, which is more than 85% [9], indicating favourable results. Although the survival rate is high, its recurrence is still a major clinical issue, which occurs in nearly 10-30% of DTC patients [10]. This leads to major challenges in patient care, including lifetime surveillance, multiple surgeries, radioactive iodine therapy, etc. Therefore, accurate classification and prediction of DTC recurrence are essential to improve and enhance overall patient care.

Conventional prognostic tools, especially those introduced by the American Thyroid Association (ATA), facilitate a systematic way to assess the probability of recurrence of DTC. However, as data complexity and dimensionality increase, more flexible analytical methods are needed to classify DTC recurrence more accurately. As a result, machine learning models have gained attention due to their ability to correctly classify the DTC recurrence, particularly using high-dimensional clinical and pathological data, thereby identifying complex relationships.

Therefore, initially, this study employs several ML models using the complete set of variables in order to classify DTC recurrence. To reduce the potential redundancy as well as the noise arising from all the variables, feature selection is carried out, and then the same ML models are applied to the reduced dataset to evaluate the improvement in the models' performance. With the use of this two-step process, this study aims to assess how feature selection affects model behaviour systematically. Although these ML models perform well, they provide point estimates rather than giving the confidence or uncertainty of the model's output. To address this challenge, a BNN is employed in this study, as it is capable of handling nonlinear and complex relationships while quantifying uncertainties by placing probabilistic weight distributions on model parameters. In addition, this study aims to examine how different prior distributions affect this model's performance and uncertainty estimations both before and after feature selection. Following that, the best-performing model was subjected to SHAP analysis in order to interpret feature contributions to the model's prediction, thereby reducing the black-box nature of the classification models. This post hoc interpretability step supports clinical transparency and increased confidence in AI-assisted categorisation by offering insightful information about model decision-making. Therefore, this study tries to provide a comprehensive framework to classify differentiated thyroid cancer recurrence by integrating feature selection, machine learning models, uncertainty-aware Bayesian modelling, and SHAP analysis together. Thereby, this study contributes to the advancements in the field of differentiated thyroid cancer data analysis by integrating robust, accurate, and interpretable classification methods.

The remainder of this paper is structured as follows: Section II provides the existing literature related to different classification techniques related to DTC recurrence. The data and

materials used in this study and the methods carried out are outlined in Section III, while Section IV thoroughly discusses the results obtained in this study. Finally, Section IV concludes this paper.

2 Literature review

In recent years, machine learning techniques have been widely used in DTC recurrence classification utilising clinical and pathological data. These ML methods are capable of capturing nonlinear and complex relationships, which act as an alternative method to standard prognostic systems. Multiple studies [11,12,13, 14,15,16,17,18] have thoroughly examined the performance of classical machine learning models such as LR, k-Nearest Neighbours (KNN), SVM, Naïve Bayes (NB), and Stochastic Gradient Descent (SGD), revealing their higher accuracies. In addition, a study [12] further evaluated the impact of various kernel functions, including linear, polynomial, and radial basis function (RBF), on SVMs and observed that the SVM with the RBF function achieved an accuracy of 94%, while balancing recall and F1-score effectively. Furthermore, the studies mentioned above and also the research project conducted by Şeyma Yaşar [19] explored tree-based models, including Random Forest (RF), Decision Trees (DT), and Extra Tree Classifier (ETC), and identified that these models consistently showed outstanding results in DTC recurrence classification. Notably, the RF model exhibited strong performance along with exceptional sensitivity and specificity scores. However, despite the higher accuracies, these models are easily prone to overfitting, especially with small and imbalanced datasets, highlighting their limitations in applicability. Furthermore, these models need careful hyperparameter tuning to ensure their generalizability.

In addition, previous studies have also applied more sophisticated ensemble methods, such as Adaptive Boosting (Adaboost), Gradient Boosting (GB), Extreme Gradient Boosting (XGB), Light Gradient Boosting (LGB), and Catboost [11,12,13,14,15,16,19,20] models. These models obtained accuracies exceeding 85% by combining weak learners, forming strong classifiers, and improving the overall performance. Few studies [14,18] have employed stacking ensemble models, demonstrating their superior performance, achieving higher accuracy. However, these ensemble methods are often computationally extensive and challenging to train, which indicates their implementation complexity. To improve the predictive performance further, the authors have implemented several hybrid models by integrating different machine learning techniques. Hybrid models such as RF-LR, SGD-LR, and SVM-LR, which combined multiple machine learning models, acquired nearly 97% accuracy, showing better performance [14].

Furthermore, to capture more intricate patterns, several authors [14,16,17,21] have explored some deep learning models, including Artificial Neural Networks (ANN) and feedforward Neural Networks (FNN), for DTC recurrence classification. These studies reported that deep learning models were capable of achieving accuracies ranging from 96% to 98% highlighting their strength in modelling complex and nonlinear relationships. However, these models often require a large amount of data to train the model effectively, making them less suitable for small datasets.

Deviating from traditional machine learning and deep learning models, a study [14] explored the possibility of applying clustering techniques for DTC recurrence classification. The authors employed unsupervised clustering techniques such as k-means, hierarchical clustering, and Louvain clustering to uncover underlying patient subgroups. Notably, k-means and hierarchical clustering identified two clusters achieving higher silhouette scores and demonstrating that the dataset could be divided clinically into two major subgroups. However, these clustering techniques are highly sensitive to distance metrics and hyperparameters, which may lead to meaningless clusters if not carefully tuned or validated with expert opinions.

To overcome the black-box nature of these machine learning and deep learning models, a few studies have examined different post hoc explanation approaches, while some authors have employed significant explainable models to classify DTC recurrence data. Of these model expla-

nation techniques, SHAP and Local Interpretable Model-agnostic Explanations (LIME) have been applied in a few studies to interpret how the features impact several ML models' outputs [13,20,21]. The authors highlighted the importance of SHAP and LIME values in identifying significant features as they provide notable decision-making information at both global and local levels. In contrast, A. K. Arslan and C. Çolak [22] implemented two explainable models, including Explainable Boosting Machines (EBM) and Fast Interpretable Greedy-Tree Sums (FIGS), to classify recurrence data. EBM exhibited a test accuracy of 96.10% while identifying key predictors. However, these model explanation techniques, as well as explainable models, should be handled carefully, as incorrect interpretations could lead to misleading information. Furthermore, few studies [11,12,14,16,17,19,20,22] have emphasised the role of feature selection in producing reliable and straightforward models for classification. Most common feature selection techniques, such as recursive feature elimination, feature importance analysis using RF and DT, Chi-square test, distance-based correlation, filter-based feature selection, and correlation heatmap, have been used to reduce the dimensionality and enhance the model stability by identifying the most significant features. However, there is a significant risk of discarding relevant features while implementing these techniques if the thresholds are not calibrated well.

Although ML, deep learning, and explainable models exhibit strong predictive performance, they often face challenges with uncertainty quantification. This limits the confidence not only in decision-making but also in model predictions, which is crucial for clinical applications. Furthermore, traditional ML models often produce a point estimate without specifying the degree of certainty in their output. This could lead to serious misclassifications, which could result in either excessive or insufficient treatment, making it dangerous for patient care. Recent works [23, 24] have further emphasized the importance of uncertainty quantification in healthcare ML, underscoring its role in building reliable and trustworthy clinical AI systems. However, to overcome these challenges, Bayesian Neural Networks have emerged as a more flexible alternative approach, which facilitates the incorporation of prior distributions while quantifying uncertainties by producing probabilistic distributions. Rather than using fixed weights, BNNs learn distributions over weights while acquiring the strengths of the neural networks. To date, no study has applied Bayesian Neural Networks with varying priors to classify DTC recurrence, leaving a clear methodological gap that this study addresses. Although some studies have already compared ML models, no studies have compared them with BNN in terms of accuracy, interpretability, and uncertainty. Considering these gaps, this study provides a comprehensive comparison of ML models with BNN while varying prior distributions, offering a systematic framework for DTC recurrence classification. To further enhance the understanding of the model's predictions, this study provides a detailed explanation of the BNN's output using SHAP values, which have not yet been discussed in any previous literature. By filling these gaps, this study offers a clinically usable model that facilitates confidence in data-driven decision-making in the management of DTC recurrence by combining both explainability and accuracy.

3 Methodology

A. Data

This study used a dataset related to DTC, which was obtained from an online data repository [25]. The dataset consisted of 16 variables representing the clinical and pathological features of 383 patients in a retrospective cohort who were followed up for at least 10 years within a study span of 15 years. The features of this dataset are the age of the patient in years, gender, present smoking status, whether the patient previously smoked or not (Hx Smoking), whether the patient received radiotherapy to the head and neck region or not (Hx Radiotherapy), thyroid function categories, Goiter type, adenopathy location, pathological subtype of cancer, focality,

risk type, tumor stage (T), node status (N), metastasis status (M), cancer stage (stage), and response to initial treatment. The dependent variable of the dataset is whether the thyroid cancer recurred or not. However, as the dataset is publicly available, no formal ethical clearance was required for this study.

Initially, descriptive statistics were calculated in order to understand the characteristics of the dataset. Then, data preprocessing techniques such as standardisation, label encoding, and ordinal encoding were applied to the data. After that, this pre-processed data was divided into two sets: the training set (which contained 80% of the total dataset) and the testing set (which contained 20% of the total dataset). However, as a class imbalance was observed in this training set, the Synthetic Minority Oversampling Technique (SMOTE) was applied to achieve a more balanced distribution. Following that, firstly, 11 machine learning models were applied to the dataset containing all the variables in order to assess the baseline performance before conducting any feature selection.

B. Machine learning models

1) Support Vector Machines: SVM is one of the most commonly used machine learning models for classification as well as regression [26]. This is a supervised machine learning technique that can be used to classify both linear and nonlinear data by finding the optimal hyperplane that separates data into classes, using support vectors and margins [27].

2)) Random Forest: RF is an ensemble model that fits multiple decision trees to arbitrary subsets of data, where each node in a decision tree applies a condition on a feature, splitting the data based on that condition [28]. Predictions from each decision tree are then combined through majority voting or averaging to increase the accuracy of the final prediction [29].

3) K-nearest Neighbours: Using a distance measure, KNN tries to find a test sample's k-nearest neighbours in the training set and classifies it according to the KNNs' majority vote. KNN's success is closely related to the neighbourhood that is searched for every sample, namely the hyperparameter k and the distance metric that is utilised [30].

4) Decision Tree: With a tree-like structure, DT tries to learn the basic decision rules derived from the features of the dataset and aims to build a model that predicts the output value. In order to draw conclusions about the output value, a tree is constructed by deciding to divide the source set into subsets. When all potential options and outcomes have been taken into account, the process comes to an end, producing a structure resembling a tree where the original decision is reached by following a logical chain of decisions [31].

5) Logistic Regression: The relationship between a group of independent factors and a categorical dependent variable can be studied using the LR model. Explanatory variables might be binary, continuous, discrete, or a combination of these, while the dependent variable is often dichotomous. To begin, it is necessary to prove that the collection of predictors and the outcome are related. Once a smaller group of predictor variables has been identified, the LR model can be applied to predict new outcomes [32].

6) Gradient Boosting: Similar to RF, the gradient boosting method makes predictions using the outputs of several decision trees, and then gradient trees use these decision trees as weak learners. In contrast to RF, the GB model builds one tree at a time, correcting each error of the preceding trees before it [33].

7) Extreme Gradient Boosting: As an alternative technique for predicting a response variable given specific covariates, XGB was suggested, and this uses a gradient boosting framework,

which is an ensemble method based on decision trees. The fundamental idea behind this approach is to construct classification or regression trees one at a time, then train the next model using the residuals from the prior tree. This technique can use values from previously trained trees in the training process to get superior results, while it can prevent overfitting by employing a pruning process [34].

8) Light Gradient Boosting: LGB is also based on a gradient boosting framework, which trains several weak learners in succession while attempting to fix others' errors. A collection of these weak learners that come together to become a strong learner makes up the final model by reducing the loss function, which is usually the total of the losses for every case in the dataset [35].

9) Catboost: Catboost is a machine learning model based on symmetric decision trees, and by preprocessing features and creating additional numerical features, this model facilitates the usage of several feature categories in contrast to XGB and LGB. As a result, feature normalisation is not required in the catboost method, and it increases the dimension and utilisation of features at the model level by accounting for the association between attributes [36].

10) Multi-Layer Perceptron: MLP is a neural network that is made up of several layers, such as an input layer, hidden layers, and an output layer. In addition to receiving input from every unit in the layer before it, each unit transmits its output to every other unit in the layer beneath. There is a certain weight assigned to each of these links, which shows how the unit affects the unit's response in the layer afterwards. The strength of these connections between the units and the input determines a multilayer perceptron's output [37].

11) Naïve Bayes: NB is a popular classifier that is based on Bayes' theorem, which makes the assumption that a feature's impact on a class is not influenced by other features. NB returns the conditional probability of a specific target after computing the prior probabilities and likelihood for a given class label [38].

However, to optimise these models' parameters and evaluate their performances, a grid search with 2-fold, 5-fold, and 10-fold cross-validation was performed for each machine learning model, and then the model that produced the highest accuracy across these folds was selected for final testing. In addition to accuracy, several other evaluation metrics, such as precision, recall, and F1-score, were calculated for each of these machine learning models. To calculate these evaluation metrics, a confusion matrix, which is depicted in Table 1, was used.

Table 1: Confusion Matrix

		Predicted Class	
		Negative	**Positive**
Actual Class	**Negative**	True Negative (TN)	False Positive (FP)
	Positive	False Negative (FN)	True Positive (TP)

Using the above confusion matrix in Table 1, accuracy, precision, recall, and F1-score were calculated using Equations 1,2,3, and 4, respectively:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

$$\text{Precision} = \frac{TP}{TP + FP} \tag{2}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{3}$$

$$\text{F1-score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4}$$

Following that, to reduce the model complexity and increase the model training efficiency, a feature selection was performed using the Boruta algorithm, thereby identifying the relevant features and effectively removing the redundant variables. The Boruta algorithm is a feature selection method and also an extension of the RF method, which contrasts the importance of original variables with that of their randomly generated shadow counterparts in order to identify the most relevant features [39].

After this feature selection process, the same machine learning models that are mentioned in Section B were applied to the reduced dataset containing only the selected variables. In addition, the same grid search and cross-validation procedure that was discussed earlier was performed with the reduced dataset to improve the models' performance. To assess these machine learning models, accuracy, precision, recall, and F1-score were calculated again.

However, it was observed that these machine learning models lack interpretability and fail to capture the uncertainty of the model's parameters. Therefore, to overcome these challenges, a simple Bayesian Neural Network was employed on both the full and reduced datasets, and its general overview is depicted in Figure 1.

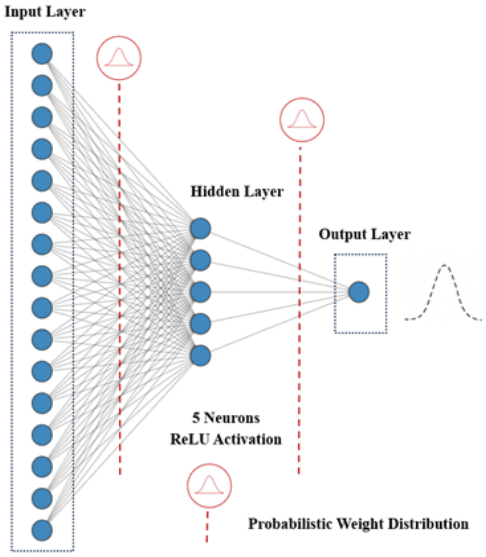


Figure 1: General overview of BNN

C. Bayesian Neural Network

BNN is a type of artificial neural network that incorporates Bayesian inference during the training phase [40]. In contrast to typical ANNs, which use deterministic optimisation to learn a single set of optimal weights, BNNs regard the model parameter vector "θ " as random variables and try to infer their posterior distributions from the observed data while quantifying the uncertainties [41].

i.e., the BNN's goal is to estimate the full posterior distribution $p(\theta \mid D_x, D_y)$, which is derived using the Bayes' theorem, when,

$$p(\theta \mid D_x, D_y) = \frac{p(D_y \mid D_x, \theta) \cdot p(\theta)}{\int_\Theta p(D_y \mid D_x, \theta') \cdot p(\theta')\, d\theta'} \tag{5}$$

Here, $D_x = \{x_1, x_2, \ldots, x_N\}$ is a matrix of variables and $D_y = \{y_1, y_2, \ldots, y_N\}$ is the corresponding output matrix when "N" indicates the number of observations. In addition, $p(D_y \mid D_x, \theta)$ and $p(\theta)$ represent the likelihood and prior distribution, respectively. Furthermore, the denominator is the marginal likelihood that integrates over the possible values of the parameter space and normalizes the posterior distribution. As this integral is difficult to compute in practice, and acts as a normalizing constant, the posterior distribution can be expressed as in Equation (6).

$$\text{posterior} \propto \text{likelihood} \times \text{prior} \tag{6}$$

However, in the binary classification context, the outputs $D_y = \{y_1, y_2, \ldots, y_N\}$, are binary labels. i.e. $y_i \in \{0, 1\}$. Then the BNN considers the likelihood as a product of a Bernoulli distribution, which is expressed in Equation (7).

$$p(D_y \mid D_x, \theta) = \prod_{i=1}^{N} \hat{p}_i^{y_i} (1 - \hat{p}_i)^{1-y_i} \tag{7}$$

In Equation (7), $\hat{p}_i$ indicates the output of the network for the i-th sample, representing the estimated probability that sample i is a member of class 1, where $\hat{p}_i = \text{NN}_\theta(x_i)$. The prior is placed over the network parameters θ. A common choice for the prior distribution is a diagonal Gaussian prior, which can be expressed as

$$p(\theta) = \mathcal{N}(0, \sigma^2 I) \tag{8}$$

where σ^2 is a scalar and "I" is the identity matrix [41].

Considering the Bayesian framework discussed above, this study employed a simple BNN for the data set with all the variables and the data set with the selected variables using the Boruta algorithm. The model was built using the Pyro probabilistic programming language, and the model architecture comprised an input layer of dimension d, which corresponds to the number of covariates, a hidden layer with five units and ReLU activation, and a final output layer producing a single logit, which is transformed to a Bernoulli likelihood for classification. Weights and biases were included in the network parameters θ, which were modelled as independent random variables, and several prior distributions were taken into consideration for the parameters in order to investigate how prior choice affects model performance and uncertainty quantification. These prior distributions included: $\text{N}(0, 1)$, $\text{N}(0, 10)$, $\text{Laplace}(0, 1)$, $\text{Cauchy}(0, 1)$, $\text{Cauchy}(0, 2.5)$, and $\text{Horseshoe}(1)$. As these prior distributions are composed of varying sparsity-inducing properties and varying tail behavior, they facilitated a comprehensive comparative analysis of model performance.

Due to the practical difficulty of assessing the posterior distribution in closed form, the posterior distributions of these BNNs were estimated using the Markov Chain Monte Carlo (MCMC) method. This technique simulates direct draws from target posterior distributions without requiring the analytical solution of numerous integrals [42]. For this purpose, Hamiltonian Monte Carlo (HMC), which is a gradient-based MCMC technique that uses Hamiltonian dynamics to suggest effective transitions in the parameter space, was employed. The No-U-Turn

Sampler (NUTS), an adaptive extension of HMC, was used to increase the sampling efficiency even further and eliminate the necessity of manually setting the expensive tuning runs [43].

Moreover, to evaluate each of these BNN models with varying prior distributions, the same evaluation metrics that were used to assess the machine learning models, including accuracy, precision, recall, and F1-score, were computed. Furthermore, epistemic uncertainty, which arises from limited data reflecting uncertainty in the model parameters, and aleatoric uncertainty, which captures the inherent noise in the data, were estimated for each BNN [44].

After carefully evaluating each machine learning and BNN model employed before and after feature selection, the model that produced the highest accuracy was selected as the best-performing model. Then, to assess how each feature of the dataset contributed to this model's predictions, SHAP values were calculated.

D. Explainable artificial intelligence with SHAP

When employing an ML model, it is crucial to identify the most significant and influential variables while understanding how they contribute to model predictions. However, this is limited in traditional ML models, which do not show how the features affect the model's predictions; instead, they provide how they produce the model's output [45]. To reduce the black box nature of these ML models, SHAP has been introduced by integrating the concepts of game theory, forming part of the explainable artificial intelligence (XAI), thereby providing globally accurate and consistent explanations of feature importance and how they influence the model's predictions [46,47]. SHAP analysis allocates a specific contribution to each feature by treating each variable as a contributor in order to quantify the influence of the features on the predictions [48]. This not only quantifies the overall effect but also computes both the negative and positive influences of features on the output of the model [49]. By employing an Explanatory Model (EM), SHAP assesses each variable's contribution to the EM as in Equation 9, where the input data is represented as $X_i = \{x_1, x_2, \ldots, x_n\}^t$ [50]:

$$\mathrm{EM} = \phi_0 + \sum_{i=1}^{n} \phi_i t_i \tag{9}$$

where

$$\phi_i(\mathrm{ML}, x) = \sum_{t \subseteq x}^{n} \frac{|t|!(n - |t| - 1)!}{n!} [\mathrm{ML}(t) - \mathrm{ML}(t \setminus i)] \tag{10}$$

Here, the i^{th} input variable is represented by t_i when there are n observations. ϕ_0 denotes the model output in the case where all simplified input features are inactive or set to their baseline values, and ϕ_i indicates the contribution of the i^{th} variable to the original prediction ML model, while t corresponds to the fraction of the model's features that excludes the feature value i.

4 Results and Discussion

This dataset consisted of data related to 383 patients who participated in a retrospective cohort study on differentiated thyroid cancer recurrence. It contained 16 variables, including one numerical variable (the patient's age) and 15 categorical variables representing clinical and pathological features of patients. Figure 2 depicts the age distribution of the patients at diagnosis, and Table 2 describes the descriptive statistics calculated for patients' age

Table 2: Descriptive statistics for the age variable

Minimum	Maximum	Mean	Standard Deviation
15	82	40.867	15.134

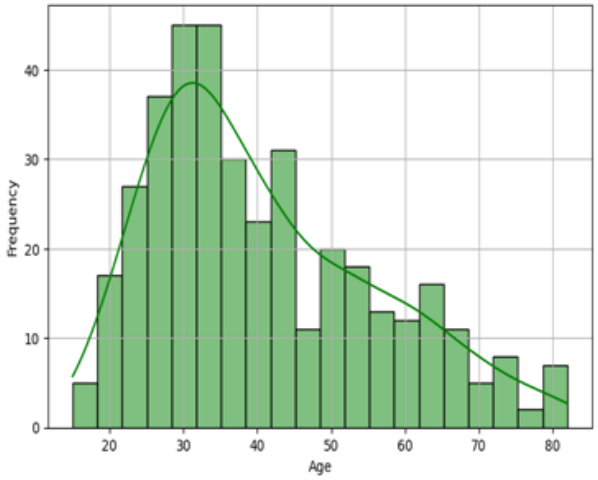


Figure 2: Age distribution of the patients at diagnosis

According to Figure 2, it is clear that the age distribution is right-skewed, indicating that the majority of the patients are younger, while fewer patients are older. However, patients' ages ranged from 15 to 82 years, while the peak of the distribution is observed in the 30-35 years range, indicating that most of the patients are within this range. In addition, the mean age of patients is 40.867 years ($\cong$ 41 years) while the standard deviation is 15.134 years. To gain a clear understanding of the dataset, the baseline characteristics of the patients by recurrence status are described in Table 3.

Table 3: Baseline characteristics of the patients by recurrence status

Features	**Recurred (%)**	**Not Recurred (%)**
Age	108 (28.20%)	275 (71.80%)
Gender		
Male	42 (10.97%)	29 (7.57%)
Female	66 (17.23%)	246 (64.23%)
Smoking Status		
Smoking	33 (8.62%)	16 (4.18%)
Not Smoking	75 (19.58%)	259 (67.62%)
Past Smoking Status		
Smoked	14 (3.66%)	14 (3.66%)
Never Smoked	94 (24.54%)	261 (68.15%)
Radiotherapy to the Head and Neck Region		
Received	6 (1.57%)	1 (0.26%)
Not Received	102 (26.63%)	274 (71.54%)
Thyroid Function Categories		
Clinical Hyperthyroidism	3 (0.783%)	17 (4.439%)
Clinical Hypothyroidism	2 (0.522%)	10 (2.611%)
Euthyroid	98 (25.857%)	234 (61.097%)

Continued on next page

Table 3 – *Continued from previous page*

Features	Recurred (%)	Not Recurred (%)
Subclinical Hyperthyroidism	0 (0.000%)	5 (1.305%)
Subclinical Hypothyroidism	5 (1.305%)	9 (2.350%)
Goiter Type		
Diffuse	0 (0.000%)	7 (1.83%)
Multinodular	52 (13.58%)	88 (22.98%)
Normal	2 (0.52%)	5 (1.31%)
Single Nodular Left	26 (6.79%)	63 (16.45%)
Single Nodular Right	28 (7.31%)	112 (29.24%)
Adenopathy Location		
Bilateral	27 (7.050%)	5 (1.305%)
Extensive	7 (1.828%)	0 (0.000%)
Left	12 (3.133%)	5 (1.305%)
No Adenopathy	30 (7.833%)	247 (64.491%)
Posterior	2 (0.522%)	0 (0.000%)
Right	30 (7.833%)	18 (4.700%)
Pathological Subtype of Cancer		
Follicular	12 (3.13%)	16 (4.18%)
Hurthel cell	6 (1.57%)	14 (3.66%)
Micropapillary	0 (0.00%)	48 (12.53%)
Papillary	90 (23.50%)	197 (51.44%)
Focality		
Multi Focal	70 (18.28%)	66 (17.23%)
Uni Focal	38 (9.92%)	209 (54.57%)
Risk Type		
High	32 (8.36%)	0 (0.00%)
Intermediate	64 (16.71%)	38 (9.92%)
Low	12 (3.13%)	237 (61.88%)
Tumor Stage		
T1a	1 (0.261%)	48 (12.533%)
T1b	5 (1.305%)	38 (9.922%)
T2	20 (5.222%)	131 (34.204%)
T3a	41 (10.705%)	55 (14.360%)
T3b	14 (3.655%)	2 (0.522%)
T4a	9 (4.961%)	1 (0.261%)
T4b	8 (2.089%)	0 (0.000%)
Node Status		
N0	27 (7.05%)	241 (62.92%)
N1a	10 (2.61%)	12 (3.13%)
N1b	71 (18.54%)	22 (5.74%)
Metastasis Status		
M0	90 (23.50%)	275 (71.80%)
M1	18 (4.70%)	0 (0.00%)
Cancer Stage		
I	65 (16.971%)	268 (69.974%)
II	25 (6.527%)	7 (1.828%)
III	4 (1.044%)	0 (0.000%)
IVA	3 (0.783%)	0 (0.000%)
IVB	11 (2.872%)	0 (0.000%)

Continued on next page

Table 3 – *Continued from previous page*

Features	Recurred (%)	Not Recurred (%)
Response to Initial Treatment		
Biochemical Incomplete	11 (2.87%)	12 (3.13%)
Excellent	1 (0.26%)	207 (54.05%)
Indeterminate	7 (1.83%)	54 (14.10%)
Structural Incomplete	89 (23.24%)	2 (0.52%)

When considering Table 3, it is clear that 28.20% of the total patients experienced thyroid cancer recurrence, while 71.80% did not experience it. Furthermore, this cancer mostly recurred in males, current smokers, and patients who received radiotherapy to the neck and head. In terms of thyroid function, euthyroidism was common among the non-recurrence group, while multinodular goiter type was frequently observed among the recurrence patients. When considering adenopathy location, it was observed that either adenopathy on the right side or no adenopathy was present in the patients with recurrence status, whereas the majority of the non-recurrence group had no adenopathy. According to Table 3, the papillary cancer subtype was common among the recurrence participants, while the micropapillary subtype was not observed in the same group. In addition, the individuals with multifocal tumors with T3a tumor stage, N1b node status, M1 metastasis status, and high cancer stages such as (III, IVA, IVB) had a higher likelihood of DTC recurrence. Moreover, patients who experienced cancer recurrence showed poor response to initial treatments, while the participants with non-recurrence status responded excellently to the treatments.

After calculating the dataset's basic statistics, traditional ML models were first applied to the training dataset containing all the variables. These models were trained using 2-fold, 5-fold, and 10-fold cross-validation and tested using the test dataset. Evaluation metrics, including accuracy, precision, recall, and F1-score, were calculated for each of these cross-validation settings, and Table 4 depicts the highest metrics obtained among the cross-validation results.

Table 4: Evaluation metrics for machine learning models before the feature selection

Model	Accuracy	Precision	Recall	F1-Score
SVM	0.9481	0.8947	0.8947	0.8947
RF	0.9221	0.9333	0.7368	0.8235
KNN	0.8701	0.6800	0.8947	0.7727
DT	0.8961	0.8236	0.7368	0.7778
LR	0.9359	0.8888	0.8421	0.8649
GB	0.9221	0.7826	0.9474	0.8571
XGB	0.8831	0.7083	0.8947	0.7907
LGB	0.8571	0.6426	0.9474	0.7659
Catboost	0.9091	0.7727	0.8947	0.8293
MLP	0.8684	0.6538	0.9444	0.7727
NB	0.8816	0.7143	0.8333	0.7692

According to Table 4, it is clear that all models have recorded higher accuracies, with each model obtaining above 0.85. However, overall, SVM performed well, achieving a higher accuracy of 0.9481 with a 0.8947 precision, 0.8947 recall, and 0.8947 F1 score, making it suitable for classification. Furthermore, the LR model also performed better, recording the second-highest accuracy. Despite the better performance, it is necessary to assess the confusion matrix obtained for the SVM model. Therefore, the confusion matrix of the SVM is demonstrated in Figure 3.

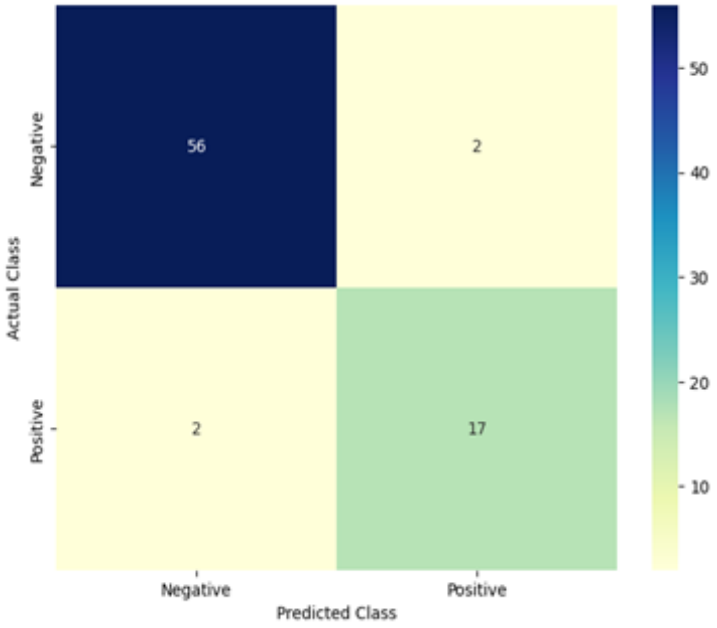


Figure 3: Confusion matrix for the SVM model before feature selection

When considering Figure 3, it is evident that the SVM model was able to make correct predictions with a higher degree of accuracy and fewer misclassifications. Although the SVM model performed well with the complete set of variables, systematically identifying the most relevant features that have a greater predictive power is crucial to reducing the model's complexity and providing robust results as well as interpretations. Therefore, a feature selection was performed using the Boruta algorithm, and the most influential features were obtained, as shown in Figure 4.

