# International Journal of Secondary Computing and Applications Research

Volume 3, Issue 3

EDITOR-IN-CHIEF: DR. MARIA HWANG

SEPTEMBER, 2026

# Proceedings of International Journal of Secondary Computing and Applications Research, Vol 3, Issue 3

September 8, 2026

## Letter from the Editor-in-Chief

It is my pleasure to welcome you to this issue of the International Journal of Secondary Computing and Applications Research (IJSCAR). The five articles gathered here reflect just how wide the reach of student-led computing research has become, spanning mobile sensing and on-device machine learning for at-home physical rehabilitation, reproducible data-engineering pipelines for behavioral survey research, lightweight transformer models for mental health text triage, the cognitive science of active versus passive learning, and lightweight machine learning for IoT intrusion detection.

Taken together, these papers show a maturing research instinct: a willingness to engage seriously with real deployment constraints, not just idealized benchmarks. Several authors built and tested working prototypes on actual hardware, released reproducible software artifacts alongside their write-ups, or ran adversarial and sensitivity checks on their own results rather than reporting a single favorable configuration—exactly the kind of self-scrutiny that distinguishes rigorous research from a class project.

At IJSCAR, our mission remains to provide a platform for students and early-career researchers to share innovative ideas, engage with the broader research community, and contribute to the advancement of knowledge. We are continually inspired by the quality of submissions we receive and by the dedication of young scholars pursuing meaningful and impactful research, and we remain committed to nurturing that curiosity issue after issue.

As Editor-in-Chief, I want to thank our authors for the care they put into this work, their mentors and teachers for the guidance behind the scenes, and our reviewers for the time they gave to strengthening each submission. On behalf of the editorial team, I am also grateful to our readers for their continued support in helping foster a community committed to learning, discovery, and scientific progress.

We hope this issue informs and inspires, and we look forward to sharing the next chapter of student computing research with you soon.

Sincerely,
Maria Hwang
Editor-in-Chief

---

Volume 3, Issue 3 September 8, 2026

DOI: 10.67149/yhjs2024.5/j4kw3mjo | https://ijscar.org/pubs/vol3-issue3.pdf

# Contents

# PostureScore: An On-Device, Phase-Aware Scoring Pipeline for At-Home Rehabilitation

Ella Wang
Emma Willard School
Troy, New York, USA
ellawang0710@hotmail.com

## Abstract

After surgery, many patients do their rehab exercises at home without knowing whether their form is correct. This project originated from the author's own post-surgical rehabilitation (ACL reconstruction, 2025; shoulder arthroscopy, six months later), which surfaced three engineering gaps in existing tools. Smartphone pose models like MediaPipe Pose only output landmarks; the tools that wrap them typically use one fixed target angle regardless of rehabilitation week; and many upload video to a server, which fellow patients met during rehabilitation were uncomfortable with. PostureScore is a prototype that addresses these three gaps in one mobile app. Its core is a scoring formula combining whether the patient holds the movement long enough and whether the movement is steady, with targets that change by rehabilitation week (wider tolerance early, narrower later). The privacy design sends no video at all; only session-summary numbers (total reps, average score, peak range of motion, and similar) reach the backend. The pipeline was measured at 38.5 fps on iPhone 16 and 31.0 fps on iPhone 14 and deployed to seven post-surgical patients over an 8- to 12-week window. The system ran across all 664 sessions and produced consistent per-rep scores. The work is reported as an engineering feasibility check on the prototype. It does not provide evidence that PostureScore improves clinical outcomes.

## CCS Concepts

• **Human-centered computing** → **Ubiquitous and mobile computing**; • **Applied computing** → *Consumer health*; • **Computing methodologies** → *Activity recognition and understanding*.

## Keywords

On-Device AI, Pose Estimation, Rehabilitation, Mobile Health, Movement Scoring, Privacy-Preserving Design

**ACM Reference Format:**
Ella Wang. 2026. PostureScore: An On-Device, Phase-Aware Scoring Pipeline for At-Home Rehabilitation. In *Proceedings of International Journal of Secondary Computing and Applications Research (IJSCAR VOL. 3, ISSUE 3)*. ACM, New York, NY, USA, 10 pages. https://doi.org/10.67149/yhjs2024.5/17iljp8h


*IJSCAR VOL. 3, ISSUE 3*


## 1 Introduction

Post-surgical rehabilitation is mostly a self-directed activity that happens at home. After a procedure such as anterior cruciate ligament (ACL) reconstruction or shoulder arthroscopy, the patient is usually discharged within a day or two with a printed list of exercises and an outpatient appointment weeks out. The period between discharge and that appointment is the longest stretch of unsupervised rehabilitation, and it is also when adherence drops the most. A 2025 prospective cohort of 240 lower-limb orthopedic postoperative patients found that only 53.7% complied with the prescribed rehabilitation protocol at one-month follow-up [1], and a 2025 case-series review across six National Health Service trusts in the United Kingdom reported that 29% of ACL patients failed to complete their rehabilitation [2]. The exercises themselves are rarely the issue. The gap is that the patient has no objective signal between clinic visits about whether they are performing them correctly.

The obvious solution is to put a movement-sensing system in the patient's pocket. Smartphones have cameras and enough compute to run pose estimation locally. MediaPipe Pose (BlazePose [3]) runs at interactive frame rates on regular phones (natively and, as measured in §6, as WebAssembly inside a mobile WebView), and its 2D joint angles are close to motion-capture references on rehabilitation movements [4, 5]. This could give the at-home patient the missing feedback signal, but an off-the-shelf pose model does not get there on its own: wrapping it in a UI leaves three engineering gaps.

*Gap 1: feedback resolution.* A pose model returns landmarks, not a score. Tools that wrap MediaPipe in a rehab UI typically collapse the landmark stream to a binary "correct / incorrect" or green/red cue, which tells the user neither how far off they were nor anything about the movement's *duration*.

*Gap 2: phase blindness.* The same exercise has very different correct ranges in week 1 and week 12: a target set for the late phase is simply out of reach in the first weeks; one set for the early phase stops challenging patients as recovery progresses. Tools that ship a single fixed target per exercise therefore either discourage early-week patients or under-stretch late-week ones.

*Gap 3: privacy boundary.* Several existing tools upload video to a cloud service for processing. Fellow patients the author met during rehabilitation (§3.3) were uncomfortable with this, and the upload is unnecessary: pose estimation runs locally on a phone already capable of much heavier ML workloads.

PostureScore is a prototype that addresses these three gaps. The app is built in React + TypeScript, packaged for iOS using Capacitor, and runs MediaPipe Pose inside the phone's WebView so camera frames are processed locally. A small Node.js / PostgreSQL backend only ever sees session-level summary numbers, never frames or per-frame data. This paper makes three contributions: a scoring formula (§4) that turns a stream of pose landmarks into a 0–100 score per repetition by combining hold time and movement steadiness; a privacy design (§5) that keeps raw video in the phone's WebView process and only sends summary numbers off-device; and a set of design choices (early-week tolerance, per-rep voice feedback, on-device inference) that come directly from the author's prior rehabilitation experience (§3.3) rather than from the published literature. The system was evaluated on seven post-surgical patients over an 8- to 12-week window (§7); across 664 sessions the pipeline ran end-to-end without recorded failures.

## 2 Related Work

This work does not try to improve MediaPipe itself. It uses MediaPipe as a tool and focuses on what can be built on top of it: a scoring rule, week-by-week targets, and a privacy-preserving app. Three things from the literature shaped that choice.

**Pose accuracy on phones.** Several studies find MediaPipe Pose's 2D joint angles close enough to motion capture for rehab feedback when the camera is set up reasonably [4, 5], though accuracy depends on viewpoint [6, 7]. That justified using it off-the-shelf rather than training a new model; the trade-off is that the underlying angle accuracy cannot be improved here, only built upon.

**Digital rehab tools and adherence.** A broader literature has looked at whether digital rehab tools help patients stick with exercises and recover better [8, 9]. Commercial systems like Sword Health publish operational cost-of-care reports [10], but none of these papers publishes its actual scoring formula in enough detail to reproduce. That is why (3) and Table 3 are written out in full, so the next student can reuse them.

**On-device vs. cloud privacy.** Two things motivated keeping frames on the phone. The author's prior rehabilitation experience (§3.3) surfaced patient discomfort with video upload, and Apple's Privacy Manifest framework [11], mandatory since 1 May 2024, requires cloud-uploading apps to declare a lot more than on-device ones. Walker *et al.* also report that tool-level trust is a real barrier to ACL rehabilitation adherence [12].

## 3 System Design

### 3.1 Overall architecture

PostureScore is a four-layer system (Figure 1). The layout is designed so that raw video stays inside the WebView on the user's phone; no code path in the app writes a frame to disk or sends it off the device.

(L1) **UI layer** — a React + TypeScript app showing exercise selection, plan navigation, on-canvas overlays, and post-session summaries.

(L2) **Native shell** — Capacitor [13] wraps the app in an iOS WebView and handles camera permission, the iOS privacy manifest, and HTTPS. Using Capacitor (instead of writing native Swift) lets the same code also run as a mobile-web app.

(L3) **Pose inference + scoring** — MediaPipe Pose v0.5 runs as WebAssembly inside the WebView; each frame is consumed and overwritten in place, and only the 33 per-frame landmarks cross into application code. The 20-frame smoothing, rep state machine, and scoring formula (3) are TypeScript modules in the same WebView context, grouped as L3 because they all touch per-frame data. L1 receives, read-only, the per-frame landmarks (to draw the on-canvas overlay) together with the per-rep score and the end-of-session summary; it never writes back to the frame or pose buffers.

(L4) **Backend** — a small Node.js + PostgreSQL backend storing week-indexed plans, the per-week standards (§4.6), and one scalar summary record per session (fields under G2 in §5). It never sees frames, landmarks, or per-frame angles; the schema has no field for them.

### 3.2 Component inventory

Table 1 lists the principal components and the maximum data scope each touches; §5 spells out the trust model behind the on-device/off-device split.

### 3.3 Patient-Informed Design Rationale

Three design choices in PostureScore come from my own ACL (June 2025) and shoulder (December 2025) post-surgical rehabilitation rather than from the literature. All hand-authored values in Table 3 were reviewed by the clinical advisor (Acknowledgements) before deployment.

*Wider angle tolerance in early weeks (15° at week 1, narrowing to 10° at week 12; full table in §4.6).* During my early ACL recovery, small deviations triggered "incorrect" feedback in the apps I tried, which made sessions demoralizing and cut my practice frequency. Week-1 tolerance is set wide for an engagement reason rather than a clinical one. Clinically, a narrower tolerance would be more correct, but early in recovery the main limiter on practice is whether a session feels achievable, so the wide setting trades strict scoring for engagement.

*Per-repetition voice feedback.* During shoulder rehabilitation I found summary feedback at session end insufficient to correct mid-session form drift: by the time I learned rep 4 was wrong, I had already done reps 5–10. The system therefore speaks the score immediately when each rep terminates (§4.4), with a 2-second debounce to avoid stacked utterances on fast reps.

*Keeping video on the device, no exceptions.* Fellow patients I met during my rehab at Fu's Orthopedic Hospital (Hangzhou) repeatedly said they were uncomfortable with

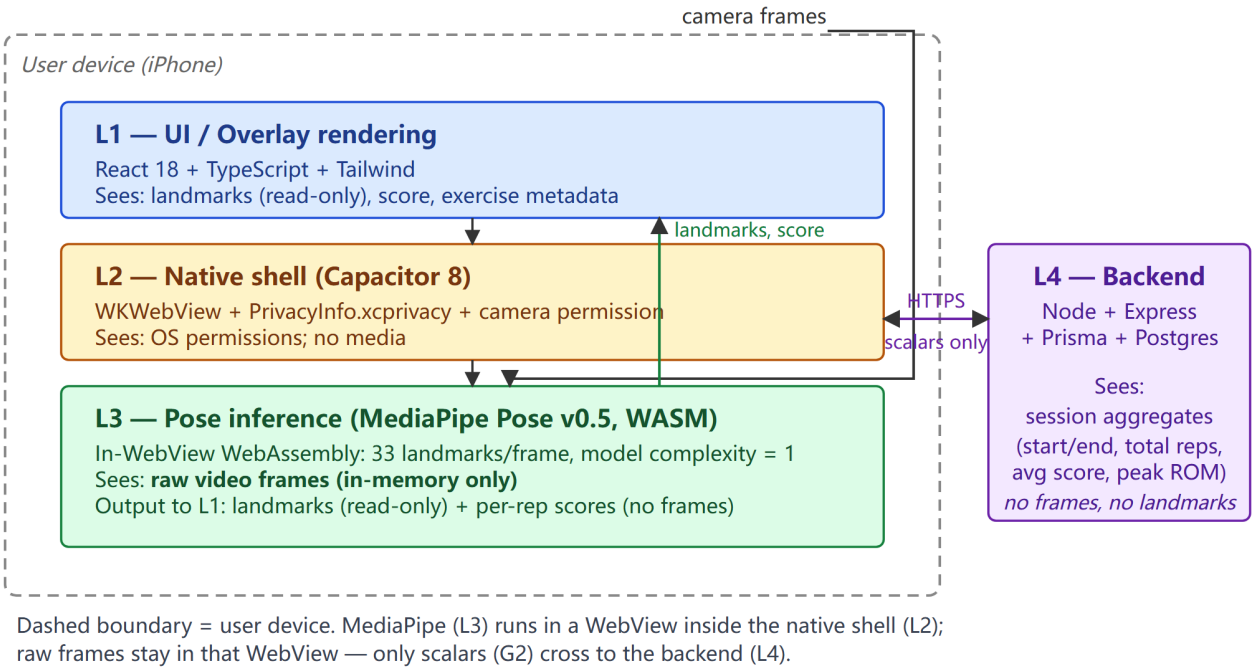


**Figure 1: Four-layer architecture and privacy boundary; the dashed line marks the user's device (§5).**

**Table 1: Component inventory.**

| Layer | Component | Technology | Data scope at runtime |
|---|---|---|---|
| L1 | UI / overlay rendering | React 18 + TypeScript + Tailwind | Landmarks (read-only), score, exercise metadata |
| L1 | TTS feedback | Web Speech API (`speechSynthesis`) | Score string only |
| L2 | Native shell | Capacitor 8 (iOS) | OS permissions; no media |
| L2 | Privacy manifest | `PrivacyInfo.xcprivacy` (NSPrivacyTracking=false) | Static metadata only |
| L3 | Pose inference | MediaPipe Pose v0.5 (WASM) | Raw frames (in-memory only) |
| L3 | Angle smoothing | 20-frame moving average (TypeScript) | Smoothed angle stream |
| L3 | Rep state machine + scoring | Custom (TypeScript) | Per-rep summaries |
| L4 | API | Node.js + Express + Prisma | Session aggregates only |
| L4 | Database | PostgreSQL | Session aggregates only |

apps that uploaded video to remote servers. For this reason, L3 runs WebAssembly inference inside the WebView instead of processing frames on a server; §5 spells out the implications.

## 3.4 User Interface Workflow

A session runs in five steps, shown for shoulder forward flexion in Figure 3. First the patient picks an exercise (Figure 3A). The screen shows the current rehabilitation week, and that week selects the $(\theta_t, \tau, \mathrm{H})$ row from the standards table (§4.6); in week 1 the row is a 60° target with $\tau = 15°$ and a 3 s hold. The camera then opens (Figure 3B). MediaPipe runs in the WebView, and the overlay draws the skeleton, marks the three landmarks for the measured angle (hip–shoulder–elbow for shoulder forward flexion), and shows the live angle, the target, and the rep count. The patient raises the arm into the target zone and holds while the state machine of §4.2 tracks it. When the rep ends, the app shows and speaks the score (Figure 3C), for example "78 points, next", after the 2 s debounce of §6. At the end a summary screen lists the totals (Figure 3D): reps, sets, average score, peak ROM, ROM achievement, voice-correction and compensation counts, and duration. These are exactly the G2 scalars sent to the backend (§5); no frame or per-frame value is shown or stored. The

same five steps repeat throughout the rehabilitation course. Only the targets change from week to week (§4.6); the steps do not.

## 4 The PostureScore Algorithm

This section describes the algorithm that turns a stream of MediaPipe pose landmarks into a single 0–100 score per repetition. The pipeline has four stages: joint angle extraction (§4.1), repetition state-machine detection (§4.2), hold-time and steadiness accumulation (§4.3), and the composite scoring function (§4.4). The function shape is justified in §4.5, and §4.6 gives the table that changes the targets across rehabilitation weeks. This work does not contribute a new pose model; it uses MediaPipe Pose v0.5 off-the-shelf. The new piece is the scoring layer on top of it.

### 4.1 Joint angle extraction

For each frame, MediaPipe Pose returns 33 landmarks with image-normalized 2D coordinates and a per-landmark visibility score in [0, 1]. For each tracked exercise the system picks three landmarks (A, B, C); for shoulder forward flexion on the right side these are the right hip, right shoulder, and right elbow (as shown in Figure 3B: R.Hip→R.Shldr→R.Elbow), with the angle measured at the shoulder vertex. It then computes the planar joint angle at vertex B using the law of cosines:

$$\theta = \frac{180}{\pi} \arccos\left(\frac{|AB|^2 + |BC|^2 - |AC|^2}{2\,|AB|\,|BC|}\right) \qquad (1)$$

The argument is clamped to $[-1, 1]$ before the inverse cosine to absorb floating-point drift near the cone limits. Frames in which any of the three landmarks has visibility below 0.5 are dropped. To suppress single-frame jitter without introducing perceptible latency, raw angles are smoothed with a 20-frame moving average; the resulting smoothing-window duration depends on the device's achieved frame rate and is quantified in §6. Some exercises measure a complementary angle (knee flexion is naturally 180° minus the inner angle); each exercise is tagged `increasing` or `decreasing`, and decreasing angles are flipped before §4.2, so the state machine is direction-agnostic.

### 4.2 Repetition state machine

Each repetition is detected by the four-state machine in Figure 2. A user begins in *Idle*. When the smoothed angle $\theta$ enters the target zone $[\theta_t, \theta_t + \tau]$ ($\theta_t$ is the prescribed target angle, $\tau$ the tolerance), the machine moves to *Holding* and a per-rep timer starts. $\tau$ is single-sided: the entry zone is $[\theta_t, \theta_t + \tau]$ and the rep-end threshold is $\theta_t - \tau$, so despite the $\pm$ notation the tolerance is not symmetric. A single $\tau$ sets both the entry-zone width and the rep-end margin, keeping each (week, movement) standard to one value. Decoupling it into a separate entry width and rep-end margin is a straightforward extension; because the rep-end margin sets how far the angle must fall to end a rep, it would shape the descent counted in `totalTime` and hence the steadiness ratio, giving the score-geometry analysis (§7) an extra parameter to vary. The upper bound $\theta_t + \tau$ makes overshooting count as leaving the zone, so it does not score higher. This caps credit and discourages forcing a joint past its prescribed range. For maximal-range movements the upper bound would be worth revisiting. A watchdog resets the machine to *Idle* and discards the in-progress rep if it has not completed within a timeout $T_{\text{idle}}$ (30 s; see below). Without it, a rep stalled in the band $[\theta_t - \tau, \theta_t)$ could sit in *Drift* indefinitely, and because later reps are reachable only through *Idle*, every one of them would go uncounted. While *Holding*, two quantities are tracked: a continuous-hold timer (time in the zone) and a maximum-hold register (the longest hold so far). Leaving the zone moves the machine to *Drift* and commits the timer to the register if it is larger. The rep ends and the machine moves to *Complete* when $\theta$ falls below $\theta_t - \tau$. At that point the score (§4.3–4.4) is computed and spoken, and the machine returns to *Idle*. Re-entering the zone from *Drift* restarts the timer without starting a new rep, so a brief overshoot-and-recover is not penalized twice (the longer prior run is preserved; Algorithm 1).

The watchdog timeout is $T_{\text{idle}} = 30$ s. In the deployment the slowest completed reps ran about 20–25 s. Setting the cutoff at 30 s gives a genuinely slow hold room to finish, and still clears a stalled rep within a few seconds. (The ˜30 s trajectory in Figure 2B is a drawn illustration, not a deployment maximum.) The timeout is a single fixed value applied to every exercise. A fast wrist movement and a slow shoulder hold have different natural durations, so a per-exercise timeout would be a reasonable extension.