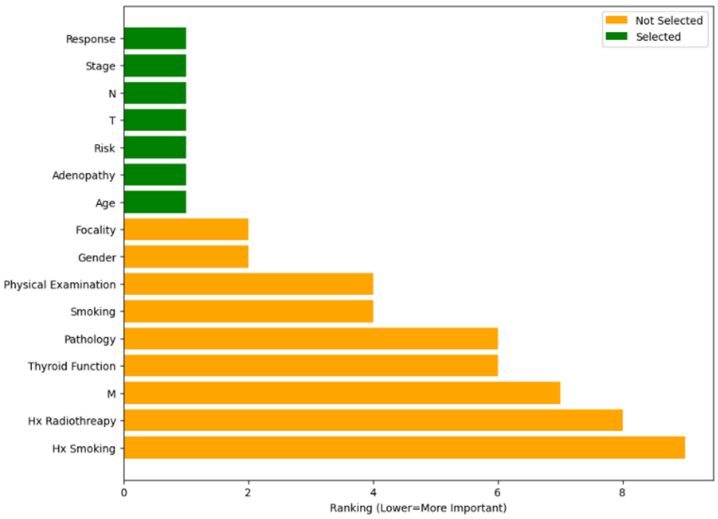


Figure 4: Feature importance by the Boruta algorithm

According to Figure 4, response to initial treatment, cancer stage, node status, tumor stage, risk type, adenopathy location, and age are the most relevant features selected by the Boruta algorithm. Considering this, the same machine learning models were applied for the selected features and trained using 2-fold, 5-fold, and 10-fold cross-validation. These models were tested using test data, and accuracy, precision, recall, and F1-score were calculated for three cross-validation settings, and Table 5 indicates the highest metrics obtained across validation.

Table 5: Evaluation metrics for machine learning models after the feature selection

Model	Accuracy	Precision	Recall	F1-Score
SVM	0.9610	0.9000	0.9474	0.9231
RF	0.9481	0.8571	0.9474	0.9000
KNN	0.8831	0.7083	0.8947	0.7907
DT	0.9091	0.8333	0.7894	0.8108
LR	0.9611	0.9444	0.8947	0.9189
GB	0.9351	0.8823	0.7894	0.8833
XGB	0.8961	0.7619	0.8421	0.8000
LGB	0.8701	0.6800	0.8947	0.7727
Catboost	0.9351	0.8500	0.8947	0.8718
MLP	0.8701	0.6667	0.9474	0.7826
NB	0.8961	0.7391	0.8947	0.8095

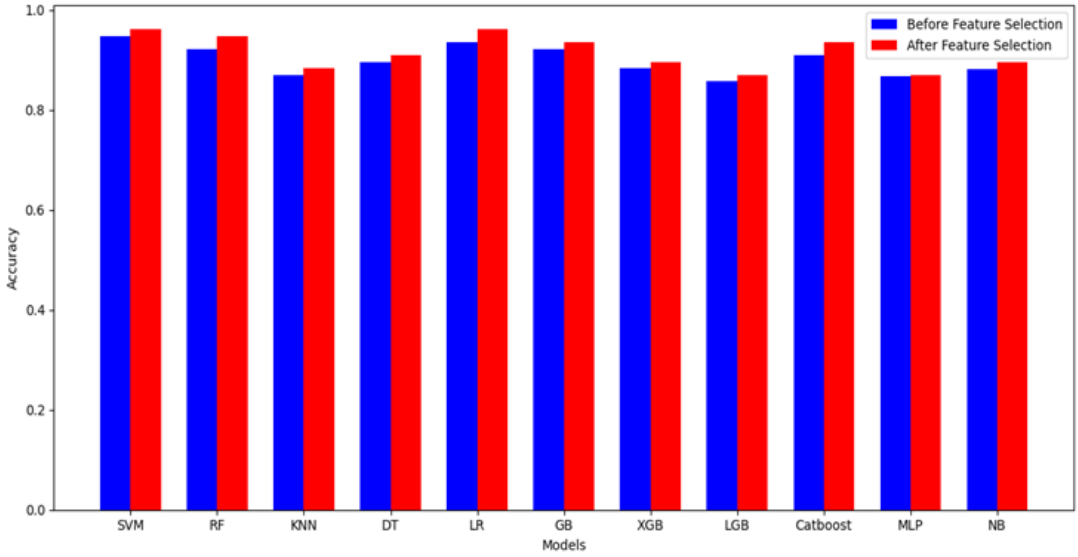


Figure 5: Model accuracies before and after feature selection

According to Table 5, the LR model outperformed the other models, achieving an accuracy of 0.9611, a precision of 0.9444, a recall of 0.8947, and an F1-score of 0.9189. Followed by that, the SVM model indicates an accuracy of 0.9610, which is slightly lower than that of the LR model. It is noted that each of the models has achieved a higher accuracy after feature selection, as shown in Figure 5, indicating that removing irrelevant features improves the model performance. Therefore, the LR model is identified as the best-performing model across all the machine learning models after feature selection, and its confusion matrix is denoted in Figure 6.

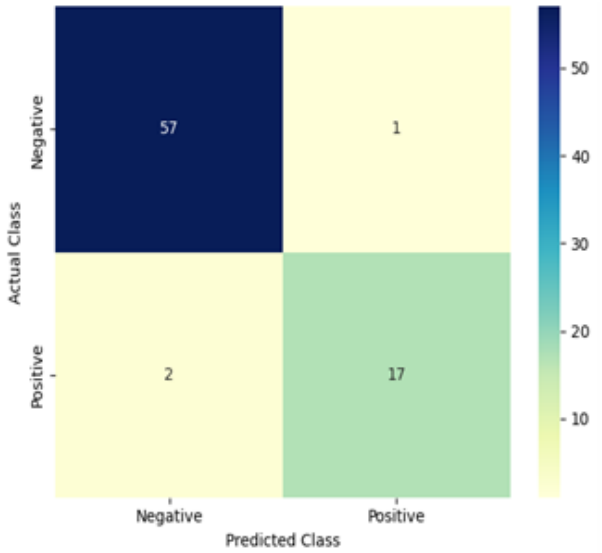


Figure 6: Confusion matrix for the LR model after feature selection

As shown in Figure 6, the LR model performs well with fewer misclassifications, achieving higher accuracy and indicating its capability in the classification of thyroid cancer recurrence. Although these traditional machine learning models performed well before and after feature selection, they lack uncertainty quantification, offering limited interpretability, particularly with small datasets. Therefore, to overcome these challenges with uncertainty quantification in traditional machine learning models, this study implemented a Bayesian Neural Network on the dataset before and after feature selection. The prior distributions of this model were varied from the standard normal distribution to the horseshoe prior, and accuracy, precision, recall, and F1-score for each model were calculated as depicted in Table 6.

Table 6: Evaluation metrics of the BNN model before feature selection

Model	**Prior**	**Accuracy**	**Precision**	**Recall**	**F1-Score**
Before feature selection	Normal (0,1)	0.9221	0.8823	0.7894	0.8333
	Normal (0,10)	0.9740	0.9474	0.9474	0.9474
	Laplace (0,1)	0.9221	0.8095	0.8947	0.8500
	Cauchy (0,1)	0.9091	0.8750	0.7368	0.8000
	Cauchy (0,2.5)	0.9351	0.9375	0.7894	0.8571
	Horseshoe (1)	0.8961	0.7200	0.9474	0.8182
After feature selection	Normal (0,1)	0.9359	0.8888	0.8421	0.8649
	Normal (0,10)	0.9870	1.0000	0.9474	0.9729
	Laplace (0,1)	0.9481	0.8947	0.8947	0.8947
	Cauchy (0,1)	0.9351	0.8182	0.9474	0.8780
	Cauchy (0,2.5)	0.9481	0.9412	0.8421	0.8889
	Horseshoe (1)	0.9091	0.8750	0.7368	0.8000

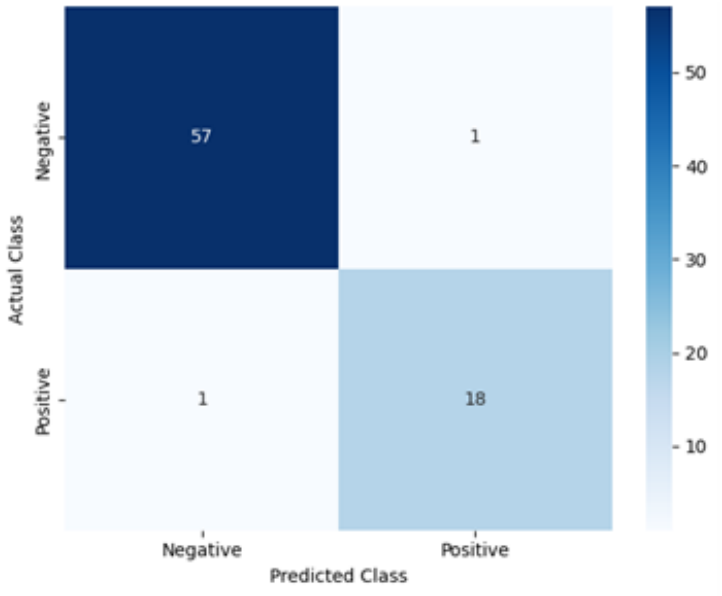


Figure 7: Confusion matrix for the BNN model with Normal (0,10) prior distribution before feature selection

According to Table 6, it is noted that the BNN model with the Normal (0,10) prior distribution achieved the highest accuracy of 0.9740 with a precision of 0.9474, recall of 0.9474, and F1-score of 0.9474 across six prior distributions before feature selection. This surpassed the SVM model, which showed a higher accuracy across all the ML models employed before the feature selection. The confusion matrix of this BNN model, which is illustrated in Figure 7, also confirms that this model performed well with fewer misclassifications.

Similarly, BNN with the same prior obtained a maximum accuracy of 0.9870 after feature selection, indicating the model's effectiveness in capturing complex relationships while quantifying uncertainties.

However, when comparing the accuracies of the BNN and machine learning models used in this study, the BNN model with Normal (0,10) prior outperformed all the other models both before and after the feature selection, which is illustrated in Figure 8.

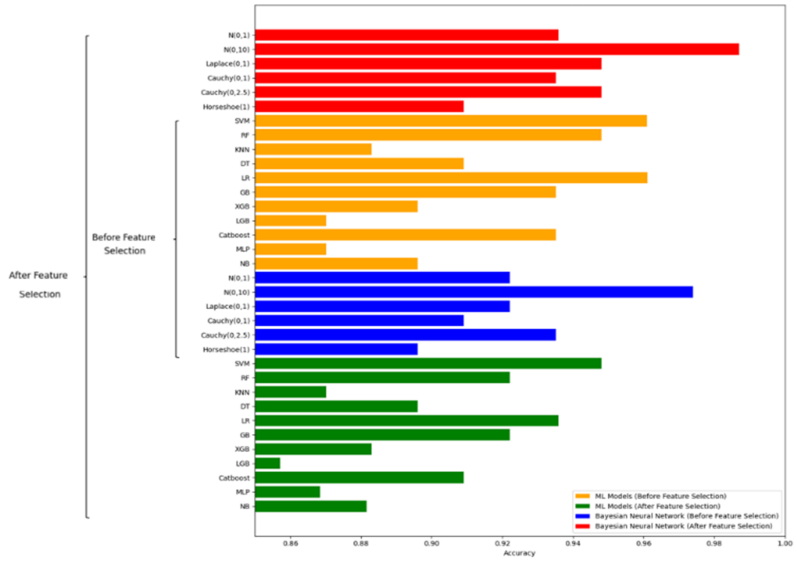


Figure 8: Accuracies of the machine learning and Bayesian models before and after feature selection

This indicates that the BNN model with a less restrictive prior enhances the predictive performance of the model by accurately estimating parameters, thereby highlighting the model's flexibility in parameter estimation through posterior distributions.

Therefore, the BNN model with Normal (0,10) prior distribution employed on the feature-selected dataset was chosen as the best-performing model for thyroid cancer recurrence classification over all the machine learning and Bayesian models discussed in this study. To further analyse, the confusion matrix of this model was generated and depicted in Figure 9.

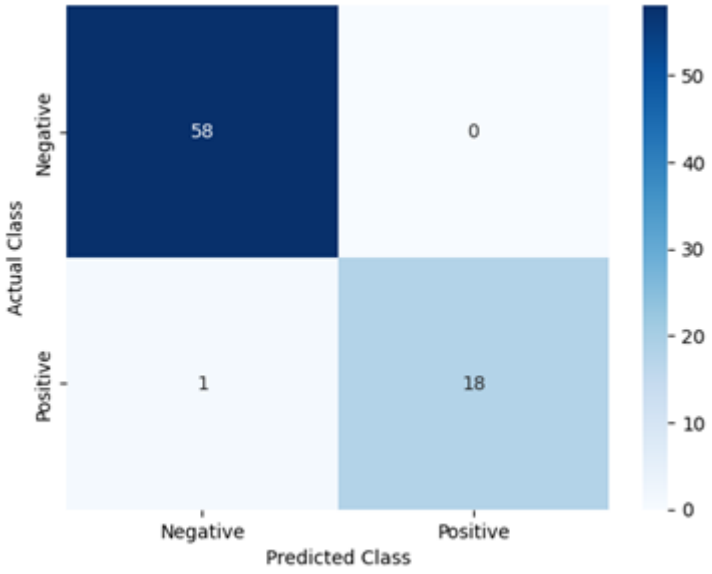


Figure 9: Confusion matrix of Bayesian Neural Network with N (0,10) prior distribution after feature selection

When considering Figure 9, it is evident that the BNN with Normal (0,10) prior performed excellently with only one misclassification while achieving a precision of 1.000. In Bayesian models, it is vital to visualise the epistemic uncertainties in order to identify the uncertainties in model predictions. In addition to epistemic uncertainties, it is equally crucial to visualise aleatoric uncertainties, which result from intrinsic noise or uncertainty in the data itself. Therefore, the posterior predictive mean with epistemic uncertainties as well as aleatoric uncertainties is depicted in Figures 10 and 11, respectively.

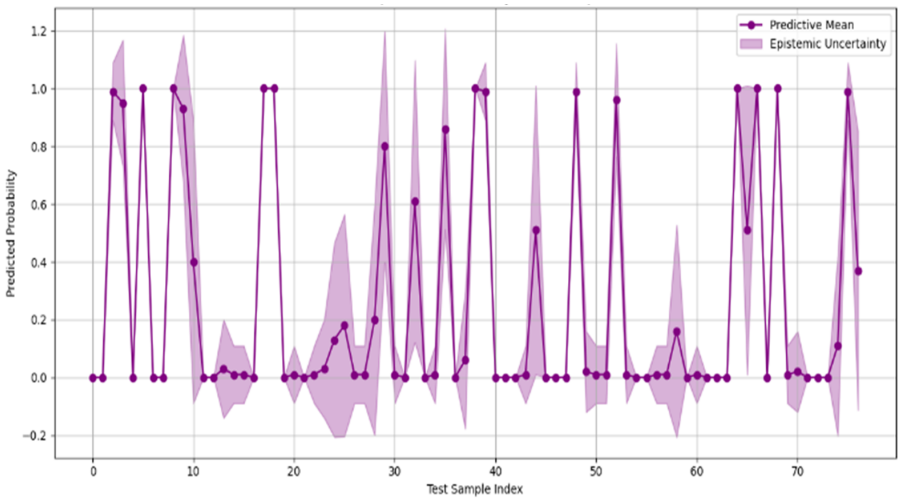


Figure 10: Posterior predictive mean with epistemic uncertainty

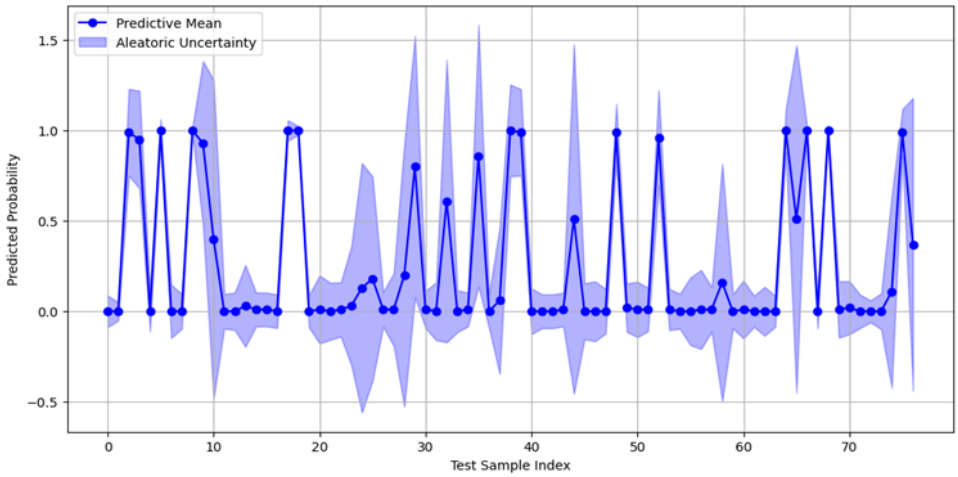


Figure 11: Posterior predictive mean with aleatoric uncertainty

In these figures, the purple and blue lines represent the mean predicted probability for the test samples, and the shaded regions indicate the epistemic uncertainty and aleatoric uncertainty. In Figure 10, the narrow-shaded area highlights the higher confidence, indicating low epistemic uncertainty, where predictions are consistent across posterior samples, while the wider regions show lower confidence, demonstrating the model's high uncertainty. Similarly, the aleatoric uncertainties depicted in Figure 11 illustrate the uncertainty inherent in the data itself through the shaded areas. In addition, several test samples in both Figures 10 and 11 depicted higher probabilities, which are closer to 1 with less uncertainty. This shows the ability of the BNN model to accurately predict whether the thyroid cancer recurred.

Furthermore, these plots depict probabilities that fall within the intermediate range, where the majority of them are accompanied by wider uncertainty. This highlights the model's reduced confidence in predicting the thyroid recurrence class and indicates that these test samples should be handled carefully and that further evaluation is needed.

However, while evaluating how confident the BNN model is in its predictions using epistemic and aleatoric uncertainty plots, it is crucial to assess why the model made these predictions, even if it performed well. Therefore, SHAP values were calculated for the selected BNN model to examine how each feature affects the final predictions, providing insights into the model's output. Initially, a global interpretation was carried out using SHAP values, in order to provide a brief understanding of the model's behaviour. Firstly, a summary bar plot of aggregated mean SHAP values was obtained as in Figure 12 to assess the overall importance of the features contributing to final predictions.

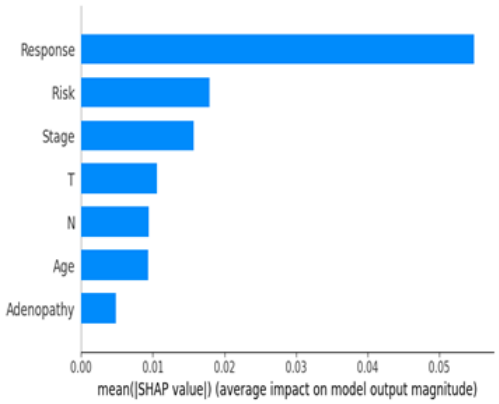


Figure 12: Bar plot of mean aggregated SHAP values for the BNN model after feature selection

According to Figure 12, it is observed that the response to initial treatment acts as the most influential factor among the seven variables in deciding whether the differentiated thyroid would recur or not. Followed by that, risk type and cancer stage contribute significantly to the final output, highlighting their role in cancer recurrence. Furthermore, tumor stage, node status, age, and adenopathy moderately affect the model's output. Although Figure 12 provided an overview of how each feature affects the final prediction, it is crucial to examine how each feature is varied across patients. Therefore, a SHAP beeswarm plot was obtained as in Figure 13, providing a global interpretation of the selected BNN model.

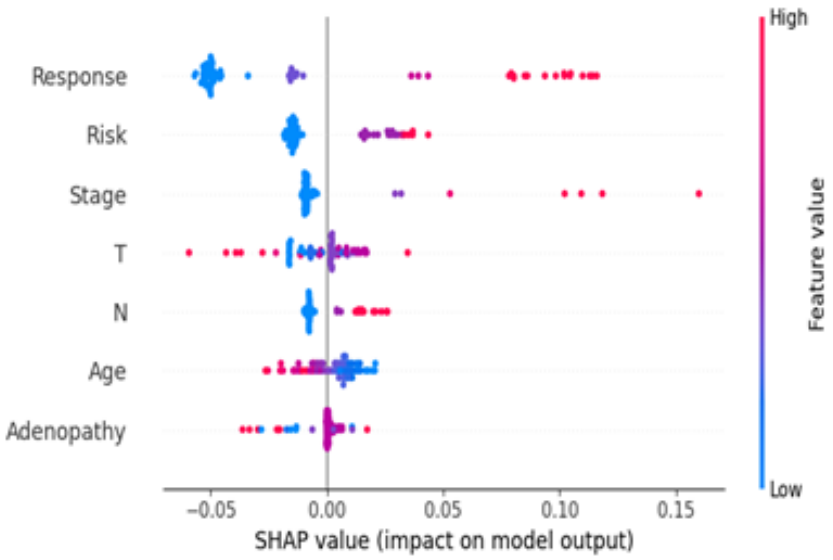


Figure 13: SHAP beeswarm plot for the selected BNN model

The beeswarm plot ranks the features according to their mean absolute SHAP value, where the features with a higher mean absolute SHAP value are located at the top, while the features with a lower value are at the bottom. Each dot in Figure 13 represents a patient, which indicates the magnitude as well as the direction of the SHAP value for a particular feature, where the instances with higher SHAP values are colored in red, while the cases with lower SHAP values are in blue. For example, response to initial treatment indicates a strong influence on the model's output, where a patient with red dots increases the prediction, showing a robust association with the recurrence of thyroid cancer. However, this demonstrates that the patients with higher severity status, such as structural incomplete responses to initial treatments, may have an increased recurrence of thyroid cancer. In contrast, low SHAP values indicate that the excellent response to initial treatments negatively affects the recurrence of thyroid cancer.

In addition, higher stages of risk type, cancer stage, and node status indicate positive SHAP values, suggesting that as the severity of these features increases, the probability of recurrence of thyroid cancer increases. However, when considering the tumor stage, it is observed that there are both red and purple plots, slightly placed on the right side of the plot, while some points are near zero, and a few are on the left side. This reveals that the larger tumors raise the risk of recurrence, while the effect of this feature is associated with other variables that need further clinical investigations.

Furthermore, the age variable shows a moderate impact on the model's predictions, where the younger age patients that are depicted in blue dots indicate a higher risk of thyroid cancer recurrence, while older patients have a lower risk or neutral effect on cancer recurrence. In addition, lower SHAP values for adenopathy demonstrate less impact on the model's output. However, this shows a mix of negative and positive contributions, highlighting that the effect of adenopathy is subtle and further investigations are required.

In addition to the SHAP global interpretation, this study focused on generating plots regarding SHAP local interpretations to provide a detailed explanation about the predictions for each instance in the dataset. Therefore, to assess how each of these features affects the predictions for each patient sequentially, a SHAP decision plot was obtained, as shown in Figure 14.

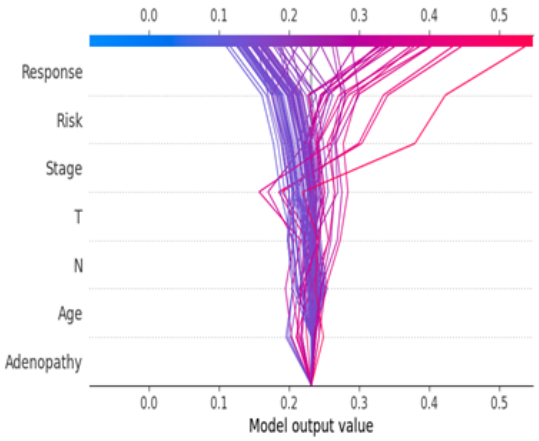


Figure 14: SHAP decision plot for the selected BNN model

When considering the decision plot in Figure 14, each prediction line starts from a common baseline value and then progressively shifts the direction of each line of all features, either increasing or decreasing the risk of thyroid recurrence. Response to initial treatment indicates a significant shift in the prediction, highlighting its effect in predicting the recurrence status. Notably, the poor response to initial treatment suggests an increased probability of cancer recurrence, which is evident in the right shift of the prediction lines in Figure 14. Similarly, risk type, cancer stage, and node status also exhibit a positive impact on the recurrence prediction, especially in patients with higher values. In addition, tumor stage and adenopathy display a deviated pattern depicting a decreased prediction risk, which suggests that these features are associated with other variables. However, age demonstrates a modest fluctuation where the prediction lines are observed in both directions. Furthermore, to examine how each of these features contributes to individual predictions, two waterfall plots for recurrence and non-recurrence classes were obtained for the same model as in Figures 15 and 16.

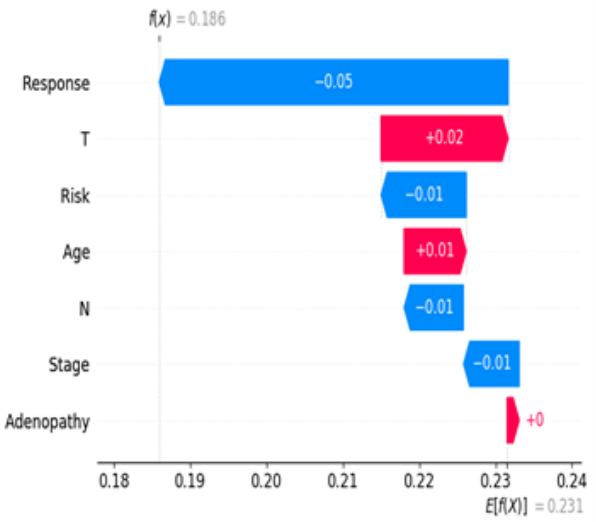


Figure 15: Waterfall plot for the non-recurrence class of the selected BNN model

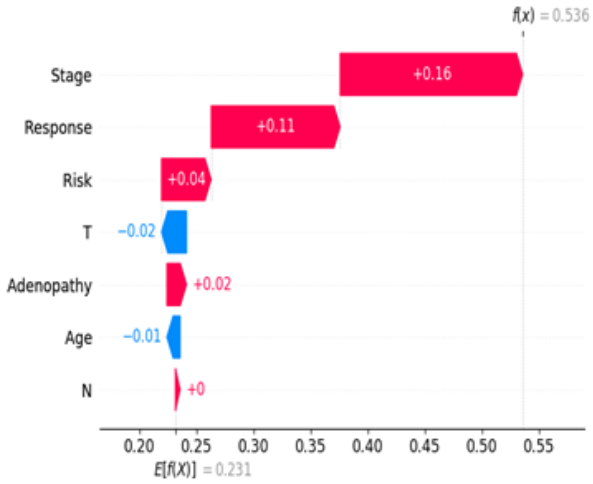


Figure 16: Waterfall plot for the recurrence class of the selected BNN model

Figures 15 and 16 provide a detailed explanation of the individual features that decide the thyroid cancer recurrence status of a patient, starting with an average recurrence probability of 0.231. According to Figure 15, the response feature shows a significant negative association with the non-recurrence prediction, indicating a SHAP value of -0.05. At the same time, it exhibits a SHAP value of +0.11, showing its contribution to recurrence prediction as in Figure 16. This shows that an excellent or indeterminate response to initial treatment notably reduces the risk of recurrence, while a poor response increases the risk. When considering Figure 15, it is observed that the lower levels of risk type, node status, and cancer stage reduce the likelihood of recurrence of thyroid cancer despite the slight increases in tumor stage and age. As depicted in Figure 16, the cancer stage has a SHAP value of +0.16, which indicates a strong positive correlation with the recurrence of the DTC. Factors such as response, risk, and adenopathy favourably impact the recurrence, while tumor stage and age depict a slight negative influence. However, these two waterfall plots highlight that the variables, such as response to initial treatment, act as a key factor that drives the decision of thyroid cancer recurrence, despite the fact that variables such as tumor stage and adenopathy need careful consideration in some contexts.

However, despite the promising results and clear interpretations of this study, there are several notable limitations and challenges that should be considered. One of the major limitations of this study is the relatively small sample size, which contained 383 patients. This may restrict the model's capability in capturing hidden patterns and complex relationships, and impact the posterior estimation as well. In addition, the data were collected from a retrospective cohort in a single clinical centre, which limits the generalizability and fails to make interpretations on a diverse population. Furthermore, the absence of an external validation is also a major limitation of this study. Although cross-validation was conducted, the lack of external validation using a diverse dataset is a significant restriction of this study. Moreover, this study implemented several machine learning models for classification, which could be used as a benchmark in future studies. Compared to these machine learning models, the BNN model employed in this study, with varying prior distributions, performed well in classification. However, only six prior distributions were considered for the BNN model by the authors of this study, which may introduce a notable bias. In addition, the use of standard prior distributions may limit the exploration and integration of the expert knowledge. However, these limitations could be addressed strategically in future studies. For example, this study could be further expanded by incorporating a larger dataset from multicentres, thereby including a diverse population to increase the interpretability and generalizability of the model. In addition, the results obtained from this study

can be confirmed, or any variations could be pointed out by considering an external validation dataset. Furthermore, this study was limited to a basic BNN architecture. Therefore, future studies could evaluate the performance of the BNN model with a deeper architecture by including additional hidden layers to capture more complex patterns. In addition, other Bayesian models, such as Bayesian logistic regression, could be used to compare the performance of the BNN model in thyroid cancer recurrence classification. Furthermore, by incorporating domain-informed priors, hierarchical priors, and expert opinions, this study can be further extended in order to provide robust results, thereby effectively classifying thyroid cancer recurrence.

5 Conclusion

This study implemented both classical ML models and a BNN model with varying prior distributions to classify differentiated thyroid cancer recurrence using a dataset containing 383 patients with 16 variables. These models were trained using the complete set of variables and the reduced set obtained using the Boruta algorithm. SVM performed well before feature selection, obtaining an accuracy of 0.9481, while the LR model outperformed with an 0.9611 accuracy after feature selection. Notably, BNN with Normal (0, 10) prior distribution depicted stronger performance than both ML and BNN models before and after feature selection. It achieved an accuracy of 0.9870 after feature selection, offering robust uncertainty quantifications. Therefore, this model was selected as the best-performing model for DTC recurrence classification. Epistemic and aleatoric uncertainties were obtained for this model, reflecting confidence in the model's predictions. Furthermore, SHAP-based interpretation was performed to enhance the explainability of the BNN model by identifying the most influential features driving its predictions. Finally, it can be concluded that this study provides a comprehensive framework by combining feature selection, ML models, Bayesian models, uncertainty quantification and SHAP interpretations.

References

[1] S. Mitra et al., "Prospective multifunctional roles and pharmacological potential of dietary flavonoid narirutin," Biomedicine & Pharmacotherapy, vol. 150, p. 112932, Jun. 2022, doi: https://doi.org/10.1016/j.biopha.2022.112932.

[2] D. Das, A. Banerjee, A. B. Jena, A. K. Duttaroy, and S. Pathak, "Essentiality, relevance, and efficacy of adjuvant/combinational therapy in the management of thyroid dysfunctions," Biomedicine & Pharmacotherapy, vol. 146, p. 112613, Feb. 2022, doi: https://doi.org/10.1016/j.biopha.2022.112613.

[3] S. Bhattacharya, R. K. Mahato, S. Singh, G. K. Bhatti, S. S. Mastana, and J. S. Bhatti, "Advances and challenges in thyroid cancer: The interplay of genetic modulators, targeted therapies, and AI-driven approaches," Life Sciences, vol. 332, p. 122110, Nov. 2023, doi: https://doi.org/10.1016/j.lfs.2023.122110.