### 4.3 Hold-time accumulation and steadiness ratio

When *Complete* is reached, the algorithm computes two summary quantities. Let `finalHold` be the larger of the most recent continuous-hold value and the maximum-hold register, and let `totalTime` be the wall-clock elapsed from rep start to rep end. The *steadiness ratio* (the term "steadiness" is used because (3) rewards sustained holding rather than total time in zone) is

$$e = \min\left(\frac{\text{finalHold}}{\text{totalTime}},\ 1\right) \in [0, 1] \qquad (2)$$

with `totalTime` lower-bounded at 0.01 s to prevent division by zero. In words, $e$ is the user's longest unbroken hold relative to total rep time: a user who reaches the target and holds it cleanly has $e \approx 1$, while one who oscillates around the boundary keeps restarting the hold timer (§4.2), so even their longest span stays small relative to the rep, giving $e \ll 1$. The two factors in (3) capture different things: whether the longest hold reached the prescribed duration $H$, and whether it was a high fraction of total rep time. Two timing details make the accumulation reproducible: the hold timer advances by the real inter-frame interval $\Delta$t (so a gap is never back-counted), and an occluded frame (any of the

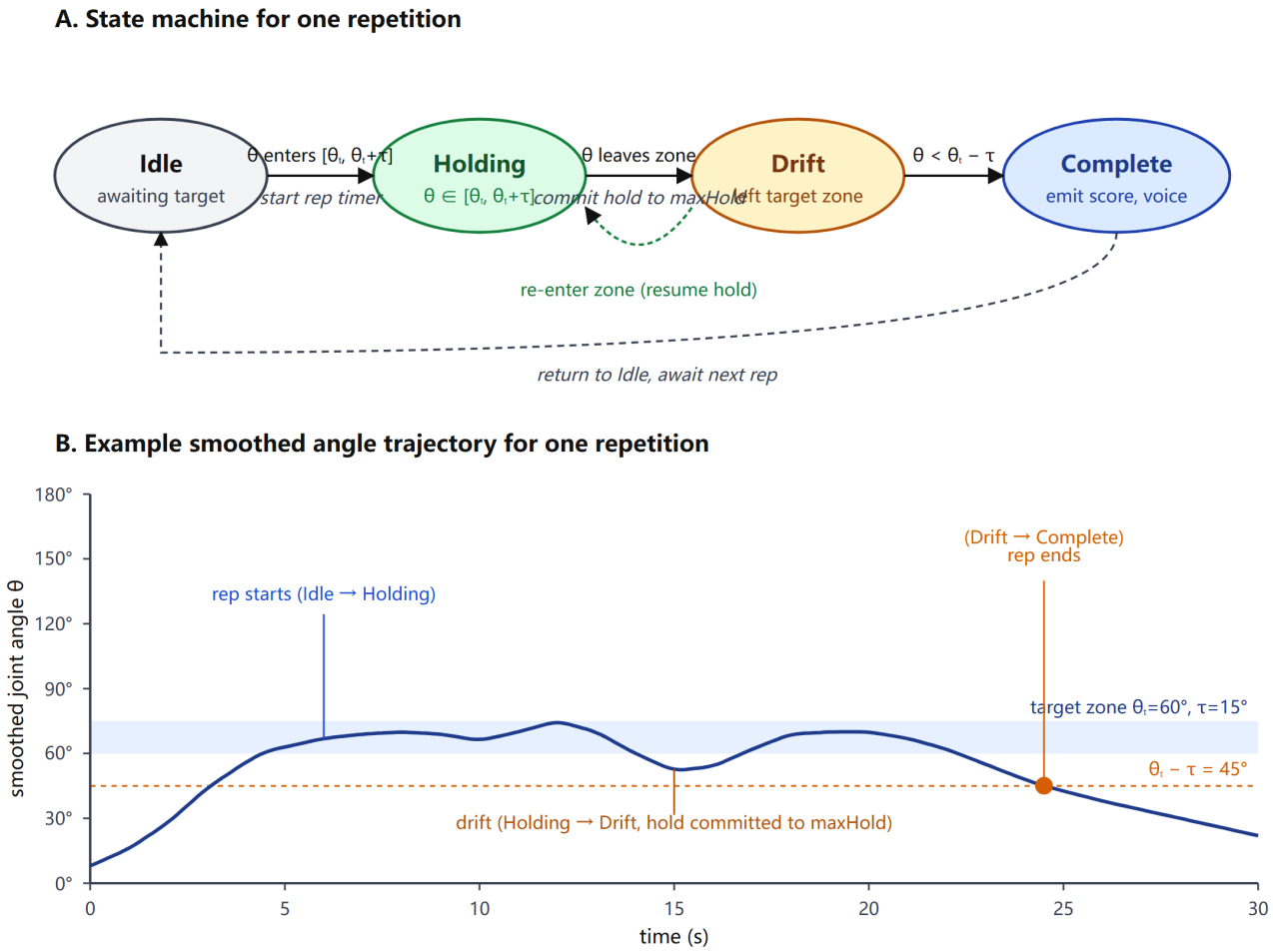


**Figure 2: (A) The four-state per-repetition machine (§4.2): Idle → Holding → Drift → Complete. (B) Example smoothed-angle trajectory ($\theta_t = 60°$, $\tau = 15°$): the blue band is the target zone $[\theta_t, \theta_t + \tau]$, the dashed line the rep-end threshold $\theta_t - \tau$. The 30 s span is a constructed illustration (deployed reps ran ≈20–25 s, below the $T_{\text{idle}} = 30$ s watchdog; §4.2).**

three landmarks below visibility 0.5) is skipped before the timer updates; `totalTime`, by contrast, is wall-clock ($t - t_{\text{start}}$) and includes occluded spans, so heavy occlusion lowers $e$. Time the camera could not verify does not count as a steady hold.

## 4.4 The composite scoring function

The final per-rep score is then

$$\text{score} = 100 \cdot \min\left(\frac{\text{finalHold}}{H},\ 1\right) \cdot \sqrt{e} \qquad (3)$$

where $H$ is the prescribed phase-hold in seconds (§4.6). The `min(·, 1)` cap means holding longer than required earns no bonus. Without the cap, users could lengthen holds to inflate the score. Scores are rounded to integers and averaged per session for reporting; Algorithm 1 summarizes the per-rep loop. The $H = 0$ branch in Algorithm 1 is retained for reach-target movements that do not require a sustained hold. The current deployment standards in Table 3 use nonzero hold durations.

## 4.5 Choice of penalty function

PostureScore uses $\sqrt{e}$ as the steadiness penalty. The alternatives are plain $e$ and the harsher $e^2$. Table 2 shows the difference: a steadiness of 0.25 scores 50 under $\sqrt{e}$ but 25 under linear $e$. All three penalties are monotonic in $e$, so the choice preserves the ordering of repetitions when the hold-time achievement factor is equal. It does not generally preserve rankings when repetitions differ in both hold-time achievement and steadiness. The penalty

```
Algorithm 1 - PostureScore per-rep scoring loop.
Input:  landmark stream {L_t} with frame times t; target theta_t; tolerance tau; phase-hold H; rep timeout T_idle = 30 s
Output: per-rep score s in {0, ..., 100}
state <- Idle; timer <- 0; maxHold <- 0; t_start <- null; t_prev <- null
for each frame at time t:
    dt <- (t - t_prev) if t_prev != null else 0;  t_prev <- t       # real inter-frame interval, advanced every frame
    if state != Idle and (t - t_start) > T_idle:                   # watchdog: abandon a stalled rep
        state <- Idle; continue                                    # partial rep discarded, no score
    if any of (A, B, C) in L_t has visibility < 0.5: continue      # occluded frame: timer not advanced
    theta_smooth <- rolling mean of last 20 angles((A,B,C))      # eqs. (1)
    in_zone <- theta_smooth in [theta_t, theta_t + tau]
    if state = Idle and in_zone:
        state <- Holding; t_start <- t; timer <- 0; maxHold <- 0
    elif state = Holding:
        if in_zone: timer <- timer + dt
        else:       state <- Drift; maxHold <- max(maxHold, timer); timer <- 0
    elif state = Drift:
        if   in_zone:                state <- Holding             # restart span
        elif theta_smooth < theta_t - tau:     state <- Complete
    if state = Complete:
        finalHold <- max(timer, maxHold)
        totalTime <- max(t - t_start, 0.01)                        # wall clock; includes any occlusion
        e <- min(finalHold / totalTime, 1)                         # eq. (2)
        achievement <- 1 if H = 0 else min(finalHold / H, 1)
        s <- round(100 * achievement * sqrt(e))                          # eq. (3)
        emit s; state <- Idle
```

**Table 2: Score multiplier under three steadiness penalties at representative $e$ (hold-time factor = 1, so each value is the full 0–100 score).**

| Steadiness e | Linear e | $\sqrt{e}$ (used) | $e^2$ |
|---|---|---|---|
| 0.25 | 25 | 50 | 6 |
| 0.50 | 50 | 71 | 25 |
| 0.81 | 81 | 90 | 66 |
| 1.00 | 100 | 100 | 100 |

function therefore determines how harshly low-steadiness repetitions are scored. The square root gives partial credit for brief instability and was selected as a design choice rather than learned from data. An empirical comparison across the three penalties would need the per-rep `finalHold` and `totalTime`, which the deployment did not store (§5).

## 4.6 Phase-aware standards

The triple ($\theta_t$, $\tau$, H) that parameterizes (3) is not constant across a rehabilitation course. It is read from the `AiCoach PhaseStandard` table at session start, indexed by the user's current rehabilitation week. Table 3 shows three representative entries: shoulder forward flexion is targeted at 60° with $\tau = 15°$ and a 3-second hold in week 1, reaches 80° / $\tau = 15°$ / 3 s by week 3, and 170° / $\tau = 10°$ / 3 s by week 12 (approximately full physiological range). Both the target angle $\theta_t$ and the tolerance $\tau$ change over the rehabilitation course (Table 3). The rising $\theta_t$ makes the zone harder to reach, while the narrowing $\tau$ (from 15° to 10° by week 12) works the other way. Tightening the parameters therefore has a *non-monotonic* effect on (3) instead of uniformly making it harder to satisfy. The §7 score trajectory must be read with this coupling in mind. PostureScore is "phase-aware" because of this split between a *fixed scoring function* and *time-varying parameters*, neither of which touches the algorithm. Correction. The values in Table 3 represent the actual configuration used during the February–April 2026 pilot study, and the §7 results were generated under that configuration. An earlier manuscript version contained preliminary documentation values that were inadvertently presented as deployment parameters; these values have been corrected here.

**Table 3: Representative phase-aware standards (`AiCoachPhaseStandard`). *Tol* = tolerance $\tau$ (both the zone width above $\theta_t$ and the rep-end margin below it, §4.2); *Hold* = phase-hold *H*.**

| Week | Movement | Target angle $\theta_t$ | Tol $\tau$ | Hold H |
|---|---|---|---|---|
| 1 | Shoulder forward flexion | 60° | 15° | 3 s |
| 3 | Shoulder forward flexion | 80° | 15° | 3 s |
| 12 | Shoulder forward flexion | 170° | 10° | 3 s |
| 1 | Knee flexion | 80° | 15° | 1 s |
| 6 | Knee flexion | 110° | 12° | 1 s |
| 12 | Knee flexion | 135° | 10° | 1 s |

# 5 Privacy Architecture

For any home rehab app that uses computer vision, one question dominates the privacy discussion: *does video leave the device?* For PostureScore the answer is no. This was a deliberate design decision. The privacy design rests on three guarantees and an explicit trust model, organized around the WebView boundary shown in Figure 1.

**(G1) Raw video frames remain in WebView memory.** The camera frame buffer is consumed by the WASM pose-inference module on the same animation tick that produces it, and is overwritten by the next frame. Under no code path in PostureScore is a frame serialized to local storage, IndexedDB, the filesystem, or the network. (G1) is enforced by the app's source code, not by the OS or WebView runtime; see the trust model below.

**(G2) Only session summary numbers cross the device-to-server boundary.** What leaves the device over authenticated HTTPS is one session record with these scalar

fields: session start/end time, total duration, total reps, total sets, average score, peak ROM angle, target ROM angle, ROM achievement ratio, voice correction count, compensation event count, and a voice-enabled flag. Per-frame landmarks, per-frame angles, and per-rep scores are not in the payload. The Prisma model `AiCoachExerciseSession` has no field for landmarks or per-frame angle data, so the schema itself enforces this.

**(G3) The system is not used for advertising tracking.** The iOS privacy manifest declares `NSPrivacyTracking = false`, lists no tracking domains, and declares only the data categories the app actually uses (account email, fitness/health data, basic usage analytics, user notes); the sensitive APIs declared are all there for app functionality, not cross-app identification.

**Trust model.** PostureScore trusts the device OS and WebView runtime: a malicious ROM, a jailbroken device, or a tampered browser engine could exfiltrate frames before the WASM module consumes them, and the architecture does not detect this; nor does it stop a malicious user uploading frames out-of-band. The only network defense is TLS in transit. The system applies no differential privacy, federated learning, or encryption beyond HTTPS. The privacy design works by limiting *what* the app sends rather than by obscuring it. The §7 aggregates are used only for summary statistics, never for training. DP and federated learning are left as future work, to be revisited if cross-user personalization is ever pursued.

**Logging for offline analysis.** The deployment stored only session aggregates (G2), so the per-rep numbers needed to re-score sessions (§4.5, §7) were never kept. Storing only aggregates was a storage decision, and the privacy design does not require it. A future build could add a few de-identified per-repetition values such as `finalHold` and `totalTime`. Those are plain numbers, so logging them would make the analyses possible with no video leaving the device. A minimal clinical study would build on this data infrastructure: it would recruit a prospective cohort, have an independent physical therapist score the recorded movements against (3), and obtain full independent ethics-board approval before making any efficacy claim.

## 6 Implementation

**Pose model and smoothing.** MediaPipe Pose v0.5 runs as WebAssembly inside the WebView, with `modelComplexity = 1` (the medium model), built-in landmark smoothing on, segmentation off, and detection/tracking thresholds at 0.5. The system uses complexity 1 instead of the faster `complexity = 0` (lite). The lite model is also real-time (Table 4), but its lighter network yields noisier landmarks, which would degrade joint-angle precision relative to the $\tau$ tolerances in Table 3. The raw 3-point angle from (1) is pushed onto a 20-frame circular buffer; at 31–38 fps (Table 4) this gives ˜525–650 ms of smoothing, which damps per-frame jitter without noticeably lagging the voice feedback. Because the window smooths over brief transitions in both directions, it can bias `finalHold` in either direction by at most approximately one smoothing window (˜0.65 s). In particular, it may over-estimate `finalHold` by masking brief out-of-zone dips or under-estimate it by attenuating brief in-zone entries. The effect is small for the slow reps here but would grow for faster ones, which bounds the movement speeds for which (3) is valid. Low-visibility frames (any landmark $< 0.5$) are dropped, with an on-screen warning to reposition the camera.

**Voice feedback and compensation detection.** Per-rep scores are spoken via the Web Speech API in Chinese or English with a 2-second debounce so fast reps do not stack utterances; users can toggle voice off and that flag is recorded per session. Compensation detection is rule-based and reuses the smoothed landmark stream from the scoring pipeline (§4.1). As a representative rule, trunk-lean during shoulder forward flexion is measured from the trunk axis between the mid-shoulder point S = midpoint(landmarks 11 and 12) and the mid-hip point P = midpoint(landmarks 23 and 24). The trunk's angular deviation from image vertical is

$$\beta = |\arctan2(S_x - P_x,\ P_y - S_y)| \times \frac{180}{\pi} \qquad (4)$$

where coordinates are MediaPipe's image-normalized values (y increasing downward, so $P_y - S_y > 0$ for an upright posture and $\beta \approx 0$). $\beta$ is smoothed with the same 20-frame moving average as the joint angles, and is taken relative to a per-rep baseline $\beta_0$ set at rep start (Idle → Holding). The baseline cancels a user's neutral standing posture or a slightly tilted camera. A trunk-lean event is counted when $|\beta - \beta_0|$ exceeds 15° for at least 0.5 s while all four trunk landmarks stay visible (visibility $\geq 0.5$). A 15° / 10° hysteresis band stops one long lean from counting several times: the event registers at 15° and does not re-arm until $|\beta - \beta_0|$ drops back below 10°. Only the per-session count reaches the backend (G2, §5); no per-frame trunk angles are stored. A single camera cannot see front-to-back lean, so the rule covers side-to-side (coronal) lean only. The thresholds were hand-authored and reviewed by the clinical advisor (Acknowledgements); a learned classifier is left for future work (§8).

**On-device performance.** Fps and p95 latency were measured on iPhone 16 (A18) and iPhone 14 (A15) via `performance.now()` in MediaPipe's `onResults` callback, averaging several 40–60 s sessions per Table 4 cell. MediaPipe dominates inference; the scoring layer adds under 1 ms/frame. At the default `complexity = 1`, iPhone 16 runs 38.5 fps and iPhone 14 31.0 fps, well above the real-time threshold. The 16–25% inter-device gap informs the §8 L3 warning about older hardware.

**Adding new exercises.** Fifteen exercises are currently active across four joints (shoulder, knee, hip, ankle); adding one needs only a config row (three landmarks, rep direction, compensation rules) plus per-week rows in `AiCoachPhaseStandard` (Table 3), with no code change.

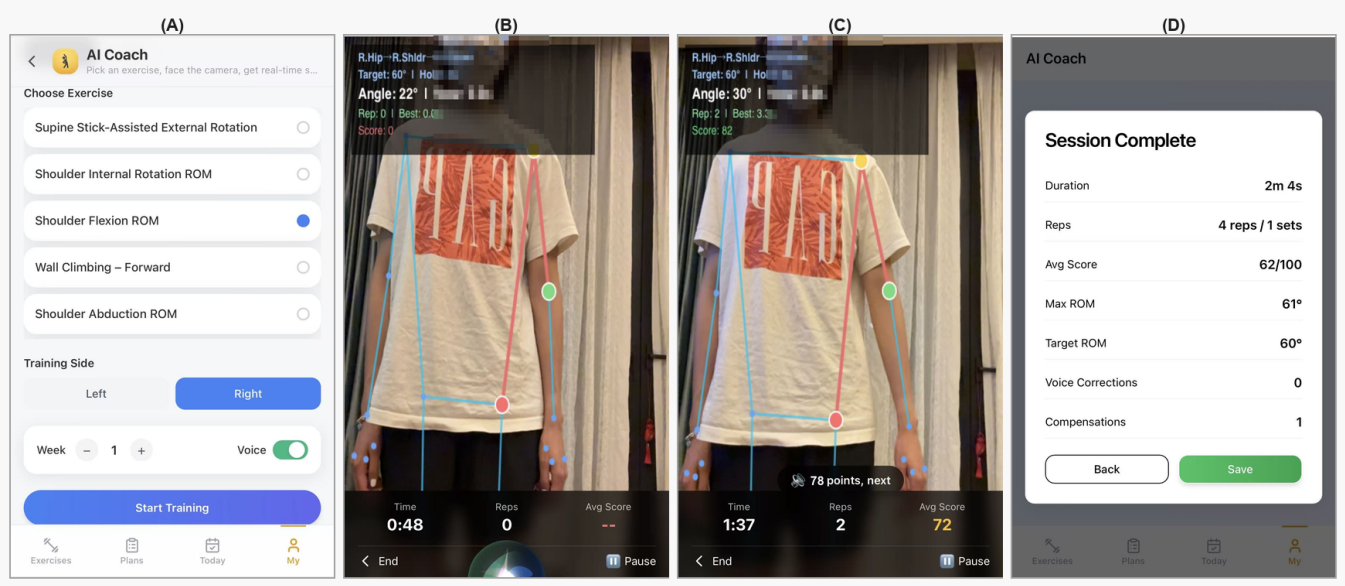


**Figure 3: The five-step session workflow (§3.4) for shoulder forward flexion: (A) exercise selection; (B) live capture with skeleton overlay; (C) per-rep score, displayed and spoken; (D) session summary (G2 scalars only, no frames).** ***Screenshots on iPhone 16; the subject's face in (B) and (C) is anonymized.***

**Table 4: End-to-end performance on two reference devices across all three `modelComplexity` settings. Each cell averages several 40–60 s exercise sessions; *fps* is the mean frame-rate, *p95 lat* the 95th-percentile per-frame inference latency.**

| `modelComplexity` | iPhone 16 — fps | iPhone 16 — p95 lat (ms) | iPhone 14 — fps | iPhone 14 — p95 lat (ms) |
|---|---|---|---|---|
| 0 (lite) | 52.8 | 19.9 | 44.3 | 25.0 |
| 1 (medium, default) | 38.5 | 30.2 | 31.0 | 33.9 |
| 2 (heavy) | 10.6 | 96.8 | 7.9 | 140.3 |

## 7 Field Evaluation

The system was tested with seven post-surgical patients to see whether it could run reliably outside the author's own testing. The deployment was a feasibility check, **not a clinical trial**. The results show only that the app worked end-to-end on real phones for several weeks, produced reasonable (non-random) scores, and respected the privacy boundary of §5. Whether a higher PostureScore actually corresponds to better clinical recovery is a separate question (§8 L5).

**Scale.** Between February and April 2026 PostureScore was distributed as a private build to seven post-surgical patients (ACL-reconstruction and shoulder-arthroscopy, ages 19–30) recruited through Fu's Orthopedic Hospital, Hangzhou. Each followed an 8- to 12-week home plan; the app recorded 664 scoring sessions.

**System stability.** All 664 sessions ran through §4's pipeline without crashing: every session produced a non-null average score, peak ROM angle, and total-rep count, and there were no recorded MediaPipe inference failures or runtime crashes. Because a session is recorded only once its summary reaches the backend, the count is a lower bound on stability: a client-side crash before upload would leave no record. Voice feedback was on by default and toggleable per session (the flag is recorded). Three of the seven users had shoulder arthroscopy and used a shoulder library shaped by the author's prior rehabilitation experience (§3.3).

**Score trajectory across rehabilitation phases.** As a basic check that (3) was producing reasonable values in real use, each user's session-average scores (the `avgScore` field in G2) were grouped by plan-week (early 1–4, mid 5–8, late 9 and beyond; plan-week is the user's own progression through their 8- to 12-week plan, not calendar week) and averaged within user, then across users (equal weight per user). The cohort means were 57 / 65 / 68 (sample SD 3.9 / 5.0 / 5.1; Figure 4). Two of seven users contributed no late-phase data (one's plan ended before the late phase; the other practiced mostly in the first half), so the late mean is n = 5 versus n = 7. Among the five users present in all three phases, each scored higher late than early, so dropout does not explain the rise. The rise should not be read as clinical improvement. Because the scoring standards ($\theta_t$, $\tau$) also tighten each week,

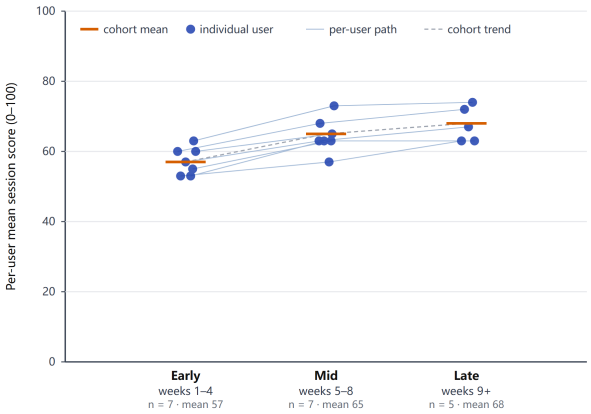


**Figure 4: Mean session scores across rehabilitation phases (dots, per-user means; bars, cohort means; thin lines, within-subject paths; n = 7 early/mid, n = 5 late); a feasibility check of (3), not evidence of clinical efficacy (§7).**

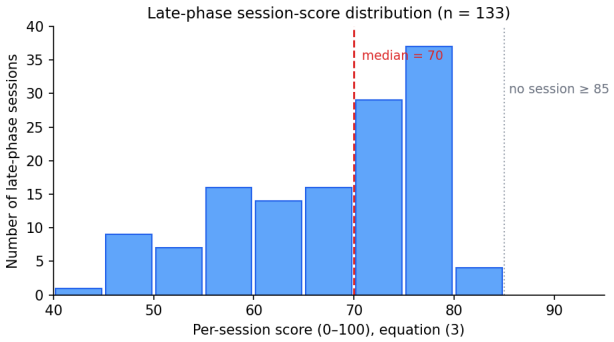


**Figure 5: Per-session score distribution across the 133 late-phase sessions (5-point bins; §7).**

the score geometry shifts independently of movement quality (§4.6): narrowing $\tau$ shortens totalTime (tending to raise the steadiness ratio e) while also lowering finalHold, so the net parameter effect on (3) is non-monotonic instead of a clean directional bias, and real change cannot be separated from parameter effects even in sign. Separating them would require re-scoring the same repetitions under a fixed ($\theta_t$, $\tau$), which needs the per-rep `finalHold` and `totalTime` the deployment did not retain (§5). The de-identified session-level dataset is publicly archived [14].

**Score distribution and ceiling effects.** The $\sqrt{e}$ factor is concave (§4.5), so it compresses scores near the top and could blur already-good repetitions if real scores piled up there. Figure 5 shows they did not. Across the 133 late-phase sessions the scores span 42–81, with a median of 70 and most in the 55–80 band; the highest was 81 (per-session; per-user means are lower, Fig. 4), well short of 85 and the 100 ceiling. The concavity therefore had no real effect here, since the scores never reached the flat part of $\sqrt{e}$. At the same time, 16 of the 133 sessions fell below 55, so the low end still separates. Because nothing reached a ceiling, saturation cannot explain the late-phase rise in the cohort means.

**Ethics.** All seven participants gave written informed consent at sign-up; the consent explicitly authorized de-identified, aggregate results to be used both for product evaluation and for academic publication. The data reported here are session-level summary numbers, not video and not landmarks. Before deployment, the Research Administration Office of Fu's Orthopedic Hospital (Hangzhou), the institution that referred the participants, reviewed this work and determined in writing that, because it analyzes only de-identified secondary records arising from ordinary product use rather than an interventional protocol, it did not require formal institutional ethics-committee review. Consistent with that determination, PostureScore was offered as a rehabilitation aid that users chose to use rather than as an interventional study; the in-app rehabilitation protocol (exercise selection, reference angles, tolerance thresholds, and per-week progression; Table 3) was reviewed for safety before deployment by the clinical advisor named in the Acknowledgements, who also referred the seven participants. Participation was entirely voluntary and did not affect any participant's clinical care or treatment. The scope of this determination (a not-required finding issued by the hospital's research administration office rather than full prospective ethics-board approval) is noted as a limitation (§8 L6).

## 8 Discussion and Limitations

Seven limitations are stated upfront, grouped as clinical, technical, and design, so readers can judge the contributions in §§4–5 against what they do not establish. The original labels (L1–L7) are retained to preserve cross-referencing with the previous review response.

**Clinical limitations.**

**L4 — Deployment scale.** Seven users show the pipeline runs end-to-end in the wild but cannot establish general robustness; all came from one hospital and were aged 19–30.

**L5 — No external validation against a clinician.** These results do not prove a higher PostureScore means better clinical recovery.

**L6 — Ethics clearance was a not-required determination, not full board approval.** The clearance was a written "no formal review required" determination issued before deployment by the research administration office of Fu's Orthopedic Hospital (the same institution that referred the participants), on the basis that the study uses only de-identified secondary data (§7); it was not a full prospective ethics-board protocol, and the determining office is not independent of the deployment, a conflict of interest for any future clinical claim. Any future study that makes clinical claims, or that collects data prospectively as research rather

than as ordinary product use, should obtain full, independent institutional ethics approval first.

**Technical limitations.**

**L2 — The system trusts the camera angle.** A user who positions the phone so a movement *looks* like it reaches the target can game the score, and (3) does not detect this; the rule-based compensation detection in §6 catches only a few common patterns, not a clever camera placement.

**L3 — WebView inference performance ceiling.** Much older hardware (e.g. iPhone 11 / A13) would run markedly slower than the tested devices, stretching the 20-frame smoothing window and lagging the voice feedback; such hardware was not available to test.

**L7 — The steadiness ratio can penalize good form.** Because `totalTime` includes the descent out of the zone (§4.3), a slow, controlled eccentric lowering (which is clinically desirable) lowers $e$ and hence the score.

**Design limitation.**

**L1 — Targets reviewed by a rehab doctor, not learned from data.** The ($\theta_t$, $\tau$, H) triples in Table 3 were hand-authored and reviewed by one rehabilitation specialist (Acknowledgements); the tolerances chosen for young-adult ACL and shoulder patients would likely need redoing for older patients or children.

Each limitation points to a next step: a larger multi-site cohort (L4), independent therapist scoring with an agreement statistic such as ICC (L5), and full independent ethics approval before any clinical-claim study (L6); on the technical side, a second camera view or inertial cross-check against camera-angle gaming (L2), benchmarking on older A11–A13-class hardware with a WebGL/Metal inference path (L3), and scoring the eccentric phase on its own tempo (L7); and refitting the hand-authored targets from multi-age data with a mixed-effects model (L1). The clinician-scoring (L5) and eccentric (L7) studies both depend on the per-rep logging in §5.

## 9 Conclusion

PostureScore is an on-device, phase-aware rehab-scoring app for home use after surgery. Its core is the scoring formula (3), combining hold time and movement steadiness into a 0–100 score, together with the week-indexed standards (Table 3) that shift targets across a rehabilitation course. The privacy design (§5) keeps raw video in the phone's WebView and sends only session summaries, enforced at the schema level. The early-week tolerance, per-rep voice feedback, and on-device-only inference come from the author's prior rehabilitation experience (§3.3). Across 664 sessions on seven patients the pipeline ran end-to-end. This is an engineering feasibility check rather than a test of clinical efficacy (§7, §8 L5). The algorithm is specified in full (Algorithm 1, equations (1)–(3), and the Table 3 schema), so it can be reimplemented without the original source; implementation code is available from the author on reasonable request.

## Acknowledgements

This project came out of my own experience recovering from ACL surgery (June 2025) and shoulder arthroscopy (December 2025), and several of its design choices come directly from problems I ran into during those rehabs. I thank Z. Wang for technical discussions on system architecture, and Dr. Youzhang Huang (Fu's Orthopedic Hospital, Hangzhou) for reviewing the rehab protocol and referring the seven users in §7.

## References

[1] J. J. Abraham, S. Kumar, and K. K. Vedavyasa Acharya, "Determinants of patient compliance with post-operative rehabilitation after lower limb orthopaedic surgery: a prospective study," *BMC Musculoskelet. Disord.*, vol. 26, no. 1, art. 1065, 2025. doi: https://doi.org/10.1186/s12891-025-09320-5

[2] N. J. Maher, C. Brogden, A. C. Redmond *et al.*, "Disparity in anterior cruciate ligament injury management: a case series review across six National Health Service trusts," *BMC Musculoskelet. Disord.*, vol. 26, no. 1, art. 363, 2025. doi: https://doi.org/10.1186/s12891-025-08572-5

[3] V. Bazarevsky, I. Grishchenko, K. Raveendran, T. Zhu, F. Zhang, and M. Grundmann, "BlazePose: On-device real-time body pose tracking," *arXiv*:2006.10204, 2020. URL: https://arxiv.org/abs/2006.10204

[4] W. Simoes, L. Reis, C. Araujo, and J. Maia Jr., "Accuracy assessment of 2D pose estimation with MediaPipe for physiotherapy exercises," *Procedia Comput. Sci.*, vol. 251, pp. 446–453, 2024. doi: https://doi.org/10.1016/j.procs.2024.11.132

[5] S. Akturk, F. B. Derdiyok, and K. Serbest, "Markerless joint angle estimation using MediaPipe with a rapid setup for joint moment calculation," *Multimed. Tools Appl.*, vol. 85, no. 1, art. 38, 2026. doi: https://doi.org/10.1007/s11042-026-21256-z

[6] S. Dill, A. Ahmadi, M. Grimmer, D. Haufe, M. Rohr, Y. Zhao, M. Sharbafi, and C. Hoog Antink, "Accuracy evaluation of 3D pose reconstruction algorithms through stereo camera information fusion for physical exercises with MediaPipe Pose," *Sensors*, vol. 24, no. 23, art. 7772, 2024. doi: https://doi.org/10.3390/s24237772

[7] R. I. Hamilton, Z. Glavcheva-Laleva, M. I. H. Milon, Y. Anil, J. Williams, P. Bishop, and C. Holt, "Comparison of computational pose estimation models for joint angles with 3D motion capture," *J. Bodywork Mov. Ther.*, vol. 40, pp. 315–319, 2024. doi: https://doi.org/10.1016/j.jbmt.2024.04.033

[8] M. MohammadNamdar, M. L. Wilson, K.-P. Murtonen, E. Aartolahti, M. Oduor, and K. Korniloff, "How AI-based digital rehabilitation improves end-user adherence: rapid review," *JMIR Rehabil. Assist. Technol.*, vol. 12, no. 1, art. e69763, 2025. doi: https://doi.org/10.2196/69763

[9] C. D'Amore, J. C. Reid, M. Chan, S. Fan, A. Huang, J. Louie, A. Tran, S. Chauvin, and M. K. Beauchamp, "Interventions including smart technology compared with face-to-face physical activity interventions in older adults: systematic review and meta-analysis," *J. Med. Internet Res.*, vol. 24, no. 10, art. e36134, 2022. doi: https://doi.org/10.2196/36134

[10] Sword Health, "The cost of digital physical therapy vs. traditional PT: an employer's guide," industry report, 3 Oct. 2025. URL: https://swordhealth.com/articles/digital-physical-therapy-costs-vs-traditional-pt-costs *(cited as operational evidence; not peer-reviewed.)*

[11] Apple Inc., "Privacy manifest files," *Apple Developer Documentation*, 2024. URL: https://developer.apple.com/documentation/bundleresources/privacy_manifest_files

[12] A. Walker, W. Hing, and A. Lorimer, "The influence, barriers to and facilitators of anterior cruciate ligament rehabilitation adherence and participation: a scoping review," *Sports Med. Open*, vol. 6, art. 32, 2020. doi: https://doi.org/10.1186/s40798-020-00258-7

[13] Ionic / Drifty Co., "Capacitor 8 documentation: cross-platform native runtime for web apps," 2024. URL: https://capacitorjs.com/docs

[14] E. Wang, "PostureScore deployment dataset (n=7, 664 sessions)," Open Science Framework, 2026. doi: https://doi.org/10.17605/OSF.IO/3N4Y8

# A Reproducible Computational Pipeline for Modelling Sequential Decision Thresholds from Stopping-Rule Data

Alex Kolesnikov
Eton College
Windsor, Berkshire, United Kingdom
alexkolesnikov2010@gmail.com

## Abstract

Stopping-rule experiments ask participants to move through ordered stages until they decide to act. These designs are useful for studying intervention thresholds, but raw survey exports are usually stored in wide form, while correct modelling requires scenario-level thresholds and stage-level at-risk rows. This paper presents a runnable R artifact for reconstructing and validating stopping-rule data. The artifact uses a JSON configuration layer, modular R source files, toy data, a second structurally different toy configuration, automated testthat tests, adversarial validation checks, reference outputs, licensing and citation metadata. In a de-identified sequential risk-decision case study, the same reconstruction logic maps 1,108 potential scenario observations from 277 consenting participants, flags 25 ambiguous response sequences, retains 1,083 clean scenario-level observations, and generates 3,047 stage-level at-risk rows. The evaluation documents local execution, automated test coverage, end-to-end preprocessing behaviour, a second configuration demonstration, adversarial failure behaviour and expected linear scaling for larger survey exports. The contribution is a reproducible data-engineering system that bridges raw stopping-rule survey exports and standard event-history or repeated-measures modelling tools, while making exclusion decisions auditable rather than hidden in manual recoding.

## Keywords

Stopping-Rule Data, Sequential Decision-Making, Computational Pipeline, R Software, Reproducibility, Data Validation, Data Engineering, Discrete-Time Hazard Models, Right-Censoring, Decision-Support Systems

**ACM Reference Format:**
Alex Kolesnikov. 2026. A Reproducible Computational Pipeline for Modelling Sequential Decision Thresholds from Stopping-Rule Data. In *Proceedings of International Journal of Secondary Computing and Applications Research (IJSCAR VOL. 3, ISSUE 3)*. ACM, New York, NY, USA, 7 pages. https://doi.org/10.67149/yhjs2024.5/3s7gdh1q


*IJSCAR VOL. 3, ISSUE 3*


## 1 Introduction

Many applied decision problems are sequential rather than static. A person, monitoring system or AI assistant may wait while evidence remains ambiguous and act only when accumulating information crosses a threshold. This structure appears in symptom checkers, safety monitors, cybersecurity alerts, financial-risk systems and user-facing decision-support tools. In each setting, the important question is not only what information is present, but when the information becomes action-relevant.

Stopping-rule surveys provide a compact experimental design for studying this behaviour. Participants see a sequence of stages and choose either to keep waiting or to act. Once action is selected, the scenario stops. The first action point defines the decision threshold. If no action occurs by the final stage, the true threshold is right-censored: it is known only to exceed the final displayed level.

The computational difficulty is that common survey platforms export this data in wide form: one row per participant and separate columns for possible scenario-stage responses. Standard modelling tools expect long, validated event-history data. Between those two forms lies a non-trivial reconstruction problem. Researchers must map survey columns to scenario structures, enforce the stopping rule, identify ambiguous response sequences, reconstruct scenario-level thresholds and expand clean cases into stage-level at-risk rows.

This preprocessing step matters because errors are easy to hide. A manual wide-to-long conversion can accidentally count post-intervention stages as if the participant were still at risk, treat right-censored observations as observed final-stage interventions, or silently retain inconsistent response sequences. Such errors can change the estimated timing of action even when the later statistical model is correctly specified. The present artifact therefore treats preprocessing as part of the scientific method, not as clerical preparation before analysis.

This paper introduces a runnable R artifact that performs this transformation. The contribution is not a new statistical estimator. It is a reproducible preprocessing and modelling pipeline that produces auditable datasets ready for existing event-history, GEE or mixed-model tools.

## 2 Related Work and Tool Gap

Discrete-time event-history models represent event occurrence across ordered opportunities and are well established

in statistics and social-science methodology [1, 2]. Generalised estimating equations provide population-average inference for clustered binary data [3, 4]. Software ecosystems such as R survival, Python lifelines, geepack and statsmodels support related model fitting once data have already been transformed into the required long form [4–7].

However, these tools do not solve the prior data-engineering problem caused by stopping-rule survey exports. They do not validate whether a response sequence obeys a first-action stopping rule, decide which post-intervention stages should be excluded, generate exclusion logs or create both scenario-level and stage-level files from a wide survey export. The present artifact fills that preprocessing gap and is designed to complement, not replace, existing modelling packages.

The closer methodological neighbours are therefore not only survival-analysis libraries but also work on reproducible workflows, tidy data and research data organisation. Tidy-data principles emphasise that each variable, observation and observational unit should be represented consistently before analysis [12]. Good-enough practices in scientific computing similarly argue that project organisation, scripted processing, versioned outputs and clear documentation are necessary for reliable computational research [13]. Data-organisation guidance for spreadsheets highlights the risk of analysis errors when raw data structures are convenient for entry but not for computation [14]. The present pipeline applies these general principles to a specific experimental structure: first-action stopping-rule surveys.

General data-validation frameworks such as validate and pointblank support reusable column-level, row-level and cross-column rules, including checks on values, missingness and consistency [15, 16]. They can therefore identify many generic data-quality problems that fall outside the scope of this artifact. However, they do not provide built-in first-action stopping-rule semantics for reconstructing thresholds, distinguishing right-censoring from ambiguous sequences, or generating stage-level at-risk rows. The present artifact is narrower and domain-specific: it performs those stopping-rule transformations and returns threshold, censoring, exclusion-log and at-risk-row outputs.

The design also follows principles from reproducible computational research: code, intermediate data products, logs and tests should make a workflow inspectable rather than relying on undocumented manual recoding [8–11]. Its specific gap is the lack of a small, reusable bridge between survey exports and event-history-ready stopping-rule datasets.

## 3 System

The artifact contains modular R files, JSON configuration files, toy datasets, testthat tests, reference output files, adversarial validation checks and software metadata. It is intentionally small so that its logic can be inspected directly and adapted to new stopping-rule designs by editing configuration rather than rewriting code. The system is organised around three design choices: study-specific structure belongs in configuration; response-sequence validity is checked before modelling; and all transformations produce auditable output files.

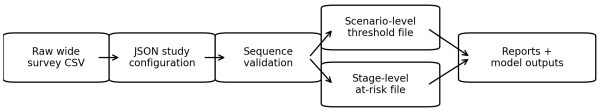


**Figure 1: R pipeline from raw wide survey export to auditable model outputs.**

### 3.1 Configuration-Driven Mapping

The primary file `config/study_config.json` specifies the participant identifier, response labels, covariates, scenario identifiers, condition labels, ordered response columns and stage scores. JSON was used because the mapping is structured, human-readable, language-independent and easy to inspect without running R. Keeping this mapping outside the R functions prevents study-specific column names from being hard-coded into the validation and expansion logic. A second file, `config/study_config_3scenario_7stage.json`, uses three scenarios, seven stages, different column names and different stage scores to demonstrate that the same reconstruction functions can be reused without source-code edits.

### 3.2 Sequence Validation

For each participant-scenario sequence, `validate_sequences.R` classifies the observation as a clean intervention, clean right-censored non-intervention, or ambiguous sequence. A clean intervention contains wait responses before the first action and no later responses after the action. A clean censored sequence contains wait responses through the final observed stage. A sequence is ambiguous if it contains unknown labels, non-wait responses before action, responses after action, missing values inside an otherwise observed sequence, or malformed stage information. This taxonomy separates errors that are analytically meaningful from ordinary right-censoring. The goal is to fail loudly or log exclusions rather than silently manufacture model-ready rows from invalid input.

### 3.3 Scenario and Stage Expansion

Clean sequences are reconstructed into scenario-level rows containing the intervention indicator, action stage, action score and censoring status. `expand_stage_level.R` then creates one row per at-risk decision opportunity. If action occurs at stage 3, only stages 1 to 3 are included; stages 4 and 5 are structurally excluded because the participant is no longer at risk. If no action occurs, all observed stages are retained with event = 0. This distinction is essential because event-history models estimate the conditional probability of acting at a stage given that action has not already occurred.

**Table 1: Scope relative to existing modelling and workflow tools.**

| Tool or approach | Primary function | Remaining gap for stopping-rule surveys |
|---|---|---|
| survival / lifelines | Event-history modelling | Requires long-form event data already reconstructed |
| geepack / statsmodels GEE | Clustered binary modelling | Requires at-risk rows and event indicators |
| General workflow tools | Project organisation and reproducibility | Do not encode first-action stopping-rule logic |
| Spreadsheet or survey exports | Store raw wide responses | Do not directly represent scenario thresholds or right-censoring |
| This R pipeline | Validation and reconstruction | Produces scenario-level and stage-level model-ready files with exclusion logs |

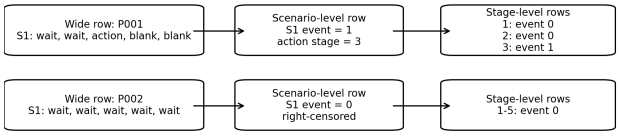


**Figure 2: Conversion from wide stopping-rule responses to scenario-level and stage-level data.**

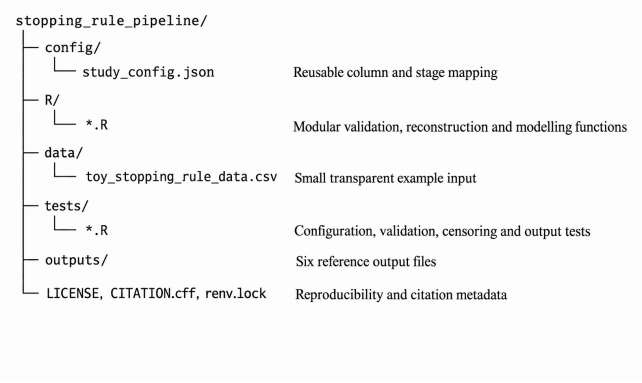


**Figure 3: Directory layout of the R software artifact supplied with the paper.**

## 3.4 Outputs and Packaging

The output module writes six files for each configured run: scenario-level data, stage-level data, an exclusions log, a reconstruction summary, a runtime report and model coefficients. The repository also includes LICENSE, CITATION.cff, .gitignore and renv.lock files so that the artifact behaves like research software rather than loose analysis scripts. Table 2 summarises the artifact structure in accessible form, and Figure 3 shows the corresponding directory layout.

# 4 Case Study

The pipeline is demonstrated using a de-identified sequential risk-decision dataset collected by the author in April and May 2026 during a broader empirical study of intervention thresholds in adolescents and adults. The original survey asked participants to move through hypothetical health-risk scenarios and choose either to continue monitoring or seek help. Participants were recruited through a convenience sample of UK secondary-school pupils aged approximately 15–18, recorded in age categories, and adult contacts, including parents and teachers. The school-age sample included participants from several independent and state secondary schools. The survey was administered online through Google Forms. The substantive empirical analysis of the Part B dataset has been published separately [17]. The case study in the present paper uses the reconstruction problem from that dataset; it does not reuse the published article's hypothesis tests as the contribution of this article.

As an independent, school-supervised project rather than university or clinical research, the study was not submitted to a university or external research ethics committee. Survey distribution and data collection involving school-age participants were reviewed and approved before distribution by the Deputy Head Pastoral and Designated Safeguarding Lead at the participating schools. Participation was voluntary, and only submissions with electronic consent were included in the analysis; two submissions without consent were excluded. The project was considered low risk because it used hypothetical scenarios, involved no intervention or medical procedure, collected no clinical or diagnostic data, and allowed participants to stop at any time. Responses were collected anonymously: no names, contact details, email addresses, IP addresses, school identifiers, personal medical histories or clinical records were included in the research dataset. Demographic information was retained only in broad categories. After collection, the twelve survey-version exports were combined into a de-identified analytical dataset. Original timestamps, submission order, row numbers, consent responses and potentially identifying metadata were removed. Each consenting participant was assigned a random anonymous identifier that did not encode survey version, timestamp or original row order, and the combined dataset was shuffled before sharing.

**Table 2: Structure of the R software artifact supplied with the paper.**

| Artifact component | Purpose |
|---|---|
| R/ | Source functions for configuration loading, validation, reconstruction, stage-level expansion, optional modelling and exporting |
| config/ | Primary and second JSON mappings for participant identifiers, response labels, scenario-stage columns and stage scores |
| data/ | Primary and second toy wide-format stopping-rule inputs used for demonstration and tests |
| tests/ | Automated checks for configuration loading, validation, expansion, output-file generation, second-configuration reuse and assumption violations |
| outputs/ | Reference outputs, second-configuration summary and adversarial stress-test matrix |
| local_records/ | Local verification records, R session information, console logs and output-file lists |
| LICENSE, CITATION.cff, renv.lock | Software licensing, citation metadata and dependency record |

**Table 3: Case-study reconstruction counts.**

| Quantity | Count |
|---|---|
| Consenting participants | 277 |
| Potential scenario observations | 1,108 |
| Ambiguous sequences flagged | 25 |
| Clean scenario-level observations | 1,083 |
| Stage-level at-risk rows | 3,047 |

The case-study dataset is used here only to demonstrate the computational reconstruction pipeline. The present article does not make substantive behavioural claims about age, escalation trajectory or information format. Its contribution is the software artifact and the auditable conversion from wide stopping-rule survey exports to scenario-level and stage-level modelling datasets.

The reconstruction logic maps 1,108 potential Part B scenario observations from 277 consenting participants. It flags 25 ambiguous response sequences, retains 1,083 clean scenario-level observations and expands these into 3,047 stage-level at-risk decision rows. These counts show the difference between raw possible observations, valid threshold observations and model-ready at-risk rows.

## 5 Evaluation

The artifact is evaluated as a computational reconstruction system rather than as a new estimator. The evaluation asks whether the system has a concrete accessible software object, whether its mapping is configuration-driven, whether output files match the paper claims, whether the automated tests cover the main intended behaviours and whether the resulting data products reduce the risk of common manual preprocessing errors.