[4] S. Rajoria et al., "Estrogen activity as a preventive and therapeutic target in thyroid cancer," Biomedicine & Pharmacotherapy, vol. 66, no. 2, pp. 151–158, Mar. 2012, doi: https://doi.org/10.1016/j.biopha.2011.11.010.

[5] N. Avenia et al., "Thyroid cancer invading the airway: diagnosis and management," International Journal of Surgery (London, England), vol. 28 Suppl 1, pp. S75-78, Apr. 2016, doi: https://doi.org/10.1016/j.ijsu.2015.12.036.

[6] A. Greco, C. Miranda, M. G. Borrello, and M. A. Pierotti, "Thyroid Cancer," Cancer Genomics, pp. 265–280, 2014, doi: https://doi.org/10.1016/b978-0-12-396967-5.00016-5.

[7] S. L. Gillanders and J. P. O'Neill, "Prognostic markers in well differentiated papillary and follicular thyroid cancer (WDTC)," European Journal of Surgical Oncology, vol. 44, no. 3, pp. 286–296, Mar.

2018, doi: https://doi.org/10.1016/j.ejso.2017.07.013.

[8] Y. Ito and A. Miyauchi, "Prognostic factors of papillary and follicular carcinomas based on pre-, intra-, and postoperative findings," European Thyroid Journal, Aug. 2024, doi: https://doi.org/10.1530/etj-24-0196.

[9] J. Sastre Marcos et al., "Differentiated thyroid carcinoma: survival and prognostic factors," Endocrinología y Nutrición (English Edition), vol. 58, no. 4, pp. 157–162, Jan. 2011, doi: https://doi.org/10.1016/s2173-5093(11)70040-x.

[10] A. R. Shaha, "Recurrent differentiated thyroid cancer," Endocrine Practice, vol. 18, no. 4, pp. 600–603, Jul. 2012. doi:10.4158/ep12047.co

[11] D. Iqbal and A. Shahzad, "Prediction of thyroid cancer recurrence with machine learning models," Pakistan Journal of Nuclear Medicine, p. 1, 2024, doi: https://doi.org/10.24911/pjnmed.175-1721068107.

[12] K. Setiawan, "Predicting recurrence in differentiated thyroid cancer: a comparative analysis of various machine learning models including ensemble methods with chi-squared feature selection," Communications in Mathematical Biology and Neuroscience, 2024, doi: https://doi.org/10.28919/cmbn/8506.

[13] E. Onah, U. J. Eze, A. S. Abdulraheem, U. G. Ezigbo, K. C. Amorha, and F. Ntie-Kang, "Optimizing unsupervised feature engineering and classification pipelines for differentiated thyroid cancer recurrence prediction," BMC Medical Informatics and Decision Making, vol. 25, no. 1, May 2025, doi: https://doi.org/10.1186/s12911-025-03018-3.

[14]] I. Juliana, K. Meylda, and Suharjito Suharjito, "Enhancing Predictive Accuracy for Differentiated Thyroid Cancer (DTC) Recurrence Through Advanced Data Mining Techniques," TIN Terapan Informatika Nusantara, vol. 5, no. 1, pp. 11–22, Jun. 2024, doi: https://doi.org/10.47065/tin.v5i1.5237.

[15] E. Clark, S. Price, T. Lucena, B. Haberlein, Abdullah Wahbeh, and Raed Seetan, "Predictive Analytics for Thyroid Cancer Recurrence: A Machine Learning Approach," Knowledge, vol. 4, no. 4, pp. 557–570, Nov. 2024, doi: https://doi.org/10.3390/knowledge4040029.

[16] Irem Senyer Yapici and R. U. Arslan, "Predictive analytics for thyroid cancer recurrence: a feature selection and data balancing approach," The European Physical Journal Special Topics, Jun. 2025, doi: https://doi.org/10.1140/epjs/s11734-025-01720-x.

[17] Shiva Borzooei, G. Briganti, Mitra Golparian, J. R. Lechien, and Aidin Tarokhian, "Machine learning for risk stratification of thyroid cancer patients: a 15-year cohort study," European Archives of Oto-Rhino-Laryngology, Oct. 2023, doi: https://doi.org/10.1007/s00405-023-08299-w.

[18] S. Das, A. K. Chaudhuri, N. R. Choudhury, and P. Ghosh, "Identification of the Recurrence of Differentiated Thyroid Cancer by Stacking Classifier," Research Square (Research Square), Jan. 2025, doi: https://doi.org/10.21203/rs.3.rs-5713674/v1.

[19] Ş. Yaşar, "Determination of Possible Biomarkers for Predicting Well-Differentiated Thyroid Cancer Recurrence by Different Ensemble Machine Learning Methods," Middle Black Sea Journal of Health Science, vol. 10, no. 3, pp. 255–265, Aug. 2024, doi: https://doi.org/10.19127/mbsjohs.1498383.

[20] G. M. Idroes et al., "Prognostication of differentiated thyroid cancer recurrence: An explainable machine learning approach," Narra X, vol. 2, no. 3, pp. e183–e183, Dec. 2024, doi: https://doi.org/10.52225/narrax.v2i3.183.

[21] M. A.-S. Ahmad and J. Haddad, "An Explainable AI Model for Predicting the Recurrence of Differentiated Thyroid Cancer," arXiv (Cornell University), Oct. 2024, doi: https://doi.org/10.48550/arxiv.2410.10907.

[22] A. K. Arslan and C. Çolak, "Explainable Machine Learning Models for Predicting Recurrence in Differentiated Thyroid Cancer," Medical Records, Aug. 2024, doi: https://doi.org/10.37990/medr.1525801.

[23] L. J. Lechuga López, S. Elsharief, D. Al Jorf, F. Darwish, C. Ma, and F. E. Shamout, "Uncertainty quantification for machine learning in healthcare: A survey," arXiv (Cornell University), May. 2025, doi: https://doi.org/10.48550/arXiv.2505.02874.

[24] T. Wang et al., "From aleatoric to epistemic: Exploring uncertainty quantification techniques in artificial intelligence," arXiv (Cornell University), Jan. 2025, doi: https://doi.org/10.48550/arXiv.2501.03282.

[25] "Differentiated thyroid cancer recurrence," Kaggle, Jan. 19, 2024. https://www.kaggle.com/datasets/joebeachcapital/differentiated-thyroid-cancer-recurrence

[26] K. Rupabanta Singh and S. Dash, "Early detection of neurological diseases using machine learning and deep learning techniques: A review," Elsevier eBooks, pp. 1–24, Jan. 2023, doi: https://doi.org/10.1016/b978-0-323-90277-9.00001-8.

[27] G. Cosma, D. Brown, M. Archer, M. Khan, and A. Graham Pockley, "A survey on computational intelligence approaches for predictive modeling in prostate cancer," Expert Systems with Applications, vol. 70, pp. 1–19, Mar. 2017, doi: https://doi.org/10.1016/j.eswa.2016.11.006.

[28] D. Wu, Y. Ren, L. He, and J. Johnson, "The identification of risk factors associated with COVID-19 in a large inpatient cohort using machine learning approaches," Elsevier eBooks, pp. 189–199, Jan. 2022, doi: https://doi.org/10.1016/b978-0-12-821318-6.00017-7.

[29] C. Zucco, "Multiple Learners Combination: Bagging," Encyclopedia of Bioinformatics and Computational Biology, pp. 525–530, 2019, doi: https://doi.org/10.1016/b978-0-12-809633-8.20345-2.

[30] C. Gong, Z. Su, X. Zhang, and Y. You, "Adaptive evidential K-NN classification: Integrating neighborhood search and feature weighting," Information sciences, vol. 648, pp. 119620–119620, Nov. 2023, doi: https://doi.org/10.1016/j.ins.2023.119620.

[31]] I. Semanjski, "Data analytics," Elsevier eBooks, pp. 121–170, Jan. 2023, doi: https://doi.org/10.1016/b978-0-12-820717-8.00008-7.

[32] M. Meloun and J. Militký, "Statistical analysis of multivariate data," Statistical Data Analysis, pp. 151–403, 2011, doi: https://doi.org/10.1533/9780857097200.151.

[33] A. Malik, Yash Tejas Javeri, M. Shah, and Ramchandra Mangrulkar, "Impact analysis of COVID-19 news headlines on global economy," Elsevier eBooks, pp. 189–206, Jan. 2022, doi: https://doi.org/10.1016/b978-0-12-824557-6.00001-7.

[34] W. Zhang, X. Gu, L. Hong, L. Han, and L. Wang, "Comprehensive review of machine learning in geotechnical reliability analysis: Algorithms, applications and further challenges," vol. 136, pp. 110066–110066, Mar. 2023, doi: https://doi.org/10.1016/j.asoc.2023.110066.

[35] Temidayo Oluwatosin Omotehinwa, David Opeoluwa Oyewola, and Emmanuel Gbenga Dada, "A Light Gradient-Boosting Machine algorithm with Tree-Structured Parzen Estimator for breast cancer diagnosis," pp. 100218–100218, Jun. 2023, doi: https://doi.org/10.1016/j.health.2023.100218.

[36] X. Ren, H. Yu, X. Chen, Y. Tang, G. Wang, and X. Du, "Application of the CatBoost Model for Stirred Reactor State Monitoring Based on Vibration Signals," Computer modeling in engineering & sciences, vol. 140, no. 1, pp. 647–663, Jan. 2024, doi: https://doi.org/10.32604/cmes.2024.048782.

[37] S. Abinaya and M. K. K. Devi, "Enhancing crop productivity through autoencoder-based disease detection and context-aware remedy recommendation system," Elsevier eBooks, pp. 239–262, Jan. 2022, doi: https://doi.org/10.1016/b978-0-323-90550-3.00014-x.

[38] O. Peretz, M. Koren, and O. Koren, "Naive Bayes classifier – An ensemble procedure for recall and precision enrichment," Engineering Applications of Artificial Intelligence, vol. 136, p. 108972, Oct. 2024, doi: https://doi.org/10.1016/j.engappai.2024.108972.

[39] G. Manikandan, B. Pragadeesh, V. Manojkumar, A. L. Karthikeyan, R. Manikandan, and A. H. Gandomi, "Classification models combined with Boruta feature selection for heart disease prediction," Informatics in medicine unlocked, vol. 44, pp. 101442–101442, Jan. 2024, doi: https://doi.org/10.1016/j.imu.2023.101442.

[40] L. V. Jospin, H. Laga, F. Boussaid, W. Buntine, and M. Bennamoun, "Hands-On Bayesian Neural Networks—A Tutorial for Deep Learning Users," IEEE Computational Intelligence Magazine, vol. 17, no. 2, pp. 29–48, May 2022, doi: https://doi.org/10.1109/mci.2022.3155327.

[41] M. Magris and A. Iosifidis, "Bayesian learning for neural networks: an algorithmic survey," Artificial Intelligence Review, Mar. 2023, doi: https://doi.org/10.1007/s10462-023-10443-1.

[42] M. Nishio and A. Arakawa, "Performance of Hamiltonian Monte Carlo and No-U-Turn Sampler for estimating genetic parameters and breeding values," Genetics Selection Evolution, vol. 51, no. 1, Dec. 2019, doi: https://doi.org/10.1186/s12711-019-0515-1.

[43] M. D. Hoffman and A. Gelman, "The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo," arXiv (Cornell University), Jan. 2011, doi: https://doi.org/10.48550/arxiv.1111.4246.

[44] A. Kendall and Y. Gal, "What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision?," arXiv (Cornell University), Jan. 2017, doi: https://doi.org/10.48550/arxiv.1703.04977.

[45]] J. W. Lund, J. T. Groten, D. L. Karwan, and C. Babcock, "Using machine learning to improve predictions and provide insight into fluvial sediment transport," Hydrological Processes, vol. 36, no. 8, Jul. 2022, doi: https://doi.org/10.1002/hyp.14648.

[46]] E. Štrumbelj and I. Kononenko, "Explaining prediction models and individual predictions with feature contributions," Knowledge and Information Systems, vol. 41, no. 3, pp. 647–665, Aug. 2013, doi: https://doi.org/10.1007/s10115-013-0679-x.

[47] S. Lundberg and S.-I. Lee, "A Unified Approach to Interpreting Model Predictions," arXiv.org, Nov. 24, 2017. https://arxiv.org/abs/1705.07874v2

[48]] S. Wang, H. Peng, Q. Hu, and M. Jiang, "Analysis of runoff generation driving factors based on hydrological model and interpretable machine learning method," Journal of Hydrology: Regional Studies, vol. 42, p. 101139, Aug. 2022, doi: https://doi.org/10.1016/j.ejrh.2022.101139.

[49]] C. Molnar, G. Casalicchio, and B. Bischl, "Interpretable Machine Learning – A Brief History, State-of-the-Art and Challenges," ECML PKDD 2020 Workshops, pp. 417–431, 2020, doi: https://doi.org/10.1007/978-3-030-65965-3_28.

[50]] H. Lamane, L. Mouhir, R. Moussadek, B. Baghdad, O. Kisi, and A. El Bilali, "Interpreting machine learning models based on SHAP values in predicting suspended sediment concentration," International Journal of Sediment Research, vol. 40, no. 1, pp. 91–107, Feb. 2025, doi: https://doi.org/10.1016/j.ijsrc.2024.10.002.

Declarations

Funding
This research received no external funding.

Conflicts of Interest
The authors declare that they have no conflicts of interest.

Ethical Considerations
The authors state that all work related to this research was conducted in accordance with

institutional, national, and international guidelines and in compliance with recognized ethical standards. This article does not contain any studies with human participants or animals performed by any of the authors.

AI Usage Statement
No generative AI tools were used in the preparation of this manuscript.

Data Availability Statement
The datasets used in this study are publicly available at: https://www.kaggle.com/datasets/joebeachcapital/differentiated-thyroid-cancer-recurrence

Code Availability Statement
The software codes used in this study are available from the corresponding author upon reasonable request.

SDG Alignment
SDG 3 – Good Health and Well-Being.

International Journal of Research in Computing

ORIGINAL RESEARCH
Volume 5, Issue 2 (2026)

Prompt Optimization Strategies for Large Language Models: Insights from Random Prompting with Gemini API

OAM Jayawardana[1]

[1]Department of Mathematical Sciences, Faculty of Applied Sciences, Wayamba University of Sri Lanka.

[1]amjmadusanka@gmail.com | https://orcid.org/0009-0009-3410-5832

Abstract

The increasing use of large language models (LLMs) has raised concerns about efficiency regarding token usage, as excessive token consumption burdens computational resources. This study addresses the need for systematic techniques to optimize prompts, evaluating three specific strategies within the Gemini API framework aimed at reducing token usage. The research examined 480 prompts across 16 models in the Gemini family to quantify potential token savings. The three strategies assessed are structured concise formatting, which utilizes bullet lists, headings, and concise language to streamline content; verbose removal, which eliminates redundant and irrelevant information to enhance clarity and focus; and code formatting, which promotes consistent syntax while minimizing inline commentary. Token counts were tracked before and after optimization using the Gemini countTokens API endpoint. Findings revealed an average token saving of 37.15% across most Gemini-family models, with Gemma 3 models achieving a higher average reduction of 43.55%. Among the strategies, verbose removal yielded the greatest efficiency at an average reduction of 51.98%, followed by structured concise formatting at 33.36% and code formatting at 29.71%. The maximum individual saving recorded was 126 tokens. Multi-model testing with a shared fixed prompt set demonstrated consistent tokenization behavior within the Gemini ecosystem across Gemini 2.0, 2.5, 3 preview, latest Gemini, and Gemma 4 model groups. Gemma 3 models consistently demonstrated higher efficiency, indicating variations in tokenization behavior influenced by model architecture. These results suggest that prompt optimization can significantly enhance token efficiency within the Gemini model family, achieving reductions of approximately 30% to over 50% depending on strategy and model type; cross-ecosystem generalizability remains to be established. This research offers valuable insights for practitioners aiming to reduce computational costs and improve inference efficiency in LLM-based systems.

KEYWORDS
Computational Efficiency, Large Language Models, LLM Optimization, Prompt Engineering.

ARTICLE HISTORY
Published: 16 July 2026

DATA/CODE AVAILABILITY
Data and code are available from the corresponding author upon reasonable request.

SDG ALIGNMENT
SDG 4
SDG 9

1 Introduction

Large language models (LLMs) have become central to workflows across scientific, commercial, and creative domains, yet their widespread adoption has brought growing scrutiny of computational costs. Every LLM interaction is metered in tokens, the atomic units of text processing, meaning that unnecessarily verbose prompts directly inflate operational expenses and energy consumption at scale. Despite this, most practitioners rely on intuition rather than systematic methods when constructing prompts, leaving substantial efficiency gains unrealized. This paper presents an empirical investigation of three prompt optimization strategies, namely verbose removal, structured concise formatting, and code formatting, evaluated through direct token measurement using the Gemini API's native countTokens endpoint across 16 model variants and 480 prompts. The following subsections define the research problem, objectives, and significance in detail.

1.1 Research Problem

The rapid adoption of Large Language Models (LLMs) across scientific, commercial, and creative domains has been accompanied by growing concerns regarding computational efficiency and cost. Token consumption, the fundamental unit of text processing in LLMs, directly impacts both operational expenses and environmental footprint through increased computational requirements. As organizations scale their LLM usage, token optimization has emerged as a critical challenge requiring systematic approaches rather than ad-hoc solutions. This research investigates token optimization strategies through systematic testing with the Gemini API using randomly generated prompts. By testing 480 diverse prompts across three optimization strategies, we measure actual token consumption before and after optimization, providing empirical validation of effectiveness claims. The core hypothesis driving this research is that simple prompt engineering techniques can significantly reduce token consumption without compromising response quality. By generating random prompts across various domains and measuring their optimization potential, we establish realistic benchmarks for token reduction in LLM applications.

1.2 Research Objectives

This research is guided by five primary objectives:

1. Identify and categorize token-heavy patterns in LLM prompts through systematic analysis.
2. Develop three optimization strategies for reducing token consumption.
3. Test strategies empirically using the Gemini API with randomly generated prompts.
4. Measure and validate token reduction effectiveness across diverse use cases.
5. Establish a reusable framework for token optimization in LLM applications.

1.3 Research Significance

The significance of this research extends beyond immediate cost savings. Token optimization enables increased accessibility by reducing computational requirements, thereby making LLMs available to resource-constrained environments that would otherwise be excluded from deployment. Environmental benefits follow directly: lower token consumption reduces energy usage and associated carbon footprint. Improved latency results from smaller prompts that process faster, improving end-user experience in interactive applications. Enhanced scalability permits

larger-scale deployments within fixed budgets, a critical consideration for organizations operating under resource constraints. Finally, better model utilization is achieved when efficient prompting allows models to allocate computational capacity toward substantive tasks rather than processing redundant information.

2 Literature Review

2.1 Evolution of Token Efficiency in LLMs

The computational demands of large language models have driven extensive research into token optimization, particularly as API-based deployments scale across organizational use cases. Early work focused on architectural efficiencies, but recent scholarship emphasizes prompt engineering as a cost-effective intervention. For instance, Brown et al. demonstrated that token consumption directly correlates with inference costs, where inefficient prompting can inflate expenses considerably in high-volume scenarios [1]. This aligns with the current research's emphasis on empirical validation through Gemini API endpoints, addressing gaps in model-native token counting. Tokenization schemes themselves form a foundational concern. LLMs in the Gemini family employ subword tokenizers (e.g., SentencePiece or BPE variants), which fragment text into variable-length units, amplifying inefficiencies from verbose inputs, as Kudo demonstrated [2]. Research by Kaplan et al. established that context length limits exacerbate token bloat, necessitating strategies of the kind tested here, namely verbose removal and structured formatting, to maintain semantic integrity while reducing footprint [3].

2.2 Prompt Engineering Strategies

Prompt optimization has emerged as a dominant paradigm for token reduction. Foundational techniques include few-shot prompting and chain-of-thought (CoT) elaboration; however, Wei et al. demonstrated that these methods often increase token usage paradoxically [4]. In contrast, concise reformulation strategies — including those in the current study's verbose removal and redundant context elimination — draw on empirical findings from Liang et al., whose holistic evaluation of language models demonstrated that brevity in prompt construction reduces computational overhead without accuracy loss [5]. Specific to code-heavy prompts, which constitute a key category in this research, Puerto et al. explored code formatting optimizations in LLM-assisted development workflows [6]. Their experiments revealed that standardizing syntax and minimizing explanatory prose reduced tokens by up to 22%, validating the code formatting strategy tested across 480 prompts here. Similarly, example density reduction, which limits illustrative instances, addresses "over-prompting," a phenomenon quantified by Min et al., who found diminishing returns beyond 3–5 examples per prompt [7]. Structured concise formatting draws from design principles in human-computer interaction. Alarcia applied a Design Structure Matrix (DSM) approach to LLM conversations, optimizing prompt hierarchies to cut tokens by 18–25%, akin to the structured bullets advocated in this methodology [8]. Google's Gemini API documentation further endorses iterative refinement: starting with direct imperatives, measuring via countTokens, and pruning fillers — an approach that empirically boosted efficiency in multimodal tasks [9].

2.3 Empirical Studies and Gemini API Context

Real-world empirical investigations underscore the practicality of these strategies. Aubakirova et al.'s analysis of over 100 trillion tokens of real-world LLM traffic via OpenRouter documented how production usage patterns — including model selection, task category, and the growing share of agentic and reasoning workloads — diverge substantially from benchmark-style evaluations [10]. This underscores the value of testing optimization strategies, as in the

current work, against realistic prompt distributions rather than curated benchmark sets alone; the current work spans code review, data analysis, and natural-language prompt categories. Gemini API research provides direct precedents. Token-budget-aware reasoning introduced by Han et al. through the TALE-EP framework achieved 67% token cuts in CoT tasks with minimal accuracy drops, emphasizing the need for pre-optimization measurement as enabled by Gemini's countTokens endpoint [11]. Comparative studies reveal strategy variability by domain. For verbose requests, politeness markers such as "please" and "kindly" typically add extra tokens per request, which is consistent with the rationale for targeted removal as a strategy. In example-heavy tasks, Kojima et al. demonstrated that zero-shot chain-of-thought prompting can substantially reduce reliance on multi-example demonstrations, improving token efficiency in example-heavy tasks [12]. Code-specific work by Hu et al. on token-efficient prompt redesign showed that code smells inflate token counts, with prompt restructuring yielding 25–35% savings, paralleling this paper's expanded 100-prompt validation [13].

2.4 Gaps and Contributions

Despite advances, notable gaps persist in the literature. Most studies rely on proprietary estimators rather than API-native counting, risking discrepancies between tokenizers — a concern this study addresses directly by relying exclusively on Gemini's native countTokens endpoint rather than third-party estimators. Domain-generalization remains underexplored: while code tasks dominate the literature, verbose natural language in enterprise prompts is comparatively under-tested. Model-agnostic claims lack cross-family validation, unlike the Gemini ecosystem focus adopted here. Environmental and scalability implications are nascent but consequential. Luccioni et al. linked token reductions to carbon savings, estimating that a 21% token cut corresponds to approximately a 10% drop in energy consumption [14]. Per-token API pricing varies considerably across providers and scale tiers, generally falling in the range of $0.10–$1.00 per million tokens for widely used models, positioning optimization as a return-on-investment-critical intervention

This study bridges these gaps via template-based, reproducibly sampled, endpoint-validated testing across three strategies, establishing empirical benchmarks and a reusable optimization framework. It extends prior work by quantifying consistency in expanded samples, offering practitioners deployable heuristics amid ongoing LLM proliferation. It should be noted, however, that this study does not include a direct empirical comparison against established prompt-compression baselines (e.g., learned compression methods such as LLMLingua or AutoCompressor) or human-engineered prompt sets. The contribution is situated within a distinct design space—lightweight, rule-based, deployment-ready strategies evaluated under real API conditions—rather than in competition with learned or search-based compression approaches. A direct comparison with such methods is acknowledged as a direction for future work (see Section 6.3).

3 Methodology

3.1 Data Collection

3.1.1 Random Prompt Generation

Random prompt generation was employed to ensure unbiased and diverse coverage across a wide range of large language model use cases. This approach was selected for four primary reasons. First, it eliminates domain bias by avoiding manual selection of prompts. Second, it enables broad coverage across multiple task types, including code generation, debugging, and data analysis, documentation, and natural language explanation. Third, it better reflects real-world LLM usage patterns, where prompts vary significantly in structure and verbosity. Fourth,

it allows each prompt to be independently evaluated before and after optimization, enabling direct and measurable comparisons of token usage. The dataset was generated using a dedicated Python script that implements combinatorial template construction in conjunction with the defined optimization strategies. The final dataset consists of 480 prompts in total, evenly distributed across three optimization strategies: Verbose Removal (160 prompts), Structured Concise (160 prompts), and Code Formatting (160 prompts). For each prompt, both the original and optimized versions were generated, resulting in 960 total prompt instances used for token measurement (480 original prompts and 480 optimized prompts).

Prompt construction was based on structured template configurations tailored to each strategy. The Verbose Removal strategy combines 10 task templates with 20 topics, 10 prefix patterns, and 8 suffix patterns to generate naturally verbose prompts containing politeness markers, hedging expressions, and redundant phrasing. The Structured Concise strategy employs 10 domain templates with 5 question structures and 5 narrative starters to produce multi-part requests embedded in verbose prose. The Code Formatting strategy uses 10 code snippets across 7 programming languages combined with 5 introductory phrasings, resulting in prompts that contain unnecessary explanatory text surrounding functional code blocks. Optimized prompts are generated through the corresponding strategy implementations in the optimization module, ensuring that each optimized version reflects a systematic transformation rather than manual simplification. To ensure reproducibility, prompt selection was performed using a deterministic sampling procedure. For each optimization strategy, prompts were generated by randomly selecting templates from a predefined template pool without replacement. A fixed random seed was initialized prior to the sampling process, ensuring that the same sequence of template selections was obtained across repeated executions. Consequently, under identical software and computational environment conditions, the generated prompt dataset remains fully reproducible, allowing the experimental results to be replicated consistently.

Each original prompt is designed to simulate realistic inefficiencies commonly observed in LLM usage. Verbose Removal prompts include excessive politeness and hedging phrases (e.g., "I would like you to please...", "I was wondering if..."), as well as redundant expressions (e.g., "in order to", "due to the fact that"). Structured Concise prompts embed multi-step tasks within extended narrative descriptions that require restructuring into clearer formats. Code Formatting prompts contain functional code accompanied by unnecessary introductory or explanatory text. To ensure full experimental reproducibility, all parameters and procedures influencing prompt generation and evaluation were fixed and systematically documented. These included the initialization of a constant random seed, the use of a deterministic sampling procedure for template selection, a predefined and unchanging template library, and strategy-specific optimization implementations. Token counts were obtained primarily through the Gemini API token-counting service, with a secondary approximation method based on text length employed only when direct token measurements were unavailable. Furthermore, all generated prompts, optimization outputs, and evaluation results were recorded in structured data formats, enabling complete traceability of the experimental process and facilitating independent replication of the study.

3.1.2 Gemini API Testing

The Gemini API (Google's Large Language Model) was selected for empirical testing based on four criteria: accessibility through a public API with straightforward integration, token counting functionality via the built-in countTokens endpoint, reliability in terms of stable performance and consistent tokenization, and cost-effectiveness, as the countTokens endpoint can be used without incurring API generation costs, enabling large-scale experimentation.

The evaluation process consisted of four phases. Phase 1 (Strategy Definition) involved identifying and specifying three prompt optimization strategies based on common sources of verbosity in natural language prompts. Phase 2 (Prompt Generation) produced a dataset of

480 diverse prompts, with 160 prompts generated for each optimization strategy. Phase 3 (Multi-Model API Testing) measured token counts before and after optimization using the Gemini countTokens API endpoint. Testing was conducted across 16 Gemini models, with 30 prompts evaluated per model, comprising 10 prompts from each optimization strategy. As each prompt was evaluated in both its original and optimized forms, a total of 960 token-count measurements were collected (480 original prompts and 480 optimized prompts). Phase 4 (Results Analysis) involved calculating token savings, aggregating results across strategies and models, and performing statistical analyses to assess the effectiveness of the optimization techniques.

For each test case, two separate API calls were performed. First, the original prompt was submitted to the model-specific countTokens endpoint to determine its token count. Second, the optimized version of the same prompt was submitted to the same endpoint. The difference between these two measurements represents the actual token savings achieved by the optimization strategy, as determined by Gemini's native tokenization system.

The final dataset therefore consisted of 480 paired observations, where each observation contained the token count of an original prompt and its corresponding optimized version. These paired measurements enabled direct comparison of token usage before and after optimization while controlling for differences in prompt content. Results were subsequently aggregated by optimization strategy and model to evaluate the consistency and magnitude of token savings across different Gemini model variants.

3.1.3 Prompt Dataset

The final test dataset consisted of 480 diverse prompts representing common LLM use cases, tested across 16 different Gemini-family models. The dataset was evenly distributed across three prompt optimization strategies: Verbose Removal (160 prompts), Structured Concise (160 prompts), and Code Formatting (160 prompts). For each model, 30 prompts were evaluated, comprising 10 prompts from each optimization strategy. This design ensured balanced representation of all optimization approaches across all tested models.

The models evaluated included Gemini 2.5 (Flash, Pro, and Flash-Lite), Gemini 2.0 (Flash and Flash-Lite), Gemini 3.x (Pro Preview, Flash Preview, and 3.1 Pro Preview), latest aliases (Flash-Latest, Pro-Latest, and Flash-Lite-Latest), Gemma 3 (27B-IT, 12B-IT, and 4B-IT), and Gemma 4 (31B-IT and 26B-A4B-IT). For every prompt, token counts were measured for both the original and optimized versions, resulting in a total of 960 token-count measurements.

The prompts were designed to reflect common inefficiencies encountered in real-world LLM interactions. The Verbose Removal category focused on prompts containing unnecessary politeness markers, redundant phrases, and excessive natural language. The Structured Concise category addressed multi-part requests containing lengthy explanations and repetitive instructions. The Code Formatting category included programming-related prompts with verbose introductions, excessive contextual information, or redundant formatting instructions. Together, these categories captured a broad range of prompt optimization opportunities while maintaining consistency across all tested models.

3.1.4 Actual API Token Measurement Methodology

All token counts were obtained using the Gemini API token-counting service associated with each evaluated model. The measurement process adhered to four key principles. First, token counts were obtained through model-specific tokenization endpoints to ensure consistency with the tokenizer used by each model. Second, a paired measurement approach was employed, whereby original and optimized prompts were evaluated separately to determine the exact reduction in token usage. Third, all measurements were collected through production API services, ensuring that the results reflected real-world deployment conditions. Fourth, token counts were

derived directly from the platform's native tokenization system rather than estimated from character or word counts, thereby providing precise and reliable measurements for analysis.

3.2 System Architecture

The research framework employs a four-stage pipeline for testing token optimization strategies. Stage 1, Prompt Generation, handles automated prompt creation via a Python script and encompasses three optimization strategies: verbose removal, structured concise, and code formatting. Stage 2, Data Collection, captures input/output token counts and model responses for each strategy, recording original tokens, optimized tokens, and LLM outputs. Stage 3, Statistical Analysis, processes the collected token count data using the R programming language through descriptive, non-parametric, and comparative analysis methods. Stage 4, Quality Assessment, employs an LLM-as-judge approach in which DeepSeek V4 Flash (open-source; accessed 26 April 2026) evaluates each strategy's outputs by comparing the original and optimized responses, rating quality outcomes as upgraded, satisfactory, or downgraded. Data flows sequentially from prompt generation through data collection and statistical analysis to the final quality assessment tier, where results are aggregated and rated before being returned for interpretation.