### 5.1 Local Verification and Public Artifact

The R artifact is available in a dedicated project OSF repository: https://osf.io/gyvmz/. The repository contains the submitted software artifact and verification materials, including modular R source code, JSON configurations, toy wide-format stopping-rule datasets, automated testthat files, six reference outputs, a second configuration demonstration, adversarial stress-test materials, software metadata and local verification records. The final primary artifact was tested locally in RStudio on R version 4.6.0. The expanded testthat suite completed successfully with 41 passes, 0 failures, 0 warnings and 0 skipped tests across test groups covering configuration loading, output-file generation, sequence validation, stage-level expansion, second-configuration reuse and assumption violations. The demonstration pipeline also completed successfully, generated all six expected output files, and recorded a runtime of 0.1449 seconds on the primary toy dataset.

Because the toy demonstration dataset is intentionally small, the model-fitting module first attempts the full GEE specification and then falls back to prespecified simpler formulas if the full model matrix is rank deficient. In the local verification run, the score-plus-trajectory exchangeable GEE model was used successfully and estimates were exported to `gee_coefficients.csv`. This fallback behaviour is part of the optional modelling wrapper rather than a change to the empirical reconstruction logic.

### 5.2 Reusability Through Configuration

The artifact now demonstrates reusability through a second structurally different toy configuration. The primary toy design uses two scenarios and five stages; the additional configuration uses three scenarios and seven stages, with a different participant identifier, different response columns and different stage scores. The same validation, reconstruction and stage-expansion functions are used for both designs. The second configuration test verifies that no source-function changes are required and reconstructs 9 clean scenario observations, 3 logged ambiguous observations and 42 stage-level at-risk rows from the alternative toy export. This does not prove universal generality, but it converts the reusability claim from an architectural argument into a checked example of configuration-driven adaptation.

### 5.3 Automated Checks and Failure Behaviour

The evaluation distinguishes successful intended behaviour from the failure mode that matters most in stopping-rule preprocessing: silent generation of apparently valid stage-level rows from invalid response sequences. The included tests cover configuration loading, presence of the six reference output files, recognition of clean intervention sequences, recognition of clean right-censored sequences, classification of action-followed-by-later-response sequences as ambiguous, exclusion of ambiguous observations from the reconstructed scenario-level dataset, stopping stage-level expansion after action, and retention of all observed stages for right-censored scenarios. The revised artifact also includes adversarial checks for a missing configured stage column, an unknown response label, a response after action, a missing value inside an otherwise observed sequence, a malformed export missing the participant identifier, and a zero-byte input file. These cases document whether the pipeline fails loudly, logs an exclusion or completes safely rather than silently producing wrong model-ready rows.

### 5.4 End-to-End Comparison With Manual Preprocessing

A manual preprocessing workflow for the toy data requires a researcher to inspect each scenario sequence, identify the first action, decide whether later responses should be ignored or treated as invalid, create a scenario-level threshold row, expand the sequence into at-risk rows and maintain a separate exclusion record. The pipeline makes those decisions explicit and repeatable. In the demonstration run, the software generated the scenario-level file, stage-level file and exclusion log from the same configured rules. The practical advantage is qualitative rather than a claim of statistical superiority: it reduces researcher discretion, makes exclusions inspectable, and prevents common event-history mistakes such as retaining post-action stages or assigning final-stage scores to right-censored non-interventions.

### 5.5 Scalability

The reconstruction algorithm is linear in the number of participant-scenario-stage cells that must be inspected. If $P$ is the number of participants, $S$ the number of scenarios per participant and $K$ the maximum number of stages, the validation step is $O(P{\cdot}S{\cdot}K)$, and the stage-level expansion produces at most $P{\cdot}S{\cdot}K$ rows. Memory use is also dominated by the resulting long-form stage-level table. This scaling is appropriate for typical survey exports because $K$ is usually small and fixed by design. The reported 0.1449-second runtime should therefore be read only as a local toy-data verification result, not as a benchmark for all future uses. For larger studies, the main practical limit is expected to be the size of the expanded stage-level file rather than the sequence-validation loop.

## 6 Discussion

The main contribution is the separation of stopping-rule reconstruction from model fitting. Existing modelling packages are powerful after the analyst has created the correct long-form data structure. The fragile step occurs earlier, when wide survey responses must be interpreted as thresholds, censoring decisions and at-risk rows. By making that step scripted, configured and logged, the artifact turns an often invisible preprocessing stage into an inspectable computational object.

The configuration-driven approach is useful because stopping-rule studies may vary in the number of scenarios, number of stages, stage scores and column names while sharing the same first-action logic. Separating these design features from the R functions makes adaptation easier and makes errors more visible: if a configured column is absent or a response label is unknown, this should be treated as a data-integrity problem rather than silently resolved by analyst judgement.

The artifact is deliberately modest. It is not a replacement for survival, geepack, statsmodels or lifelines. Its purpose is to produce the data structures those tools require. Its novelty is therefore practical rather than mathematical: it packages validation, reconstruction, exclusion logging and stage-level expansion for a class of sequential survey designs that otherwise invite ad hoc recoding.

## 7 Limitations and Future Work

The artifact currently assumes ordered stages and a first-action stopping rule. Designs that allow repeated actions, reversible decisions, competing event types or multiple possible interventions would require additional state logic. For example, a study in which participants can first choose to call a friend, later call a doctor and later call emergency services would need a competing-risk or multi-state extension rather than the single first-action rule used here.

The case-study dataset has also been used in a published empirical analysis of information-format effects on intervention thresholds in the same sequential health-decision task [17]. That article examines substantive behavioural associations between risk-information format and intervention timing, whereas the present article addresses the distinct computational problem of reproducible stopping-rule reconstruction, validation, exclusion logging and stage-level expansion. The dataset is therefore shared between the two articles, while the analyses, research questions and results are entirely separate.

The evaluation remains limited. The automated tests address the main intended behaviours and a small set of adversarial failure cases, but they do not prove robustness to every possible survey-platform export. Future work could turn the script-based artifact into a formal R package with continuous integration, vignettes, more bundled configuration templates and a public issue tracker. Larger simulated benchmarks would also be useful for measuring runtime and

**Table 4: Automated checks and adversarial stress-test behaviour.**

| Automated or adversarial check | Observed or required behaviour | Why it matters |
|---|---|---|
| Configuration loading | `study_config.json` loads and exposes scenario-stage mapping | Prevents analysis from proceeding without an explicit study map |
| Second configuration | 3-scenario × 7-stage configuration uses same source functions and reconstructs 9 clean scenarios, 3 exclusions and 42 stage rows | Demonstrates configuration-driven reuse rather than only asserting it |
| Reference outputs | All six expected primary output files are present | Confirms that the runnable artifact produces the documented files |
| Clean intervention sequence | First action is recognised and action stage is reconstructed | Defines the observed decision threshold |
| Right-censored sequence | All-wait sequence is recognised as non-intervention rather than final-stage action | Preserves right-censoring rather than inventing an event |
| Response after action | Sequence is classified as ambiguous and logged | Prevents impossible post-action rows from entering the at-risk dataset |
| Missing value inside sequence | Sequence is classified as ambiguous with reason `missing_inside_sequence` | Prevents interpolation or silent compression of stages |
| Unknown response label | Sequence is classified as ambiguous with reason `unknown_response` | Prevents unrecognised survey labels from entering model-ready data |
| Missing configured stage column | Reconstruction fails loudly rather than guessing absent data | Protects against malformed survey exports |
| Malformed export missing participant identifier | Reconstruction fails loudly rather than producing model-ready output | Prevents structurally incomplete survey exports from being processed as valid data |
| Zero-byte input file | Reconstruction fails loudly with "no lines available in input" rather than producing model-ready output | Prevents an empty survey export from being treated as valid input |
| Malformed configuration | Configuration loading fails when required fields are absent | Prevents reconstruction without a valid study map |
| Stage expansion | Expansion stops after action and retains all stages for censored scenarios | Creates valid event-history rows |

memory use across thousands of participants, many scenarios and longer stage sequences.

## 8 Conclusion

Stopping-rule designs are useful for studying sequential decision thresholds, but they create a specific data-engineering problem. Raw wide survey exports must be mapped, validated, reconstructed and expanded before they can be modelled correctly. This paper presents a runnable R artifact that performs this transformation using configuration-driven mapping, modular reconstruction functions and auditable outputs.

By turning stopping-rule responses into scenario-level thresholds and stage-level at-risk rows, the pipeline provides a bridge between behavioural survey exports and standard event-history or repeated-measures modelling tools. This makes the timing of action under uncertainty easier to analyse reproducibly in digital health, AI warning systems and other decision-support settings.

## Data Availability

The complete R software artifact and reproducibility materials are publicly available through the dedicated OSF project at https://osf.io/gyvmz/ and via the persistent DOI https://doi.org/10.17605/OSF.IO/GYVMZ. The repository contains all computational materials supporting this manuscript, including the modular R source code, primary and secondary JSON configurations, toy datasets, automated tests, adversarial validation checks, reference outputs, software metadata and local verification records.

## References

[1] Paul D. Allison. 1982. Discrete-time methods for the analysis of event histories. *Sociological Methodology* 13, 61–98.

**Table 5: Software-artifact evaluation summary.**

| Evaluation item | Artifact status |
|---|---|
| Public accessibility | OSF link supplied for code, toy data, tests, outputs and local verification records |
| Configurable mapping | Study-specific columns, labels, scenario definitions and stage scores stored in JSON rather than hard-coded in reconstruction functions |
| Second configuration | Additional 3-scenario × 7-stage toy configuration demonstrates reuse with changed column names, participant identifier and stage scores |
| Adversarial checks | Stress matrix covers missing stage column, unknown response, post-action response, internal missing value, malformed export and zero-byte input |
| Core source modules | Configuration loading, validation, scenario reconstruction, stage-level expansion, optional modelling and exporting |
| Reference outputs | Six primary files exported in outputs/ and verified in local run records |
| Testing | testthat checks cover primary reconstruction logic, second configuration and assumption-violation behaviour |
| Existing-tool comparison | Pipeline fills preprocessing gap before survival/geepack/lifelines-style modelling |
| Scalability | Linear expected scaling in participant-scenario-stage cells; toy runtime reported only as local verification |

[2] Judith D. Singer and John B. Willett. 2003. *Applied Longitudinal Data Analysis: Modeling Change and Event Occurrence*. Oxford University Press.

[3] Kung-Yee Liang and Scott L. Zeger. 1986. Longitudinal data analysis using generalized linear models. *Biometrika* 73, 1, 13–22.

[4] Ulrich Halekoh, Soren Hojsgaard and Jun Yan. 2006. The R package geepack for generalized estimating equations. *Journal of Statistical Software* 15, 2, 1–11.

[5] Terry M. Therneau. 2024. *A Package for Survival Analysis in R*. R package survival.

[6] Cameron Davidson-Pilon. 2019. lifelines: Survival analysis in Python. *Journal of Open Source Software* 4, 40, 1317.

[7] Statsmodels Developers. 2024. statsmodels: Statistical modelling and econometrics in Python.

[8] Roger D. Peng. 2011. Reproducible research in computational science. *Science* 334, 6060, 1226–1227.

[9] Victoria Stodden, Friedrich Leisch and Roger D. Peng, eds. 2014. *Implementing Reproducible Research*. CRC Press.

[10] Mark D. Wilkinson et al. 2016. The FAIR Guiding Principles for scientific data management and stewardship. *Scientific Data* 3, 160018.

[11] Karthik Ram. 2013. Git can facilitate greater reproducibility and increased transparency in science. *Source Code for Biology and Medicine* 8, 7.

[12] Hadley Wickham. 2014. Tidy data. *Journal of Statistical Software* 59, 10, 1–23.

[13] Greg Wilson, Jennifer Bryan, Karen Cranston, Justin Kitzes, Lex Nederbragt and Tracy K. Teal. 2017. Good enough practices in scientific computing. *PLOS Computational Biology* 13, 6, e1005510.

[14] Karl W. Broman and Kara H. Woo. 2018. Data organization in spreadsheets. *The American Statistician* 72, 1, 2–10.

[15] Mark P. J. van der Loo and Edwin de Jonge. 2021. Data Validation Infrastructure for R. *Journal of Statistical Software* 97, 10, 1–31.

[16] Richard Iannone, Mauricio Vargas and June Choe. 2026. pointblank: Data Validation and Organization of Metadata for Local and Remote Tables. R package version 0.12.4. DOI: 10.32614/CRAN.package.pointblank.

[17] Alex Kolesnikov. 2026. A color-coded urgency bar and intervention thresholds in sequential health decisions. *Journal of High School Science* 10, 3, 394–413. https://doi.org/10.64336/001c.166227.

# Precision versus Efficiency: A Quantized DistilBERT Framework for Automated Mental Health Text Triage

Ayaan Goswami
Carnegie Vanguard High School
Houston, Texas, USA
ayaangoswami17@gmail.com

Clayton Greenberg
Saarland University
Saarbrücken, Germany

## Abstract

This study addresses the challenge of automatically triaging online mental health data by engineering a text-analysis system capable of accurately and efficiently categorizing human psychological distress in a lightweight, web-deployable format. As more people turn to social media and other online platforms when experiencing psychological distress, for both interaction with others and to seek support, the volume of such disclosure increasingly outpaces the capacity of moderators and clinicians to review it, forming a bottleneck to accurate and efficient triaging. To overcome the bottleneck, this study proposes an automated classification system, deployed as a web application where users submit text directly. Within this approach, an optimization framework was utilized, targeting the classification head, encoder, inference runtime, and numeric precision of a bidirectional deep learning transformer (BERT). Because inference cost is dominated by the encoder rather than the classifier, optimizing the classification head yielded no speedup, so the encoder itself was replaced with a distilled model and quantized. The final framework was able to successfully categorize raw user text into seven different psychological states, at an overall accuracy of about 82% on unseen test data. Additionally, the distilled and quantized model measurably reduced response time, lowering latency from 186.4 to 43.8 ms and model size from 438.7 to 68.3 MB, with per-class performance reported for every configuration. The result quantifies what each optimization costs: the runtime change is numerically lossless but contributes under 10% of the speedup, distillation and quantization supply the rest, and quantization damages the smallest classes severely for one encoder while leaving the other's per-class F1 intact. The deployed system is offered as a research prototype for prioritizing human review of written text, not as a diagnostic or standalone screening tool.

## Keywords

Natural Language Processing, Text Classification, Mental Health Triage, Machine Learning, BERT, Optimization, Sentiment Analysis


*IJSCAR VOL. 3, ISSUE 3*


**ACM Reference Format:**
Ayaan Goswami and Clayton Greenberg. 2026. Precision versus Efficiency: A Quantized DistilBERT Framework for Automated Mental Health Text Triage. In *Proceedings of International Journal of Secondary Computing and Applications Research (IJSCAR VOL. 3, ISSUE 3).* ACM, New York, NY, USA, 15 pages. https://doi.org/10.67149/yhjs2024.5/5wcbbc4m

## 1 Introduction

### 1.1 Context and Background

In recent years, the overlap between digital communication and mental health has become a critical focal point for both psychological research and computational linguistics. Individuals who experience psychological distress increasingly turn to online platforms and social media to facilitate social interaction, access peer support, and articulate their struggles to seek mental health help [13]. Dizavandi and colleagues define mental health triage as the process of "assessing whether the patient... needs mental health care [and assessing] what level of urgency [they] fall into" [2]. However, the large amount of digital mental health data has created a severe bottleneck in mental health triage, as the volume of online disclosure far outpaces the capacity of moderators and clinicians to review it. Because of this bottleneck, "rapidly and accurately assessing clinical characteristics to determine the severity of psychiatric emergencies [has become] one of the most critical initial steps to providing appropriate treatment" [19].

To meet the demand for increasingly rapid and accurate assessments, efforts have been made to develop automated tools, such as artificial intelligence (AI) models, for mental health analysis and triage. However, rapid automated tools that can accurately filter and prioritize massive volumes of unstructured text to identify who needs immediate help remain scarce, and there is a further gap in balancing the accuracy and efficiency of such tools, since the most accurate tools are the most computationally expensive to run, and the cheapest tools have subpar accuracy, making both types unsatisfactory for deployment. These gaps lead to the core research question driving this study: under a limited computational budget, how can an automated text-analysis system be optimized to categorize human psychological distress accurately and efficiently in a web-deployable format, and what does each optimization cost?

### 1.2 Service and Target Audience

To address the research question, this project presents a web-based automated triage application designed to analyze user-inputted text and classify its underlying psychological state. The core users of this application include platform moderators who desire automated safety filters to flag high-risk posts, clinical organizations looking for preliminary evaluation tools, and general users looking for a quick reflection of how an automated classifier reads their own text. Users submit text to the application directly. The system does not monitor or collect posts from social media, and is engineered to minimize the computational cost of each request so that it remains responsive on low-cost hosting.

### 1.3 Problem Formulation

This study is structured as a supervised classification problem, wherein the system is trained on over 42,000 samples of complex language data from dedicated online mental health support communities [18]. This input consists of raw, unstructured human text, with the output being a set of categorical labels predicting one of seven discrete mental health statuses. The system is intended for risk screening rather than diagnosis, and the labels it predicts are inherited from the categories under which the training data was originally collected rather than from clinical assessment.

## 2 Dataset

### 2.1 Dataset Source

This study utilized the public "Sentiment Analysis for Mental Health" dataset compiled by Suchintika Sarkar on Kaggle [18], containing 53,043 samples of raw textual data compiled from social media platforms and public forums such as Reddit and Twitter. The dataset has two primary variables: "statement" (the independent feature), the raw user-generated text to be parsed by the pipeline, and "status" (the dependent feature), a categorical label mapping each statement into one of seven mutually exclusive classes: Normal, Depression, Suicidal, Anxiety, Stress, Bipolar, and Personality disorder.

The corpus is an aggregation of nine previously published datasets: 3k Conversations Dataset for Chatbot, Depression Reddit Cleaned, Human Stress Prediction, Predicting Anxiety in Mental Health Data, Mental Health Dataset Bipolar, Reddit Mental Health Data, Students Anxiety and Depression Dataset, Suicidal Mental Health Dataset, and Suicidal Tweet Detection Dataset. Each was scraped and labeled independently under its own scheme. The compiler applied no unified annotation pass across them and does not document how each source maps onto the seven classes. The seven statuses were therefore inherited from the source files rather than assigned by clinicians, and the dataset carries no diagnostic assessment, annotation codebook, or inter-annotator agreement statistic. The labels indicate the category under which a statement was originally collected, which correlates with psychological state but does not measure it.

### 2.2 Preprocessing

Missing data (NaN values) were removed, eliminating 362 rows and leaving 52,681 usable samples. Next, the "status" labels were converted to numerical IDs using label encoding. The dataset was then divided into a 20% testing set and an 80% training set, producing 42,144 training samples and 10,537 testing samples. As seen in Figure 1, the dataset exhibits a clear class imbalance, with Normal (≈31.0%) and Depression (≈29.2%) comprising the majority of entries, followed by Suicidal (≈20.2%), while the specialized clinical conditions of Anxiety, Bipolar, Stress, and Personality disorder together account for under 20%. Because of the extreme class imbalance, overall accuracy is weakly informative on this task, and all results are additionally reported per class. To account for the imbalance, stratified sampling was utilized to maintain the original proportional distribution of each status across both the train and test sets. Finally, a pre-trained Hugging Face tokenizer was applied to the "statement" column, truncating long text to 128 tokens, padding shorter sequences to a uniform length, and generating the input and attention-mask tensors the model requires [20]. The 128-token cap was selected as a deployment tradeoff, since sequence length drives inference cost quadratically in the attention mechanism. Under this cap, 33.5% of statements are truncated, and no ablation at longer sequence lengths was run. Section 3 returns to what this choice may cost in accuracy.

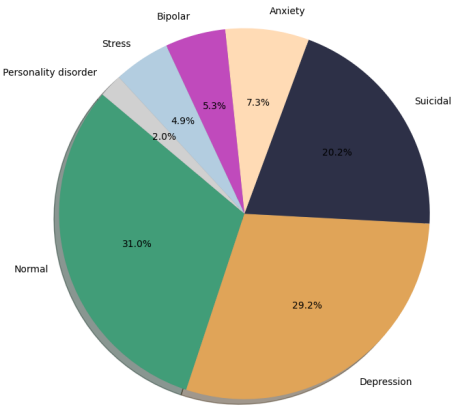


**Figure 1: Distribution of Mental Health Conditions in the Sarkar (2024) Dataset.**

### 2.3 Corpus Quality

Because the corpus is an aggregation of independently collected sources, it was audited for duplication and label consistency before any model was trained. Statements were normalized by lowercasing, removing non-alphanumeric characters, and collapsing whitespace. Under this normalization, the 52,681 usable rows contain 50,960 unique statements, so 1,721 rows (3.27%) are duplicates of another row. In addition, 1,453 distinct statements appear more than once after

**Table 1: Label pairs in conflict among repeated statements. Depression/Suicidal is the most frequent disagreement in the corpus.**

| Conflicting labels | Statements | Rows |
|---|---|---|
| Depression / Suicidal | 17 | 35 |
| Anxiety / Depression | 9 | 24 |
| Normal / Stress | 2 | 4 |
| Depression / Normal | 1 | 2 |
| Depression / Normal / Stress / Suicidal | 1 | 13 |
| Total | 30 | 78 |

normalization. A further 494 statements normalize to fewer than ten characters, and three normalize to the empty string.

A portion of the corpus is not mental health disclosure at all: the most common repeated entries include "what do you mean" (22 occurrences), "why not" (16), and a study recruitment advertisement appearing 10 times under the Stress label, none containing clinical content.

Of the 1,453 repeated statements, 30 (2.06%) carry conflicting labels, affecting 78 rows. The most extreme case is a statement normalizing to "name," which appears 13 times under four different statuses, including one instance labeled Suicidal. This is a lower bound on label noise rather than an estimate of it, since disagreement is only observable where a statement happens to repeat, and repeats are unlikely to be representative. What the audit establishes is that the mapping from text to label is not deterministic.

The distribution of conflicting labels, however, is more informative than their frequency. As shown in Table 1, 17 of the 30 conflicting statements (57%), covering 35 of the 78 affected rows, are disagreements between Depression and Suicidal, likely due to their high level of comorbidity [8]. This double-labeling of statements could be indicative that some Depression/Suicidal confusion when modeling may reflect inconsistency in the supervision rather than error by the model.

## 2.4 Train/Test Leakage

Because duplicates are distributed across the corpus, stratified splitting can place copies of the same text in both partitions, inflating measured performance. The test set was therefore audited for overlap with the training set. A test row counted as leaked if its normalized text exactly matched a training row, or if its maximum character 3–5-gram TF-IDF cosine similarity against the training set reached 0.90. Cosine similarity over-fires on very short texts, where character overlap is high but content unrelated ("very beautiful" and "is it beautiful" score highly while sharing little substance), so near-duplicate matches were confirmed only when the pair additionally shared at least half of its words and the test statement exceeded 30 normalized characters.

Under this rule, 517 test rows are exact duplicates of training rows and a further 101 are confirmed near-duplicates, giving 618 leaked rows, or 5.87% of the test set. Omitting the word-overlap confirmation raises the count to 705 rows (6.69%), so that step removes 87 short-text collisions. The estimate is insensitive to the threshold: varying the cosine cutoff from 0.80 to 0.99 moves the leakage rate only between 6.24% and 5.22%.

**Table 2: Train/test leakage by class. Contamination concentrates in the smallest classes, and the Suicidal class is least affected.**

| Class | Test rows | Leaked | Leaked % |
|---|---|---|---|
| Personality disorder | 215 | 54 | 25.12 |
| Stress | 517 | 93 | 17.99 |
| Bipolar | 556 | 97 | 17.45 |
| Anxiety | 768 | 76 | 9.90 |
| Depression | 3081 | 134 | 4.35 |
| Normal | 3269 | 125 | 3.82 |
| Suicidal | 2131 | 39 | 1.83 |
| Overall | 10537 | 618 | 5.87 |

Leakage is distributed unevenly across classes, as depicted in Table 2. The three smallest classes are most contaminated, with roughly a quarter of Personality disorder test items appearing in training. Suicidal, the class of greatest consequence for triage, is least contaminated at 1.83%, so performance on the safety-critical class is not materially inflated by memorization.

## 2.5 Consequences for Evaluation

Because leakage concentrates in the smallest classes, it distorts macro-averaged metrics far more than overall accuracy. Across the configurations evaluated in Section 5, removing leaked rows lowers accuracy by 0.30 to 0.46 percentage points but lowers macro-F1 by 1.48 to 1.96 points, an effect roughly four to five times larger. Overall accuracy is therefore doubly misleading on this corpus, as it under-weights the minority classes and conceals most of the inflation that duplication introduces.

All accuracy, macro-F1, and per-class results reported in this paper are computed on the leakage-filtered test set of 9,919 samples, with full test set figures given alongside for comparability with prior work. The calibration analysis of Section 5.8 is the one exception, and is reported on the full test set for the reason given there. The training set was left intact, as deduplicating it would alter both the size and class distribution of the training data, yielding models different from the ones deployed, whereas filtering the test set corrects the measurement without changing what is measured. Duplication internal to the training set may still encourage memorization that a test-side filter cannot detect, so this correction is partial.

# 3 Related Work

## 3.1 Classification on the Sarkar Corpus

The dataset has been analyzed in several prior publications, allowing this work to be positioned against reported

results on identical data. Kallstenius et al. compared three approaches on 52,681 statements: a traditional pipeline combining TF-IDF features with a support vector machine, a prompt-engineered GPT-4o-mini, and a fine-tuned GPT-4o-mini [10]. The TF-IDF pipeline achieved the highest overall accuracy at 95%, the fine-tuned model reached 91%, and prompting alone reached 65%. Rehman et al. paired several embedding schemes with recurrent and transformer architectures on 51,074 entries of the same corpus, reporting 95.71% for BERT with Word2Vec embeddings and 94.47% for RoBERTa with FastText, against 81.08% for a bidirectional LSTM [16].

The accuracy reported in this study is substantially lower than the strongest of these figures, a difference that warrants explanation. Several methodological differences prevent direct comparison. Kallstenius et al. augmented the corpus by back-translating statements through French without stating whether augmentation preceded the train/test split; if it did, augmented variants of the same statement could appear on both sides of it. That study also describes the corpus as 52,681 unique statements, whereas the normalization in Section 2.3 finds 50,960 distinct statements among those rows. Rehman et al. removed exact duplicates before modeling but report no split ratio and do not examine near-duplicates. Neither study reports the sequence-length limit used, which materially affects both accuracy and inference cost. A further difference is internal to this work: the 128-token cap truncates the longest statements in the corpus, and no ablation at longer sequence lengths was run, so some part of the gap may be attributable to this deployment-motivated choice rather than to methodology in the compared studies.

This work therefore does not claim to improve on the results of the literature. What the comparison shows is that neither study reports a near-duplicate audit or any latency, throughput, or model size figure, and only Kallstenius et al. report per-class metrics. The two axes central to this paper, per-class behavior under compression and measured inference cost, are therefore absent from prior work on this corpus, as is any quantification of the leakage it contains.

## 3.2 Data Quality in Social Media Corpora

The audit reported in Section 2.4 follows an established line of work rather than introducing a new method. Elangovan et al. showed that overlap between training and test partitions in public NLP datasets inflates reported results, so measurements interpreted as generalization partly reflect memorization [3]. Mu et al. examined twenty datasets widely used in computational social science, found most contain duplicate and near-duplicate samples producing label inconsistencies and leakage, and concluded that duplication affects existing claims of state-of-the-art performance [12]. Their finding that few dataset papers document any deduplication step is consistent with what is observed here. This study applies those methods to a corpus actively used for mental health classification, while establishing baselines that account for inference cost and compression.

## 3.3 Efficient Transformer Inference

Reducing the cost of transformer inference is a well-developed area, as seen in the literature surrounding this topic. Knowledge distillation is a method that trains a smaller student model to reproduce the behavior of a larger teacher. In the case of BERT models, DistilBERT retains most of BERT's performance with roughly 40% fewer parameters, and is accordingly the distilled encoder used in this study [17]. Quantization reduces the numeric precision of weights and activations, typically from 32-bit floating point to 8-bit integers, which shrinks the model and allows integer arithmetic at inference [9]. Zafrir et al. applied 8-bit quantization to BERT specifically, reporting a fourfold reduction in model size with limited accuracy loss [21].

Prior work on this corpus has appealed to efficiency without measuring it: Kallstenius et al. argue their pipeline is preferable partly because it is smaller and faster than a large language model, but report no latency, throughput, or model size [10]. This study treats these as measurements rather than assumptions, and reports them for every configuration.

## 3.4 Per-Class Effects of Compression

Compression can damage some classes far more than aggregate metrics suggest. Hooker et al. found that pruned and quantized image classifiers retain comparable overall accuracy while diverging sharply on a narrow subset of inputs, with the cost falling disproportionately on underrepresented parts of the distribution [5]. They report quantization as considerably less damaging than pruning, but the risk remains that 8-bit quantization alone can cost a minority class substantial performance while overall accuracy barely moves.

# 4 System

## 4.1 Text Representation

Since the data were raw textual statements, an algorithm was needed to translate them into a mathematical format a computer can process. Several were tested during development, ranging from word-frequency methods such as Bag-of-Words and TF-IDF to neural models such as BERT [1, 11].

The first approaches tested were frequency-based. A Bag-of-Words (BoW) model counts how often each unique word in the training set appears and classifies new inputs on those frequencies. TF-IDF (Term Frequency-Inverse Document Frequency) refines this by weighting each term frequency by the logarithm of its inverse commonness, reducing the impact of filler words such as "and" and "the." Both are computationally inexpensive, but are rigid and context-free, unable to account for word order, syntax, or negation.

To overcome these limitations this study used neural network-based techniques, specifically embeddings and BERT. Where TF-IDF assigns static values to words, embeddings represent them as high-dimensional vectors that capture conceptual relationships rather than frequencies. BERT (Bidirectional Encoder Representations

from Transformers) was chosen for its bidirectionality and contextual awareness: unlike unidirectional models it reads the words before and after a token to interpret its use, and its attention mechanism weights each word by those around it. These features make it far more nuanced than BoW or TF-IDF, but also much less efficient, which constrains its deployment on the low-cost hardware available to this project.

## 4.2 Optimization Axes

BERT's efficiency limitation was therefore treated as an optimization problem rather than an architectural one. A fine-tuned BERT model with its native classification head serves as the reference configuration, and four independent axes were identified along which its inference cost might be reduced: the classification head, the encoder, the inference runtime, and the numeric precision of the weights. Each axis was varied separately so its contribution could be attributed, and every configuration was evaluated under the protocol in Section 4.4.

The first axis is the classification head. An earlier version of this system replaced BERT's linear output layer with an optimized Random Forest classifier, using BERT solely as a static feature extractor over 768-dimensional embeddings. Optimizing this axis cannot reduce inference cost, because every input must still pass through the full encoder to produce that embedding, and substituting 300 decision trees for a single linear layer adds work rather than removing it. Section 5 reports the measured outcome. The native head was retained and optimization was directed at the encoder instead.

The second axis is the encoder itself. Two options were tested. The first was to skip fine-tuning and extract embeddings from a pre-trained encoder, which removes the cost of task-specific training. However, as Section 5 shows, this also substantially degrades the model's accuracy. The second was to replace the encoder with a smaller one, namely the released DistilBERT checkpoint described in Section 3 [17]. Rather than distilling from the BERT model directly, this study fine-tuned that checkpoint on the mental health corpus.

The third and fourth axes concern how the encoder is executed rather than what it contains. Both encoders were exported to the ONNX interchange format and served through ONNX Runtime, replacing the PyTorch execution path without altering any weights. Each was then quantized dynamically to 8-bit integers, a standard technique that reduces model size and permits integer arithmetic at inference in place of floating-point operations [9, 21]. Dynamic quantization computes activation scales at run time, so its behavior depends on the composition of each batch. This property is examined in Section 5.

## 4.3 Model Selection Criteria

Compressing a model trades quality for cost, so a criterion was required to determine which configurations remained acceptable. To avoid selecting a threshold that favored a preferred outcome, this study adopted the tolerance used by the MLPerf Inference benchmark, which requires that an optimized submission achieve at least 99% of the accuracy of the unmodified fp32 reference model, with quantization the permitted form of optimization [15]. The same 99% relative tolerance is applied here in two places: to overall accuracy, and to recall on the Suicidal class. The second constraint reflects a failure asymmetry, in which failing to flag a user at risk is a more consequential error than raising a false alarm, and it prevents a configuration from qualifying by trading away performance on the class that matters most for triage.

Among the configurations satisfying both constraints, the one with the lowest per-request latency was selected for deployment. These criteria were fixed before the configurations were compared.

## 4.4 Measurement Protocol

All configurations were measured under a single fixed protocol, since latency figures obtained under differing thread counts, batch sizes, or hardware are not comparable. Every measurement was taken on the CPU, matching the deployment environment of Section 6, on an Intel Core i7-14650HX with the thread budget of both PyTorch and ONNX Runtime pinned to two threads to approximate the allocation available to the hosted application. Inputs were truncated to 128 tokens. Accuracy was computed over the complete hold-out test set in batches of 32, while latency was measured one request at a time, matching the way the deployed application serves traffic. Batching of 32 was adopted for tractability; Section 5.9 shows the fp32 figures are insensitive to this choice, while quantized BERT varies by 2.55 accuracy points across batch sizes and the deployed model by 0.28. Each configuration was timed over 300 single requests drawn from the test set, discarding the first 10 as warm-up, and the reported figure is the mean, in ms. Throughput was instead measured under batched inference at batch size 32, since a deployed service amortizes cost across concurrent requests, so it is not the reciprocal of the single-request latency and the two columns are not directly comparable. The small reversals between the fp32 PyTorch and ONNX rows of Table 4 fall within the run-to-run variation of that measurement and should not be read as an ordering. Model size is that of the deployed artifact, comprising weights, configuration, and tokenizer, excluding optimizer state and training checkpoints.

Absolute latencies are specific to this processor, which combines performance and efficiency cores and is subject to thermal limits under sustained load, but comparisons between configurations hold because all were measured on the same machine under identical settings.

**Table 3: Per-class performance of the deployed quantized DistilBERT model on the leakage-filtered test set ($n = 9{,}919$).**

| Class | Precision | Recall | F1 | Support |
|---|---|---|---|---|
| Anxiety | 0.8342 | 0.8873 | 0.8599 | 692 |
| Bipolar | 0.8416 | 0.8105 | 0.8257 | 459 |
| Depression | 0.7766 | 0.7665 | 0.7715 | 2947 |
| Normal | 0.9479 | 0.9612 | 0.9545 | 3144 |
| Personality disorder | 0.7000 | 0.6522 | 0.6752 | 161 |
| Stress | 0.6986 | 0.7217 | 0.7100 | 424 |
| Suicidal | 0.7228 | 0.7103 | 0.7165 | 2092 |
| Macro average | 0.7888 | 0.7871 | 0.7876 | 9919 |
| Accuracy | | | 0.8231 | 9919 |

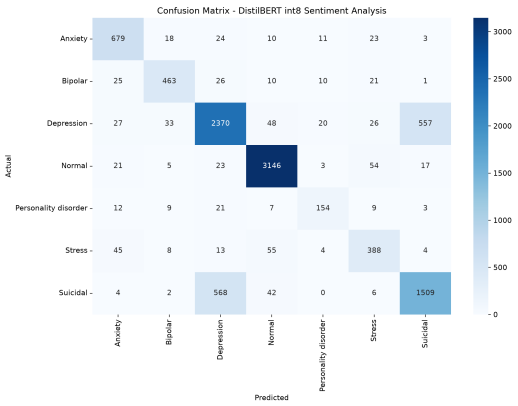


**Figure 2: Confusion Matrix of the Deployed Quantized DistilBERT Model.**

## 5 Results and Discussion

All results in this section are reported on the leakage-filtered test set of 9,919 samples described in Section 2.4, with full test set figures given alongside where comparability with prior work matters. The calibration analysis of Section 5.8 is the single exception and is reported on the full test set. Unless stated otherwise, every configuration was measured under the protocol in Section 4.4.

### 5.1 Model Performance

The deployed quantized DistilBERT model achieved an overall accuracy of 82.31% on the leakage-filtered evaluation set and 82.65% on the full test set, where accuracy is the proportion of correctly predicted statuses out of all predictions. Because overall accuracy is weakly informative under the class imbalance of Section 2.2, precision, recall, and F1 are reported for every class in Table 3, giving a macro-averaged F1 of 0.7876. Figure 2 shows the confusion matrix, with the principal diagonal representing correct predictions.

The model was reasonably accurate across all statuses, with the most prominent confusion coming from mixing up depression and suicidal. The encoder relies on contextual embeddings, and highly similar contexts mapping to distinct but related psychological states blur its decision boundaries. As reported in Section 2.3, this same pair accounts for 57% of the labeling conflicts in the corpus itself, so part of the confusion likely reflects inconsistency in the supervision rather than model error.

Of the 2,092 suicidal statements, 553 were classified as depression and 41 as normal. Section 7.3 takes up the difference between those two failures.

### 5.2 Hyperparameter Selection

The model was tuned using the Hugging Face "TrainingArguments" module over 3 epochs to prevent overfitting on the 42,144 training samples. A learning rate of 2e-5 ensured stable gradient descent, training and evaluation batch sizes were set to 16, a weight decay of 0.01 penalized overly-large weights, and 16-bit (FP16) mixed precision training accelerated training. The same settings were used for DistilBERT, so any difference between the encoders is attributable to the encoder rather than its training schedule.

### 5.3 Comparison with Frequency Baselines

To establish what a transformer actually purchases, the Bag-of-Words and TF-IDF approaches of Section 4 were evaluated under the same protocol, and appear in the frequency block of Table 4. The strongest frequency baseline, TF-IDF with logistic regression, reaches 77.58% accuracy and 0.6883 macro-F1 against 82.31% and 0.7876 for the deployed model. The transformer buys 4.73 percentage points of accuracy and 9.93 of macro-F1, at roughly 45 times the latency and 15 times the size.

The gap is wider on the minority classes than the overall figures suggest: on Personality disorder the best baseline reaches 0.5135 F1 against 0.6752, and on Stress 0.4384 against 0.7100, so the advantage of contextual embeddings concentrates in the classes with the least training data.

### 5.4 Optimization Results

Table 4 reports every configuration retained for deployment consideration, and the two abandoned axes are reported in Section 5.5. Exporting either encoder to ONNX Runtime is numerically lossless, reproducing the PyTorch predictions exactly, and reduces latency by only 5-9%. Quantizing to 8-bit integers reduces it by a further factor of 2.02 for both encoders. The speedup is therefore attributable almost entirely to the encoder and the numeric precision, not the runtime.

Applying the selection criteria of Section 4.3, the reference is fine-tuned BERT in fp32, at 0.8312 accuracy and 0.7132 Suicidal recall. Quantized BERT reaches only 96.33% of reference accuracy and is rejected, while DistilBERT in fp32 reaches 98.73% and is also rejected. Quantized DistilBERT reaches 99.02% of reference accuracy and 99.60% of reference Suicidal recall, satisfying both constraints, and is the fastest to do so. It was therefore deployed, reducing latency from 186.4 ms to 43.8 ms and size from 438.7 MB to 68.3 MB.

**Table 4: All configurations retained for deployment consideration. The two abandoned optimizations are reported in Section 5.5. Accuracy, macro-F1, and Suicidal recall are reported on the leakage-filtered test set. Latency, throughput, and size are measured under the protocol of Section 4.4. The deployed configuration is highlighted in bold.**

| | Configuration | Acc. | Macro-F1 | Suic. R | ms/req | req/s | MB |
|---|---|---|---|---|---|---|---|
| freq. | BoW + logistic regression | 0.7608 | 0.6841 | 0.6711 | 0.84 | 9694.9 | 4.2 |
| | TF-IDF + logistic regression | 0.7758 | 0.6883 | 0.6788 | 0.98 | 9055.9 | 4.6 |
| | TF-IDF + multinomial naive Bayes | 0.6509 | 0.4209 | 0.4990 | 1.03 | 9684.0 | 7.4 |
| BERT | fp32, PyTorch | 0.8312 | 0.7964 | 0.7132 | 186.4 | 3.8 | 438.7 |
| | fp32, ONNX Runtime | 0.8312 | 0.7964 | 0.7132 | 170.3 | 3.6 | 439.2 |
| | int8, ONNX Runtime | 0.8007 | 0.7406 | 0.7787 | 84.4 | 7.0 | 111.4 |
| DistilBERT | fp32, PyTorch | 0.8206 | 0.7839 | 0.7424 | 93.2 | 7.5 | 268.8 |
| | fp32, ONNX Runtime | 0.8206 | 0.7839 | 0.7424 | 88.4 | 7.0 | 268.9 |
| | **int8, ONNX Runtime** | **0.8231** | **0.7876** | **0.7103** | **43.8** | **13.1** | **68.3** |

The margin is narrow. The deployed configuration clears the 99% threshold by 0.02 percentage points on the filtered test set and by 0.08 on the full one, but a paired bootstrap with 5,000 resamples places the 95% confidence interval on that ratio at [98.31%, 99.74%] and the probability that it exceeds the threshold at 0.51. The criterion is therefore met at the point estimate but not robustly, and the separation between the quantized and unquantized DistilBERT configurations should not be treated as established.

One consequence of this criterion deserves comment. The 99% tolerance is applied to overall accuracy, a metric this paper elsewhere argues is weakly informative under class imbalance. It is retained as the primary gate because it is externally specified rather than chosen here, and because a criterion defined on a single class is trivially gameable. The cost is visible: quantized BERT attains the highest Suicidal recall of any configuration evaluated, 0.7787, yet is rejected for falling to 96.33% of reference accuracy, and its expected triage cost is the worst of the four transformer configurations at 0.3843, because that recall gain is purchased by damage to Personality disorder and Stress. A rule keyed to Suicidal recall alone would therefore have deployed the model with the worst aggregate cost-sensitive behavior. This tension between a class-specific safety metric and an aggregate one is not resolved here, and a principled multi-objective criterion is left to future work.

## 5.5 Optimizations That Did Not Succeed

Two of the four axes were investigated and abandoned, both evaluated under the canonical configuration and on the leakage-filtered test set. Replacing the native head with an optimized Random Forest trained on the fixed 768-dimensional embeddings from the fine-tuned model's [CLS] token produced essentially the same accuracy, 0.8300 against 0.8312, and 1.67 points lower Suicidal recall. It did not reduce inference cost: with the embedding already computed, evaluating the 300 class-balanced decision trees of depth 25 is more expensive than a single 768-to-7 linear layer, and both are negligible beside the encoder forward pass preceding them [14].

The second abandoned optimization was to skip fine-tuning and extract embeddings from a pre-trained encoder, removing the cost of task-specific training. This proved far more damaging than any compression step, reducing accuracy by 18.24 points, macro-F1 by 35.36, and Suicidal recall by 33.41. Together these results locate both the cost and the capability in the encoder: the head is too cheap for its replacement to matter, and the fine-tuned representations too important to discard. The only remaining way to reduce cost without destroying accuracy is a smaller encoder.

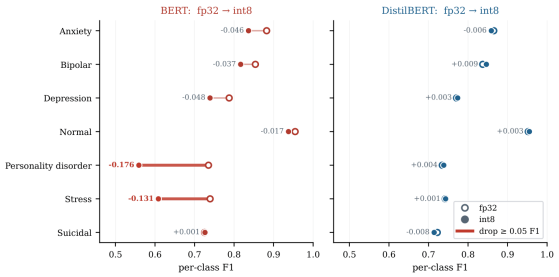


**Figure 3: Change in per-class F1 under 8-bit quantization for both encoders, on the leakage-filtered test set. Hollow markers are fp32 and filled markers are int8. Drops of at least 0.05 F1 are highlighted. Quantization is near-lossless for DistilBERT but costs BERT heavily on the two smallest classes.**

## 5.6 Per-Class Effects of Compression

Aggregate accuracy conceals the most important consequence of quantization. Figure 3 shows per-class F1 for both encoders before and after quantization.

Quantizing BERT costs 3.05 points of overall accuracy, which appears tolerable, but the loss is not evenly distributed: Personality disorder falls by 0.129 F1 and Stress by 0.122, while the largest class loses 0.015. The redistribution

is not uniformly negative: recall on Suicidal rises from 0.7132 to 0.7787, apparently because quantization noise shifts the Depression/Suicidal boundary toward the higher-risk class. The gain is paid for by the two smallest classes and by a 3.05-point accuracy loss, which is why the configuration is nonetheless rejected. The identical procedure applied to DistilBERT leaves per-class F1 essentially unchanged and in several classes marginally improves it, so two configurations separated by 2.24 points of overall accuracy differ by 12.9 points of F1 on the smallest class. The exception is Suicidal recall, which falls from 0.7424 to 0.7103, but is offset in F1 as precision rose to compensate. However, the recall loss is still the safety-relevant quantity and is taken up in Sections 5.7 and 7.3.

This extends the findings reviewed in Section 3: 8-bit quantization alone can remove a substantial fraction of a minority class's performance, and the damage depends on which architecture is quantized [5, 6].

## 5.7 Cost-Sensitive Evaluation

Overall accuracy also treats every error as equally serious, which is inappropriate for triage. An expected triage cost (ETC) was therefore computed by assigning a cost to each type of misclassification and averaging over the test set: classifying a suicidal statement as normal raises no flag at all, and is assigned 10; classifying it as another form of distress still routes the user to an elevated-risk category and is assigned 3; missing any other distress is assigned 3; confusing one distress category for another is assigned 1; and a false alarm on a normal statement is assigned 1. These weights encode an ordering over error types rather than a calibrated clinical cost, and were fixed before configurations were compared.