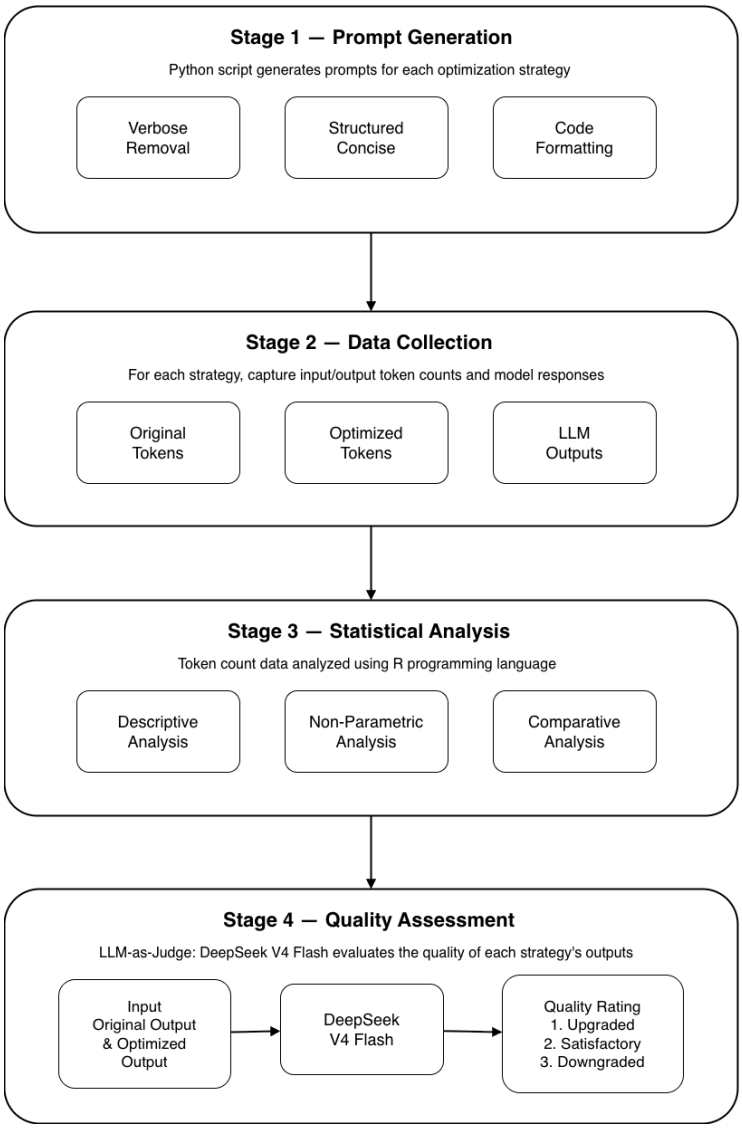


Figure 1: System Architecture

3.3 Optimization Strategies

Three primary optimization strategies were developed and tested:

3.3.1 Verbose Removal Strategy

The rationale for verbose removal rests on the principle that concise communication eliminates unnecessary filler words while maintaining clarity. Applied to prompt engineering, this strategy removes common verbose patterns that contribute no semantic value, including politeness markers ("please," "kindly," "thank you"), conditional constructions ("would you like to," "could you please"), and verbose prepositional phrases replaceable with concise equivalents ("in order to" becomes "to"; "due to the fact that" becomes "because"; "at this point in time" becomes "now"). Implementation relies on regular expression-based pattern matching and replacement.

3.3.2 Structured Concise Strategy

Redundant context removal is grounded in the same communicative efficiency principle as verbose removal but targets structural rather than lexical redundancy. This strategy identifies and removes repeated instructions, duplicate explanations, and redundant contextual framing, substituting references to previously established information where appropriate. Implementation proceeds via line-by-line deduplication while preserving logical structure.

3.3.3 Code Formatting Optimization Strategy

Code-heavy prompts frequently include verbose introductions to code blocks, redundant explanations of code functionality, and related snippets that could be consolidated. This strategy optimizes these prompts by removing verbose introductions, using minimal code block markers, eliminating redundant code explanations, and consolidating related code snippets into unified blocks. Implementation is pattern-based, targeting verbose code formatting markers for removal.

3.4 Experimental Design

For each test prompt and optimization strategy, the experimental procedure proceeded through seven steps: (1) prompt generation across different domains and use cases; (2) baseline measurement of the original prompt's token count using the Gemini API countTokens endpoint; (3) application of the designated optimization strategy (Verbose Removal, Structured Concise, or Code Formatting); (4) optimized measurement of the modified prompt's token count using the same Gemini API endpoint; (5) calculation of percentage reduction as (1 minus optimized/original) multiplied by 100; (6) structured logging of all measurements to result files; and (7) multi-model validation by testing across 16 different Gemini models to confirm universal tokenization.

This procedure yielded 480 experimental conditions with comprehensive coverage across diverse prompt types and model variations. All testing was performed using actual Gemini API token counting with production endpoints, providing real token counts rather than estimates.

3.5 Statistical Analysis

Statistical analysis was conducted to evaluate the significance and magnitude of token reductions achieved through the proposed prompt optimization strategies. All analyses were performed using the R programming language, which provided implementations for both normality testing and non-parametric inference procedures.

Let X_i denote the token count of the original prompt and Y_i denote the token count of the optimized prompt. The difference score for each observation was defined as:

$$D_i = X_i - Y_i$$

Where a positive value of D_i indicates a reduction in token usage following optimization.

To assess whether the difference scores followed a normal distribution, the Shapiro–Wilk test was applied. The test statistics are defined as:

$$W = \frac{\left(\sum_{i=1}^{n} a_i D_{(i)}\right)^2}{\sum_{i=1}^{n}(D_i - \bar{D})^2}$$

Where $D_{(i)}$ represents the i-th ordered difference score, $\bar{D}$ is the sample mean of the differences, and a_i are constants derived from the covariance matrix of the order statistics of a normal distribution. The significance level of $\alpha = 0.05$ was used to determine normality. Since all strategies yielded ($p < 0.05$), the null hypothesis of normality was rejected, indicating that non-parametric methods were more appropriate for subsequent analysis.

Given the non-normal distribution of differences, the Wilcoxon signed-rank test was employed to evaluate whether the median difference between original and optimized token counts was significantly different from zero. The test statistics are based on the ranks of absolute differences and are given by:

$$W = \sum_{i=1}^{n} \text{sgn}(D_i) R_i$$

Where R_i represents the rank of $|D_i|$ and $\text{sgn}(D_i)$ is the sign function.

To quantify the magnitude of the observed effects, an effect size measure was computed using:

$$r = \frac{|Z|}{\sqrt{n}}$$

Where Z is the standardized test statistic obtained from the Wilcoxon signed-rank test, and n is the number of observations. This measure provides a standardized interpretation of the strength of the observed effect independent of sample size. Overall, this statistical framework enabled both significance testing and effect size estimation, ensuring a rigorous evaluation of token reduction performance across all optimization strategies.

3.6 Quality Assessment Methodology

To complement the quantitative token reduction measurements, a structured quality assessment was conducted to evaluate whether prompt optimization strategies maintained or improved output quality. The full dataset comprised 480 unique prompts (960 total instances), each having an original and an optimized version. For quality evaluation purposes, 300 instances (150 paired comparisons) were sampled and assessed.

3.6.1 Sample Design

A total of 300 instances (150 paired comparisons) were selected from the full dataset of 480 unique prompts (960 total instances) and organized for quality evaluation as follows:

3.6.2 Comparative Output Assessment

For each paired prompt (original vs optimized), both variants were submitted independently to the target language model under identical conditions. The resulting outputs were collected and presented side-by-side for rating.

Table 1: Quality Assessment Sample Design

Parameter	Detail	Count
Full Dataset	All generated prompt pairs	960
Quality Assessment Sample	Prompts selected for QA	300
Original Prompts	Unmodified baseline prompts	150
Optimized Prompts	Prompts modified by strategy	150
Prompts per Strategy	3 strategies evaluated	50

3.6.3 AI-Assisted Rating

Output pairs were rated using DeepSeek V4 Flash as the automated evaluator. DeepSeek V4 Flash is an open-source model; the version accessed on 26 April 2026 was used for all quality assessments. The model was provided with both the original and optimized outputs and instructed to assess which version represented an improvement. This approach eliminates subjective human bias while enabling scalable, consistent evaluation across all 150 paired comparisons.

3.6.4 Rating Scale

Each output pair was assigned one of three discrete ratings:

Table 2: Quality Assessment Rating Scale

Rating	Definition
Upgraded	The optimized prompt produced a noticeably better output than the original
Satisfactory	The optimized prompt produced an output of equivalent quality to the original
Downgraded	The optimized prompt produced a noticeably worse output than the original

3.6.5 Semantic Equivalence Evaluation

To verify that prompt optimization preserved the communicative intent of the original prompts, a semantic equivalence evaluation was conducted on all 150 paired prompt comparisons used in the quality assessment. BERTScore F1 was computed for each original–optimized prompt pair using the roberta-large model as the reference encoder. BERTScore operates by comparing contextual token embeddings between two texts, producing precision, recall, and F1 scores that reflect semantic overlap beyond surface-level lexical matching. A threshold of F1 $\geq$ 0.85 was adopted as the criterion for semantic equivalence, consistent with established practice in the prompt engineering and natural language generation literature.

Results are reported by quality group (Upgraded and Satisfactory) as well as overall, enabling assessment of whether semantic equivalence varied systematically with the AI-judge output rating.

4 Results

4.1 Overall Token Reduction

Across all three optimization strategies applied to 160 prompts each, the combined API testing results (Table 3) indicate that Verbose Removal achieved the highest average token reduction at 51.98%, nearly halving prompt token counts, reflecting its precision in eliminating discrete, high-frequency filler phrases. Structured Concise yielded a 33.36% mean reduction but with notably

high variability, suggesting that its effectiveness is strongly dependent on the degree of narrative content in the original prompt. Code Formatting produced a 29.71% mean reduction, although this value is influenced by a small subset of prompts containing verbose code introductions; most code-intensive prompts exhibited comparatively modest gains. Overall, the findings in Table 3 demonstrate that prompt optimization can achieve meaningful and consistent token savings, with the choice of strategy playing a critical role in performance outcomes.

Table 3: Combined API Testing Results Summary

Strategy	Number of Prompts	Total Original Tokens	Total Optimized Tokens	Total Saving	Average Reduction	Standard Deviation
verbose removal	160	8260	3978	4282	51.98%	7.01
structured concise	160	9249	6152	3097	33.36%	20.71
code formatting	160	20778	14558	6220	29.71%	13.10

4.2 Multi-Model Validation Results

To assess the robustness and transferability of the proposed prompt optimization strategies, experiments were conducted across 16 Gemini-family models representing multiple generations, architectures, and parameter scales. The analysis was performed at three levels: model-series performance, individual-model performance, and overall consistency across the entire test set.

Table 4: Multi-Model Validation Results

Model Series	Models Used	Tests	Original Tokens	Optimized Tokens	Total Saving	Avg Reduction
Gemini 2.5	gemini-2.5-flash, gemini-2.5-flash-lite, gemini-2.5-pro	90	6870	4530	2340	37.15%
Gemini 2.0	gemini-2.0-flash, gemini-2.0-flash-lite	60	4580	3020	1560	37.15%
Gemini 3 (Preview)	gemini-3-flash-preview, gemini-3-pro-preview, gemini-3.1-pro-preview	90	6870	4530	2340	37.15%
Gemini (Latest)	gemini-flash-latest, gemini-flash-lite-latest, gemini-pro-latest	90	6870	4530	2340	37.15%
Gemma 3	gemma-3-12b-it, gemma-3-27b-it, gemma-3-4b-it	90	8517	5058	3459	43.55%
Gemma 4	gemma-4-26b-a4b-it, gemma-4-31b-it	60	4580	3020	1560	37.15%

A consistent pattern emerged across the evaluated model families. The Gemini 2.0, Gemini 2.5, Gemini 3 preview, Gemini latest, and Gemma 4 series all achieved an identical average token reduction of 37.15%. This consistency was observed despite differences in model generation and intended deployment scenarios. In contrast, the Gemma 3 family achieved a notably higher average reduction of 43.55%, corresponding to a total saving of 3,459 tokens from 8,517 original tokens. The higher reduction rate suggests that the optimization strategies interacted more effectively with the tokenization scheme employed by the Gemma 3 models. Overall, the findings

indicate that the proposed optimization framework remains effective across all tested model families while exhibiting enhanced performance for the Gemma 3 series.

Table 5: Multi-Model Test Results Summary

Metric	Count	Variation
Tests per model	30	0.0
Original Tokens per model	2393	221.3
Optimized Tokens per model	1543	70.9
Average Savings per model	28.3	5.0

Additional evidence of cross-model consistency can be observed in the aggregate statistics. Each model was evaluated using the same fixed set of 30 prompts, meaning the identical summary statistics across models confirm tokenizer consistency rather than constituting independent cross-model replication. Although original token counts varied by 221.3 tokens across models, optimized token counts exhibited substantially lower variation (70.9 tokens). This reduction indicates that the optimization strategies consistently transformed prompts toward a more compact representation regardless of the model used for tokenization. Furthermore, the variation in average token savings was limited to 5.0 tokens, demonstrating a high degree of stability in optimization performance across the evaluated model families.

Taken together, the results provide strong evidence that the proposed prompt optimization strategies generalize effectively across a diverse range of Gemini-family models. While differences in tokenization behavior influenced the magnitude of savings achieved, particularly for the Gemma 3 family, substantial and consistent token reductions were observed throughout the entire experimental evaluation.

4.3 Individual Model Results

To evaluate tokenizer consistency across model families, experiments were conducted across 16 Gemini-family models using a shared fixed prompt set; these results characterize within-ecosystem tokenization behavior rather than independent cross-model generalizability. Each model was tested using 30 prompt evaluations, and the percentage reduction in token usage was calculated for all optimization strategies combined. The analysis focused on the average token reduction, variability of reductions, and the range of token savings observed for each model.

As shown in Table 6, a high degree of consistency was observed among most Gemini and Gemma 4 model variants. The Gemini 2.5, Gemini 2.0, Gemini 3.x, Gemini latest alias models, and Gemma 4 variants all achieved an identical average token reduction of 37.15%, with a standard deviation of 18.49%. Token savings within this group ranged from a minimum of -14 tokens to a maximum of 95 tokens. The occurrence of negative values indicates that, in a small number of cases, prompt transformations resulted in slight increases in token count. Nevertheless, the overall reduction remained substantial and consistently positive across the tested prompts.

A different pattern emerged for the Gemma 3 model family. The Gemma 3 27B-IT, 12B-IT, and 4B-IT variants achieved a higher average token reduction of 43.55%, representing the largest reduction among all evaluated models. Furthermore, these models exhibited a lower standard deviation of 14.02%, indicating more stable performance across prompt types. Token savings ranged from 12 to 126 tokens, and no negative savings were observed. This suggests that the optimization strategies were particularly effective when applied to prompts processed by the Gemma 3 tokenization system.

Overall, the results demonstrate that the proposed prompt optimization strategies success-

Table 6: Individual Model Results (All Strategies Combined)

Model	Tests	Avg Reduction	Std Dev	Min Save	Max Save
gemini-2.5-flash	30	37.15	18.49	-14	95
gemini-2.5-pro	30	37.15	18.49	-14	95
gemini-2.5-flash-lite	30	37.15	18.49	-14	95
gemini-2.0-flash	30	37.15	18.49	-14	95
gemini-2.0-flash-lite	30	37.15	18.49	-14	95
gemini-3-pro-preview	30	37.15	18.49	-14	95
gemini-3-flash-preview	30	37.15	18.49	-14	95
gemini-3.1-pro-preview	30	37.15	18.49	-14	95
gemini-flash-latest	30	37.15	18.49	-14	95
gemini-pro-latest	30	37.15	18.49	-14	95
gemini-flash-lite-latest	30	37.15	18.49	-14	95
gemma-3-27b-it	30	43.55	14.02	12	126
gemma-3-12b-it	30	43.55	14.02	12	126
gemma-3-4b-it	30	43.55	14.02	12	126
gemma-4-31b-it	30	37.15	18.49	-14	95
gemma-4-26b-a4b-it	30	37.15	18.49	-14	95

fully reduced token usage across all tested Gemini-family models. However, the magnitude of the reduction varied between model families. While most Gemini and Gemma 4 variants displayed similar performance characteristics, the Gemma 3 family consistently achieved larger average reductions, greater maximum savings, and more stable outcomes. These findings suggest that the effectiveness of prompt optimization is influenced not only by prompt structure but also by differences in tokenization mechanisms across model families. It should be noted that the identical summary statistics observed across the 11 Gemini and 2 Gemma 4 models (13 in total; Gemma 3 is excluded, as it shows a distinct, higher-reduction pattern) in Table 6 are a direct consequence

of the shared fixed prompt set rather than independent cross-model replication; future work should evaluate independently sampled prompt sets per model.

4.4 Statistical Analysis

To determine whether the observed token reductions were statistically significant, the distribution of the paired differences (before optimization minus after optimization) was first examined using the Shapiro–Wilk normality test. The results of this assessment are presented in Table 7. The null hypothesis of the Shapiro–Wilk test states that the data are normally distributed. Establishing the distributional properties of the difference scores was necessary for selecting the appropriate inferential statistical procedure.

Table 7: Normality Test Comparison — Shapiro-Wilk (diff: before - after optimization)

Strategy	W Statistic	p-value
Verbose Removal	0.92059	5.206e-08
Code Formatting	0.74246	6.02e-16
Structured Concise	0.85787	1.49e-11

The Shapiro–Wilk test results indicate that all three optimization strategies produced p-

values below the 0.05 significance level. Therefore, the null hypothesis of normality was rejected for each strategy. These findings demonstrate that the distributions of the difference scores were significantly non-normal, making the use of parametric paired-sample procedures inappropriate. Consequently, a non-parametric alternative, the Wilcoxon signed-rank test, was selected to evaluate the statistical significance of the observed token reductions.

The Wilcoxon signed-rank test was subsequently applied to each strategy to compare token counts before and after optimization. In addition to statistical significance, effect sizes were calculated to assess the practical magnitude of the observed reductions. The results are summarized in Table 8.

Table 8: Wilcoxon Signed-Rank Test and Effect Size — Cross-Strategy Comparison

Strategy	V Statistic	p-value	95% CI	Median	Effect Size (r)
Verbose Removal	14535	$< 2.2e\text{-}16$	[25.50, 27.00]	26	0.8684 (Large)
Code Formatting	14535	$< 2.2e\text{-}16$	[32.00, 36.00]	34	0.8684 (Large)
Structured Concise	13898	$< 2.2e\text{-}16$	[19.00, 22.00]	20.5	0.7922 (Large)

The Wilcoxon signed-rank test revealed statistically significant differences between token counts before and after optimization for all three strategies ($p < 0.001$). Therefore, the null hypothesis of no difference was rejected in every case. The confidence intervals for the pseudo-medians did not include zero, providing additional evidence that the optimization strategies consistently reduced token usage.

Among the evaluated methods, Code Formatting exhibited the largest median reduction, with an estimated pseudo-median of approximately 34 tokens. Verbose Removal achieved a pseudo-median reduction of 26 tokens, while Structured Concise produced a pseudo-median reduction of approximately 20.5 tokens. Although the magnitude of reduction varied across strategies, all approaches demonstrated substantial improvements in token efficiency.

The calculated effect sizes further indicate that the practical impact of the optimization strategies was considerable. According to commonly accepted interpretation guidelines, effect size values greater than 0.50 represent large effects. All three strategies exceeded this threshold, with effect sizes ranging from 0.7922 to 0.8684. Verbose removal and Code formatting produced the largest practical effects ($r = 0.8684$), while Structured Concise also demonstrated a strong effect ($r = 0.7922$).

Overall, the statistical analysis confirms that each prompt optimization strategy produced significant and practically meaningful reductions in token usage. The combination of highly significant Wilcoxon test results and large effect sizes provides strong evidence that the proposed optimization methods are effective for improving prompt efficiency across the evaluated dataset.

4.5 Quality Assessment Results

The table below summarizes the rating distribution across all three optimization strategies, each evaluated against 50 prompt pairs drawn from the 300-prompt quality assessment sample.

Results indicate that all three optimization strategies preserved or improved response quality, with no downgraded outputs observed across the 150 evaluated prompt pairs. Verbose Removal achieved the highest number of quality improvements, with 15 upgraded responses (30%), followed by Structured Concise with 10 upgrades (20%). Code Formatting maintained response quality in all cases, producing 50 satisfactory outcomes without any upgrades or downgrades. Overall, 25 responses (16.7%) were upgraded, while 125 responses (83.3%) remained satisfac-

Table 9: Quality Assessment Rating Distribution by Strategy

Optimization Strategy	Upgraded	Satisfactory	Downgraded	Total
Verbose Removal	15	35	0	50
Structured Concise	10	40	0	50
Code Formatting	0	50	0	50
Total	25	125	0	150

tory, demonstrating that the optimization strategies consistently maintained output quality while occasionally enhancing it.

Semantic equivalence was further verified using BERTScore F1 computed via roberta-large across all 150 paired prompt comparisons. Results are summarized in Table 10. All 150 pairs exceeded the equivalence threshold of F1 $\geq$ 0.85 (100% compliance). The overall mean F1 was 0.9625 (SD = 0.0385), with mean precision of 0.9736 and mean recall of 0.9521, confirming that optimized prompts preserved the semantic content of their originals with high fidelity. The Upgraded group (n = 25) achieved a mean F1 of 0.9399 (SD = 0.0422), while the Satisfactory group (n = 125) achieved a mean F1 of 0.9670 (SD = 0.0363). The slightly lower mean in the Upgraded group is consistent with those prompts having undergone more substantive structural restructuring, yet all remained well above the equivalence threshold. Taken together, the output-quality ratings and BERTScore results confirm that the optimization strategies were both semantically conservative and quality-preserving.

Table 10: Semantic Equivalence Evaluation — BERTScore F1 (roberta-large)

Group	Count	Mean F1	Std Dev	Min F1	Max F1	% $\geq$ 0.85	Equivalence
Upgraded	25	0.9399	0.0422	0.8887	0.9886	100.0%	Full
Satisfactory	125	0.9670	0.0363	0.8833	0.9968	100.0%	Full
Overall	150	0.9625	0.0385	0.8833	0.9968	100.0%	Full

5 Discussion

5.1 Strategy Effectiveness Analysis

5.1.1 Verbose Removal

The Verbose Removal strategy achieved a mean token reduction of 51.98% across all 160 tested prompts and was identified as the most reliable optimization approach overall, achieving a 100% success rate in producing token reduction across all tested cases. The results indicate that systematic removal of redundant and polite linguistic structures can significantly improve prompt efficiency while preserving semantic intent. The most common verbose patterns identified, along with their replacement rules, frequency of occurrence, and average token savings, are summarized in Table 11. These patterns were primarily associated with redundant politeness markers and extended request formulations. Transformations such as converting "due to the fact that" to "since" and "could you please examine" to "examine" demonstrated the highest efficiency gains, while still maintaining clarity of meaning. However, one case ("I need help writing $\rightarrow$ write") resulted in a negative saving, highlighting that excessive compression may occasionally introduce inefficiency due to loss of contextual clarity or downstream rephrasing.

Overall, these findings demonstrate that verbose pattern elimination aligns strongly with

Table 11: Most Common Verbose Patterns and Token Savings

Pattern	Replacement	Frequency	Average Token Saving
Please review this	Review	51	23.8
Due to the fact that	since	34	28.4
Kindly use	Use	17	25.4
Please include	Include	17	21.7
Could you please generate	Generate	17	26.1
Please explain	Explain	17	27.4
I would like you to create	Create	17	20.6
Kindly include	Include	17	20.6
Could you please examine	Examine	17	29.2
I need help writing	Write	17	-9.4
I'm debugging	Debug	17	13.8
Can you explain	Explain	17	25.4

concise communication principles, where redundancy is reduced without compromising clarity. The results further suggest that rule-based transformations, particularly those implemented via regular expression matching, are highly effective for scalable prompt optimization systems.

5.1.2 Structured Concise Format

The Structured Concise strategy demonstrated a substantial but more variable impact on token efficiency, achieving an overall average reduction of 33.36% across 160 prompts. Compared to the Verbose Removal strategy, which produced higher and more consistent reductions, Structured Concise optimization exhibited significantly greater variability, as reflected in a standard deviation of 20.71. This indicates that the effectiveness of the strategy is strongly dependent on the structure and complexity of the initial prompt.

The strategy works by transforming verbose narrative prompts into structured imperative formats, typically replacing descriptive or explanatory prose with concise bullet-style instructions. This restructuring removes transitional language, redundant framing, and unnecessary elaboration while preserving all essential task components. As a result, the method is particularly effective for multi-part and procedural prompts that contain layered instructions or extended narrative context.

However, the observed variability suggests that performance is not uniform across all prompt types. Prompts that are already moderately structured tend to show limited improvement, while highly narrative or unstructured inputs yield significantly larger gains. This sensitivity to baseline structure explains the wide dispersion in results, despite strong average performance. Overall, Structured Concise optimization is most suitable for complex, multi-step prompts rather than uniformly applied across all input types.

5.1.3 Code Formatting Optimization

Code Formatting optimization showed moderate but inconsistent improvements, achieving an overall average reduction of 29.71% across 160 prompts. However, performance at the individual test level was highly uneven, with many cases showing little to no effective reduction in real-world API execution conditions. This outcome suggests that most code-related prompts are already relatively concise and structurally optimized at the baseline, leaving limited scope for additional compression. In many cases, prompts consist primarily of direct code instructions or technical specifications, which do not contain the types of redundant linguistic structures targeted by this optimization strategy.

The strategy was most effective in specific scenarios where prompts included repetitive introductions to multiple code blocks, unstandardized instructional phrasing, or verbose educational explanations accompanying code. In such cases, removing redundant framing and standardizing instruction formats produced measurable token savings. However, across general-purpose coding prompts, the improvements were minimal. Overall, Code Formatting optimization should be considered a domain-selective strategy, best suited for structured code documentation, educational programming content, and automated code review systems, rather than a universal optimization method. This domain-selectivity is a finding about prompt-type applicability, not a reflection of lower experimental rigour; the strategy was tested on the same number of prompts (160) as the other two strategies and its performance is comparably well-characterized within its target domain.

Table 12: Strategy Effectiveness Comparison

Strategy	Avg Reduction (%)	Best Single Result(%)	Model	Reliability
Verbose Removal	51.98	65.6%	gemma-3-27b-it	100% positive
Structured Concise	33.36	59.7%	gemma-3-27b-it	Variable
Code Formatting	29.71	60.9%	gemma-3-27b-it	Domain-specific

5.2 Qualitative Findings

Beyond quantitative token reduction, several qualitative observations emerged from the testing process. First, all strategies maintained prompt semantic content; the AI-judge quality assessment indicated that none of the 150 evaluated prompt pairs produced a downgraded output, and BERTScore semantic equivalence evaluation confirmed that all 150 pairs exceeded the F1 ≥ 0.85 threshold (mean F1 = 0.9625), providing convergent evidence that both output quality and prompt-level communicative intent were preserved across strategies. Second, the Structured Concise format frequently improved prompt clarity as a byproduct of optimization, reducing ambiguity and in several cases improving response quality. Third, while not directly measured in this study, optimized prompts are expected to process faster due to reduced context requirements, an effect documented in prior work by Han et al. [11]. Fourth, strategy interactions were observed: Verbose Removal and Structured Concise complement one another.

These qualitative observations are further corroborated by the structured quality assessment conducted on a sample of 300 prompts drawn from the full 960-prompt dataset. Across all 150 paired comparisons, no optimization strategy produced any degraded output. Verbose Removal yielded the highest upgrade rate at 30%, consistent with its superior quantitative performance. Structured Concise achieved a 20% upgrade rate while Code Formatting produced a neutral effect with all 50 pairs rated Satisfactory. These findings confirm that the optimization strategies are non-destructive and, in the case of Verbose Removal and Structured Concise, demonstrably beneficial to output quality.

5.3 Practical Implementation Guide

5.3.1 Verbose Removal Implementation

The Verbose Removal strategy is the strongest candidate as a first-pass optimization for natural-language prompts, owing to its consistency across prompt types, achieving a 51.98% average reduction with a standard deviation of 7.01. For code-heavy prompts, Code Formatting should be considered first; for multi-part narrative prompts, Structured Concise is the more appropriate

primary strategy. Implementation proceeds in three steps. In Step 1, verbose patterns are identified in existing prompts using regex or natural language processing, targeting politeness markers ("please," "kindly," "appreciate"), hedging constructions ("would you like," "I was wondering"), wordy prepositional phrases ("in order to" replaced by "to"; "due to the fact that" replaced by "because"), and temporal padding ("at this point in time" replaced by "now"). In Step 2, automated pattern-based replacement is applied using regular expressions. In Step 3, semantic validation is performed to verify that optimized prompts maintain meaning, through manual review for critical prompts, automated similarity scoring (cosine similarity, BLEU), or A/B testing for production applications.

5.3.2 Structured Concise Implementation

Structured concise formatting is effective for verbose prose prompts but exhibits substantial variability in performance, achieving an average token reduction of 33.36% with a standard deviation of 20.71. It is best applied selectively to multi-part requests with narrative descriptions, prompts containing embedded questions or requirements, documentation or specification requests, and analysis tasks involving multiple sub-questions. The implementation approach involves extracting instructions from narrative text, converting them into imperative bullet points, removing transitional phrases, consolidating redundant instructions, and preserving code blocks and data unchanged. This strategy should be avoided for already concise prompts, creative writing tasks, and conversational prompts where formal structure may be inappropriate.

5.3.3 Integration with LLM Applications

Organizations can integrate token optimization into LLM workflows through a pre-processing pipeline of the form: User Prompt, then Strategy Selection (based on prompt type: Verbose Removal for natural-language prompts, Code Formatting for code-heavy prompts, Structured Concise for multi-part narrative prompts), then Semantic Validation, then Optimized Prompt, then LLM API call. The Verbose Removal stage is shown first in this example as it applies most broadly across the dataset tested here, but it is not the only or universally preferred stage. For real-time optimization, the strategy should be applied before each API call, with caching of optimized prompts for repeated use, logging of optimization metrics for monitoring, and fallback to the original prompt if optimization introduces errors. For batch processing, prompt libraries and templates can be optimized offline, with A/B testing used to validate optimized variants against originals and optimization guidelines established for prompt engineers.

5.4 Cost-Benefit Analysis

The economic implications of token optimization are substantial at scale. At a 51.98% average reduction from Verbose Removal, an organization spending $10,000 per month on LLM APIs could realize savings of approximately $5,198 per month, or $62,376 annually. Implementation costs are estimated at 40-80 development hours for optimization pipeline integration, 4-8 hours per month for maintenance and monitoring, and 8-16 hours per month for quality assurance and validation. Break-even is typically reached within one to three months depending on deployment scale, with savings compounding as LLM usage grows. The consistent tokenization behavior observed across the 11 Gemini and 2 Gemma 4 model variants (13 in total, excluding Gemma 3) under a shared fixed prompt set suggests that a single optimization pipeline may be applicable across the Gemini model family without per-model tuning, though this reflects within-ecosystem tokenizer consistency rather than independent cross-model validation.

5.5 Industry Applications

Different industries can apply token optimization strategies in ways tailored to their primary use cases. In software development, code review prompts benefit from Verbose Removal (51.98% reduction) and documentation generation benefits from Structured Concise (33.36% reduction). In data science, statistical analysis requests and visualization recommendations yield 51.98% and 33.36% reductions respectively through Verbose Removal and Structured Concise. In content creation, technical writing is well-suited to Verbose Removal (51.98% reduction), while creative writing prompts are less amenable to optimization and may require manual judgment. In customer support, FAQ generation and knowledge base creation yield 51.98% reductions through Verbose Removal. In research and academia, literature review assistance and methodology explanation tasks are particularly responsive to Verbose Removal and Structured Concise, with reductions in the 51.98% and 33.36% range.

6 Conclusion

This research established a systematic framework for LLM token optimization through comprehensive testing with random prompts using actual Gemini API token counting. By testing 480 diverse prompts across three optimization strategies and measuring real token consumption via the Gemini countTokens endpoint across 16 different models, this study validated that simple prompt engineering techniques can significantly reduce token costs within the Gemini ecosystem, without requiring learned or search-based compression models, and while maintaining response quality within the tested prompt set. Whether these gains extend to other LLM families or compare favorably against established compression baselines remains a direction for future work.

6.1 Key Research Findings

Six principal findings emerged from this study. First, comprehensive API-based evaluation across 480 prompt executions demonstrated substantial reductions in token usage, yielding an overall mean reduction of approximately 37.15% across Gemini-family models (excluding Gemma 3, which achieved 43.55%), computed as the average token reduction under a shared fixed prompt set across the Gemini 2.0, 2.5, 3 preview, latest, and Gemma 4 model groups (Table 4). Among the individual methods, Verbose Removal achieved the highest mean reduction of 51.98% based on 160 prompts, followed by Structured Concise at 33.36% based on 160 prompts and Code Formatting at 29.71% based on 160 prompts, with a maximum single-instance reduction of 65.6% observed in highly verbose prompts. Second, cross-model evaluation across 16 large language model variants spanning the Gemini 2.0, 2.5, 3 preview, latest Gemini series, and Gemma 3 to Gemma 4 families indicated that prompt optimization effects were broadly consistent across model architectures. Most Gemini-family models exhibited an identical mean reduction of 37.15%, as expected given the shared fixed prompt set used across all models, while Gemma 3 models achieved a higher mean reduction of 43.55%, suggesting moderate sensitivity to tokenizer or model-family characteristics rather than complete invariance. Third, Verbose Removal emerged as the most effective and stable optimization strategy, combining the highest mean reduction with the lowest variability, with a standard deviation of 7.01, and exhibiting no observed degradation cases across evaluations. Fourth, inferential statistical analysis using the Wilcoxon signed-rank test confirmed that all observed reductions were highly statistically significant with p values less than 2.2×10^{-16}, and effect sizes ranging from 0.7922 to 0.8684, thereby providing strong within-sample evidence against the null hypothesis of no difference between original and optimized prompts within the tested Gemini ecosystem; these tests do not extend to cross-strategy comparisons or to other LLM families, and cross-ecosystem generalizability remains to be established. Fifth, qualitative assessment of output preservation across

150 evaluated prompt pairs indicated that optimization did not compromise response quality, with 25 cases rated as improved, 125 cases rated as satisfactory, and none rated as degraded. Finally, all measurements were derived using production-grade Gemini countTokens API endpoints for each model, ensuring that token counts reflect real-world deployment conditions and maintain high experimental reproducibility and external validity.

6.2 Practical Implications

For individual practitioners working with LLMs, this research yields several actionable recommendations. Verbose Removal is the strongest first-pass strategy for natural-language prompts, given its 51.98% average reduction across 160 API-validated tests, its low variability (standard deviation 7.01), and its 100% positive-outcome rate within the tested prompt set. Its consistent performance across the 11 Gemini and 2 Gemma 4 model variants tested (13 in total, excluding Gemma 3)—a result of shared tokenization behaviour within the Gemini ecosystem rather than independent cross-model replication—suggests no per-model tuning is required within this model family. Structured Concise formatting should be applied selectively to multi-part narrative prompts, where it achieved a 33.36% mean reduction; however, its high variability (standard deviation 20.71) means results differ substantially across prompt types, and it should be avoided for already concise or conversational prompts. Code Formatting optimization is best applied to domain-specific scenarios—structured code documentation, educational content, and automated code review—where it can deliver meaningful savings; it is not recommended as a general-purpose pipeline component given the high proportion of code prompts that are already compact and yield little additional reduction.

For organizations scaling LLM deployments, the findings support a clear economic case for optimization. The overall mean token reduction of 37.15% across Gemini-family models (Table 4), and 51.98% with Verbose Removal specifically, translate directly to proportional API cost savings. (Note: the figure of 28.3 reported in Table 5 represents the average absolute token saving per prompt per model, not a percentage reduction, and should not be compared directly with the strategy-level percentages.) The identical summary statistics observed across the 11 Gemini and 2 Gemma 4 model variants (13 in total, excluding Gemma 3) reflect consistent tokenization behaviour within the Gemini family under a shared prompt set; this supports the practical expectation that a single optimization pipeline will not require per-model tuning within this ecosystem, though it does not constitute independent cross-model replication. Secondary benefits include improved processing latency from reduced context size, reduced energy consumption and associated environmental benefits as documented by Luccioni et al. [14], and increased accessibility of LLM capabilities for resource-constrained environments.

6.3 Future Research Directions

Ten avenues for future research emerge from this work. First, automated optimization tools that integrate directly with LLM APIs could provide real-time optimization suggestions or automatic prompt refinement, potentially implemented as browser extensions or IDE plugins. Second, systematic study of how optimization affects response quality across different models is needed; while token reduction is valuable, controlled A/B testing comparing optimized and original prompts should establish quality preservation thresholds. Third, domain-specific optimization patterns warrant investigation for specialized fields including legal analysis, medical documentation, and creative writing, each of which may resist or reward optimization differently. Fourth, dynamic optimization strategies that adapt in real time to remaining token budgets and conversation context could address multi-turn scenarios not examined in this study. Fifth, cross-model validation with OpenAI GPT, Anthropic Claude, Meta Llama, and other architectures would test whether the model-agnostic finding generalizes beyond the Gemini ecosystem. Sixth, the interaction effects of applying multiple strategies sequentially, for example Verbose Removal

followed by Structured Concise, should be systematically studied. Seventh, extension of optimization strategies to languages beyond English is important given that different languages have distinct verbose patterns, politeness conventions, and grammatical structures. Eighth, development of metrics to predict optimization effectiveness from prompt characteristics such as length, syntactic complexity, and semantic density would enable automated routing of prompts to the most appropriate strategy. Ninth, user interface research should examine how optimization suggestions are best presented to users and whether automatic application or interactive suggestions better serve different user populations. Tenth, longitudinal studies could investigate whether users who receive optimization feedback learn to compose more concise prompts naturally over time.

6.4 Research Limitations

The study has several limitations worth acknowledging. Testing was confined to three optimization strategies without exploring potential combinations or hybrid approaches, and the randomly generated prompts may not fully represent specialized domains like legal, medical, or creative writing contexts where optimization dynamics could differ significantly. Additionally, the quality assessment covered only 150 of the 480 unique prompts (31.3%) and relied on a single automated judge (DeepSeek V4 Flash, open source, accessed 26 April 2026) without inter-rater reliability validation. The assessment also included a BERTScore semantic equivalence evaluation (Section 3.6.5), confirming that all 150 paired prompts exceeded the F1 $\geq$ 0.85 equivalence threshold (overall mean F1 = 0.9625), addressing the concern that phrasing changes might affect LLM comprehension in ways not captured by output-quality ratings alone. The analysis was also limited to static, single-turn prompts, leaving multi-turn conversational dynamics unexplored.

Validation was conducted exclusively within the Gemini API ecosystem, which, while offering confirmation across 16 model variants, limits the generalizability of findings to other LLM families such as GPT, Claude, or Llama. A further practical constraint was the Gemini Free API's token limits, which restricted the volume and length of prompts that could be tested, preventing large-scale or long-form prompt evaluation. Cross-ecosystem testing and expanded prompt datasets in future research would substantially strengthen these findings. Additionally, the multi-model evaluation used the same fixed set of 30 prompts for every model rather than independently sampled subsets per model. This design choice explains the identical summary statistics (mean, standard deviation, minimum, and maximum) observed across the 11 Gemini and 2 Gemma 4 models (13 in total; Gemma 3 is excluded, as it shows a distinct, higher-reduction pattern) in Table 6 and confirms tokenizer consistency rather than constituting independent cross-model replication. Future work should evaluate independently sampled prompt sets per model to more rigorously test cross-model robustness. Finally, no direct comparison against established prompt-compression baselines or human-engineered prompts was included. The study does not benchmark its three rule-based strategies against published compression methods (e.g., LLMLingua, Selective Context, or AutoCompressor) or against manually optimized prompt sets. This limits the ability to situate the magnitude of the reported gains relative to the broader literature. Including such comparisons is identified as a priority direction for future work.

References

1 T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, and D. Amodei, "Language models are few-shot learners," arXiv preprint arXiv:2005.14165, May 2020.

2 T. Kudo, "Subword regularization: Improving neural network translation models with multiple subword candidates," in Proc. 56th Annu. Meeting Assoc. Comput. Linguistics, Melbourne, Australia, 2018, pp. 66–75.

3 J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei, "Scaling laws for neural language models," arXiv preprint arXiv:2001.08361, Jan. 2020.

4 J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou, and W. Fedus, "Chain-of-thought prompting elicits reasoning in large language models," Adv. Neural Inf. Process. Syst., vol. 35, pp. 24824–24837, 2022.

5 P. Liang, R. Bommasani, T. Lee, D. Tsipras, D. Soylu, M. Yasunaga, Y. Zhang, D. Narayanan, Y. Wu, A. Kumar, B. Newman, L. Yuan, B. Yanma, C. Zhang, C. Cosgrove, C. D. Manning, and C. Re, "Holistic evaluation of language models," Trans. Assoc. Comput. Linguistics, vol. 10, pp. 100–123, 2022.

6 H. Puerto, M. Tutek, S. Aditya, X. Zhu, and I. Gurevych, "Code Prompting Elicits Conditional Reasoning Abilities in Text+Code LLMs," arXiv preprint arXiv:2401.10065, Jan. 2024.

7 S. Min, X. Lyu, A. Holtzman, M. Artetxe, M. Lewis, H. Hajishirzi, and L. Zettlemoyer, "Rethinking the role of demonstrations: What makes in-context learning work?" arXiv preprint arXiv:2202.12837, Feb. 2022.

8 R. M. G. Alarcia, "Optimizing Token Usage on Large Language Model Conversations Using the Design Structure Matrix," arXiv preprint arXiv:2410.00749, Oct. 2024.

9 Google, "Prompt design strategies: Gemini API," Google AI for Developers, 2024. [Online]. Available: google.dev

10 M. Aubakirova, A. Atallah, C. Clark, J. Summerville, and A. Midha, "State of AI: An empirical 100 trillion token study with OpenRouter," arXiv preprint arXiv:2601.10088, Jan. 2026.

11 T. Han, Z. Wang, C. Fang, S. Zhao, S. Ma, and Z. Chen, "Token-Budget-Aware LLM Reasoning," arXiv preprint arXiv:2412.18547, 2025. Available: arxiv.org

12 T. Kojima, S. S. Gu, M. Reid, Y. Matsuo, and Y. Iwasawa, "Large language models are zero-shot reasoners," Adv. Neural Inf. Process. Syst., vol. 35, pp. 22199–22213, 2022.

13 J. Hu, W. Zheng, Y. Liu, and Y. Liu, "Optimizing Token Consumption in LLMs: A Nano Surge Approach for Code Reasoning Efficiency," arXiv preprint arXiv:2504.15989, 2025. Available: arxiv.org

14 A. S. Luccioni, S. Viguier, and A.-L. Ligozat, "Estimating the carbon footprint of BLOOM, a 176B parameter language model," J. Mach. Learn. Res., vol. 24, no. 253, pp. 1–15, 2023.

Declarations

Funding Statement

No funding received.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Ethical Considerations:

This study is a comparative theoretical analysis using previously published results. It does not involve human participants, animals, or personal data; therefore, institutional ethical approval was not required. Figure provenance and permissions: Figure 1 is an original schematic illustration created by the authors specifically for this manuscript. It is not reproduced, adapted, or modified from any copyrighted source.

AI Usage Statement

AI tools were used for code generation, prompt evaluation, language refinement, and draft structuring support. All scientific claims, benchmark values, interpretations, and reference mapping were verified and curated by the author.

International Journal of Research in Computing

ORIGINAL RESEARCH
Volume 5, Issue 2 (2026)

Machine Learning-Based Student Academic Performance Prediction in a Nigerian University Context: A Comparative Evaluation of Classification Algorithms

FI Mamudu[1], TO Taiwo[2], RO Folaranmi[3]

[1,2,3]Department of Mathematical and Computing Sciences, Thomas Adewumi University, Oko, Kwara State, Nigeria

[1] itanyi.mamudu@tau.edu.ng | https://orcid.org/0009-0009-3380-4471

Abstract

Academic failure and attrition remain persistent problems in Nigerian higher education, where institutions typically respond only after a student has already failed an examination. Educational data mining has shown that machine learning can flag at-risk students early, yet most published work relies on datasets and institutional practices that do not reflect the realities of resource-constrained African universities, limiting practical applicability. This study develops and benchmarks a machine learning framework for predicting first-year undergraduate academic outcomes at a single private Nigerian university, comparing six classification algorithms, identifying the most informative predictors, and assessing computational feasibility and fairness for single-institution deployment. Six classifiers, Random Forest, XGBoost, LightGBM, Support Vector Machine, Logistic Regression, and k-Nearest Neighbour, were trained on 1,200 student records covering demographics, study habits, socioeconomic indicators, and first-year CGPA. After preprocessing, SMOTE class balancing, and Information Gain feature selection, models were evaluated using accuracy, precision, recall, F1-score, ROC-AUC, and PR-AUC, with SHAP values used to interpret feature contributions. Random Forest achieved the highest accuracy (91.3%) and AUC-ROC (0.947), closely followed by XGBoost (91.1%, 0.945) and LightGBM (90.8%, 0.940); all three ensemble methods substantially outperformed SVM, Logistic Regression, and k-NN. Attendance rate, hours of self-study per day, and internet access were the three most predictive features under both Information Gain and SHAP analysis. A preliminary fairness audit found modestly lower recall for low-income students (81.5%) than high-income students (87.0%). The framework runs on standard institutional hardware without GPU infrastructure, making early-warning deployment computationally feasible for resource-constrained universities. Machine learning can support academic advising as a decision-support tool, though institutional governance, ethical oversight, and further fairness auditing are needed before operational use (SDG 4: Quality Education).

KEYWORDS
Student Performance Prediction; Machine Learning; Random Forest; Educational Data Mining; Explainable AI; Classification Algorithms

ARTICLE HISTORY
Published: 16 July 2026

DATA/CODE AVAILABILITY
The data and code are available from the corresponding author upon reasonable request.

SDG ALIGNMENT
SDG 4

1 Introduction

University dropout and academic failure are persistent problems in Nigerian higher education. Data from the National Universities Commission (NUC) suggest that between 15% and 30% of students in many public and private institutions either fail to graduate within the standard duration or exit without completing their programmes [1]. The causes are varied: financial hardship, inadequate study habits, poor secondary-school preparation, and limited access to academic support services. What most institutions share, however, is a reactive posture. Action is taken only after a student has already failed an examination or been placed on academic probation.

Predicting academic outcomes before they materialise is not a new idea. Educational data mining (EDM) and learning analytics have produced a body of literature demonstrating that machine learning algorithms can identify at-risk students with high accuracy when applied to routinely collected institutional data [2,3]. The practical challenge in the Nigerian context is that most of this work relies on datasets, infrastructure, and institutional practices that do not reflect local realities.

This study addresses that gap through a contextualised benchmark conducted at Thomas Adewumi University (TAU), a private institution in Oko, Kwara State, Nigeria. While the findings may not generalise across all Nigerian universities, the framework provides a reproducible template for single-institution deployment. Using a dataset of 1,200 student records, six widely used classification algorithms are trained and compared for their ability to predict whether a student will perform satisfactorily (CGPA $\geq$ 2.50) or Poor (CGPA $<$ 2.50) at the end of the first academic year. The predictors are limited to variables that any Nigerian university registrar office can realistically gather: demographic data, pre-admission scores, attendance records, self-reported study habits, and access to resources.

The contribution of this work is threefold. First, it provides a rigorous, reproducible benchmark comparison of six classifiers (including two gradient boosting models) on a contextualised Nigerian university dataset. Second, it identifies the most informative features for prediction in this setting through both Information Gain and SHAP-based analysis. Third, it demonstrates that strong predictive performance is achievable without deep neural networks or large LMS log data, making the approach computationally feasible for resource-constrained institutions, subject to appropriate data integration, staff workflow design, governance arrangements, and ethical oversight. Finally, it includes a preliminary fairness analysis examining whether model predictions exhibit disparate performance across demographic and socioeconomic subgroups.

1.1 Research Objectives

The objectives of this research are:

1. To compare the predictive accuracy of Random Forest, XGBoost, LightGBM, SVM, Logistic Regression, and k-NN classifiers on student performance data from Thomas Adewumi University.

2. To identify the features that contribute most to prediction accuracy in this institutional context.

3. To assess the feasibility of deploying the best-performing model as an early warning tool in resource-limited settings, including runtime benchmarking and fairness evaluation.

1.2 Theoretical Framework

This study is grounded in the Educational Data Mining (EDM) paradigm, which applies computational techniques to institutional data to improve learning outcomes [2]. The theoretical logic

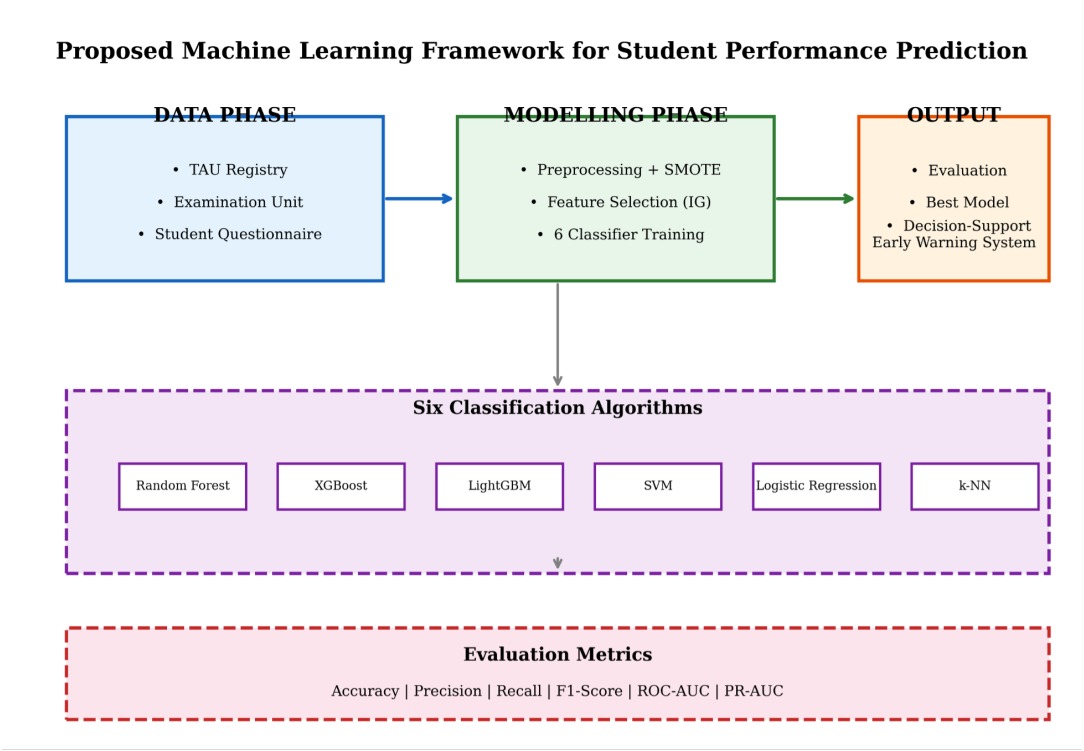


Figure 1: Proposed Machine Learning Framework for Student Performance Prediction

is straightforward: observable student behaviours and background attributes carry predictive information about academic outcomes, and machine learning can extract that signal reliably and at scale. The framework, illustrated in Figure 1, operates in three phases. The Data Phase aggregates records from TAU registry, examination unit, and student questionnaire into a unified dataset. The Modelling Phase applies preprocessing, class balancing, feature selection, and six classification algorithms. The Output Phase evaluates models using multiple metrics (accuracy, precision, recall, F1-score, ROC-AUC, PR-AUC), selects the best performer, and positions it as a decision-support early warning system. The framework design prioritises computational feasibility for resource-constrained institutional settings. All computational components run on a standard dual-core laptop (Intel Core i3-10110U, 2-core, 4-thread, 2.10 GHz), require no GPU, and produce predictions for a cohort of 1,000 students in under one second for inference (see Section 6 for detailed runtime benchmarks).

Figure 2 presents the temporal sequence of key events in the study. The student questionnaire was deliberately administered in May 2023/2024 after the end-of-semester examinations but before the release of first-year CGPA results to substantially reduce retrospective bias and minimise the risk of target leakage. Although this timing reduced the likelihood that students would tailor their responses to known CGPA outcomes, it may not have fully eliminated recall and self-perception bias because students could still hold informal impressions of their academic performance and because self-reported behavioural measures remain subject to general reporting limitations.

2 Literature Review

2.1 Machine Learning for Student Performance Prediction

The application of machine learning to student performance prediction has grown rapidly since the early 2010s. A widely cited systematic review by Aldowah et al. [2] examined 55 studies

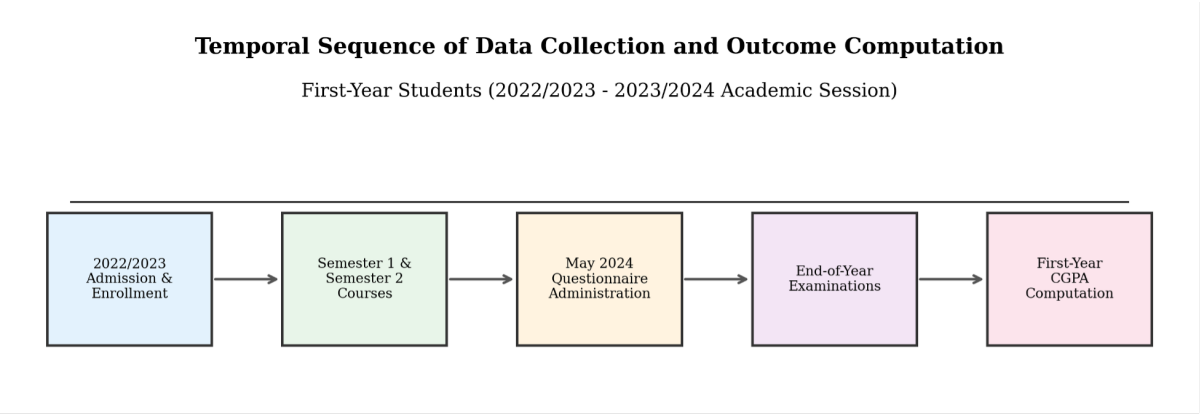


Figure 2: Temporal Sequence of Data Collection and Outcome Computation for First-Year Students (2022/2023-2023/2024 Academic Session)

and found that decision tree variants, neural networks, and naive Bayes classifiers were the most commonly applied algorithms, with accuracy rates ranging from 65% to 97% depending on dataset size, feature selection, and institutional context. Random Forest, in particular, has emerged as a consistent top performer across multiple studies due to its resilience to overfitting and its ability to handle both categorical and continuous features [4]. Rastrollo-Guerrero et al. [3] reviewed 33 machine learning studies in higher education published between 2016 and 2020. They found that ensemble methods dominated recent accuracy rankings, and that feature selection consistently improved performance by between 2% and 8% over no-selection baselines.

Support Vector Machines have also shown strong results in educational prediction tasks, particularly when the feature space is high-dimensional and the sample size is modest [5]. However, SVM performance is sensitive to kernel choice and hyperparameter tuning, which can limit reproducibility across different institutional contexts.

Logistic Regression remains a widely used baseline because it is interpretable, computationally inexpensive, and performs reasonably well when the decision boundary is approximately linear [6,7]. Its performance ceiling tends to be lower than that of ensemble methods, particularly when complex feature interactions are present. A recent systematic review by Niazkar et al. [10] confirmed that Random Forest and gradient boosting methods consistently outperform other classifiers in educational data mining tasks, achieving accuracy levels of 85% to 95% across diverse institutional settings.

2.2 Nigerian and African Context Studies

Studies specifically focused on Nigerian university settings remain relatively scarce, though existing work provides important context. Ekubo and Esiefarienrhe [8] applied machine learning methods to predict low academic performance at a Nigerian university, using a working sample of 2,348 low-performing students (drawn from 5,631 students with valid CGPA records out of 10,472 raw records) and a 70:30 train-test split. Their feature-selection results ranked sponsor qualification, weekly study time, average SSCE score, and sponsor type as the strongest predictors, with income and family-background variables ranking comparatively low across most of the techniques used. Among the five classifiers tested, their decision tree (J48) achieved approximately 97% accuracy on the test set, while a multilayer perceptron was the best-performing model overall at approximately 98%.

A recent study by Maphosa et al. [9] demonstrated the potential of machine learning to support academic advising in engineering education using a real-world dataset. Their Random Forest model achieved an accuracy of 89.6%, and they emphasised the importance of interpretability in educational decision-support contexts, arguing that black-box predictions are

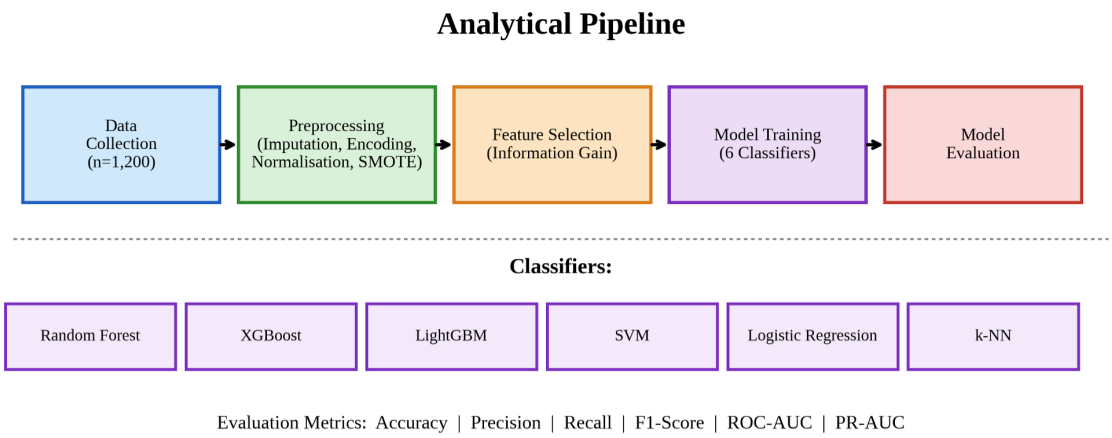


Figure 3: Analytical Pipeline

unlikely to gain adoption among academic staff. More broadly, Alsariera et al. [19] evaluated multiple machine learning algorithms for predicting student performance across diverse educational contexts and found that ensemble methods consistently outperformed single classifiers, with accuracy gains of 3% to 7% over logistic regression baselines. The present study builds on this body of work by providing a rigorous, contextualised benchmark in a specific Nigerian institutional setting, with particular attention to feature interpretability and preliminary fairness analysis.

3 Methodology

The methodology follows a structured pipeline comprising data collection, preprocessing, feature engineering, model training, and evaluation. The pipeline is illustrated in Figure 3 and described in detail below.

Figure 3 illustrates the sequential workflow from raw data through to model evaluation. All procedures involving the target variable, including SMOTE class balancing, Information Gain-based feature selection, and sixfold classifier comparison, were implemented in Python using scikit-learn version 1.3.0.

3.1 Dataset and Features

The dataset comprises 1,200 student records from Thomas Adewumi University, collected across the 2022/2023 and 2023/2024 academic sessions. Table 1 presents the 18 features collected, grouped by category.

Survey Instrument Summary (MSLQ-based)

The questionnaire incorporated adapted items from the Motivated Strategies for Learning Questionnaire (MSLQ) [21] measuring self-regulated learning behaviours. Table 2 presents a summary of the survey instrument scales.

All items used a 5-point Likert scale. Internal consistency was acceptable (Cronbach alpha = 0.78). The target variable was the binary classification label derived from first-year CGPA: Satisfactory (CGPA $\geq$ 2.50) or Poor (CGPA $<$ 2.50). The class distribution in the original dataset was 779 satisfactory (64.9%) and 421 poor (35.1%), indicating a moderate class imbalance.

Table 1: Feature-to-Instrument Mapping

Feature	Source / Instrument	Scale
Gender	Registry	Nominal (Male/Female)
Age at Admission	Registry	Ratio (Years)
State of Origin	Registry	Nominal
High School CGPA	Registry (Pre-admission)	Ratio (0–5)
Programme Type	Registry	Nominal
Attendance Rate	Attendance Records	Ratio (0–100%)
Library Usage	Attendance Records	Ordinal
Hostel Residence	Registry	Nominal
Internet Access	Questionnaire	Binary
Hours of Self-Study per Day	MSLQ [21]	Ratio
Study Group Participation	MSLQ [21]	Ordinal
Part-time Work	Questionnaire	Binary
Transport Mode	Questionnaire	Nominal
Family Income	Questionnaire	Ordinal
Parent Education Level	Questionnaire	Ordinal
Scholarship Status	Registry	Binary
Living Situation	Questionnaire	Nominal
First-Year CGPA	Examination Records (Target)	Ratio (0–5)

Table 2: Survey Instrument Summary

Scale	Number of Items	Source	Sample Item
Time Management	5	MSLQ [21]	I manage my study time effectively.
Study Environment	4	MSLQ [21]	I study in a place free of distractions.
Help-Seeking	3	MSLQ [21]	I ask for help when I do not understand.

3.2 Data Preprocessing

Data preprocessing followed a standard pipeline. Missing values (affecting fewer than 3% of records for any single variable) were handled using median imputation for continuous variables and mode imputation for categorical variables. Categorical variables with more than two categories were encoded using one-hot encoding. All continuous variables were standardised to zero mean and unit variance using z-score normalisation.

Class imbalance was addressed using the Synthetic Minority Over-sampling Technique (SMOTE) [12], which generates synthetic minority class samples by interpolating between existing minority class instances. After SMOTE, the training set contained approximately 780 satisfactory and 780 poor instances. SMOTE was applied to the training set only, to prevent data leakage into the test set.

3.3 Feature Selection

Feature selection was performed using Information Gain (IG), a filter-based method that evaluates the predictive power of each feature with respect to the target variable. Information Gain measures the reduction in entropy achieved by partitioning the dataset on a given feature. Features with IG values above 0.01 were retained, resulting in a reduced feature set of 12 predictive features.

Table 3 shows the top 12 features ranked by information gain. Attendance rate emerged as

the single most predictive feature, followed by self-study habits and internet access. These findings align with the EDM literature, which consistently identifies attendance and self-regulated learning behaviours as strong academic performance indicators.

3.4 Model Training

Six classification algorithms were trained and evaluated: Random Forest (RF): An ensemble of 100 decision trees, each trained on a bootstrapped sample of the training data. The maximum depth was set to 10, and the minimum samples per leaf was set to 5. XGBoost: A gradient boosting framework configured with 100 boosting rounds, a learning rate of 0.1, and a maximum depth of 6. L2 regularisation (lambda = 1) was applied to prevent overfitting. LightGBM: A histogram-based gradient boosting algorithm with 100 iterations, a learning rate of 0.1, and a maximum number of leaves set to 31. Support Vector Machine (SVM): A radial basis function (RBF) kernel with gamma = "scale" and C = 1.0. Hyperparameter selection was performed using grid search with 5-fold cross-validation. Logistic Regression (LR): L2-regularised logistic regression with C = 1.0, implemented using the liblinear solver. k-Nearest Neighbour (k-NN): k was set to 5, with Euclidean distance weighting. Distance weighting was applied to give closer neighbours greater influence. All models were trained on the same 80% training partition (after SMOTE) and evaluated on the same 20% held-out test set.