The deployed configuration scores 0.3533, against 0.3441 for the fp32 reference and 0.3323 for unquantized DistilBERT, the best transformer configuration. Quantized BERT is worst at 0.3843. The deployed model is therefore not what this metric would select, and the 41 suicidal statements it assigns to the normal class compare with 31 for unquantized DistilBERT. This is a tradeoff of the deployment decision: the criteria prioritize the latency benefits that make the system practical, subject to the accuracy and recall constraints, at a cost of 0.0210 ETC.

## 5.8 Calibration

Because the interface displays prediction probabilities to users, their reliability was evaluated on the full test set using expected calibration error (ECE), Brier score, and negative log-likelihood. This is the one analysis reported on the unfiltered set. Because the leaked rows are ones the model has effectively already seen, retaining them can only make stated confidence appear better matched to accuracy than it is, so the overconfidence reported below is a lower bound. Every configuration is overconfident, in that mean confidence exceeds accuracy, which Figure 4 shows as bars sitting below the diagonal across most of the confidence

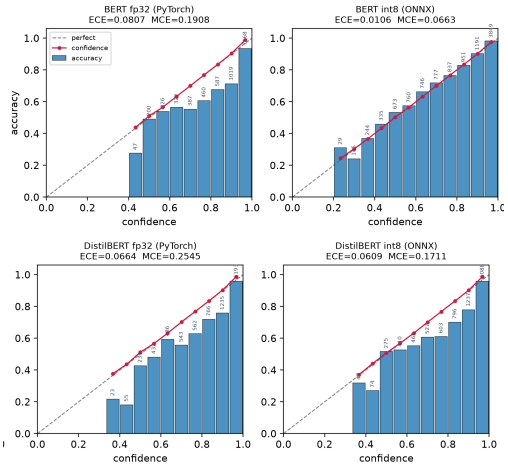


**Figure 4: Reliability diagrams for each configuration on the full test set, 15 equal-width bins. Bars give observed accuracy, the line gives mean confidence, and the dashed diagonal is perfect calibration. Bars falling below the diagonal indicate overconfidence.**

range. The deployed model assigns confidence of at least 0.99 to 38.2% of predictions while being correct on 82% of them, though its maximum confidence is 0.9992, so the interface never displays a probability of 100%.

Quantization leaves DistilBERT's calibration essentially unchanged, moving ECE from 0.0664 to 0.0609 and Brier score from 0.2510 to 0.2521. Its effect on BERT is far larger, reducing ECE from 0.0807 to 0.0106, but this does not indicate improvement, as ECE measures only whether stated confidence matches observed accuracy, and quantization noise flattens the confidence distribution of a previously overconfident model. Brier score and negative log-likelihood, which reward correct predictions as well as calibrated ones, rank quantized BERT last at 0.2763 and 0.5200. Selecting on calibration error alone would therefore have chosen the least accurate of the transformer configurations.

## 5.9 Robustness Checks

Three properties of the evaluation were examined to determine how far the reported figures depend on measurement choices. Dynamic quantization computes activation scales at run time from tensors whose padding depends on batch composition: across batch sizes of 1, 32, and 64, quantized DistilBERT varies by 0.28 accuracy points while quantized BERT varies by 2.55, so evaluation batching should match the serving configuration.

The same bootstrap distinguishes which reported differences exceed evaluation noise. The deployed model's accuracy deficit against the fp32 reference is reliable at $-0.0082$ (95% CI $[-0.0141, -0.0021]$), whereas its deficits on macro-F1 ($-0.0088$, CI $[-0.0182, +0.0006]$) and on Suicidal recall ($-0.0029$, CI $[-0.0192, +0.0132]$) are not distinguishable from zero. Compression is therefore measurably costly

in aggregate accuracy while leaving the safety-critical metric statistically indistinguishable from the reference.

The effect of the leakage documented in Section 2.4 was also examined, and is quantified there. The single exception is quantized BERT, which scores marginally lower on leaked rows than on unseen ones.

## 5.10 Analysis of Edge-Case Classifications

To evaluate the boundary limitations and pragmatic capabilities of the deployed system beyond static test metrics, a qualitative edge-case analysis was performed. Because a probe set written and interpreted by the same authors risks selecting for known weaknesses, the suite was authored externally. Two collaborators who were not involved in developing the system wrote fifty probes, ten for each of five linguistic phenomena, with the label an external human reader would assign. The suite was frozen before any item was run. All results were recorded, and are visible in Appendix A, which lists every probe.

The deployed model classified 24 of the 50 probes correctly, with accuracy varying sharply by phenomenon: 80% on hyperbole, 60% on minimized distress, 40% on figurative idiom, and 30% on both sarcasm and temporal blindness. Idiom, sarcasm, and temporal hedging all require the model to override the literal sentiment of the tokens present, whereas hyperbole is handled comparatively well because its exaggeration is lexically overt.

These failures share a direction, as of the 26 misclassified probes, 20 were assigned to the Normal class and none to Suicidal. The model therefore does not over-escalate on figurative language, it under-reacts to distress expressed indirectly. Nine of the ten sarcastic probes were classified as normal, seven of them incorrectly and at a mean confidence of 0.985, and every idiom failure went to normal. Temporal blindness fails in both directions, with four ongoing disclosures classified as normal and three resolved narratives still assigned a clinical label, indicating that the model keys on clinical vocabulary rather than on tense or trajectory. Minimized distress is handled better, at 60%, but two of its four failures still resolve to normal.

The safety consequence appears in the three probes whose intended label was Suicidal. All three were missed. A sarcastic disclosure of suicidal ideation was classified as normal at 0.919, a temporally hedged one as normal at 0.968, and a minimized one as depression, with 0.305 assigned to the Suicidal class, a distribution whose consequence for the crisis-resource threshold is traced in Section 7. The model is also confidently wrong rather than uncertain, as mean confidence on the twenty misclassifications that resolved to normal was 0.902, so these errors would not be caught by thresholding on the displayed probability.

These results identify a cost the aggregate metrics do not expose. The deployed configuration sits within 0.8 accuracy points of the fp32 reference and is statistically indistinguishable from it on Suicidal recall, yet answers fewer than half of a pragmatics suite correctly and fails in precisely the direction that matters for triage. With ten probes per category these rates carry wide confidence intervals and indicate which phenomena are difficult rather than precisely estimating performance. Pragmatic robustness is nonetheless a further axis, alongside per-class F1 and calibration, on which overall accuracy is uninformative.

## 5.11 Limitations

Several limitations must also be acknowledged. The training data was sourced from public social media platforms like Reddit and Twitter, biasing the model toward the demographic makeup of those platforms, often younger and internet-literate populations, and lowering its ability to generalize to older populations or to non-digital communication such as spoken transcripts. It is consequently suited to prioritizing which texts a human reviews first rather than to standalone screening. Section 7 sets out the intended human-in-the-loop configuration and the uses the system does not support. The model also classifies text into seven mutually exclusive categories, whereas clinical conditions are often comorbid. Displaying probabilities for all statuses partly addresses this, but the singular classification framework remains a limitation, as do the edge-case failures reported above. A further limitation concerns conversational context. Every item in the corpus is a single statement stripped of the thread it appeared in, and the deployed interface likewise classifies isolated text. A sentence such as “if that passes I’m going to throw myself off a bridge” is a disclosure of intent in one context and political hyperbole in another, and neither the training labels nor the classifier has access to the distinction. The system therefore cannot resolve context-dependent risk, a further argument for using it as a prioritization signal for human review rather than as a filter.

Three limitations concern the evaluation rather than the model. All figures come from a single training run on a single split, without cross-validation or repeated seeds. The bootstrap reported above quantifies the uncertainty arising from the finite test set, but it cannot capture the variance that retraining under a different seed would introduce, so the selection margin in particular should be read as provisional. The leakage audit removed contaminated rows from the test set but not from the training set, so duplication within training may still encourage memorization this study does not quantify. Finally, no concurrency or load testing was performed, so the latency figures describe single-request behavior on one processor.

# 6 Site Features

The web application is deployed as a two-tier system. The first tier is a static frontend hosted on GitHub Pages at https://jaukg9.github.io/mental-health-sentiment-analysis, while the second tier is a Python inference backend hosted on Hugging Face Spaces [4, 7]. The backend serves the quantized

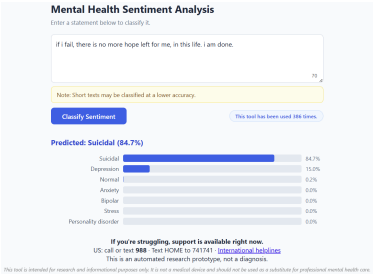


**Figure 5: Visualization of a High Risk Classification Using the Website.**

DistilBERT model of Section 5 through ONNX Runtime, requiring neither PyTorch nor Scikit-Learn at inference, keeping the deployed image small enough for the CPU-only tier. Every result is accompanied by a notice stating that the tool is a research prototype and not a diagnosis, and results identified as high risk additionally display crisis resources, as described in Section 7. Below are primary use cases that illustrate the platform's core functionality.

## 6.1 Self Reflection

The first use case supports individuals who wish to see how an automated classifier reads their own writing. A user is presented with a text-input module where they can submit unstructured personal logs, such as a journal entry, for example: "I'm confused, I'm not feeling good lately. Every time I want to sleep, I always feel restless." Upon submission, the application routes the text to the inference model. The interface then updates to display the predicted classification as a probability distribution over the seven categories, accompanied by a notice that it is a research prototype and not a diagnosis. As Section 7.5 sets out, the output is not intended as an assessment of the user's condition, and the self-reflection case is supported only in that weaker sense.

## 6.2 Digital Content Triage and Moderation

The application also functions as a triage tool for community moderators managing online social forums. A moderator who identifies a concerning post and needs a severity assessment inputs its raw text into the site, and if the model identifies severe emotional distress it outputs a high-risk categorization such as "Suicidal," letting the moderator prioritize the case and respond with emergency hotline numbers or platform-specific support protocols. An example of a high-risk classification can be visualized in Figure 5, which also shows the crisis resources the interface displays alongside such a result.

## 6.3 Clinical Pre-Screening

Finally, the platform can be used within a telehealth or clinical pre-appointment process. A patient may be prompted before a virtual session to write a brief summary of their current mental state, which the application processes into a preliminary risk category, allowing the provider to tailor their initial line of questioning. That category is a screening signal for the clinician, not a diagnosis, and is not shown to the patient as an assessment of their condition.

# 7 Ethics and Safety

A publicly deployed classifier that assigns mental health categories to text raises questions that performance metrics do not address. This section documents data provenance, the review status of the study, the asymmetry between kinds of error, the safeguards in the deployed application, and the uses for which the system is and is not intended.

## 7.1 Data Provenance and Consent

The corpus was not collected by the authors. It is a publicly released aggregation of nine previously published datasets, each scraped from public posts and forum comments by its original compiler, and no interaction with any participant took place at any stage of this work.

The people whose statements appear in the corpus wrote them for other readers, not for research. Public visibility is not consent to research use, and the individuals concerned were not asked and cannot practically be asked. Neither the compiler of the aggregated dataset nor those of its constituents document whether platform terms of service permitted automated collection and redistribution, or whether any opt-out was offered, so this study cannot verify the conditions under which its training data was obtained. No attempt was made to identify any author, link statements across posts, or recover metadata beyond the text and its label, none of which is present in the released dataset. The code, audit scripts, and edge-case suite supporting this study are available at https://github.com/JaukG9/mental-health-sentiment-analysis/.

## 7.2 Ethical Review

This study was not submitted for institutional review board approval. It involves no intervention, recruitment, or interaction with human participants, and uses only previously published public data, which ordinarily falls outside human-subjects review. That determination was made by the authors rather than a board, and the deployed application is a component a board might reasonably have examined, since it accepts text from the public.

The deployed application maintains a count of the number of classifications performed, which is not associated with any user. Submitted text is processed to produce a classification and is not retained or logged.

## 7.3 Failure Asymmetry

The errors this system can make are not equivalent: failing to flag a person at risk removes the possibility of a response, whereas flagging one who is not merely consumes reviewer attention. Overall accuracy treats the two as interchangeable, which is why per-class metrics are reported alongside it throughout Section 5. Of the 2,092 suicidal statements

in the filtered test set, the deployed model assigns 553 to depression and 41 to normal. The first group is misclassified but still receives an elevated-risk category and would be routed for review, but the second receives no flag at all, and represents 1.96% of suicidal statements. This is the failure mode that matters most, and the one the expected triage cost of Section 5.7 is weighted to penalize.

As Sections 5.4 and 5.7 establish, the deployed configuration is not the safest available: unquantized DistilBERT and quantized BERT each miss fewer suicidal statements, but neither satisfies the accuracy criterion of Section 4.3 at comparable cost. The deployed model was selected because it meets that criterion at less than half the inference cost, a trade defensible for a prototype on donated compute but one that must be acknowledged.

## 7.4 Deployment Safeguards

Every classification is presented with a notice stating that the tool is an automated research prototype and not a diagnosis, and a footer stating it is for research purposes, is not a medical device, and is not a substitute for professional care. High-risk classifications additionally display crisis resources beneath the result: the 988 Suicide and Crisis Lifeline, the Crisis Text Line (text HOME to 741741), and a link to international helpline directories, as shown in Figure 5.

Crisis resources are displayed whenever the model assigns a probability of at least 0.25 to the Suicidal class, rather than only when Suicidal is the top-ranked label. Because recall on that class is 0.7103, a rule keyed to the top-ranked label alone would leave roughly 29% of suicidal statements without crisis information. The probability threshold extends coverage to cases ranked as depression that still carry substantial suicidal probability.

The threshold narrows the gap without closing it. Of the three probes in the edge-case suite of Section 5 whose intended label was Suicidal, one clears 0.25: the minimized disclosure, at 0.305, would display crisis resources despite being ranked as depression. The other two do not, having been assigned suicidal probabilities below 0.01. Indirect disclosure expressed sarcastically or with temporal hedging produces confident normal predictions rather than uncertain ones, and no threshold on a probability the model does not assign can recover those cases. This is the clearest limitation of the deployed system, and argues for human review of flagged and unflagged text alike rather than reliance on the classifier as a filter.

## 7.5 Intended Use and Human Oversight

The system is intended to prioritize human attention, not replace it. Its outputs are screening signals indicating which texts a moderator, clinician, or support worker should read first, and every workflow in Section 6 places a person between the classification and any action affecting the individual concerned. It produces no diagnosis, and its categories are inherited from the collection conditions of the training corpus rather than from clinical assessment.

Several uses are outside what this work supports. The system should not inform consequential decisions about a person, including those concerning care not voluntarily sought, employment, education, or insurance; should not screen individuals without their knowledge; and its outputs should not be presented to a person as an assessment of their condition. It has not been validated against clinical outcomes, has not been evaluated beyond the English-language social media users in its training data, and its probabilities are systematically overconfident.

Finally, a tool that identifies distress at low cost can be turned toward ends its authors did not intend, including identifying vulnerable users for exploitation rather than support. It is released as a research artifact with that risk acknowledged, and the reporting here is intended in part to make its error characteristics legible to anyone evaluating it for deployment.

# 8 Conclusions

This study developed an automated natural language processing framework to detect and categorize mental health distress from unstructured text, treating deployment as an optimization problem over four axes: the classification head, the encoder, the inference runtime, and numeric precision. A BERT model fine-tuned on 42,144 social media samples served as the reference, and the deployed system classifies inputs into seven psychological states at 82.31% accuracy and 0.7876 macro-F1 on a leakage-filtered hold-out set.

Two axes yielded no benefit. Replacing the classification head with a Random Forest left accuracy unchanged while adding inference cost, since every input must still traverse the full encoder, and skipping fine-tuning cost 18.24 accuracy points. Replacing the encoder and quantizing it cut per-request latency from 186.4 ms to 43.8 ms and the deployable artifact from 438.7 MB to 68.3 MB, a 4.3× and 6.4× improvement over the fp32 BERT reference, while retaining 99.02% of the reference accuracy and 99.60% of its Suicidal recall. Distillation and quantization contribute comparably to that speedup, the distilled encoder halving latency and quantization halving it again (2.02×), while the runtime change alone accounts for only 5-9%.

Overall, this study demonstrates that a mental health triage system can be lightweight and affordable without fixating on a single accuracy metric. By prioritizing measurable efficiency, this framework produces a fast, deployable model, still with transparent strengths and limitations.

# References

[1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *CoRR* abs/1810.04805 (2018). arXiv:1810.04805 http://arxiv.org/abs/1810.04805

[2] Fatemeh Rajab Dizavandi, Razieh Froutan, Hossein Karimi Moonaghi, Abbas Ebadi, and Mohammad Reza. 2023. Mental Health Triage from the Viewpoint of Psychiatric Emergency Department Nurses; a Qualitative Study. *PubMed* 11 (01 2023), e70–e70. https://doi.org/10.22037/aaem.v11i1.2080

[3] Aparna Elangovan, Jiayuan He, and Karin Verspoor. 2021. Memorization vs. Generalization: Quantifying Data Leakage in NLP Performance Evaluation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*. Association for Computational Linguistics, 1325–1335. https://aclanthology.org/2021.eacl-main.113/

[4] GitHub. 2024. GitHub Pages Documentation. https://docs.github.com/en/pages.

[5] Sara Hooker, Aaron Courville, Gregory Clark, Yann Dauphin, and Andrea Frome. 2019. What Do Compressed Deep Neural Networks Forget? *arXiv preprint arXiv:1911.05248* (2019). https://arxiv.org/abs/1911.05248

[6] Sara Hooker, Nyalleng Moorosi, Gregory Clark, Samy Bengio, and Emily Denton. 2020. Characterising Bias in Compressed Models. *arXiv preprint arXiv:2010.03058* (2020). https://arxiv.org/abs/2010.03058

[7] Hugging Face. 2024. Hugging Face Spaces Documentation. https://huggingface.co/docs/hub/spaces-overview.

[8] Erkki Isometsä. 2014. Suicidal Behaviour in Mood Disorders—Who, When, and Why? *The Canadian Journal of Psychiatry* 59 (03 2014), 120–130. https://doi.org/10.1177/070674371405900303

[9] Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. 2018. Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2704–2713.

[10] Thomas Kallstenius, Andrea Johansson Capusan, Gerhard Andersson, and Adam Williamson. 2025. Comparing traditional natural language processing and large language models for mental health status classification: a multi-model evaluation. *Scientific Reports* 15, 1 (2025), 24102. https://doi.org/10.1038/s41598-025-08031-0

[11] Christopher D Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press.

[12] Yida Mu, Mali Jin, Xingyi Song, and Nikolaos Aletras. 2024. Enhancing Data Quality through Simple De-duplication: Navigating Responsible Computational Social Science Research. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Miami, Florida, USA, 12477–12492. https://aclanthology.org/2024.emnlp-main.694/

[13] John A. Naslund, Ameya Bondre, John Torous, and Kelly A. Aschbrenner. 2020. Social Media and Mental Health: Benefits, Risks, and Opportunities for Research and Practice. *Journal of Technology in Behavioral Science* 5 (04 2020), 245–257. https://doi.org/10.1007/s41347-020-00134-x

[14] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830. https://jmlr.org/papers/v12/pedregosa11a.html

[15] Vijay Janapa Reddi, Christine Cheng, David Kanter, Peter Mattson, Guenther Schmuelling, Carole-Jean Wu, Brian Anderson, Maximilien Breughe, Mark Charlebois, William Chou, et al. 2020. MLPerf Inference Benchmark. In *ACM/IEEE 47th Annual International Symposium on Computer Architecture (ISCA)*. 446–459. https://doi.org/10.1109/ISCA45697.2020.00045

[16] Abdul Rehman, Muzamil Ahmed, Hikmat Ullah Khan, Amal Bukhari, Ali Daud, and Hussain Dawood. 2025. Mental Health Sentiment Analysis: Exploring an Optimized BERT with Deep Encodings. *Engineering, Technology & Applied Science Research* 15, 5 (2025), 26242–26248. https://doi.org/10.48084/etasr.10469

[17] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108* (2019). https://arxiv.org/abs/1910.01108

[18] Suchintika Sarkar. 2024. Sentiment Analysis for Mental Health. https://www.kaggle.com/datasets/suchintikasarkar/sentiment-analysis-for-mental-health/data

[19] Samanvith Thotapalli, Musa Yilanli, Ian McKay, William Leever, Eric Youngstrom, Karah Harvey-Nuckles, Kimberly Lowder, Steffanie Schweitzer, Erin Sunderland, Daniel I Jackson, and Emre Sezgin. 2025. Potential of ChatGPT in youth mental health emergency triage: Comparative analysis with clinicians. *PubMed* 4 (09 2025), e70159–e70159. https://doi.org/10.1002/pcn5.70159

[20] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, Qun Liu and David Schlangen (Eds.). Association for Computational Linguistics, Online, 38–45. https://doi.org/10.18653/v1/2020.emnlp-demos.6

[21] Ofir Zafrir, Guy Boudoukh, Peter Izsak, and Moshe Wasserblat. 2019. Q8BERT: Quantized 8Bit BERT. In *Fifth Workshop on Energy Efficient Machine Learning and Cognitive Computing (EMC2), NeurIPS Edition*. https://arxiv.org/abs/1910.06188

# A Externally Authored Edge-Case Suite

This appendix lists the complete edge-case suite discussed in Section 5, together with a sample of ordinary hold-out items for comparison. The suite consists of fifty probes, ten for each of five linguistic phenomena, written by collaborators not involved in developing the system and frozen before any item was run. Probabilities are the softmax outputs of the deployed quantized model, computed one request at a time to match the serving configuration of Section 4.4. For each probe, P(int.) is the probability the model assigns to the label a human reader intends and P(pred.) the probability it assigns to the label it actually returns. The two coincide wherever the model is correct. The suite and the code that produced these results are available at https://github.com/JaukG9/mental-health-sentiment-analysis/.

**Table 5: Full prediction probabilities on a sample of hold-out test items. The Suicidal item, classified as Depression, is the only misclassification in the sample and reflects the Depression/Suicidal confusion discussed in Section 5.**

| Input Text | Actual | Predicted | Prediction Probabilities | |
|---|---|---|---|---|
| If you really have to go to the ICU, it's okay, the important thing is to get better quickly and be healthy again | Normal | Normal | Normal | 97.4% |
| | | | Anxiety | 1.9% |
| | | | Stress | 0.3% |
| | | | Bipolar | 0.1% |
| | | | Depression | 0.1% |
| | | | Suicidal | 0.1% |
| | | | Personality disorder | 0.0% |
| Housing crisis Global warming Lack of employment Coronavirus No future Despair Hopeless | Suicidal | Depression | Depression | 90.3% |
| | | | Suicidal | 8.2% |
| | | | Normal | 1.1% |
| | | | Anxiety | 0.2% |
| | | | Stress | 0.1% |
| | | | Personality disorder | 0.0% |
| | | | Bipolar | 0.0% |
| Medication I haven't taken my medication in months because I lacked health insurance and could not find someone to sign off on prescriptions. I have insurance now and can afford my medication. Has any of you started medication after being off it for some time? What was it like? Should I be worried I'll have an episode? | Bipolar | Bipolar | Bipolar | 89.9% |
| | | | Anxiety | 9.0% |
| | | | Depression | 0.6% |
| | | | Stress | 0.2% |
| | | | Normal | 0.1% |
| | | | Personality disorder | 0.1% |
| | | | Suicidal | 0.1% |
| If I worry about people, my chest always hurts. Please don't worry :( | Anxiety | Anxiety | Anxiety | 98.8% |
| | | | Normal | 0.4% |
| | | | Stress | 0.4% |
| | | | Depression | 0.2% |
| | | | Bipolar | 0.1% |
| | | | Suicidal | 0.1% |
| | | | Personality disorder | 0.0% |

**Table 6: Externally authored edge-case suite, probes 1-25, with the deployed model's predictions. A tick marks agreement with the label a human reader assigns. Where the model is wrong, the gap between P(int.) and P(pred.) shows how confidently it prefers the wrong class. For probes whose intended label is Suicidal, P(int.) is the quantity governing the crisis-resource display described in Section 7.**

| Phenomenon | | Probe | Intended | P(int.) | Predicted | P(pred.) |
|---|---|---|---|---|---|---|
| Hyperbole | ✓ | I'm so hungry I could eat a horse right now. | Normal | 99.9% | Normal | 99.9% |
| | × | This deadline is actually going to kill me, I've got way too much on my plate this week. | Stress | 12.7% | Normal | 68.5% |
| | ✓ | I laughed so hard I nearly died, that video was hilarious. | Normal | 99.3% | Normal | 99.3% |
| | ✓ | My heart is pounding out of my chest, I'm so nervous about this exam I could throw up. | Anxiety | 99.2% | Anxiety | 99.2% |
| | ✓ | I feel like the weight of the entire world is crushing me, I can't shake this sadness no matter what I do. | Depression | 94.4% | Depression | 94.4% |
| | × | I have a thousand things to do and only twenty-four hours in the day, I'm losing it. | Stress | 13.6% | Depression | 38.2% |
| | ✓ | That concert was insane, best night of my entire life, I'm still buzzing. | Normal | 98.1% | Normal | 98.1% |
| | ✓ | I'm so anxious about the flight tomorrow I feel like I'm going to explode. | Anxiety | 99.3% | Anxiety | 99.3% |
| | ✓ | I feel like I'm carrying around a two hundred pound weight of sadness that never goes away no matter how much I try to shake it off. | Depression | 96.7% | Depression | 96.7% |
| | ✓ | I have so much on my plate right now I feel like I'm about to combust, this week has been absolute chaos. | Stress | 56.3% | Stress | 56.3% |
| Idiom | ✓ | I killed two birds with one stone today, finished my report and picked up groceries. | Normal | 99.0% | Normal | 99.0% |
| | ✓ | That joke had me dying, so funny. | Normal | 99.8% | Normal | 99.8% |
| | × | I'm swamped at work this week, drowning in emails and back-to-back meetings. | Stress | 0.4% | Normal | 98.7% |
| | × | My stomach's in knots waiting for the test results to come back. | Anxiety | 0.1% | Normal | 99.8% |
| | × | I've been feeling down in the dumps for weeks now, nothing seems to lift my mood. | Depression | 0.2% | Normal | 98.7% |
| | ✓ | Break a leg tonight, you're going to do great. | Normal | 99.8% | Normal | 99.8% |
| | × | I'm running on fumes trying to juggle work and the kids this week. | Stress | 0.1% | Normal | 99.8% |
| | ✓ | I've been walking on eggshells all week worried about the layoffs at work. | Anxiety | 98.6% | Anxiety | 98.6% |
| | × | One week I'm on top of the world starting five new projects and the next I can't even get off the couch, it's like living with two different people in my head. | Bipolar | 13.5% | Normal | 65.5% |
| | × | I burn bridges the second I feel someone pulling away, better to torch it myself than wait around to get left, that's just how I'm wired I guess. | Personality disorder | 0.5% | Normal | 89.4% |
| Sarcasm | × | Oh great, another all-nighter, exactly what I needed with three deadlines this week. | Stress | 0.1% | Normal | 99.6% |
| | ✓ | Wonderful, it's raining again the one day I decided to walk to work. | Normal | 99.9% | Normal | 99.9% |
| | × | Fantastic, my hands are shaking again right before the presentation, love that for me. | Anxiety | 0.1% | Normal | 99.8% |
| | × | Great, another day where I can't find the motivation to get out of bed, really thriving over here. | Depression | 0.2% | Normal | 98.9% |
| | ✓ | Perfect, spilled coffee all over my shirt right before the meeting, off to a great start. | Normal | 99.8% | Normal | 99.8% |

**Table 7: Externally authored edge-case suite, probes 26-50.**

| Phenomenon | | Probe | Intended | P(int.) | Predicted | P(pred.) |
|---|---|---|---|---|---|---|
| Sarcasm | × | Love how my inbox somehow doubled overnight, definitely going to be a relaxing week. | Stress | 0.0% | Normal | 99.7% |
| | × | Oh good, my mind is racing through worst case scenarios again at 2am, exactly what I needed. | Anxiety | 0.1% | Normal | 99.6% |
| | × | Sure, I'm 'doing great,' that's why I've cancelled plans with everyone three weekends in a row. | Depression | 0.0% | Normal | 99.8% |
| | × | Oh I'm SO fine, definitely not lying awake calculating how much easier it would be for everyone if I just wasn't here anymore, nothing to see here. | Suicidal | 0.9% | Normal | 91.9% |
| | ✓ | Great, another 'friend' who disappears the second things get hard, must be something about me right? Awesome pattern I've got going, really love collecting people just to lose them. | Personality disorder | 49.6% | Personality disorder | 49.6% |
| Temporal blindness | × | I used to struggle with anxiety attacks in college, but I haven't had one in years and feel completely fine now. | Normal | 6.8% | Anxiety | 86.8% |
| | × | I went through a rough depressive episode after my divorce, but that was five years ago and I'm honestly in a really good place now. | Normal | 9.4% | Depression | 46.3% |
| | ✓ | I've been anxious about this move for weeks now and it's still eating at me every single day. | Anxiety | 98.7% | Anxiety | 98.7% |
| | × | Ever since the new project started last month, I've been stressed out non-stop and it hasn't let up. | Stress | 17.3% | Normal | 81.1% |
| | × | I've felt this heavy sadness since the breakup two months ago and it still hasn't gone away. | Depression | 1.9% | Normal | 68.2% |
| | × | I had panic attacks daily back in high school, but that stopped once I graduated and I've been panic-free for almost a decade. | Normal | 24.5% | Anxiety | 41.7% |
| | ✓ | I had a manic episode last year and I'm still cycling between highs and crashes most months. | Bipolar | 97.8% | Bipolar | 97.8% |
| | ✓ | I used to have really bad social anxiety, but after a few years of practice I actually enjoy parties now. | Normal | 98.9% | Normal | 98.9% |
| | × | I told my therapist I 'used to' think about not waking up, but honestly some nights it's still there, I just don't say it out loud anymore. | Suicidal | 0.1% | Normal | 96.8% |
| | × | I used to tell people my relationships 'just didn't work out,' but looking back, every single one ended the same way, me either clinging so hard they ran or pushing them away before they could leave first, and honestly that hasn't really changed. | Personality disorder | 9.2% | Normal | 55.1% |
| Minimized distress | × | I've been swamped with work lately, staying late most nights and barely keeping up, but I think I just need a lighter week. | Stress | 3.3% | Normal | 95.9% |
| | × | I'm okay I guess, just haven't felt like myself in weeks and everything feels kind of pointless. | Depression | 28.9% | Anxiety | 39.1% |
| | ✓ | It's nothing, I just get really panicky before flights, my heart races and I can't sit still. | Anxiety | 89.6% | Anxiety | 89.6% |
| | × | No big deal, just been crying more than usual and struggling to get out of bed most mornings. | Depression | 1.3% | Normal | 96.7% |
| | ✓ | It's fine honestly, just thinking too much right now and feeling constantly worried about stuff. | Anxiety | 98.9% | Anxiety | 98.9% |
| | ✓ | Not a big thing, I just can't stop worrying about everything lately, my mind won't slow down. | Anxiety | 98.8% | Anxiety | 98.8% |
| | ✓ | It's nothing serious, I just go through these weeks where I barely sleep and work way too much, then crash hard after. | Stress | 71.7% | Stress | 71.7% |
| | ✓ | I'm managing, just feel numb most days and nothing really excites me like it used to. | Depression | 57.5% | Depression | 57.5% |
| | × | It's nothing really, I've just been having thoughts about wanting to end my life, but don't make a big deal out of it, I'll be fine. | Suicidal | 30.5% | Depression | 46.6% |
| | ✓ | I know this probably isn't a big deal compared to what others are going through, and I hate to take up anyone's time, but I've been feeling pretty overwhelmed lately. | Stress | 78.3% | Stress | 78.3% |

# The Effects of Active and Passive Learning Methods

Jimin Lee
Branksome Hall Asia
Jeju, South Korea
jimin2026lee@gmail.com

## Abstract

While active learning is widely recognized as the more instructional approach for improving student engagement and knowledge retention, traditional passive learning methods remain prevalent in modern educational settings. The study examined the effects of active and passive training methods on immediate comprehension and application of new knowledge, evaluating whether study duration influenced learning outcomes. Using a 2-by-2 factorial design, participants were assigned to either active or passive training conditions and to a 10- or 20-minute time constraint prior to completing a post-test. The analysis of the post-test outcomes showed that those in active training conditions achieved significantly higher scores than those in passive training conditions, with a statistically significant effect of the training method (p=0.007). In contrast, study duration showed no significant effect on performance. These findings suggest the importance of learners' active engagement with learning materials for immediate comprehension and application of new material, supporting how active training methods play a critical role in educational environments.

## Keywords

Active vs. Passive Training, Educational Research, Comprehension and Application, Study Duration

**ACM Reference Format:**
Jimin Lee. 2026. The Effects of Active and Passive Learning Methods. In *Proceedings of International Journal of Secondary Computing and Applications Research (IJSCAR VOL. 3, ISSUE 3).* ACM, New York, NY, USA, 5 pages. https://doi.org/10.67149/yhjs2024.5/76v9axa9

## 1 Introduction

Educational systems have undergone significant changes over the past decades to foster improvement in instructional practices and learning outcomes (Yue, 2024). Despite these past advancements and developments of new training methods, determining the most effective method for immediate comprehension and application of new knowledge remains an important challenge in the field of education. As the learning environment becomes increasingly diverse and complex, and as new learning methods emerge in contemporary educational settings, educators continue to seek instructional approaches that most effectively encourage student engagement and enhance academic performance.

Among the most widely discussed approaches in education, two of the prominent methods include active and passive learning (Bonwell & Eison, 1991). Passive learning, more traditionally characterized by lectures and instructor-led explanation, positions students as recipients of information; in contrast, active learning is often characterized by the learners' direct engagement in the learning process in forms of discussions, questioning, problem-solving, and self-assessments. As such, active learning encourages proactive construction of knowledge and understanding by applying concepts during the learning process (Chi & Wylie, 2014).

Recent educational research has suggested that active training methods may lead to improved student outcomes (Freeman et al., 2014). Despite the benefits, lecture-based passive training methods continue to play a significant role in many educational settings, primarily for their efficiency in delivering masses of information to large numbers of people (Bligh, 2000), as well as their familiarity to both instructors and learners alike (Burgan, 2006). This raises important questions about the comparative effectiveness of such approaches and the settings in which each method can be deemed beneficial.

This study aims to investigate the effects of active and passive training methods on students' immediate comprehension and application of new material. Specifically, participants were either administered an active training method in which they answered comprehension questions throughout the learning process, or a passive training method in which they studied the same material without interruption. Additionally, the study examined whether the amount of study time influenced learning outcomes by comparing participants who were given either 10 or 20 minutes to study the material. The study measured the effectiveness of both training methods with a post-test, measuring the participants' ability to understand and apply the new knowledge.

Understanding the strengths and limitations of active and passive training methods is essential for educators and curriculum designers who aim to enhance the quality of education. As demands of modern education continue to evolve, identifying evidence-based teaching practices will be critical in fostering meaningful and lasting learning experiences.


*IJSCAR VOL. 3, ISSUE 3*

## 2 Literature Review

The effectiveness of active training methods compared to traditional passive training methods has been a major focus of educational research over the past several decades. Research has increasingly suggested that active training promotes deeper understanding and stronger retention of knowledge by encouraging students to further construct knowledge proactively rather than simply receiving information (Lovett et al., 2023).

A previous study conducted by Scott Freeman et al. (2014) examined the effectiveness of active learning methods by performing a meta-analysis of 225 studies comparing active learning with traditional lecturing in science, technology, engineering, and mathematics (STEM) education. The study showed that students in active learning environments achieved significantly higher examination scores. It also showed that students in lecture-based courses experienced substantially higher failure rates. These findings provided strong empirical evidence that active learning consistently produces higher academic outcomes in a range of academic contexts.

Michelene T.H. Chi and Ruth Wylie (2014) also proposed the Interactive, Constructive, Active, Passive (ICAP) Framework, which explained how different forms of cognitive engagement produce different levels of learning outcomes. The proposed framework suggests that the effectiveness of learning increases as students move away from passive activities, such as listening to lectures, and move toward active behaviors that contribute to generating new ideas or explanations. Chi and Wylie argued that higher levels of engagement promote deeper cognitive understanding and construction of knowledge, such that the training method results in improved understanding and information retention. The ICAP framework provides a strong theoretical basis for the effectiveness of active training methods by demonstrating the critical role of cognitive engagement in maximizing learning outcomes.

In a study by Douglas B. Markant et al. (2016), active learning was found to enhance retention and memory by increasing engagement with information and encouraging self-directed exploration of the study material. The researchers argued that active participation strengthens the processes by which learners obtain and encode new information into memory and therefore improves long-term retention. Learners who actively interact with the material are more likely to develop meaningful cognitive connections that aid in later recall in comparison to learners who passively receive information.

A study conducted by Louis Deslauriers et al. (2019) found that students in active learning environments demonstrated significantly greater learning gains than students in traditional, passive settings despite reporting lower feelings of learning. Authors of the study suggested that while passive learning may accompany greater self-reported feelings of learning, active learning introduces productive cognitive effort referred to as "desirable difficulties" that make learning feel more challenging while ultimately improving understanding. These findings thereby highlight an important distinction between perceived ease and actual effectiveness in learning.

Thus, existing literature consistently indicates how active training methods outperform passive training approaches in knowledge retention and academic achievement by encouraging meaningful engagement with the material. The present study aims to build upon the existing body of research by examining whether active training continues to provide meaningful advantages over passive training under different study time conditions.

## 3 Method

### 3.1 Participants

Participants were recruited through Prolific, an online research platform. Respondents participating in the experiment were directed to a custom-built survey website that administered the study.

Participants provided informed consent and were compensated through Prolific at the platform's standard rate. All 85 participants who completed the study were included in the research.

### 3.2 Technology

The study materials and the training methods were administered through a web application using the Python Flask framework and a client-server architecture. Participants were able to access the application through any modern web browser and a variety of compatible devices. The implementation enabled features such as automated random assignment, standardized delivery of treatment conditions, and efficient data collection for subsequent analysis.

### 3.3 Research Design

The study employed a 2-by-2 factorial experimental design examining the effects of training method and study duration on knowledge comprehension and application (using the knowledge obtained in challenging exercises). The two independent variables were the administered training methods (active training vs. passive training) and time allocation (10 minutes vs. 20 minutes), resulting in four treatment conditions:

- Condition A: Active training method with a 10 minute time limit
- Condition B: Active training method with a 20 minute time limit
- Condition C: Passive training method with a 10 minute time limit
- Condition D: Passive training method with a 20 minute time limit

Participants were randomly assigned to one of the four conditions upon entering the study. Randomization was handled automatically through the survey platform.

### 3.4 Study Material

The study material consisted of an informational article published by Red Hat discussing the concepts of generative artificial intelligence and agentic artificial intelligence. The article was selected based on its moderate technical complexity. This enabled participants to be presented with unfamiliar material that they may encounter in the media. The topic was considered suitable for evaluating training methods in immediate comprehension and application of knowledge. It required participants to process and understand new conceptual material and apply what they have learned in a practical exercise.

### 3.5 Respondent Demographics

Random assignment resulted in 20, 21, 20, and 24 respondents allocated to conditions A, B, C, and D, respectively. The participants' ages ranged from 18 to 72, with a mean age of 36.7 and a median age of 36. The sample was predominantly male (63.5%), with 34.1% identifying as female and 2.4% as non-binary. Participants also self-reported their level of technology proficiency, with 39 (45.9%) identifying as advanced, 35 (41.2%) identifying as intermediate, and 11 (12.9%) identifying as basic. Daily internet usage was also collected, given its potential relevance to the study material, with 58.8% of participants spending between 5 and 10 hours online per day, while 20.0% reported more than 10 hours online per day.

A pre-test was not conducted to preserve the novelty of the study material. Therefore, baseline equivalence rests on random assignment, and demographic features were similar across the four conditions (mean ages of 38.0, 34.5, 37.2, and 37.2 years for Conditions A–D).

### 3.6 Procedure

After being assigned to their respective conditions, participants were instructed to study the article within their allocated time limits of either 10 or 20 minutes. A visible countdown timer remained on screen throughout the study period.

Participants in the passive training conditions (Conditions C and D) were instructed to read the article without interruption or interaction. In contrast, participants in the active training conditions (Conditions A and B) were instructed to read the article in sections. Following each section, participants completed short multiple-choice comprehension questions designed to reinforce understanding and encourage active engagement with the material.

After the allocated study period expired, participants were directed to a 10-question test administered uniformly across the four treatment conditions. The final questionnaire was designed to assess critical thinking and the application of knowledge acquired from the article. The areas in which the respondents were tested included application in real-life situations, comparison in context, and evaluation to extend upon simple factual recall. Participants' final percentage scores on the post-test were recorded as the primary outcome measure for subsequent analysis.

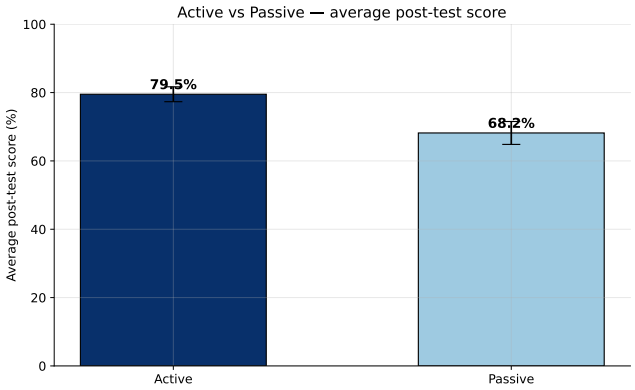


**Figure 1: Average post-test score by training method. Bars show the group mean percentage score for the Active (n = 41) and Passive (n = 44) training conditions, collapsed across study duration; error bars denote ±1 standard error of the mean. The active group averaged 79.5%, versus 68.2% for the passive group, an 11.3 percentage-point difference.**

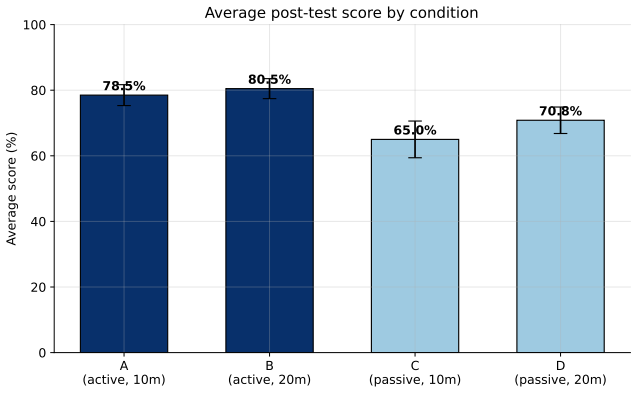


**Figure 2: Average post-test score by treatment condition. Bars show the group mean percentage score for each of the four conditions (A: active, 10 min, n = 20; B: active, 20 min, n = 21; C: passive, 10 min, n = 20; D: passive, 20 min, n = 24); error bars denote ±1 standard error of the mean. Both active conditions outscored their passive, same-duration counterparts: A exceeded C by 13.5 percentage points, and B exceeded D by 9.7 percentage points.**

## 4 Results

Figure 1 shows the average test score by training methods, with the active and passive training method groups scoring an average score of 79.5% and 68.2%, respectively. The difference of 11.3 percentage points suggests better comprehension and application of new knowledge for the active training group.

Figure 2 shows the performance across the four treatment conditions. Active training groups in both 10- and 20-minute

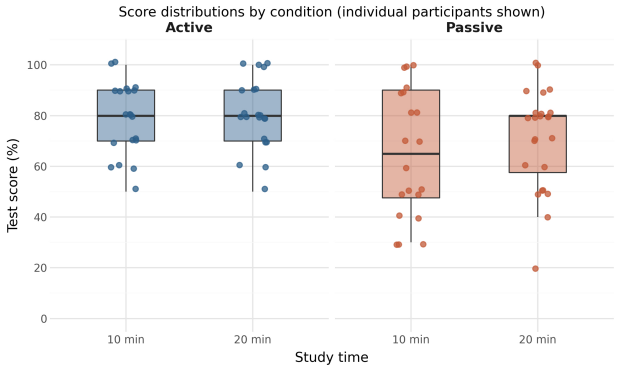


**Figure 3: Score distributions by condition and study duration, with individual participants shown. Box plots give the median and interquartile range (IQR) of post-test scores (%) for the Active and Passive training methods at each study duration (10 min, 20 min); whiskers extend to the most extreme values within 1.5×IQR, and each point is one participant's score. Passive-condition scores show visibly greater spread, including more low-scoring outliers, than active-condition scores at both durations.**

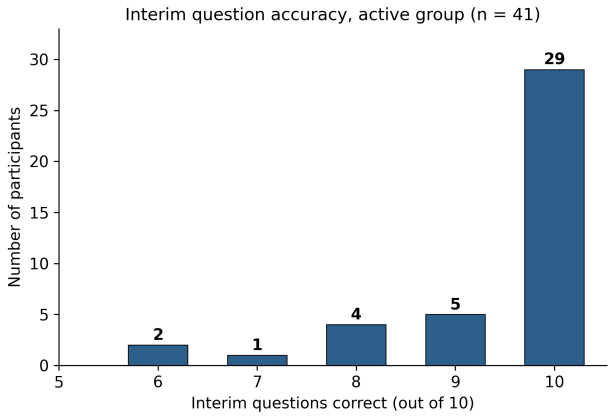


**Figure 4: Distribution of interim comprehension-question accuracy for the active training group (n = 41). Bars show the number of participants who answered each possible count (6 through 10) of the ten embedded interim questions correctly; no participant in this group scored below 6 of 10. Most participants (29 of 41) answered all ten interim questions correctly.**

**Table 1: Analysis of Variance**

| Source of Variation | df | SS | F | p |
|---|---|---|---|---|
| Training Method (Active vs. Passive) | 1 | 27.25 | 7.71 | .007 |
| Training Time (10 min vs. 20 min) | 1 | 3.33 | 0.94 | .335 |
| Training Method × Study Duration | 1 | 0.79 | 0.22 | .639 |
| Residual | 81 | 286.34 | — | — |

Note. ANOVA was conducted on raw test scores (0–10 scale).

time constraints scored consistently higher than their passive counterparts, with a 13.5% difference between group A and C and a 9.7% difference between group B and D.