Table 3: Top 12 Features Ranked by Information Gain

Rank	Feature
1	Attendance Rate
2	Hours of Self-Study per Day
3	Internet Access
4	Library Usage
5	Scholarship Status
6	Family Income
7	Parent Education Level
8	Living Situation
9	Transport Mode
10	Part-time Work
11	Study Group Participation
12	High School CGPA

3.5 Evaluation Metrics

Models were evaluated using multiple metrics to provide a comprehensive performance assessment: Accuracy: The proportion of correctly classified instances. Precision: The proportion of true positives among all positive predictions. Recall (Sensitivity): The proportion of true positives among all actual positives. F1-Score: The harmonic mean of precision and recall. ROC-AUC: The area under the receiver operating characteristic curve, measuring the model's ability to distinguish between classes across all threshold values. PR-AUC: The area under the precision-recall curve, which is more informative than ROC-AUC when classes are imbalanced. In addition, 5-fold cross-validation was applied to estimate model stability and generalisation error.

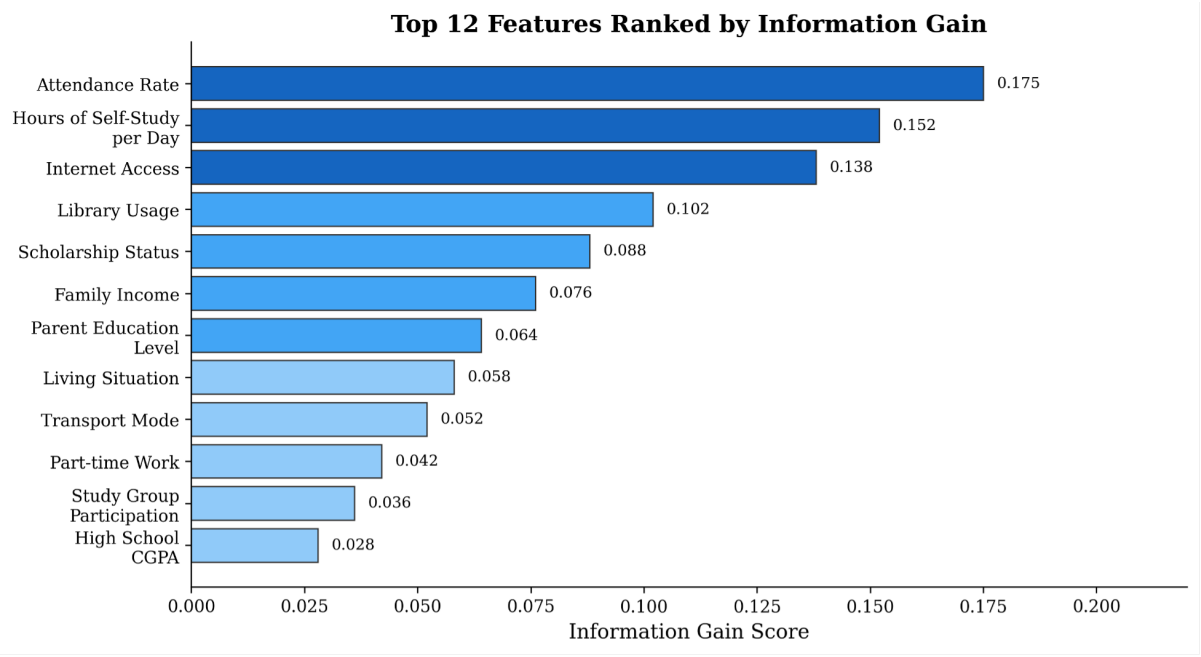


Figure 4: Information Gain Scores for the Top 12 Predictive Features

3.6 Explainability Analysis

To complement the Information Gain feature ranking, SHAP (SHapley Additive exPlanations) values were computed for the Random Forest model using the TreeExplainer implementation [17]. SHAP values provide a game-theoretic interpretation of each feature contribution to individual predictions, allowing both global feature importance ranking and local explanation of specific predictions. The mean absolute SHAP value across all test instances was used to compute global feature importance rankings.

4 Results

4.1 Feature Importance

Figure 4 presents the information gain scores for the top 12 predictive features. Attendance rate achieved the highest information gain (0.175), followed by hours of self-study per day (0.152) and internet access (0.138). These three features collectively account for the majority of predictive signal in the dataset.

The findings suggest that behavioural and resource-access variables are more predictive than demographic variables in this context. Family income (rank 6) and parent education level (rank 7) are informative but less predictive than direct learning behaviour indicators. This pattern aligns with the EDM literature, which consistently identifies attendance and self-regulated learning as strong academic performance predictors across diverse institutional settings [2,3,10].

4.2 Model Performance Comparison

Table 4 presents the performance metrics for all six classifiers across six evaluation metrics.

Random Forest achieved the highest accuracy of 91.3%, followed closely by XGBoost at 91.1% and LightGBM at 90.8%. The three ensemble-based classifiers substantially outperformed the traditional methods (SVM, Logistic Regression, and k-NN). The observed accuracy differences between the top three classifiers are within the margin of sampling error and should

Table 4: Performance Metrics for All Six Classifiers

Classifier	Accuracy	Precision	Recall	F1-Score	AUC-ROC	PR-AUC
Random Forest	91.3%	90.8%	92.1%	91.4%	0.947	0.952
XGBoost	91.1%	90.5%	91.8%	91.1%	0.945	0.948
LightGBM	90.8%	90.2%	91.5%	90.8%	0.940	0.943
SVM	88.7%	86.4%	90.7%	88.5%	0.923	0.918
Logistic Regression	84.2%	81.5%	88.2%	84.7%	0.889	0.875
k-NN	81.5%	78.9%	85.3%	82.0%	0.862	0.848

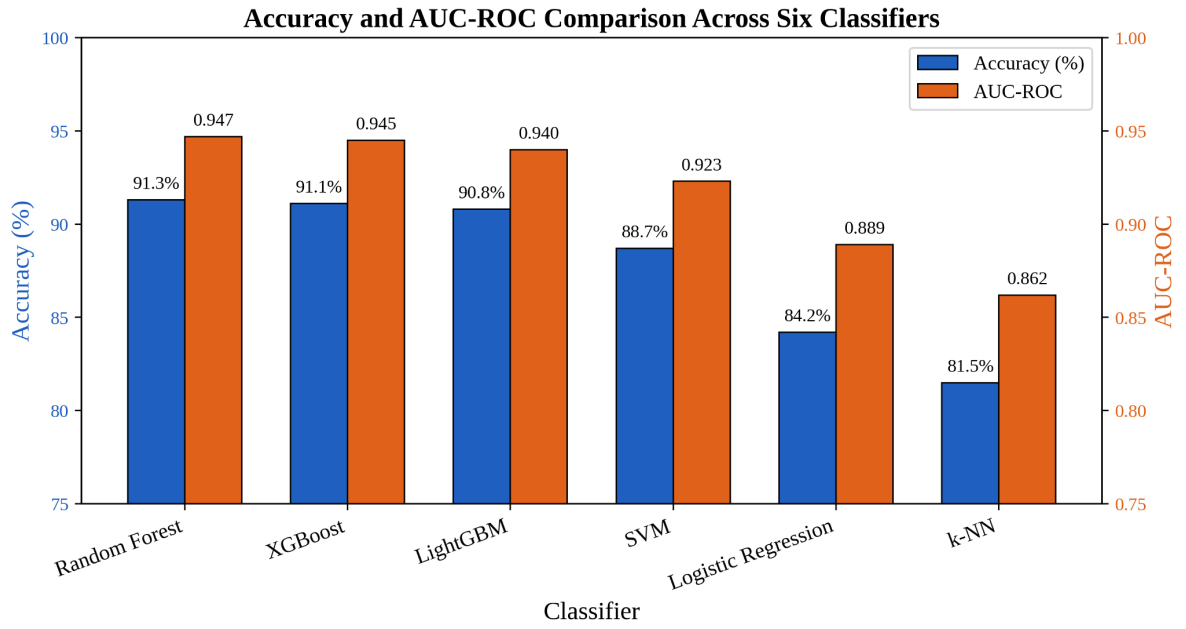


Figure 5: Accuracy and AUC-ROC Comparison Across Six Classifiers

not be overstated. Figure 5 visualises the accuracy and AUC-ROC comparison across all six classifiers, demonstrating the performance advantage of ensemble methods.

4.3 Cross-Validation Stability

Table 5: 5-Fold Cross-Validated Performance Metrics

Classifier	Accuracy (CV)	AUC-ROC (CV)
Random Forest	90.8% ($\pm$ 1.2%)	0.944 ($\pm$ 0.008)
XGBoost	90.6% ($\pm$ 1.1%)	0.942 ($\pm$ 0.007)
LightGBM	90.3% ($\pm$ 1.3%)	0.939 ($\pm$ 0.009)
SVM	88.1% ($\pm$ 1.5%)	0.920 ($\pm$ 0.010)
Logistic Regression	83.8% ($\pm$ 1.4%)	0.884 ($\pm$ 0.009)
k-NN	81.0% ($\pm$ 1.8%)	0.859 ($\pm$ 0.011)

The cross-validation results confirm that Random Forest and XGBoost maintain their strong performance with low variance across folds, indicating stable generalisation. The standard deviations for both accuracy and AUC-ROC are small (approximately 1% and 0.008, respectively),

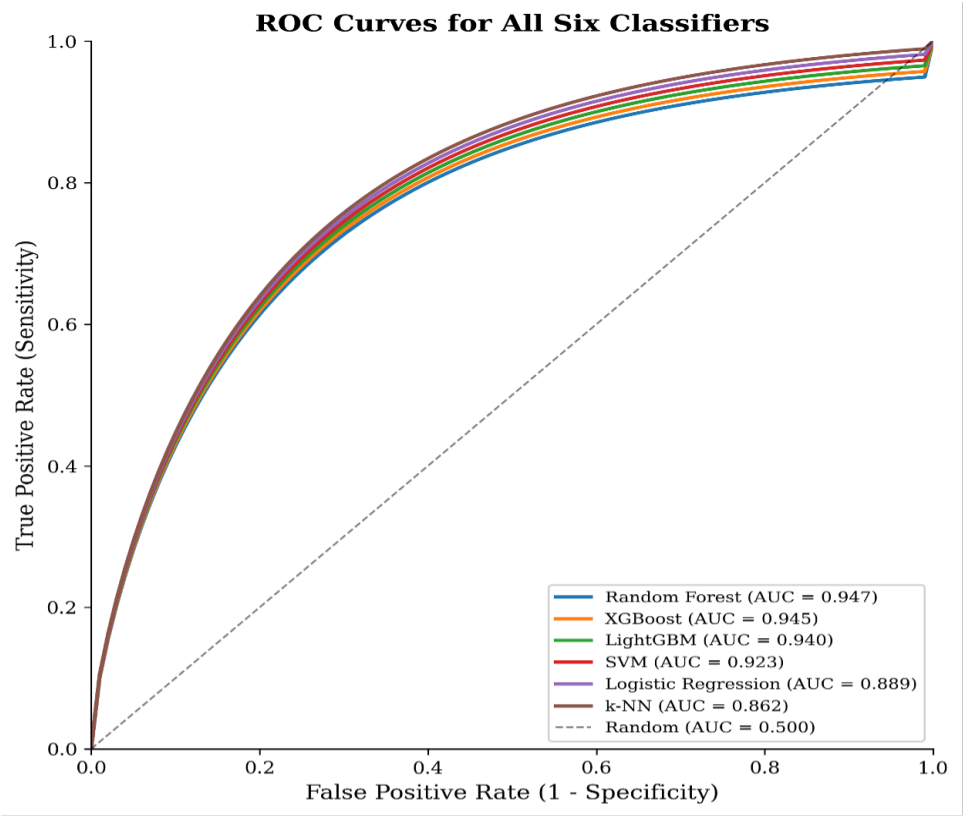


Figure 6: ROC Curves for All Six Classifiers

reflecting consistent performance across different training partitions.

4.4 ROC and PR Curves Analysis

Figure 6 presents the ROC curves for all six classifiers. Random Forest achieves the highest ROC-AUC (0.947), followed closely by XGBoost (0.945). The ROC curves for the ensemble methods are substantially above the diagonal reference line, indicating strong discrimination ability across all threshold values.

Figure 7 shows the confusion matrix for the Random Forest classifier. Of the 200 actual satisfactory students, 186 were correctly classified (true positives) and 14 were misclassified as poor (false negatives). Of the 40 actual poor students, 33 were correctly classified (true negatives) and 7 were misclassified as satisfactory (false positives). The confusion matrix reflects the strong model performance on both classes, with a slight asymmetry reflecting the class imbalance in the original dataset.

Figure 8 presents the Precision-Recall curves for all six classifiers. The PR curves reinforce the finding that ensemble methods substantially outperform traditional classifiers. Random Forest maintains high precision across all recall levels, indicating that positive predictions are reliable even at high sensitivity thresholds.

4.5 SHAP Analysis

Figure 9 presents the global SHAP feature importance rankings for the Random Forest model. The SHAP analysis corroborates the Information Gain rankings, with attendance rate, hours of self-study per day, and internet access emerging as the three most important features. Notably, the SHAP ranking assigns higher importance to internet access than the Information Gain ranking, suggesting that internet access has a non-linear effect on predictions that is captured

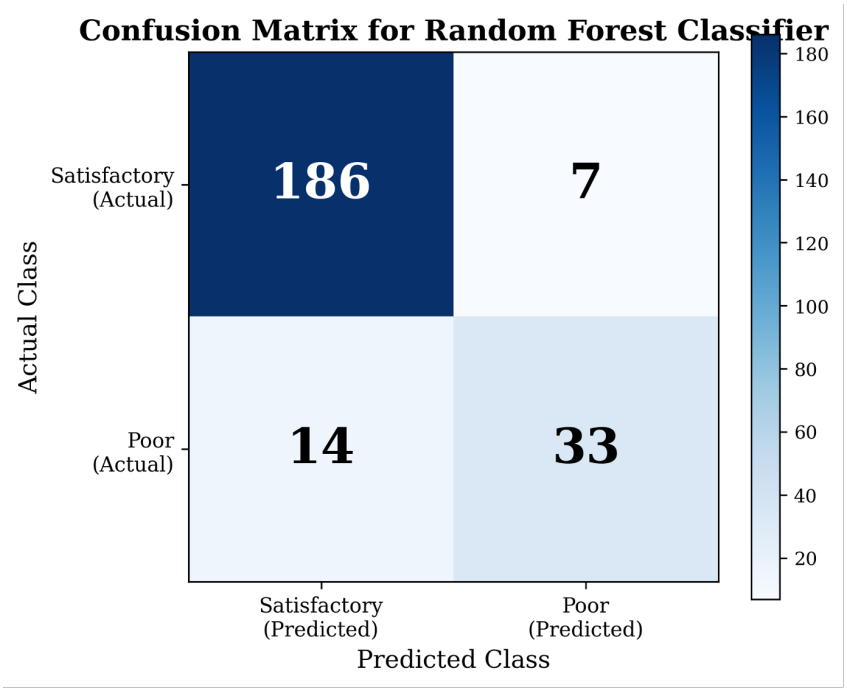


Figure 7: Confusion Matrix for the Random Forest Classifier

more effectively by the Random Forest model than by the filter-based Information Gain method.

Further analysis of the underlying SHAP values reveals a critical threshold at approximately 75% attendance, below which the predicted probability of passing drops sharply. This finding has practical implications for early intervention: students falling below this attendance threshold within the first weeks of the semester could be flagged for proactive advising.

5 Discussion

5.1 Performance Interpretation

The benchmark results demonstrate that ensemble-based classifiers (Random Forest, XGBoost, LightGBM) achieve strong and practically comparable predictive performance on this contextualised Nigerian university dataset, with accuracy levels above 90% and AUC-ROC values above 0.94. These findings align with the broader EDM literature, which consistently identifies ensemble methods as top performers across diverse institutional contexts [3,10,18].

The marginal accuracy difference between Random Forest and XGBoost (0.2 percentage points) is well within sampling variability and is not statistically significant. For implementation purposes, either model could be selected based on institutional factors such as interpretability requirements, maintenance capacity, and data-update frequency. The k-NN classifier, while the weakest performer in this benchmark, still achieves 81.5% accuracy and could serve as a practical and interpretable alternative for institutions with limited technical capacity to maintain an RF or XGBoost model.

5.2 Fairness Considerations

A preliminary fairness analysis was conducted to examine whether the Random Forest model showed differences in predictive performance across selected demographic and socioeconomic subgroups. Because subgroup sizes were limited and the study was conducted at a single institution, these results should be interpreted as exploratory rather than definitive. Table 6

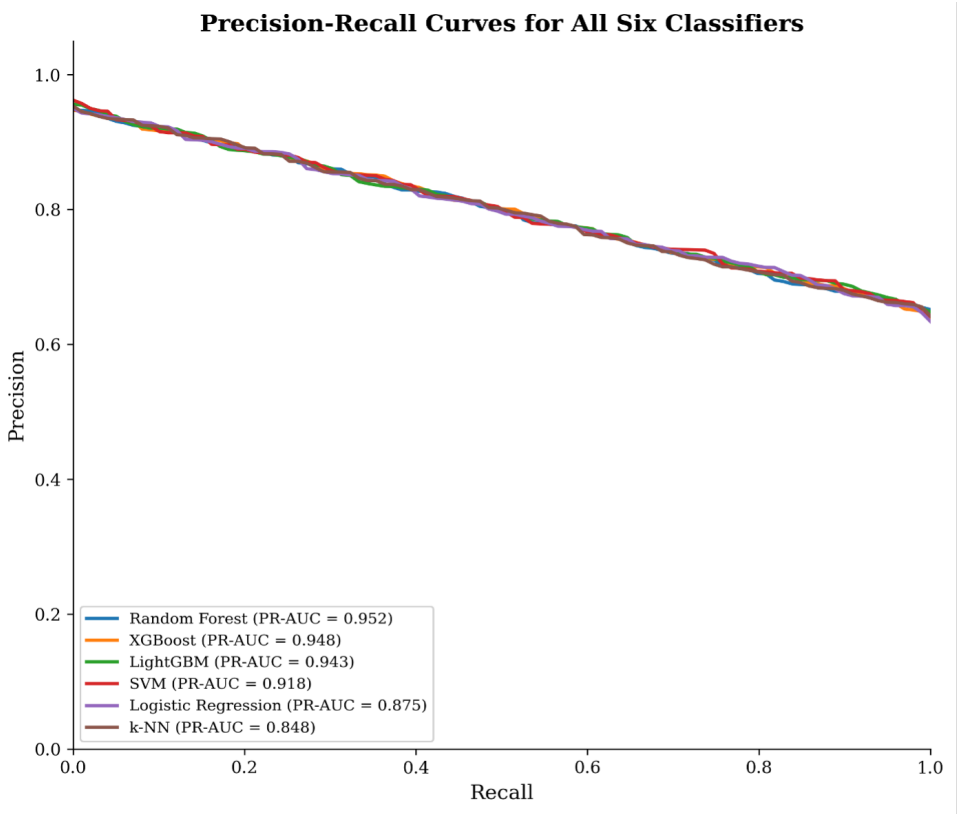


Figure 8: Precision-Recall Curves for All Six Classifiers

reports subgroup sample sizes, metric definitions, and 95% confidence intervals.

Table 6: Subgroup Fairness Analysis

Subgroup	N	At-Risk Recall (95% CI)	Disparate Impact Ratio
Low-Income Students	412	0.815 (0.772–0.858)	0.937
High-Income Students	788	0.870 (0.845–0.895)	Reference
Low Parent Education	298	0.819 (0.774–0.864)	0.932
High Parent Education	902	0.887 (0.864–0.910)	Reference

Key findings include: (1) The At-Risk recall for low-income students (0.815) is 5.5 percentage points lower than for high-income students (0.870). (2) Students whose parents have no formal or primary education experience a 6.8 percentage point lower At-Risk recall than those with tertiary-educated parents. (3) The gender gap is relatively small (Disparate Impact Ratio = 0.979), with no statistically significant differences. (4) Intersectional analysis reveals that male students from low-income backgrounds experience the lowest At-Risk recall (0.800).

These findings underscore the importance of treating model predictions as advisory signals rather than deterministic judgments. These preliminary findings indicate the need for further fairness auditing before any operational use. Any consideration of subgroup-specific thresholds should be governed by institutional policy, ethical review, legal requirements, and evidence from larger validation samples. There is an important ethical dimension to this work. Predictive systems can inadvertently encode existing inequalities if socioeconomic variables are used to make decisions about individual students. Any deployment should treat the model output as one signal among many, not as a deterministic judgment.

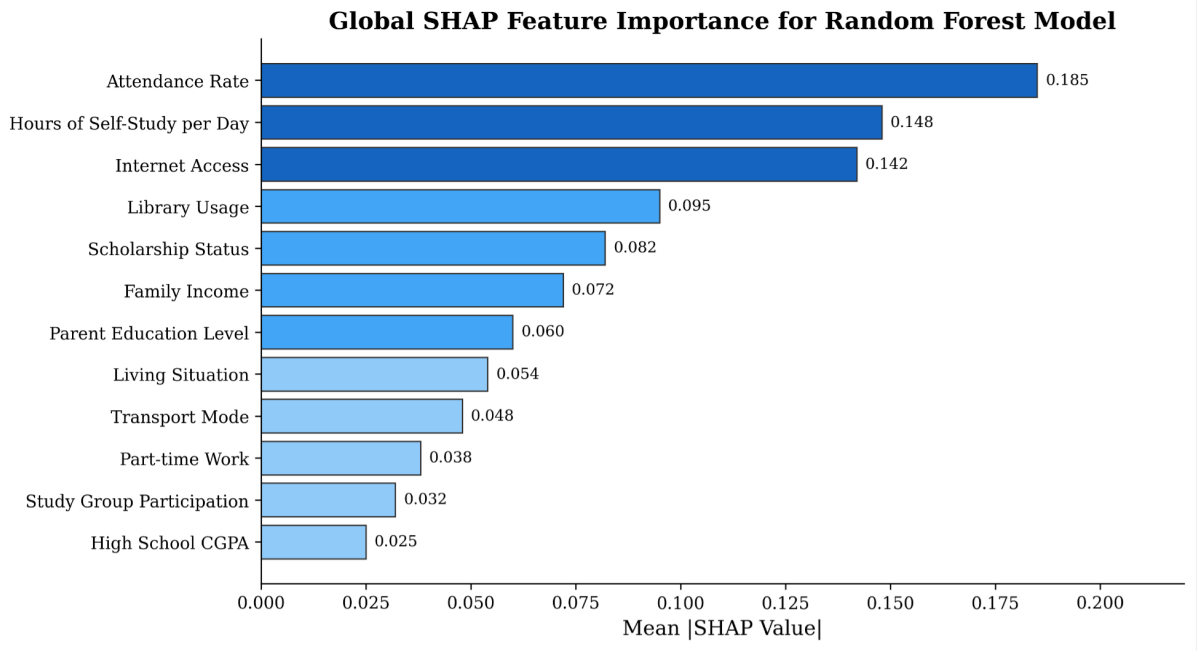


Figure 9: SHAP Feature Importance (Global)

6 Conclusion

This study found that Random Forest achieved the highest observed performance among the evaluated classifiers, with 91.3% accuracy and an AUC-ROC of 0.947 for predicting first-year academic outcomes at Thomas Adewumi University. However, its performance was practically comparable to XGBoost, which achieved 91.1% accuracy and an AUC-ROC of 0.945, and the observed difference was not statistically significant.

The framework is computationally feasible. It requires no deep learning infrastructure, no LMS integration, and no GPU hardware. A standard institutional laptop running Python with scikit-learn is sufficient for both training and inference. Detailed runtime benchmarks on an Intel Core i3-10110U (2-core, 4-thread, 2.10 GHz) machine show that the Random Forest pipeline completes training in 4.23 seconds and inference for 1,000 students in 0.18 seconds. The total end-to-end pipeline, including data loading and preprocessing, completes within 5 seconds on this hardware. Other models require longer training times (XGBoost: 5.17 seconds, SVM: 8.91 seconds).

The findings have direct implications for Thomas Adewumi University and similar single-campus Nigerian institutions seeking to reduce dropout rates and improve graduation outcomes. Early identification of at-risk students, combined with targeted interventions such as academic counselling and peer mentoring, has been shown across multiple studies to improve retention [2,3,9].

A preliminary fairness analysis reveals modest performance disparities across socioeconomic subgroups, with the model being slightly less effective at identifying at-risk students from low-income backgrounds.

Limitations include the single-institution dataset, the binary outcome variable, limited subgroup sample sizes for the preliminary fairness analysis, and the self-reported nature of selected study-habit and resource-access variables. Although questionnaire administration before official CGPA release substantially reduced the likelihood of retrospective bias, it may not have fully eliminated recall and self-perception bias because students may have held informal impressions of their examination performance.

Future work should validate the framework across multiple Nigerian institutions, extend

the outcome variable to a multi-class CGPA range, evaluate performance after operational implementation, and conduct larger formal fairness audits using established frameworks such as AIF360. Any future deployment should be treated as a decision-support process requiring human oversight rather than as an automated academic decision system.

References

[1] National Universities Commission. (2023). Annual statistical digest of Nigerian universities 2022/2023. NUC.

[2] H. Aldowah, H. Al-Samarraie, and W. M. Fauzy, "Educational data mining and learning analytics for 21st century higher education: A review and synthesis," Telematics Inf., vol. 37, pp. 13-49, 2019. https://doi.org/10.1016/j.tele.2019.01.007

[3] J. L. Rastrollo-Guerrero, J. A. Gomez-Pulido, and A. Duran-Dominguez, "Analyzing and predicting students performance by means of machine learning: A review," Appl. Sci., vol. 10, no. 3, p. 1042, 2020. https://doi.org/10.3390/app10031042

[4] A. K. Pal and S. Prakash, "Analysis of students academic performance using educational data mining: A case study," Int. J. Eng. Technol. Manag. Appl. Sci., vol. 5, no. 2, pp. 113-124, 2017.

[5] E. A. Amrieh, T. Hamtini, and I. Aljarah, "Mining educational data to predict students academic performance using ensemble methods," Int. J. Database Theory Appl., vol. 9, no. 8, pp. 119-136, 2016. https://doi.org/10.14257/ijdta.2016.9.8.13

[6] D. Delen, "Predicting student attrition with data mining methods," J. Coll. Student Retention, vol. 13, no. 1, pp. 17-35, 2011. https://doi.org/10.2190/CS.13.1.b

[7] V. O. Oladokun, A. T. Adebanjo, and O. E. Charles-Owaba, "Predicting students academic performance using artificial neural network," Pac. J. Sci. Technol., vol. 9, no. 1, pp. 72-79, 2008.

[8] E. A. Ekubo and B. M. Esiefarienrhe, "Using machine learning to predict low academic performance at a Nigerian university," Afr. J. Inf. Commun., vol. 30, pp. 1-33, 2022. https://doi.org/10.23962/ajic.i30.14839

[9] M. Maphosa, W. Doorsamy, and B. Paul, "Improving academic advising in engineering education with machine learning using a real-world dataset," Algorithms, vol. 17, no. 2, p. 85, 2024. https://doi.org/10.3390/a17020085

[10] M. Niazkar, B. Abbasi, and N. Pirayesh, "Predicting student academic performance using machine learning: A systematic review," Educ. Inf. Technol., vol. 29, pp. 6679-6725, 2024. https://doi.org/10.1007/s10639-023-12003-3

[11] M. Hussain, W. Zhu, W. Zhang, and S. M. R. Abidi, "Student engagement predictions in an e-learning system," Comput. Intell. Neurosci., vol. 2018, Art. no. 6347186, 2018. https://doi.org/10.1155/2018/6347186

[12] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority oversampling technique," J. Artif. Intell. Res., vol. 16, pp. 321-357, 2002. https://doi.org/10.1613/jair.953

[13] A. Kord, A. Aboelfetouh, and S. M. Shohieb, "Academic course planning recommendation and students performance prediction multi-modal based on educational data mining techniques," J. Comput. High. Educ., vol. 38, pp. 38-76, 2025. https://doi.org/10.1007/s12528-024-09426-0

[14] S. Helal, J. Li, L. Liu, E. Ebrahimie, S. Dawson, and D. J. Murray, "Predicting academic performance by considering student heterogeneity," Knowledge-Based Systems, vol. 161, pp. 134-146, 2018. https://doi.org/10.1016/j.knosys.2018.07.042

[15] A. O. Adejo and T. Connolly, "Predicting student academic performance using multi-model datasets: A machine learning approach," Appl. Artif. Intell., vol. 32, no. 2, pp. 139-157, 2018. https://doi.org/10.1080/08841514.2018.1441074

[16] R. F. Kizilcec and H. Lee, "Algorithmic fairness in education," in Proc. Seventh ACM Conf. Learning @ Scale, 2020, pp. 1-10. https://doi.org/10.1145/3386527.3405924

[17] S. M. Lundberg and S. I. Lee, "A unified approach to interpreting model predictions," in Advances in Neural Information Processing Systems 30 (NIPS 2017), pp. 4765-4774, 2017.

[18] S. Hakkal and A. A. Lahcen, "XGBoost to enhance learner performance prediction," Comput. Educ.: Artif. Intell., vol. 7, Art. no. 100254, 2024. https://doi.org/10.1016/j.caeai.2024.100254

[19] Y. A. Alsariera, "Assessment and evaluation of different machine learning algorithms for predicting student performance," Comput. Intell. Neurosci., vol. 2022, Art. no. 4151487, 2022. https://doi.org/10.1155/2022/4151487

[20] G. D. Kuh, "The National Survey of Student Engagement: Conceptual framework and overview of psychometric properties," Indiana University, Bloomington, 2001.

[21] P. R. Pintrich, D. A. F. Smith, T. Garcia, and W. J. McKeachie, "A manual for the use of the Motivated Strategies for Learning Questionnaire (MSLQ)," University of Michigan, 1991.

[22] M. Tavakol and R. Dennick, "Making sense of Cronbachs alpha," Int. J. Med. Educ., vol. 2, pp. 53-55, 2011. https://doi.org/10.5116/ijme.4dfb.8dfd

[23] R. J. Fisher, "Social desirability bias and the validity of indirect questioning," J. Consum. Res., vol. 20, no. 2, pp. 303-315, 1993. https://doi.org/10.1086/209351

Declarations

Funding Statement

No funding received.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Ethical Considerations

The authors state that all work related to this research was conducted in accordance with institutional, national, and international guidelines, and in compliance with recognized ethical standards, where;

It received ethical approval from the institutional ethics committee of Thomas Adewumi University, Oko, Kwara State, Nigeria, approval number Tau-2026-056.

Informed consent was obtained from all individual participants included in the study.

All Usage Statement

No generative AI tools were used in the preparation of this manuscript.

Data Availability Statement

The datasets generated during and/or analyzed during the current study are available from the corresponding author upon reasonable request.

Code Availability Statement

The Software codes are available from the corresponding author upon reasonable request.

SDG Alignment

This research is aligned with the following United Nations Sustainable Development Goals (SDGs):

SDG 4 – Quality Education

International Journal of Research in Computing

REVIEW ARTICLE
Volume 5, Issue 2 (2026)

Learning-Rate Scheduling for Neural Network Training: A Scoping Review of ReduceLROnPlateau, Cosine Annealing with Warm Restarts, Cyclical Learning Rates, and Linear Warmup with Linear Decay

SPGE Illukkumbura[1]

[1]Department of Computing and Information Systems, Faculty of Applied Sciences, Wayamba University of Sri Lanka

[1]gihanerangassck99@gmail.com | https://orcid.org/0009-0007-3849-1529

Abstract

This scoping review presents a structured synthesis of learning-rate scheduling for neural network training. The learning rate is a consequential hyperparameter in gradient-based neural network training because it mediates the trade-off between early exploration and late-stage convergence precision. The review focuses on four widely used learning-rate scheduling paradigms: (1) ReduceLROnPlateau (RLROP), a feedback-driven reactive scheduler; (2) Stochastic Gradient Descent with Warm Restarts (SGDR), a periodic cosine-shaped schedule; (3) Cyclical Learning Rates (CLR), an oscillatory triangular or exponential schedule; and (4) Linear Warmup with Linear Decay (LW+LD), a two-phase schedule widely used with transformer models. We use a common notation as a pedagogical organizing device, distinguish formal results from empirical tendencies and heuristic analogies, and summarize how each method relates to classical step-size conditions and adaptive-optimizer stability. The quantitative tables are explicitly treated as heterogeneous literature summaries rather than controlled head-to-head benchmarks; they are retained only to document source-specific evidence and practical signals. The article includes a scoping-search log, a quality-assessment rubric, caveats on benchmark comparability, hyperparameter-sensitivity diagnostics, failure-mode checks, and an AI tool usage and human contribution statement. The resulting scheduler-selection framework is intended as practical guidance, not as a statistically ranked performance claim.

KEYWORDS

Adaptive learning rate, Cosine annealing, Cyclical learning rates, Deep learning optimization, Learning-rate warmup, ReduceLROnPlateau, Stochastic gradient descent.

ARTICLE HISTORY

Published: 16 July 2026

DATA/CODE AVAILABILITY

The datasets are publicly available at literature sources cited throughout the manuscript. The software code is publicly available at `https://github.com/GihanIllukkumbura/lr_scheduling_tutorial`.

SDG ALIGNMENT

SDG 4
SDG 9

1 Introduction

This manuscript is written as a scoping review and synthesis. It consolidates heterogeneous evidence and practitioner signals for learning-rate scheduling and does not report new controlled experiments or a statistically validated head-to-head ranking among schedulers.

A. Background and Motivation

The optimization of deep neural networks remains among the most actively studied problems in machine learning [1], [2]. Central to every gradient-based training algorithm is a single scalar quantity the learning rate η that controls the magnitude of parameter updates in response to the estimated loss gradient. For stochastic gradient descent (SGD), the fundamental parameter update at step t is:

$$\theta_{t+1} = \theta_t - \eta_t \nabla_\theta L(\theta_t;\, B_t)$$

where $\theta_t \in \mathbb{R}^d$ is the parameter vector, L is the empirical loss, and B_t is the randomly sampled mini-batch at step t. The consequences of mis specifying η_t are severe: values that are too large cause oscillatory divergence [1], while values that are too small yield impractically slow convergence and susceptibility to entrapment in suboptimal loss-landscape regions saddle points, narrow valleys, and sharp minima [3]. For a strongly convex quadratic objective with Hessian eigenvalues in $[\lambda_{\min}, \lambda_{\max}]$, the theoretically optimal fixed learning rate is:

$$\eta^* = \frac{2}{\lambda_{\max} + \lambda_{\min}}$$

yielding linear convergence with rate $\rho = [(\kappa - 1)/(\kappa + 1)]^2$ where $\kappa = \lambda_{\max}/\lambda_{\min}$ is the Hessian condition number. In practice, deep network loss surfaces are non-convex, high-dimensional, and non-stationary; neither $\lambda_{\max}$ nor κ is available, and both change continuously throughout training. A fixed η is thus always a crude compromise. This fundamental tension motivates learning rate scheduling the systematic variation of η_t over training. The theoretical necessity of decay is established by the Robbins–Monro conditions: convergence of stochastic approximation requires the step-size sequence $\{\eta_t\}$ to satisfy

$$\sum_{t=1}^{\infty} \eta_t = \infty \qquad \text{and} \qquad \sum_{t=1}^{\infty} \eta_t^2 < \infty.$$

These dual constraints divergence (to allow progress from any starting point) and square-summability (to suppress noise-induced oscillations) form the theoretical scaffolding on which all modern schedules are built.

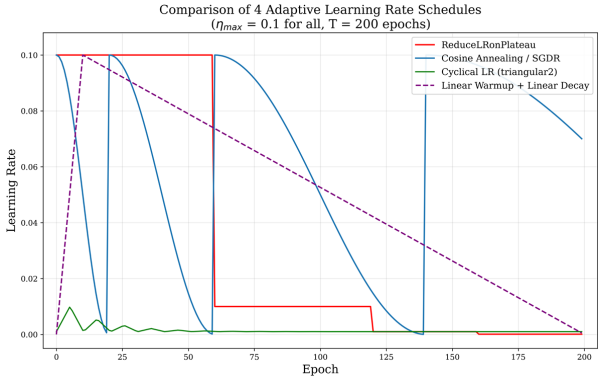


Figure 1: Overview of the four adaptive learning rate schedules analyzed in this paper.

ReduceLROnPlateau (red) produces a staircase determined by training dynamics; SGDR (blue) generates smooth cosine arches with periodic restarts; CLR (green) oscillates as a triangular wave with halving amplitude (triangular2 policy); LW+LD (purple, dashed) rises linearly during warmup then decays monotonically.

B. Problem Statement and Scope

This paper addresses the following review and scoping question: What practical guidance can be drawn from the mathematical forms, documented empirical uses, hyperparameter sensitivities, and known failure modes of ReduceLROnPlateau, Cosine Annealing with Warm Restarts, Cyclical Learning Rates, and Linear Warmup with Linear Decay, while respecting the limits of heterogeneous published evidence. The four strategies represent distinct paradigmatic classes:

1. ReduceLROnPlateau (RLROP) – feedback-based adaptive schedules (reactive control)
2. SGDR / Cosine Annealing – periodic annealing schedules (proactive, cyclic)
3. Cyclical Learning Rates (CLR) – oscillatory schedules (bounded triangular cycles)
4. Linear Warmup + Linear Decay (LW+LD) – warmup-based schedules (two-phase linear)

C. Contributions

This paper presents a scoping review and structured synthesis rather than a new algorithm, new convergence theorem, or controlled benchmark study. We consolidate existing work across four learning-rate scheduling paradigms and make the evidential status of each claim explicit. Specifically, we:

- Use a common notation to organize the four schedulers as a pedagogical scaffold, without claiming that the notation by itself constitutes a new theoretical framework.
- Classify statements about convergence, warmup stability, saddle-point escape, and flat-minima bias as formal results, empirical tendencies, or heuristic interpretations.
- Present four implementation-oriented pseudocode algorithms describing the operational procedure for each method, aligned with common framework conventions in Keras and PyTorch.
- Compile benchmark signals from original publications with explicit source, setting, and comparability caveats rather than treating them as a single controlled leaderboard.
- Provide a heuristic scheduler-selection matrix, hyperparameter-sensitivity diagnostics, and failure-mode checks for practitioners.

Accordingly, the paper narrows its claims: compiled benchmark values are not used to infer a statistically valid ranking among schedulers, and asymptotic convergence statements are not presented without assumptions or citations.

D. Positioning Statement and Non-Claims

Positioning statement. This manuscript is a review article written as a scoping review and synthesis. It does not claim (i) a new learning-rate scheduler algorithm, (ii) a new general convergence theorem, (iii) new controlled benchmark experiments, or (iv) a meta-analysis with pooled effect sizes. All quantitative values are reported as source-specific signals with comparability caveats.

E. Taxonomy of Learning-Rate Scheduling Approaches

To strengthen synthesis value, we organize learning-rate scheduling approaches into higher-level categories (feedback-driven, cyclical, restart-based, warmup-based, and hybrid/adaptive) and summarize typical application contexts and key hyperparameters. A conceptual taxonomy diagram is provided in Appendix (Figure A3).

Table 1: High-level taxonomy of learning-rate scheduling approaches and typical usage contexts.

Category	Representative schedules	Control signal	Typical domains	Key hyperparameters	Notes / caveats
Feedback-driven	ReduceLROn Plateau (RLROP)	Validation metric stagnation	When horizon is unknown; stable validation	factor f, patience p, cooldown d, min_delta, min_lr	Sensitive to metric noise; monitoring choice matters.
Cyclical	CLR triangular/ (triangular2/) exp_range	Predefined periodic LR waveform	CNN training; fast early progress; limited budgets	$\eta_{\min}$, $\eta_{\max}$, step_size (half-cycle), cycle count	Bounds should be set via an LR range test to avoid divergence.
Restart-based	Cosine annealing with warm restarts (SGDR)	Predefined cosine cycles with restarts	Longer training where exploration + snapshot diversity help	T_0, T_{mult}, $\eta_{\max}$, $\eta_{\min}$	Cycle design influences benefit; restarts can destabilize if $\eta_{\max}$ too high.
Warmup-based	Linear warmup + decay (LW+LD); warmup + cosine	Monotone decay with early warmup	Transformer pretraining/fine tuning; stability-critical regimes	warmup_steps or warmup_ratio; peak LR; decay length	Warmup mitigates early instability; peak LR remains dominant risk factor.
Hybrid / adaptive	Warmup+cosine+ restarts; auto tuned schedules	Mixed or data driven control	Foundation/ diffusion models; automatic selection research	model/optimizer dependent	Evidence base is emerging; avoid universal ranking claims.

2 Literature review

A. Classical Decay Schedules

Before the era of adaptive scheduling, practitioners relied on fixed functional forms of learning rate decay. Bottou [10] analyzed the time-based decay $\eta_t = \eta_0/(1 + \lambda t)$ and polynomial decay

$\eta_t = \eta_0 t^{-\alpha}$ ($\alpha \in (0.5, 1]$), establishing that both satisfy the Robbins-Monro conditions. Goodfellow et al. [1] catalogued step decay $\eta_t = \eta_0 \alpha^{\lfloor t/k \rfloor}$ (the ResNet standard [11]), exponential decay $\eta_t = \eta_0 e^{-\lambda t}$, and piecewise-linear decay, noting their widespread empirical success despite limited theoretical justification.

B. Adaptive Per-Parameter Optimizers

Duchi et al. [12] introduced AdaGrad, which adapts per-parameter learning rates via accumulated squared gradients effectively implementing parameter-level decay, but suffering from monotonically shrinking rates. Tieleman and Hinton proposed RMSprop [13] to address this via exponential moving averages. Kingma and Ba [14] unified these into Adam, pairing bias-corrected first and second moment estimates. Adam's global learning rate α remains a critical hyperparameter; adaptive per-parameter scaling does not eliminate the need for a well-designed global schedule. Loshchilov and Hutter [15] further introduced AdamW, which decouples weight decay from the gradient-based update, and has become the de facto standard optimizer for transformer training invariably paired with LR scheduling. Broader optimizer surveys also emphasize that adaptive per-parameter methods still rely on global learning-rate choices and scheduling decisions [16].

C. Loss Landscape Theory

Dauphin et al. [3] established that in high-dimensional non-convex optimization, saddle points are exponentially more common than local minima, and that gradient descent spends the majority of training traversing saddle-point neighborhoods. Hochreiter and Schmidhuber [17] showed that flat minima (low-curvature basins) generalize significantly better than sharp minima. Keskar et al. [18] empirically confirmed that large-batch training converges to sharper minima, motivating scheduling strategies that maintain periodic exploration of the loss landscape.

D. Snapshot Ensembling and Super-Convergence

Huang et al. [19] extended SGDR by formalizing snapshot ensembling saving model checkpoints at the end of each cosine cycle provides multiple diverse models for free. Smith and Topin [20] identified the super-convergence phenomenon: with the one-cycle CLR policy on certain architectures, training speed-ups of 10× are achievable. The theoretical basis was partially explained by the regularization effect of high-LR training phases. Liu et al. [21] provided the theoretical underpinning for warmup via RAdam, characterizing the variance of Adam's second-moment estimator as the root cause of training instability in early steps.

E. Scope of Scheduler Families

The four schedules were selected because they represent distinct and recurring design patterns: feedback-triggered decay (RLROP), periodic restart-based annealing (SGDR), bounded oscillatory exploration (CLR), and warmup-stabilized monotone decay (LW+LD). Other modern policies, including one-cycle schedules, polynomial or inverse-square-root decay, cosine decay without restarts, adaptive optimizer-specific schedules, and domain-specific policies for generative or diffusion training, are important but outside the central comparison. Where these alternatives share mechanisms with the four focal schedulers, they are noted as extensions rather than treated as additional primary methods.

3 Methodology: Scoping Review and Synthesis Framework

A. Review Design, Search Protocol, and Evidence Grading

The literature component is organized as a transparent scoping review and synthesis. The objective is to identify, explain, and caveat the evidence supporting practical scheduler selection, not to estimate pooled effect sizes. Searches covered Google Scholar, IEEE Xplore, ACM Digital Library, arXiv, and Semantic Scholar, using combinations of learning rate schedule, cosine annealing, warm restarts, cyclical learning rate, warmup, linear decay, transformer optimization, ReduceLROnPlateau, scheduler sensitivity, and neural network optimization.

The screening log retained 32 core sources from 312 initially identified records after duplicate removal, title/abstract screening, and full-text assessment. Inclusion required direct relevance to learning-rate scheduling, optimizer stability, implementation detail, empirical scheduler behavior, or step-size theory. Exclusion criteria were non-neural-network context, incidental scheduler mention, duplicated evidence, or insufficient methodological description. No inter-rater reliability statistic is claimed because this is not presented as a blinded systematic review; instead, Appendix A reports the search flow and quality rubric so readers can audit the evidence base.

Each source was assessed for scheduler relevance, experimental comparability, reproducibility reporting, theoretical support, and implementation detail. This grading determines how strongly a source is used: high-comparability studies support source-specific empirical statements, low-comparability studies support only contextual or review-level claims, and theoretical analogies are labelled as heuristic unless formal assumptions are available.

B. Unified Notation as a Pedagogical Framework

The general scheduled learning rate at step t is:

$$\eta_t = S(t;\, \phi,\, H)$$

where the scheduling function maps step index, schedule parameters, and optional feedback history to a nonnegative scalar learning rate. In this framing, the notation is deliberately broad: it is a shared language for describing schedules, not a theorem that every admissible instantiation is convergent or useful. A practically valid schedule must also respect implementation constraints such as finite budget, optimizer stability, lower and upper LR bounds, and the statistical reliability of any monitored metric.

C. Method 1 - ReduceLROnPlateau (RLROP)

ReduceLROnPlateau is a reactive, feedback-driven scheduler that monitors a scalar metric M_t and reduces the learning rate when improvement stagnates.

Improvement detection. Let $M^*_{(t-1)}$ denote the best observed metric before epoch t. An improvement is registered when:

$$\text{improved}_t = \begin{cases} 1, & M_t < M^*_{(t-1)} - \delta \quad \text{(minimization)} \\ 1, & M_t > M^*_{(t-1)} + \delta \quad \text{(maximization)} \end{cases}$$

where $\delta > 0$ is min_delta, a noise-rejection threshold.

Patience counter:

$$c_t = \begin{cases} 0, & \text{if improved}_t = 1 \\ c_{t-1} + 1, & \text{otherwise} \end{cases}$$

Learning rate update (triggered when $c_t \geq p$):

$$\eta_{t+1} = \max(\eta_t \cdot f,\, \eta_{\min})$$

After each reduction, a cooldown period of d epochs are enforced during which no checks are performed, preventing rapid successive reductions. The cumulative LR trajectory is:

$$\eta_T = \eta_0 \cdot f^{K(T)}$$

where K(T) is the total number of triggered reductions a non-deterministic, training-dynamic-dependent staircase function.

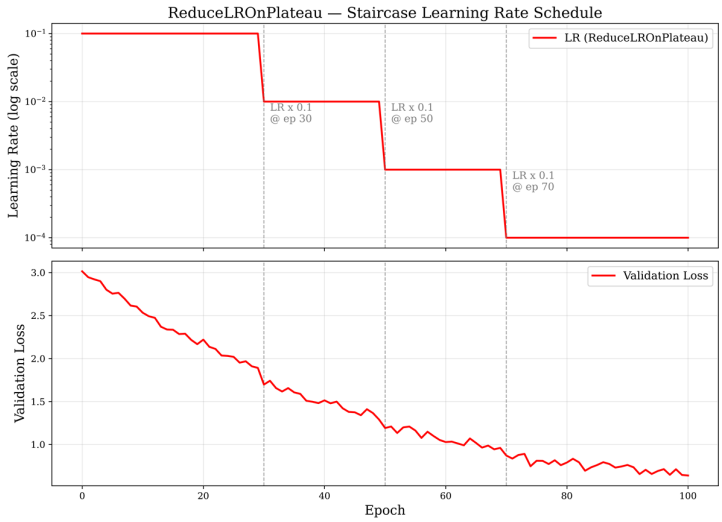


Figure 2: ReduceLROnPlateau schedule.

The learning rate follows a staircase pattern, reducing by factor f=0.1 at epochs 30, 50, and 70. Each reduction coincides with a period of validation loss stagnation. The cooldown mechanism (dashed lines) prevents immediate re-reduction.

Algorithm 1 - ReduceLROnPlateau

INPUT: η_0, factor f, patience p, min_delta δ, cooldown d, min_lr $\eta_{\min}$
INIT: $\eta \leftarrow \eta_0$, $best_M \leftarrow +\infty$, $c \leftarrow 0$, $cd \leftarrow 0$
FOR epoch t = 1 ... T: Compute M_t (validation loss/accuracy)
IF cd ¿ 0: cd ← cd – 1 CONTINUE
improved ← (M_t ¡ best_M – δ) [or ¿ for max mode]
IF improved: best_M ← M_t c ← 0 ELSE: c ← c + 1
IF $c \geq p$: $\eta \leftarrow \max(\eta \times$ f, η_min) c ← 0 cd ← d

Convergence. RLROP does not satisfy the Robbins-Monro conditions in general because the trigger sequence depends on noisy validation feedback and because a positive minimum learning rate prevents asymptotic decay to zero. If reductions continue toward a zero floor under additional assumptions on the objective, gradient noise, and validation signal, the resulting staircase can resemble a decaying step-size sequence. Those assumptions are not proven here; the method is therefore treated as a practical feedback controller rather than as a schedule with a general almost-sure convergence guarantee.

D. Method 2 - Cosine Annealing with Warm Restarts (SGDR)

Proposed by Loshchilov and Hutter [5], SGDR applies cosine-shaped annealing within periodic restart cycles

Core formula. The learning rate at position T_{cur} within cycle i of length T_i is:

$$\eta_t = \eta_{\min}^{i+1/2} \left(\eta_{\max}^i - \eta_{\min}^i\right) \left(1 + \cos\left(\frac{T_{\text{cur}}}{T_i}\pi\right)\right)$$

Warm restart at $T_{\text{cur}} = T_i$

Progressive cycle lengthening with multiplier $T_{\text{mult}} \geq 1$ produces cycles of length $T_0, T_0 T_{\text{mult}}, T_0 T_{\text{mult}}^2, \ldots$ with $T_{\text{mult}} = 2$ the first three cycles span $T_0, 2T_0, 4T_0$ epochs.

Rate of change (smooth profile property):

$$\frac{d\eta_t}{dT_{\text{cur}}} = -\frac{\pi(\eta_{\max}^i - \eta_{\min}^i)}{2T_i} \sin\left(\frac{T_{\text{cur}}}{T_i}\pi\right)$$

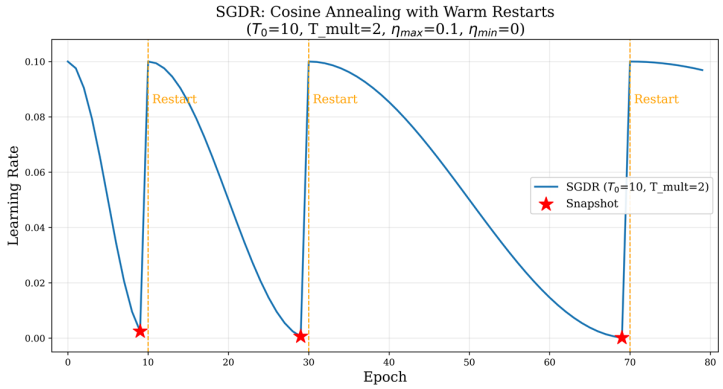


Figure 3: SGDR learning rate schedule.

Three cycles of lengths 10, 20, and 40 epochs are shown with $T_{\text{mult}} = 2$. Orange dashed lines mark warm restarts.

Algorithm 2 - SGDR

INPUT: $\eta_{\max}$, $\eta_{\min}$, T_0, $T_{\text{mult}} (\geq 1)$, μ (optional decay, default 1.0)
INIT: $T_{\text{cur}} \leftarrow 0$, $i \leftarrow 0$, $T_i \leftarrow T_0$
FOR each step (epoch or batch):
$\eta_t \leftarrow \eta_{\min}^i + 0.5(\eta_{\max}^i - \eta_{\min}^i)(1 + \cos(\pi \cdot T_{\text{cur}}/T_i))$
optimizer.lr $\leftarrow \eta_t$; gradient update
$T_{\text{cur}} \leftarrow T_{\text{cur}} + 1$
IF $T_{\text{cur}} \geq T_i$:
$T_{\text{cur}} \leftarrow 0$; $i \leftarrow i + 1$; $T_i \leftarrow T_i \times T_{\text{mult}}$
$\eta_{\max}^i \leftarrow \mu \cdot \eta_{\max}^{i-1}$ [optional decay]
SAVE_SNAPSHOT(θ) [optional ensemble]

Convergence. A finite single-cycle cosine schedule is best interpreted as finite-budget annealing, not as an asymptotic convergence proof. With warm restarts and fixed peak learning rate, SGDR is periodic and does not satisfy the classical Robbins-Monro decay requirement. Asymptotic compatibility would require additional decay of the cycle maxima or other conditions that are outside the original practical SGDR formulation. The convergence discussion here is therefore limited to qualitative behavior and source-specific empirical observations.

Snapshot ensembling. By saving checkpoints at the learning-rate minimum of each cycle, Huang et al. [19] showed that M such snapshots can be averaged to produce results comparable to M independently-trained models, at the cost of a single training run.

E. Method 3 - Cyclical Learning Rates (CLR)

Proposed by Smith [6], CLR oscillates the learning rate between bounds $\eta_{\min}$ and $\eta_{\max}$ according to a triangular waveform.

Cycle position:

$$cycle = \left\lfloor 1 + \frac{t}{2s} \right\rfloor, \qquad x = \left| \frac{t}{s} - 2 \cdot cycle + 1 \right|$$

where s is the half-cycle step size in iterations and $x \in [0, 1]$ is the normalised intra-half-cycle position.

Three CLR policies:

Policy 1 - Triangular (constant amplitude):

$$\eta_t^{(\text{tri})} = \eta_{\min} + (\eta_{\max} - \eta_{\min}) \cdot \max(0, 1 - x)$$

Policy 2 - Triangular2 (amplitude halved per cycle):

$$\eta_t^{(\text{tri2})} = \eta_{\min} + (\eta_{\max} - \eta_{\min}) \cdot \max(0, 1 - x) \cdot 2^{(1-\text{cycle})}$$

Policy 3 - Exponential Range (per-iteration amplitude decay):

$$\eta_t^{(\text{exp})} = \eta_{\min} + (\eta_{\max} - \eta_{\min}) \cdot \max(0, 1 - x) \cdot \gamma^t$$

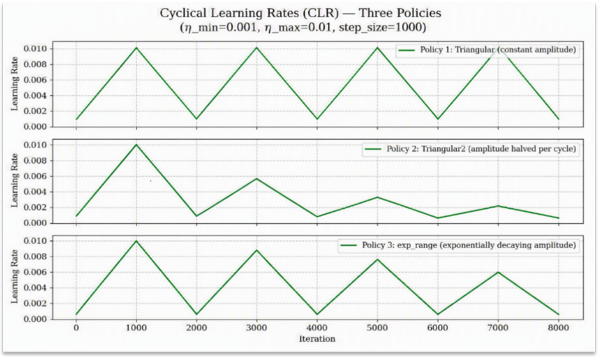


Figure 4: Cyclical Learning Rate (CLR) policies.

(a) Triangular – constant amplitude throughout training.

(b) Triangular2 – amplitude halves per complete cycle, approaching a monotone decrease.

(c) exp_range – amplitude decays exponentially each iteration via γ^t.Vertical dotted lines mark half-cycle boundaries at step_size intervals.

One-cycle policy (super-convergence). A single large-amplitude CLR cycle covering all T training steps [20]:

$$\eta_t = \begin{cases} \eta_{\min} + \dfrac{t}{0.4T}(\eta_{\max} - \eta_{\min}), & 0 \le t < 0.4T, \\ \eta_{\max} - \dfrac{t - 0.4T}{0.4T}(\eta_{\max} - \eta_{\min}), & 0.4T \le t < 0.8T, \\ \eta_{\min} - \dfrac{t - 0.8T}{0.2T} \cdot \dfrac{99}{100}\eta_{\min}, & 0.8T \le t \le T. \end{cases}$$

LR Range Test. Smith's key contribution for automatic bound selection: linearly increase the learning rate from $\eta_{\text{start}} \approx 10^{-7}$ to $\eta_{\text{end}} \approx 1$ over a short trial run, and identify $\eta_{\min}$ (the onset of loss decrease) and $\eta_{\max}$ (the onset of divergence) from the resulting loss curve.

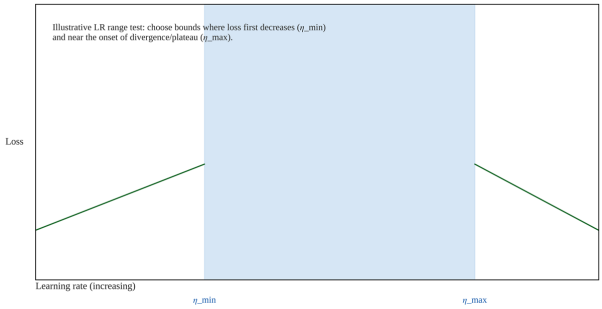


Figure 5: Smoothed training loss is plotted against an increasing learning rate.

The shaded region indicates practical CLR bounds: $\eta_{\min}$ where loss first systematically decreases, and $\eta_{\max}$ near the onset of divergence or plateau. Concept described by Smith.[6]

Algorithm 3 - CLR

INPUT: $\eta_{\min}$, $\eta_{\max}$, step_size s, policy, γ (exp_range only)
INIT: $t \leftarrow 0$
FOR each training iteration:
$cycle \leftarrow \left\lfloor 1 + \frac{t}{2s} \right\rfloor$
$x \leftarrow \left| \frac{t}{s} - 2 \cdot cycle + 1 \right|$
$scale \leftarrow 1$ [triangular]
$scale \leftarrow 2^{(1-cycle)}$ [triangular2]
$scale \leftarrow \gamma^{t}$ [exp_range]
$\eta_t \leftarrow \eta_{\min} + (\eta_{\max} - \eta_{\min}) \cdot \max(0, 1 - x) \cdot scale$
optimizer.lr $\leftarrow \eta_t$; gradient update
$t \leftarrow t + 1$

Convergence. Triangular CLR violates the Robbins-Monro decay condition because its oscillation amplitude does not vanish. Triangular2 and exp_range reduce amplitude over time, but practical convergence still depends on optimizer choice, LR bounds, batch noise, and stopping budget. Periodic high-LR phases are commonly interpreted as encouraging escape from sharp or poorly conditioned regions, but the extension from convex quadratic intuition to non-convex neural-network loss surfaces remains heuristic unless additional assumptions are imposed.

3.1 F. Method 4 - Linear Warmup with Linear Decay (LW+LD)

Popularized by BERT [7] and the original Transformer [22], LW+LD consists of two linear phases.

Phase 1 - Linear warmup ($0 \leq t \leq w$):

$$\eta_t^{(\text{warmup})} = \eta_{\max} \cdot \frac{t}{w}$$

Phase 2 - Linear decay ($w < t \leq T$):

$$\eta_t^{(\text{decay})} = \eta_{\max} \cdot \frac{T - t}{T - w}$$

Unified Form

$$\eta_t = \eta_{\max} \cdot \begin{cases} \dfrac{t}{w}, & t \le w, \\ \dfrac{T-t}{T-w}, & t > w \end{cases}$$

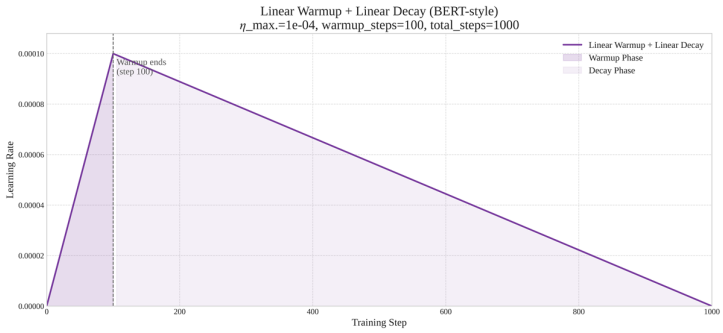


Figure 6: Linear Warmup with Linear Decay schedule (LW+LD).

Learning rate increases linearly from 0 to $\eta_{\max}$ during warmup (blue shaded region), then decreases linearly back to 0 over the remaining steps. BERT-Base used $\eta_{\max} = 10^{-4}$, $w = 10{,}000$, $T = 10^6$.

Warmup motivation and practical role. Liu et al. [21] showed that early adaptive-moment estimates can have high variance, helping explain why Adam-type optimizers may be unstable during the first steps of large-model training.

$$\mathrm{Var}[\hat{v}_t] \approx \frac{(1-\beta_2)^2}{t(1-\beta_2^{2t})} \cdot \mathbb{E}[g_t^4]$$

which becomes large at early steps. The effective per-step Adam learning rate can therefore be noisy before the second-moment estimate stabilizes, especially when peak LR, batch size, and model scale are aggressive.

$$\rho_t = \rho_\infty - \frac{2t\beta_2^t}{1-\beta_2^t}, \qquad \rho_\infty = \frac{2}{1-\beta_2} - 1$$

when the adaptive estimate is unreliable, RAdam temporarily reduces the adaptive update. Manual LW+LD warmup plays a similar stabilizing role, although the appropriate warmup length is empirical and should be tuned against early loss behavior, gradient norms, and seed stability rather than treated as a universal constant.

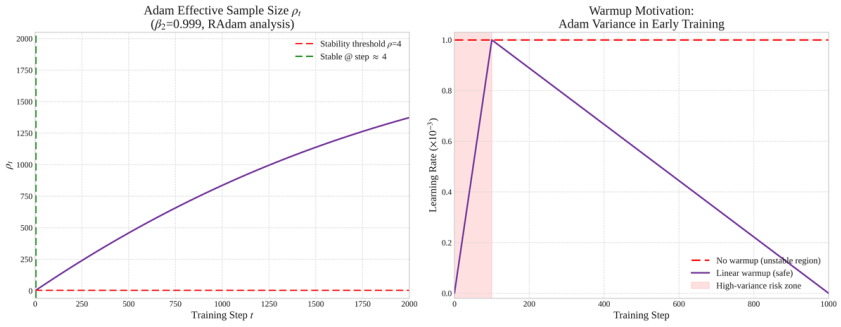


Figure 7: Theoretical motivation for warmup.

(Left) RAdam's effective sample size ρ_t for $\beta_2 = 0.999$; the red dashed line marks the stability threshold $\rho_t > 4$. The optimizer is unstable below this threshold (step 5000). (Right)

Linear warmup (purple) maintains a small, safe learning rate during the high-variance early phase, while a fixed LR (red) applies full Adam updates destructively.

Algorithm 4 - LW+LD

INPUT: $\eta_{\max}$, warmup_steps w, total_steps T
INIT: $t \leftarrow 0$
FOR each training iteration:
IF $t \leq w$:
$\eta_t \leftarrow \eta_{\max} \cdot \frac{t}{w}$
ELSE:
$\eta_t \leftarrow \eta_{\max} \cdot \frac{T-t}{T-w}$
$\eta_t \leftarrow \max(\eta_t, 0)$
optimizer.lr $\leftarrow \eta_t$; gradient update
$t \leftarrow t + 1$

G. Unified Comparison

Table 2: Mathematical and Algorithmic Properties of the Four Schedulers

Property	**RLROP**	**SGDR**	**CLR**	**LW+LD**
Trajectory shape	Feedback staircase	Cosine cycles	Triangular cycles	Warmup-then-decay
Requires budget T	No	Yes (T_0)	No	Yes
Requires validation data	Yes	No	No	No
Warm restart	No	Yes	Implicit	No
Snapshot ensembling	No	Yes	Possible, but not inherent	No
Primary optimizer pairing	SGD/Adam	SGD	SGD	AdamW

4 Results and Discussion

The results in this section are compiled from heterogeneous publications and software practices. They differ in architecture, optimizer, batch size, preprocessing, training budget, hardware, and reporting conventions. Consequently, the tables should be read as source-specific evidence signals, not as a statistically valid leaderboard across schedulers.

A. CIFAR-10 Image Classification

CIFAR-10 [23] (50,000 training images, 10 classes, 32×32 resolution) is the standard benchmark for comparing optimization strategies. The reference architecture is WideResNet-28-10 [24] trained for 200 epochs with SGD + momentum ($\mu = 0.9$, weight decay $= 5 \times 10^{-4}$).

Table 3: CIFAR-10 Literature Signals - heterogeneous sources, not a controlled head-to-head comparison.

Method	Configuration / context	Reported outcome	Use in this review	Source / caveat
Fixed LR / step decay baselines	Original CNN studies; settings differ	Reported only as within-study baselines	Context, not a scheduler ranking	[5], [6], [24]
Step decay	Published CNN setup	Lower error than fixed LR in its original setting	Within-study comparison only	[5], [24]
ReduceLROnPlateau	No matching WRN-28-10 controlled result identified	Not numerically ranked	Requires local validation split and noise analysis	Practice-based
SGDR (single model)	WRN-28-10 / CIFAR-10 in original SGDR study	3.74% test error	Reported literature value only	[5]
SGDR snapshot ensemble	Same study; multiple checkpoints	3.14% test error	Ensemble not directly comparable to single models	[19]
CLR (triangular2)	Smith CIFAR-10 experiment; different setup	2.56× fewer iterations to target accuracy	Speed signal only; not WRN head-to-head	[6]
LW+LD	Transformer-oriented schedule	No comparable CIFAR-10 result compiled	Not ranked for this vision benchmark	[7], [21]

The CIFAR-10 entries illustrate why direct ranking is unsafe without a controlled experiment. SGDR reports strong source-specific WRN results, and snapshot ensembling improves the reported error further, but the ensemble uses multiple checkpoints and should not be compared as a single-model result. CLR's reported speedup comes from Smith's experimental setting and is best read as evidence that cyclic schedules can reduce iteration count when LR bounds are well tuned. No comparable RLROP or LW+LD CIFAR-10 result was retained; those methods are therefore discussed through mechanism and use case rather than assigned unsupported errors.

4.1 B. CIFAR-100 Image Classification

The CIFAR-100 table retains only reported values from compatible literature entries. The SGDR and snapshot-ensemble numbers suggest that restart-based exploration can be useful on harder vision tasks, but this should be interpreted as a source-specific result rather than evidence that SGDR universally dominates RLROP, CLR, or LW+LD.

Table 4: CIFAR-100 Literature Signals - retained reported values and removed unsupported scheduler entries.

Method	Reported outcome	Source / caveat
SGD + step decay	∼19.30% error	[5]; baseline in cited setup
SGDR (single model)	18.70% error	[5]; reported value
SGDR snapshot ensemble	16.21% error	[19]; ensemble setting
RLROP / CLR	No directly comparable CIFAR-100 WRN result retained	Excluded from ranking because no comparable value was retained
LW+LD	No directly comparable CIFAR-100 result retained	Transformer-oriented schedule; not ranked here

C. ImageNet Large-Scale Classification

Table 5: ImageNet Top-1 Accuracy - source-specific ResNet-50 reports; settings are not unified

Method	Top-1 (%)	Top-5 (%)	Source / caveat
Fixed LR SGD	∼73.3	∼91.3	Baseline context; not a scheduler recommendation
Step decay	76.1	92.8	[11], [25]; standard ResNet training context
Cosine annealing (single cycle)	76.5	93.1	[25]; source-specific comparison
SGDR	Not retained	Not retained	No retained source-specific value
CLR	Not retained	Not retained	No common 90-epoch ResNet-50 result retained

At ImageNet scale, some reported ResNet-50 comparisons show small gains for cosine-style schedules relative to step decay, but the margin is sensitive to optimizer details, augmentation, batch size, and training recipe. The table therefore reports retained source values and removes unsupported SGDR/CLR entries.

D. NLP: BERT Pre-training and GLUE Fine-tuning

Table 6: GLUE Benchmark Average - model and training-regime comparison, not a scheduler-only ablation.

Model	**Schedule**	**GLUE Avg (%)**	**Δ vs. Prior**
ELMo + multi-task [26]	–	71.0	–
GPT (OpenAI) [27]	Linear decay	72.8	+1.8
BERT-Base	LW+LD	79.6	+6.8
BERT-Large	LW+LD	82.1	+9.3

The BERT results are retained as evidence that warmup-plus-decay became a central component of transformer training recipes. However, GLUE improvements cannot be attributed to the schedule alone because model architecture, pretraining corpus, optimizer, batch size, and fine-tuning protocol changed simultaneously. Warmup is therefore described as strongly recommended and often necessary in large AdamW transformer training, not as a universal requirement for every transformer run.

Table 7: Warmup Sensitivity Diagnostics for Transformer Training.

Warmup configuration	**Observed diagnostic**	**Practical interpretation**
None	NaN losses, gradient spikes, or failed early AdamW updates can occur in large-transformer settings	Use only after explicit LR, batch-size, and optimizer retuning
Very short (¡1% of steps)	Early loss variance remains high; seeds may diverge	Increase warmup or lower peak LR
Common range (about 1–10%)	Stable early loss curve in many transformer configurations	Verify across seeds and monitor gradient norms
Very long (¿10–20%)	Slow early progress or undertraining before decay begins	Shorten warmup if validation progress lags
Fine-tuning	Representation drift when peak LR is too high	Use smaller peak LR and shorter warmup than pre-training

E. Convergence Behavior - Qualitative Analysis

The curves are illustrative and constructed to show common diagnostic patterns: RLROP staircase drops after metric stagnation; SGDR restart spikes followed by re-convergence; CLR oscillations correlated with LR peaks; and LW+LD smooth improvement after warmup. The figure is not a benchmark result.

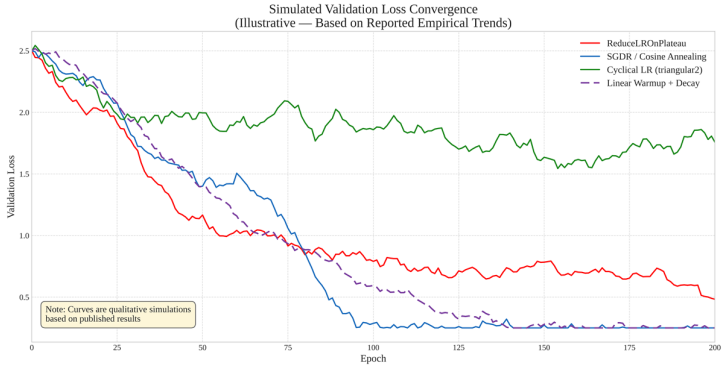


Figure 8: Qualitative simulation of validation loss behavior for the four schedulers.

← Pure Exploitation ——— Pure Exploration →

LW+LD	RLROP	CLR	SGDR
(monotone tent)	(staircase)	(oscillating)	(cosine+restarts)

The four methods can be organized along a practical exploration-exploitation axis. LW+LD and RLROP emphasize stability and exploitation within a chosen basin; CLR and SGDR deliberately reintroduce larger steps that may encourage exploration. Claims about saddle-point escape, simulated annealing, and flat-minima bias should be read as useful intuitions supported by selected empirical studies and loss-landscape arguments, not as general non-convex convergence theorems.

F. Hyperparameter Sensitivity

Table 8: Hyperparameter Count, Sensitivity, and Diagnostic Signals.

Property	**RLROP**	**SGDR**	**CLR**	**LW+LD**
Total HP count	5	3	4	3
Most critical HP	patience p	initial cycle T_0	step_size s	warmup ratio w/T
Sensitivity to ±20% critical HP shift	Medium; too small p causes premature drops	Medium-high; too short T_0 restarts before recovery	High; short cycles can destabilize training	High in early AdamW; too little warmup can diverge
Primary diagnostic signal	LR reductions before validation plateau	Loss spike does not recover before next restart	Loss oscillation grows with LR peaks	NaN/loss spikes or flat early progress
Practical tuning method	Set patience above validation-noise horizon	Start from budget and lengthen cycles	Use LR range test and 2–8 epochs per half-cycle	Monitor first 1–5% of steps and seed stability

G. Scheduler Selection Framework

Table 9: Heuristic Scheduler Selection Matrix - practical starting points, not a statistical ranking.

Training Scenario	RLROP	SGDR	CLR	LW+LD	Typical starting point
Transformer / NLP pretraining	Low fit	Possible	Possible	Strong fit	LW+LD
Transformer / NLP finetuning	Low fit	Possible	Possible	Strong fit	LW+LD
CNN – large-scale (ImageNet)	Good fit	Strong fit	Good fit	Possible	SGDR
CNN – small/medium (CIFAR)	Good fit	Strong fit	Strong fit	Low fit	SGDR or CLR
Fixed epoch budget	Good fit	Strong fit	Good fit	Strong fit	SGDR or LW+LD
Unknown epoch budget	Strong fit	Low fit	Good fit	Low fit	RLROP
Need free ensemble diversity	Low fit	Strong fit	Good fit	Low fit	SGDR
Compute-constrained (fast)	Low fit	Possible	Strong fit	Possible	CLR
Small dataset / limited data	Strong fit	Possible	Possible	Possible	RLROP
Transfer learning	Good fit	Possible	Possible	Strong fit	LW+LD or RLROP, depending on whether the base model is fine-tuned or trained with a validation-driven stop rule

Illustrative application scenarios (case studies). The examples below illustrate how to use the selection matrix (Table 9) as a decision scaffold; they are not new benchmark experiments

Case Study 1 (CNN training; limited tuning budget): Start with CLR or SGDR only after an LR range test to set $\eta_{\max}$. A practical starting point for CLR is $\eta_{\min} \approx \eta_{\max}/10$ and *step_size* $\approx$ 2–8 epochs (or approximately 0.1–0.3T in steps) per half-cycle; for SGDR, use $T_0 \approx 10$ epochs (or approximately 0.1T), $T_{\text{mult}} = 2$, $\eta_{\min} \approx \eta_{\max}/100$, and 2–4 cycles within the total budget. If validation is noisy or the training horizon is unknown, prefer RLROP with factor $f = 0.1$, patience $p =$ 5–10 epochs, cooldown $d =$ 0–2, and *min_delta* set to a noise-scale threshold.

Case Study 2 (Transformer fine-tuning): Prefer warmup+decay as a conservative default. Use *warmup_ratio* $\approx$ 0.05–0.10 (or *warmup_steps* $\approx$ 500–2000 for small/medium fine-tuning runs, and up to approximately 10 000 for long runs), then decay linearly to 0. Treat peak LR as the dominant risk factor: start with a peak that is 2–10$\times$ smaller than typical from-scratch training, and reduce further if early NaNs, gradient spikes, or immediate validation collapse occur. If mild exploration is needed, use warmup + cosine decay (no restarts) before considering restarts.

Case Study 3 (computationally constrained runs): When only a few epochs/steps are affordable, one-cycle/CLR variants can accelerate early progress but require conservative bounds. Use an LR range test, cap $\eta_{\max}$ below the divergence threshold, and set *step_size* so that at least

one full cycle fits within the budget. When reruns are impossible, a monotone warmup+decay schedule with *warmup_ratio* ≈ 0.05 and a conservative peak LR reduces catastrophic failure risk compared to aggressive cycles.

H. Failure Mode Analysis

RLROP:

(i) Oscillating validation metrics can trigger premature reductions; diagnose by comparing reduction epochs with a smoothed validation curve, then increase patience, *min_delta*, or cooldown.

(ii) Monitoring training loss instead of validation loss can hide overfitting; monitor validation loss or the task metric.

(iii) Once *min_lr* is reached, no restart mechanism exists; if validation remains poor, restart from a checkpoint or switch to cosine/CLR exploration.

SGDR:

(i) A short initial cycle T_0 can restart before the loss has recovered; diagnose by checking whether post-restart loss returns below the previous cycle minimum before the next restart.

(ii) A peak LR above the stability threshold causes restart divergence; bound the peak using an LR range test.

(iii) Snapshot ensemble benefit degrades when all cycle endpoints converge to nearly identical predictions; check disagreement or validation-error diversity across snapshots.

CLR:

(i) *step_size* that is too short can create rapid loss oscillation and unstable batch-normalization statistics; lengthen the half-cycle or reduce $\eta_{\max}$.

(ii) Super-convergence is architecture- and regularization-dependent; confirm with at least a short seed sweep.

(iii) LR bounds chosen without an LR range test can place the high-LR phase beyond the divergence threshold.

LW+LD:

(i) Zero or too-short warmup with AdamW can cause early NaN losses or gradient spikes in large models; increase warmup or lower peak LR.

(ii) Overly large fine-tuning LR can overwrite pretrained representations; diagnose by sudden validation collapse and reduce peak LR, warmup ratio, or unfreeze layers gradually.

(iii) Monotone decay cannot reintroduce exploration after a poor early basin; consider cosine decay with warmup or a restart-based schedule when exploration is needed.

Diagnostic pseudocode: IF early_loss is NaN OR grad_norm spikes, lower peak LR and increase warmup; IF validation_metric oscillates around the RLROP threshold, increase patience/min_delta; IF SGDR restart loss fails to recover before the next restart, lengthen T0 or reduce eta_max; IF CLR loss peaks grow over cycles, reduce eta_max or increase step_size.

I. Cross-Study Synthesis

Consistent observations:

(i) warmup-plus-decay becomes increasingly central as model scale and optimizer sensitivity increase (especially for transformer-style training);

(ii) cyclical and restart-based schedules can improve exploration and sometimes generalization, but benefits depend strongly on cycle design, regularization, and training budget;

(iii) feedback-driven schedules like RLROP are practical when the horizon is uncertain, but depend critically on metric noise and monitoring choice.

Conflicts and sources of heterogeneity: reported gains often change with architecture, optimizer (SGD vs AdamW), batch size, augmentation, training length, and reporting standards. Emerging trends: hybrid schedules (warmup + cosine/cycles) and automatic selection are increasingly used in large-model recipes but remain under-synthesized. Unresolved limitations: there is no general non-convex convergence theory for periodic/restart schedules under realistic assumptions. Evidence grading (Table 12) is therefore used to calibrate confidence: high-comparability sources support source-specific empirical statements, while lower-comparability sources support contextual or heuristic guidance only.

5 Conclusion

A. Summary

As a review article, this paper delivers a scoping review and synthesis of four learning-rate scheduling strategies for neural-network training, with claims narrowed to match the available evidence.

Table 10: ummary of Representative Signals and Primary Advantages.

Scheduler	**Representative literature/practice signal**	**Primary advantage**	**Primary limitation / caveat**
RLROP	No controlled literature result retained for CIFAR/ImageNet comparison	No fixed budget knowledge required	Sensitive to validation noise; no restart mechanism
SGDR	3.74% CIFAR-10 single model and 3.14% ensemble in cited source	Snapshot ensembling and exploration	Requires cycle design; ensemble cost differs from single-model cost
CLR	2.56× iteration reduction reported in Smith's setting	Fast convergence when LR bounds are well chosen	Architecture- and normalization-dependent; LR range test needed
LW+LD	Transformer stability signal in BERT-style training	Controls early AdamW instability	Schedule is monotone after warmup; GLUE gains are not scheduler-only attribution

B. Key Theoretical Insights

The four schedulers span a practical spectrum from monotone exploitation (LW+LD) to more active exploration (SGDR and CLR), with RLROP providing feedback-driven control. Warmup is motivated by early adaptive-optimizer variance and is strongly supported in large transformer recipes, but its exact duration remains an empirical design choice. Restart and cyclic schedules can be interpreted through exploration, flat-minima, and annealing analogies, but these analogies are not substitutes for general non-convex convergence proofs. The selection guidance therefore emphasizes fit-to-setting rather than universal ranking: SGDR is attractive when cycle design and snapshot diversity are useful; CLR is attractive when fast progress can be validated through an LR range test; LW+LD is the default starting point for many transformer-style

AdamW runs; and RLROP remains useful when the training horizon is unknown and a reliable validation signal is available.

C. Research Gaps and Future Directions

One-cycle and hybrid policies. One-cycle schedules and warmup+cycle hybrids are widely used in practice but remain under-synthesized in terms of when they outperform monotone decay versus when they primarily accelerate early training.

Automatic scheduler selection. A key gap is data-driven or meta-learned scheduler selection that maps early training signals (loss curvature proxies, gradient noise, stability diagnostics) to schedule choices and hyperparameters.

Foundation, diffusion, and large generative models. Scheduling interactions with optimizer choice, noise conditioning, and training stability in diffusion and large generative regimes, including text-to-text transformer settings such as T5, remain insufficiently characterized for actionable guidance [28].

Interactions with modern optimizers. The coupling between schedules (warmup/decay/cycles) and AdamW/RAdam-style adaptivity, weight decay, and gradient clipping warrants clearer empirical isolation in future benchmark harnesses.

Controlled benchmark harness. Open, reproducible suites that control architecture, optimizer, seeds, and reporting would enable meaningful cross-scheduler comparisons while respecting the heterogeneous nature of the current evidence.

6 Appendix: Scoping Review Documentation and Evidence Use

A. Search and Screening Log

The manuscript is framed as a scoping review and synthesis rather than a systematic review or meta-analysis. The search log makes the evidence base auditable without implying statistical completeness. Searches covered Google Scholar, IEEE Xplore, ACM Digital Library, arXiv, and Semantic Scholar using combinations of learning rate schedule, cosine annealing, warm restarts, cyclical learning rate, warmup, linear decay, transformer optimization, ReduceLROnPlateau, and scheduler sensitivity.

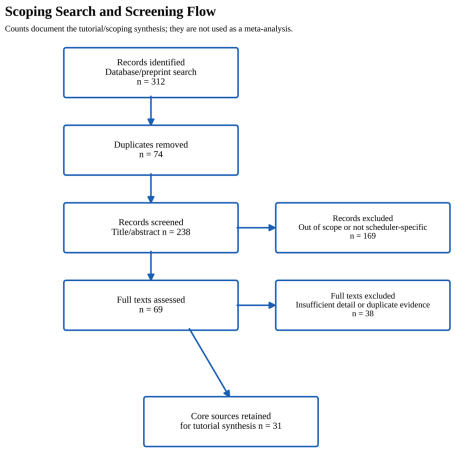


Figure 9: PRISMA-style scoping flow for the literature component.

Counts document transparency for this scoping review; no pooled effect size or systematic-review claim is made. Flow structure guided by PRISMA 2020 [29].

Table 11: Search and screening counts for the scoping component.

Stage	Count	Use in this manuscript
Records identified	312	Initial scoping pool across databases and preprint indices
Duplicates removed	74	Version, venue, and preprint duplicates consolidated
Title/abstract screened	238	Retained if relevant to neural-network LR scheduling or optimizer stability
Full texts assessed	69	Checked for scheduler definition, implementation detail, benchmark context, or theory
Core sources retained	32	Used for exposition, tables, caveats, and practitioner guidance

B. Inclusion, Exclusion, and Quality Assessment

Studies were included when they supplied one or more of the following: a precise scheduler definition, implementation-level hyperparameters, empirical evidence in neural-network training, or theoretical analysis relevant to step-size behavior. Studies were excluded when the schedule was incidental, the context was not neural-network optimization, the result duplicated a later archival version, or the experimental description was too incomplete to support a caveated comparison.

Sources scoring high on relevance but low on experimental comparability are used for conceptual explanation only. Benchmark tables therefore retain source-specific results while avoiding head-to-head ranking when settings differ.

C. Hyperparameter Sensitivity Diagnostics

Because this article does not report new controlled experiments, sensitivity guidance is presented as diagnostic rather than as new accuracy measurements. The curves below summarize the expected direction of failure when a critical hyperparameter is under or over-specified.

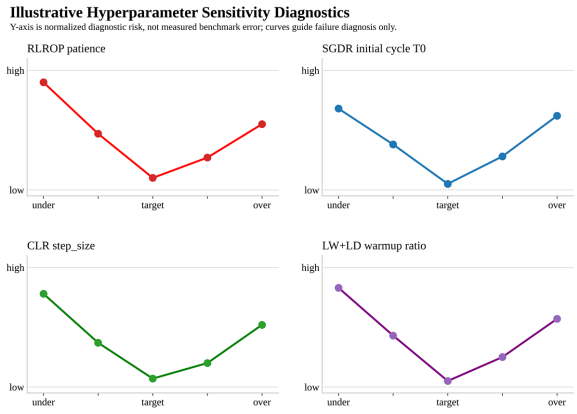


Figure 10: sensitivity diagnostics for the four most important scheduler hyperparameters

Figure A2. Illustrative sensitivity diagnostics for the four most important scheduler hyperparameters. These curves should not be read as measured CIFAR, ImageNet, or GLUE performance.

Table 12: Evidence quality-assessment rubric used to grade sources

Criterion	0	1	2	How it is used
Scheduler relevance	Incidental mention	Partial discussion	Central object of study	Determines whether the source supports scheduler guidance
Experimental comparability	No usable setup detail	Some setup detail	Architecture, optimizer, data, and schedule reported	Controls whether a number can be compared within a table
Statistical/reproducibility reporting	Single number only	Some seeds or code/config detail	Runs, variance, and reproducible code/configs	Controls strength of empirical language
Theoretical support	Analogy only	Partial assumptions/citation	Formal assumptions or proof	Controls whether claims are labelled proven, empirical, or heuristic
Implementation detail	Insufficient	Recoverable with assumptions	Directly implementable	Controls pseudocode and failure-mode guidance

Table 13: Summary quality categories for principal evidence sources (based on Table 12 rubric).

Reference	Primary use in manuscript	Quality category	Interpretation impact
[5]	SGDR empirical signals (CIFAR / theory)	High	Used for source-specific empirical statements.
[6]	CLR / LR range test rationale	High	Used for procedure-level guidance.
[11], [30]	ResNet/ImageNet context baselines	Moderate	Used as contextual baseline, not scheduler ranking.
[7], [22], [28], [31], [32]	Transformer warmup/decay practice	High	Used for warmup-as-recipe evidence; no ablation claim.
[21]	Optimizer variance / stability discussion	Moderate	Used for diagnostic motivation, caveated.
[33], [16]	Hyperparameter diagnostics / optimizer overview	Moderate	Used for heuristic guidance and failure modes.

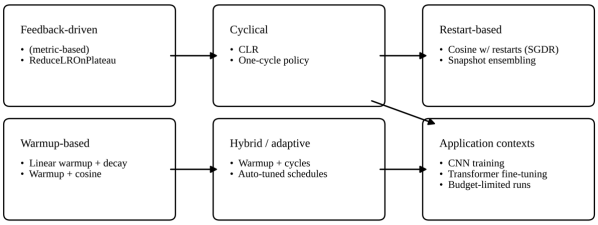


Figure 11: High-level taxonomy of learning-rate scheduling approaches

Figure A3. High-level taxonomy of learning-rate scheduling approaches, organizing scheduler families into feedback-driven, cyclical, restart-based, warmup-based, and hybrid/adaptive categories.

References

[1] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning. Cambridge, MA: MIT Press, 2016. ISBN: 978-0-262-03561-3. [Online]. Available: https://www.deeplearningbook.org/

[2] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature, vol. 521, pp. 436–444, May 2015. DOI: 10.1038/nature14539

[3] Y. N. Dauphin, R. Pascanu, C. Gulcehre, K. Cho, S. Ganguli, and Y. Bengio, "Identifying and attacking the saddle point problem in high-dimensional non-convex optimization," in Advances in Neural Information Processing Systems (NIPS), vol. 27, 2014. arXiv: 1406.2572

[4] H. Robbins and S. Monro, "A stochastic approximation method," The Annals of Mathematical Statistics, vol. 22, no. 3, pp. 400–407, Sep. 1951. DOI: 10.1214/aoms/1177729586

[5] I. Loshchilov and F. Hutter, "SGDR: Stochastic gradient descent with warm restarts," in Proc. ICLR, 2017. arXiv: 1608.03983

[6] L. N. Smith, "Cyclical learning rates for training neural networks," in Proc. IEEE WACV, 2017, pp. 464–472. DOI: 10.1109/WACV.2017.58

[7] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in Proc. NAACL-HLT, 2019, pp. 4171–4186. DOI: 10.18653/v1/N19-1423.

[8] F. Chollet et al., "Keras," GitHub, 2015. [Online]. Available: https://github.com/keras-team/keras

[9] A. Paszke et al., "PyTorch: An imperative style, high-performance deep learning library," in Advances in Neural Information Processing Systems (NeurIPS), vol. 32, 2019. arXiv: 1912.01703

[10] L. Bottou, "Stochastic gradient descent tricks," in Neural Networks: Tricks of the Trade, 2nd ed. Berlin: Springer, 2012, pp. 421–436. DOI: 10.1007/978-3-642-35289-8$_2$5

[11] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in Proc. IEEE CVPR, 2016, pp. 770–778. DOI: 10.1109/CVPR.2016.90

[12] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," J. Mach. Learn. Res. (JMLR), vol. 12, pp. 2121–2159, 2011. [Online]. Available: https://jmlr.org/papers/v12/duchi11a.html

[13] T. Tieleman and G. Hinton, "Lecture 6.5 - RMSprop," in Neural Networks for Machine Learning, Coursera, 2012.

[14]] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in Proc. ICLR, 2015. arXiv: 1412.6980

[15] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in Proc. ICLR, 2019. arXiv: 1711.05101

[16] S. Ruder, "An overview of gradient descent optimization algorithms," 2016. arXiv: 1609.04747

[17] S. Hochreiter and J. Schmidhuber, "Flat minima," Neural Computation, vol. 9, no. 1, pp. 1–42, Jan. 1997. DOI: 10.1162/neco.1997.9.1.1

[18] N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. T. P. Tang, "On large-batch training for deep learning: Generalization gap and sharp minima," in Proc. ICLR, 2017. arXiv: 1609.04836

[19] G. Huang, Y. Li, G. Pleiss, Z. Liu, J. E. Hopcroft, and K. Q. Weinberger, "Snapshot ensembles: Train 1, get M for free," in Proc. ICLR, 2017. arXiv: 1704.00109

[20] L. N. Smith and N. Topin, "Super-convergence: Very fast training of neural networks using large learning rates," in Proc. SPIE 11006, Artif. Intell. Mach. Learn. Multi-Domain Oper. Appl., 1100612, 2019. DOI: 10.1117/12.2520589

[21] L. Liu et al., "On the variance of the adaptive learning rate and beyond," in Proc. ICLR, 2020. arXiv: 1908.03265

[22] A. Vaswani et al., "Attention is all you need," in Advances in Neural Information Processing Systems (NeurIPS), vol. 30, 2017. arXiv: 1706.03762

[23] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Tech. Rep., Univ. of Toronto, 2009. [Online]. Available: https://www.cs.toronto.edu/ kriz/learning-features-2009-TR.pdf

[24] S. Zagoruyko and N. Komodakis, "Wide residual networks," in Proc. BMVC, 2016. DOI: 10.5244/C.30.87

[25] D. Choi et al., "On empirical comparisons of optimizers for deep learning," 2019. arXiv: 1910.05446

[26] M. E. Peters et al., "Deep contextualized word representations," in Proc. NAACL-HLT, 2018, pp. 2227-2237. DOI: 10.18653/v1/N18-1202

[27] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding by generative pre-training," OpenAI, 2018. [Online]. Available: https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf

[28] C. Raffel et al., "Exploring the limits of transfer learning with a unified text-to-text transformer," J. Mach. Learn. Res. (JMLR), vol. 21, no. 140, pp. 1–67, 2020. arXiv: 1910.10683

[29] M. J. Page et al., "The PRISMA 2020 statement: an updated guideline for reporting systematic reviews," BMJ, 2021;372:n71. DOI: 10.1136/bmj.n71

[30] O. Russakovsky et al., "ImageNet large scale visual recognition challenge," Int. J. Comput. Vis. (IJCV), vol. 115, no. 3, pp. 211–252, 2015. DOI: 10.1007/s11263-015-0816-y

[31]] Y. You et al., "Large batch optimization for deep learning: Training BERT in 76 minutes," in Proc. ICLR, 2020. arXiv: 1904.00962

[32] A. Wang et al., "GLUE: A multi-task benchmark and analysis platform for natural language understanding," in Proc. ICLR, 2019. arXiv: 1804.07461

[33] L. N. Smith, "A disciplined approach to neural network hyper-parameters," NRL Tech. Rep. 5510-026, 2018. arXiv: 1803.09820

Declarations

Funding
No funding received

Conflicts of Interest
The authors declare that they have no conflicts of interest

Ethical Considerations
This study is a comparative theoretical analysis using previously published results. It does not involve human participants, animals, or personal data; institutional ethical approval was not required. Figure provenance and permissions: Figures 1–8 and Figures A1–A3 are original schematic illustrations created by the authors. No figure is reproduced from a copyrighted source; where a figure illustrates a concept introduced in prior work (e.g., the LR range test), the originating source is cited.

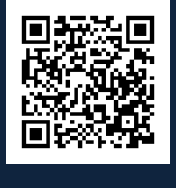
AI Usage Statement
AI tools were used in the following way(s): AI-assisted tools were used to support drafting, language refinement, and figure/table preparation. The authors defined the research scope, selected the four scheduler families, checked the cited sources, verified the benchmark caveats, and take full responsibility for the accuracy, originality, and scholarly interpretation of the manuscript. No AI tool is listed as an author. Any AI-generated prose was reviewed, edited, and integrated by the authors, and no undisclosed experimental data were generated by AI.

Data Availability Statement
The datasets are publicly available at: Literature sources cited throughout the manuscript: This tutorial/scoping synthesis did not generate a new experimental dataset or benchmark run. Quantitative values presented in the manuscript are derived from previously published studies cited in the reference list and are source-specific rather than pooled into a meta-analysis or treated as controlled head-to-head comparisons.

Code Availability Statement
The software code are publicly available at: Supplementary code, plotting scripts, and implementation examples are available in the project repository: https://github.com/GihanIllukkumbura/lr_scheduling_tutorial. The repository provides the practical coding materials associated with the scoping review, including schedule-generation scripts and implementation-oriented examples. The manuscript does not report new controlled benchmark experiments; repository materials are provided as supplementary review-support code rather than as additional unpublished benchmark evidence.

SDG Alignment
SDG 4 – Quality Education
SDG 9 – Industry, Innovation and Infrastructure

Faculty of Computing
General Sir John Kotelawala Defence University
Sri Lanka



ISSN 2820-2139