Box plots in Figure 3 show distributions of participant post-test scores (%) across treatment conditions. Compared to passive training participants, active training participants showed less variability in test performance. This suggests that the use of active training methods not only raised average performance but also produced more consistent results.

Figure 4 shows the performance of the active training group in the interim quiz, with the majority of participants (29 out of 41) answering all 10 questions correctly. Very few participants received a score below 8 out of 10, suggesting that the participants actively interacted with the material and completed the task as intended, thereby indicating a high level of engagement.

A two-way Analysis of Variance (ANOVA) was conducted to examine the effects of training method (active vs. passive) and training time (10 minutes vs. 20 minutes) on participants' performance on the questionnaire. The training method effect corresponded to partial $\eta^2 = .087$ (Cohen's d = 0.61), a medium-sized effect. Effect sizes for duration and the interaction were small (partial $\eta^2 = .011$ and .003, respectively). The analysis revealed a statistically significant main effect of the training method ($F = 7.71$, $p = 0.007$), indicating that participants who engaged in active training achieved significantly higher test scores than those who participated in passive training. This finding is consistent with the descriptive results presented in Figures 1 and 2. In contrast, the main effect of training time was non-significant ($F = 0.94$, $p = 0.335$).

Additionally, the interaction between training method and training time was non-significant ($F = 0.22$, $p = 0.639$). The result indicates that the advantage of active training remained relatively consistent and stable regardless of whether participants were given 10 or 20 minutes to study the material. Considering these results, the experiment suggests that the method of training had a greater influence on post-test outcomes than the amount of study time.

After Levene's test, there was a meaningful difference in variances across conditions, reflecting the greater variability in the passive conditions visible in Figure 3. A Welch's t-test for the method contrast, which does not assume equal variances, confirmed the effect ($t(73.0) = 2.83$, $p = .006$).

## 5 Conclusion

The study examined the effect of active and passive training methods in the acquisition and application of new knowledge and whether these advantages are influenced by the amount of time available for study. Participants in the active training conditions achieved higher average scores (79.5%) than those in passive training conditions (68.2%), with confirmation of a statistically significant effect of training method via ANOVA ($p = 0.007$). In contrast, the study duration was shown not to have a significant effect on performance.

The findings of the study suggest that the methods by which learners engage with the study material can be more effective than the amount of time spent on studying it; as active training strategies require learners to process and apply information, it is able to promote stronger understanding and application than passive exposure alone. The results are also consistent with the findings of previous research that identified active training methods as an effective instructional approach in various educational settings relative to passive training methods.

The study further proposes a practical implication for educators and instructional designers, highlighting the importance of incorporating proactive training methods into learning, especially when instructional time is limited. While increasing the study time is often assumed to improve learning outcomes significantly, the findings of the study suggest that the quality of learner engagement may be a more critical factor than the duration of study. However, since the study compared only two durations using a single article, the absence of a duration effect may also indicate that 10 minutes was already sufficient to process the particular text used in the study; the present data cannot distinguish between these explanations. Hence, incorporating opportunities for learners to reflect on or test their understanding of the material in between may yield greater benefits than simply extending exposure to content.

It is also important to note that the active manipulation displayed in this study is a form of interpolated retrieval practice rather than active learning in general, as interpolated retrieval practice includes answering comprehension questions during reading, whereas active learning in general includes discussions or problem-solving. Although the active participants had less raw reading time due to the interim questions taking up a part of the fixed study window, the active participants still showed outperformance compared to the passive participants.

Within the ICAP framework (Chi & Wylie, 2014), answering interim multiple-choice questions is categorized under the active tier of engagement, rather than the constructive or interactive tier. Since a meaningful advantage was observed at said modest tier, it is suggested that higher tiers of engagement, such as self-explanation or peer discussion, may yield greater benefits.

Several limitations of the study can additionally be acknowledged. The prominent methodological limitations of the study include limitations in the scope of the study material, as well as applicability for longer retention durations. The study was also conducted in an online environment, which may enable more proactive training via person-to-person interaction that affects the effectiveness of the training method. These factors may limit the generalizability of the findings in other educational contexts not explored in the study.

Further research could investigate whether the advantages observed in this study can persist over a longer period of time across different subject areas. As the post-test was conducted immediately after the participants' study, the present results do not represent long-term retention, but rather immediate comprehension. The testing-effect literature predicts the active-learning advantage to grow with delay (Roediger & Butler, 2011). It can also be valuable to compare specific forms of active learning, such as peer discussion, direct question-and-answer sessions, and interactive exercises, to determine which approaches are most effective for different types of content and learners.

Overall, the study provides further evidence that active training methods are an effective approach for promoting understanding and application of knowledge in the learning process. As the educational environment continues to adapt to increasingly complex and dynamic subject matter, modifying strategies to encourage active engagement with the material at hand may play an important role in improving learning outcomes.

## References

[1] D. A. Bligh. 2000. *What's the Use of Lectures?* Jossey-Bass.

[2] C. C. Bonwell and J. A. Eison. 1991. *Active Learning: Creating Excitement in the Classroom.* ASHE-ERIC Higher Education Report No. 1. George Washington University.

[3] M. Burgan. 2006. In defense of lecturing. *Change: The Magazine of Higher Learning* 38, 6, 30–34. https://doi.org/10.3200/CHNG.38.6.30-34

[4] Michelene T. H. Chi and Ruth Wylie. 2014. The ICAP Framework: Linking Cognitive Engagement to Active Learning Outcomes. *Educational Psychologist* 49, 4, 219–243. https://doi.org/10.1080/00461520.2014.965823

[5] Louis Deslauriers et al. 2019. Measuring actual learning versus feeling of learning in response to being actively engaged in the classroom. *Proceedings of the National Academy of Sciences* 116, 39, 19251–19257. https://doi.org/10.1073/pnas.1821936116

[6] Scott Freeman et al. 2014. Active learning increases student performance in science, engineering, and mathematics. *Proceedings of the National Academy of Sciences* 111, 23, 8410–8415. https://doi.org/10.1073/pnas.1319030111

[7] Marsha C. Lovett, Michael W. Bridges, Michele DiPietro, Susan A. Ambrose, and Marie K. Norman. 2023. *How Learning Works: Eight Research-Based Principles for Smart Teaching*, 2nd ed. Jossey-Bass, San Francisco, CA.

[8] Douglas B. Markant et al. 2016. Enhanced memory as a common effect of active learning. *Mind, Brain, and Education* 10, 3, 142–152. https://doi.org/10.1111/mbe.12117

[9] H. L. Roediger and A. C. Butler. 2011. The critical role of retrieval practice in long-term retention. *Trends in Cognitive Sciences* 15, 1, 20–27. https://doi.org/10.1016/j.tics.2010.09.003

[10] S. Yue. 2024. The evolution of pedagogical theory: From traditional to modern approaches and their impact on student engagement and success. *Journal of Education and Educational Research* 7, 3, 226–230. https://doi.org/10.54097/j4agx439

# A Five-Run Comparison of Lightweight Machine Learning Models for IoT Intrusion Detection

Pradyumn Singh
John F. Kennedy High School
La Palma, California, US
contactprady@gmail.com

Russ Alizadeh
Cypress College
Cypress, California, US
rafaculty@gmail.com

## Abstract

IoT intrusion-detection models are useful only if they detect attacks without becoming too costly to run. We compared Logistic Regression, Decision Tree, Random Forest, and XGBoost on binary CICIoT2023 intrusion detection using a cleaned file with 21,005,260 rows and 37 numeric features. The main experiment used five attack-resampling balanced outer runs: each outer run reused all 1,047,367 benign records and sampled an equal number of attack records, followed by five inner train-validation-test splits. XGBoost had the highest default-threshold F1-score in the main experiment (0.983109 ± 0.000205), followed by Decision Tree (0.982860 ± 0.000257), Random Forest (0.982237 ± 0.000371), and Logistic Regression (0.974215 ± 0.000272). Because the main outer runs share the benign records, we interpret the paired tests as evidence from repeated attack resampling rather than as five fully independent datasets. To address this limitation, we added a disjoint sensitivity analysis in which both benign and attack rows were resampled into non-overlapping 100,000-per-class outer subsets. The same general pattern remained: XGBoost reached 0.982572 ± 0.000612 F1, while Logistic Regression reached 0.974407 ± 0.000920. A feature-ablation check dropping `Number` and `HTTPS` caused only small F1 decreases and did not change the main ordering. Per-record inference latency on the main balanced test split ranged from 0.615 $\mu$s for Logistic Regression to 1.124 $\mu$s for Random Forest, with XGBoost at 0.647 $\mu$s. Overall, the results support tree-based models for this controlled binary benchmark, while also showing why latency, footprint, and dataset-specific shortcuts must be considered before making deployment claims.

## Keywords

IoT Security, Cybersecurity, Intrusion Detection, Machine Learning, XGBoost

**ACM Reference Format:**
Pradyumn Singh and Russ Alizadeh. 2026. A Five-Run Comparison of Lightweight Machine Learning Models for IoT Intrusion Detection. In *Proceedings of International Journal of Secondary Computing and Applications Research (IJSCAR VOL. 3, ISSUE 3)*. ACM, New York, NY, USA, 7 pages. https://doi.org/10.67149/yhjs2024.5/br0qqg04


*IJSCAR VOL. 3, ISSUE 3*


## 1 Introduction

IoT devices are now part of homes, factories, hospitals, cars, and public infrastructure. That spread makes them useful, but it also gives attackers more places to look for weak points. Many IoT devices are always online, lightly protected, and difficult to patch after deployment [3, 4, 7]. Intrusion detection is therefore an important layer of defense, especially when the hardware itself cannot easily be redesigned.

Machine-learning intrusion detection is appealing because it can learn traffic patterns that rule-based systems may miss [4, 8]. For IoT systems, however, accuracy is not the whole problem. A detector may need to run near the edge, where storage, memory, and latency are limited [3, 10]. This paper therefore treats deployment cost as an offline measurement target, not as a claim that the models have already been certified on edge hardware.

This study compares four common tabular models: Logistic Regression, Decision Tree, Random Forest, and XGBoost. We chose this group because it creates a practical range from a compact linear baseline to more flexible tree-based models. Logistic Regression is fast and easy to interpret, but it can only learn a linear decision boundary. The tree-based models can capture nonlinear interactions, which are often useful in cybersecurity datasets [6, 8, 10]. The question is not whether a flexible model can win once. The question is whether the gain is repeatable and large enough to justify the added cost.

CICIoT2023 is a strong benchmark for this comparison because it was designed around large-scale IoT security traffic, with 105 devices and 33 attacks across seven attack families [5, 9]. The main experiment in this paper uses a balanced binary version of the task, where all malicious traffic is grouped into one attack class. This makes the model-family comparison easier to interpret because the majority attack class does not dominate the result. We also include a supplementary imbalanced validation to check whether the main ranking still appears when the original attack-heavy distribution is sampled directly.

The research question is: do tree-based models consistently outperform Logistic Regression on binary IoT attack detection, and is the improvement worth the extra compute? The two hypotheses are straightforward. **H1**: tree-based models achieve higher F1-scores than Logistic Regression. **H2**: those gains come with higher deployment cost,

measured through latency, model size, and memory footprint.

The study contributes a repeated-run comparison of four lightweight tabular models on CICIoT2023, pairs predictive metrics with cost measurements, adds an imbalanced validation, and includes reviewer-driven sensitivity checks for shared benign rows and possible shortcut features. The goal is not to propose a new algorithm or a final deployment system. It is to make a careful binary benchmark comparison that is useful when deciding which model family deserves further testing in a practical IoT intrusion-detection pipeline.

## 2 Related Work

Recent reviews of IoT intrusion detection reach a similar conclusion: machine learning can improve detection, but the best model depends on the deployment setting [4, 7, 8]. Lightweight methods remain important because many IoT environments cannot support large deep-learning pipelines or expensive real-time processing [3, 10]. At the same time, benchmark studies often emphasize peak accuracy, which can hide whether a result is stable across repeated sampling and whether it remains practical under latency and footprint constraints.

CICIoT2023 has become a common dataset for IoT intrusion-detection research [9]. Prior work has reported high binary-detection performance on this dataset, so the purpose of this paper is not to claim a new state of the art. The original CICIoT2023 benchmark reported very high binary accuracy for Random Forest and deep neural networks [9]. Yiltirak and Ozturk compared several standard machine-learning algorithms and reported Random Forest as the strongest method, including 96.79% F1 for binary classification and lower F1 for the 7-class and 33-class settings [11]. Adewole et al. evaluated ensemble methods with rule induction and reported XGBoost as the strongest CICIoT2023 binary model in their setup, with 98.55% F1 [1]. Almahaqeri et al. reported a more optimized gradient-boosting pipeline with feature selection, where XGBoost reached 0.995 macro-F1 and 0.355 $\mu$s/sample inference latency for binary detection [2]. Table 1 summarizes these prior results alongside this paper's main and disjoint-sensitivity F1 scores for direct comparison.

This paper therefore takes a narrower approach than many prior studies. It compares four familiar tabular models under the same preprocessing, threshold tuning, repeated-run structure, and deployment-cost measurements. The main experiment uses attack-resampling outer runs, and the revision adds a disjoint sensitivity analysis where both benign and attack rows are resampled. This framing is more limited than a full multiclass benchmark, but it directly addresses whether the ranking among lightweight model families is stable in a controlled binary setting.

## 3 Experimental Setup

The study used the CICIoT2023 benchmark dataset [9]. The raw CSV file contained 45,019,243 rows. Duplicate removal discarded 24,013,983 rows, or 53.34% of the raw file, leaving 21,005,260 cleaned rows with 37 numeric features. The cleaned file contained 1,047,367 benign records and 19,957,893 attack records. Duplicate removal affected the attack labels unevenly: for example, 4,985,687 DDOS-ICMP-FLOOD rows, 3,216,938 DDOS-UDP-FLOOD rows, and 2,745,436 DDOS-TCP-FLOOD rows were removed, while only 4,006 benign rows were removed. Because the cleaned file was strongly attack-heavy, the main experiment used a balanced binary task to compare model families under controlled class proportions.

For each of five outer random seeds (101, 202, 303, 404, and 505), we built one balanced binary subset with all 1,047,367 benign rows and an equal number of sampled attack rows. Each outer run therefore contained 2,094,734 rows. This design should be described as *attack-resampling*, not as five fully independent datasets, because the benign records were reused across outer runs. Within each outer run, five inner stratified train-validation-test trials were created with different split seeds. The split proportions were 70% training, 15% validation, and 15% testing, giving 1,466,313 training rows, 314,210 validation rows, and 314,211 test rows per trial. This produced 25 trials per model and 100 total model fits across the four models.

To address the shared-benign limitation, we added a reviewer-requested disjoint sensitivity analysis. This analysis used five non-overlapping outer subsets with 100,000 benign rows and 100,000 attack rows per outer seed. It used three inner train-validation-test splits per outer seed, producing 15 trials per model. We also repeated this disjoint analysis after dropping `Number` and `HTTPS`, the two features that ranked highest in the Random Forest importance analysis, to test whether the main conclusion depended on these possible dataset-specific shortcuts.

Preprocessing was fitted inside each trial to avoid information leakage. Numeric features were converted to numeric values, infinite values were replaced with missing values, and missing values were imputed inside the training pipeline. Standardization was also fitted only on the training split and then applied to the validation and test splits. In other words, validation and test records did not affect the fitted preprocessing steps.

The comparison included Logistic Regression, Decision Tree, Random Forest, and XGBoost. Logistic Regression used the `lbfgs` solver with `C=1.0` and `max_iter=1000`. Decision Tree used the Gini criterion with `max_depth=12`. Random Forest used 100 trees, `max_depth=14`, `max_features=sqrt`, bootstrapping, and `n_jobs=-1`. XGBoost used 100 estimators, depth 6, learning rate 0.1, subsampling and column-sampling rates of 0.8, `logloss` evaluation, and `n_jobs=-1`. These were fixed modest configurations rather than a full hyperparameter search, so the results compare reproducible model configurations rather than each model family's maximum possible performance.

F1-score was the primary metric because it balances precision and recall. Accuracy, precision, recall, ROC-AUC,

**Table 1: Selected CICIoT2023 results from prior work. Results are not directly comparable because preprocessing, feature selection, class balancing, tuning, and hardware differ across studies.**

| Study | Task | Reported result | Relevance to this paper |
|---|---|---|---|
| Pinto Neto et al. [9] | Binary and multiclass CICIoT2023 benchmark | Random Forest binary accuracy 99.68%; DNN binary accuracy 99.44% | Establishes that binary CICIoT2023 detection can reach near-ceiling accuracy. |
| Yiltirak and Ozturk [11] | 33-class, 7-class, and binary CICIoT2023 classification | Random Forest binary F1 96.79%; 7-class F1 94.31%; 33-class F1 94.84% | Shows that performance depends strongly on task granularity. |
| Adewole et al. [1] | Binary ensemble IDS with rule induction | XGBoost CICIoT2023 F1 98.55%; accuracy 98.54% | Gives a close ensemble-method comparison point for binary detection. |
| Almahaqeri et al. [2] | Optimized binary, 8-class, and 34-class gradient-boosting IDS | XGBoost binary macro-F1 0.995; inference latency 0.355 $\mu$s/sample | Shows what is possible with optimized feature selection and tuning. |
| This paper | Controlled binary benchmark with repeated runs and cost reporting | XGBoost main F1 0.983109; disjoint sensitivity F1 0.982572 | Focuses on repeated sampling, transparency, and model-family tradeoffs rather than peak optimization. |

PR-AUC, false positives, false negatives, and threshold-optimized F1 were also recorded. For deployment cost, the notebook measured training time, inference time for the full held-out test split, per-record inference latency, serialized model size, and memory deltas during fitting and prediction. For the main attack-resampling experiment, paired tests were computed using the five outer-run means, but the resulting $p$-values are interpreted cautiously because the benign records are shared across outer runs. The disjoint sensitivity analysis provides a stronger check because both classes were resampled into non-overlapping outer subsets. Paired t-tests were reported with bootstrap confidence intervals. Wilcoxon signed-rank tests were also computed, but with only five paired outer observations their two-sided $p$-values are limited and should be read cautiously.

# 4 Results and Discussion

## 4.1 Main balanced experiment

Table 2 summarizes the cleaned dataset and the main evaluation setup. The full cleaned CICIoT2023 file was strongly attack-heavy, with 19,957,893 attack rows and 1,047,367 benign rows. Duplicate removal removed more than half of the raw rows, mostly from high-volume attack labels. We used balanced binary subsets for the main comparison so that the results would reflect model behavior rather than the original class ratio, while also reporting the shared-benign limitation directly.

The main performance results are shown in Table 3 and Figure 1. XGBoost had the highest default-threshold F1-score across the 25 balanced trials (0.983109 $\pm$ 0.000205). Decision Tree was very close (0.982860 $\pm$ 0.000257), followed by Random Forest (0.982237 $\pm$ 0.000371). Logistic Regression was lower at 0.974215 $\pm$ 0.000272. The gap was small in absolute terms, but it appeared consistently

**Table 2: Dataset and evaluation setup for the main balanced experiment.**

| Item | Value |
|---|---|
| Raw CICIoT2023 rows | 45,019,243 |
| Rows removed as duplicates | 24,013,983 (53.34%) |
| Cleaned CICIoT2023 rows | 21,005,260 |
| Features after cleaning | 37 |
| Full cleaned benign rows | 1,047,367 |
| Full cleaned attack rows | 19,957,893 |
| Main task | Balanced binary detection |
| Main outer-run design | Attack-resampling; benign rows reused |
| Rows per balanced outer run | 2,094,734 |
| Benign rows per balanced run | 1,047,367 |
| Attack rows per balanced run | 1,047,367 |
| Training rows per trial | 1,466,313 |
| Validation rows per trial | 314,210 |
| Test rows per trial | 314,211 |
| Outer runs / inner trials | 5 / 5 |
| Total repeated fits per model | 25 |
| Disjoint sensitivity check | 100,000 benign + 100,000 attack rows per outer run |

across the attack-resampling outer runs. Compared with Logistic Regression, the mean outer-run F1 improvements were 0.008894 for XGBoost, 0.008645 for Decision Tree, and 0.008022 for Random Forest.

These results support H1, but with an important design qualification. The paired t-tests used five outer-run means, but those outer runs are not fully independent because all of them contain the same benign records. They are therefore best interpreted as repeated attack-resampling comparisons. Under that framing, the pattern was still consistent: every tree-based model beat Logistic Regression in mean F1.

**Table 3: Predictive performance across 25 balanced trials. Values are mean ± SD across repeated trials. $p$-values compare the five attack-resampling outer-run mean F1-scores for each tree-based model against Logistic Regression and should be interpreted cautiously because benign rows are shared across outer runs.**

| Model | Accuracy | Precision | Recall | F1-score | $p$ vs. LR |
|---|---|---|---|---|---|
| Logistic Regression | 0.974846 | 0.999277 | 0.950379 | 0.974215 ± 0.000272 | – |
| Decision Tree | 0.983119 | 0.998176 | 0.968007 | 0.982860 ± 0.000257 | $2.46 \times 10^{-8}$ |
| Random Forest | 0.982542 | 0.999662 | 0.965410 | 0.982237 ± 0.000371 | $3.32 \times 10^{-8}$ |
| XGBoost | 0.983370 | 0.998781 | 0.967922 | 0.983109 ± 0.000205 | $4.30 \times 10^{-9}$ |

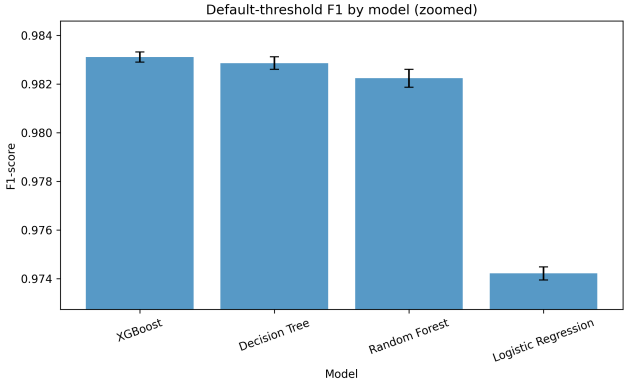


**Figure 1: Default-threshold F1-score across 25 balanced trials, shown with a narrowed y-axis so the error bars are visible. Error bars show standard deviation across repeated trials.**

Bootstrap 95% confidence intervals for the mean F1 improvement were fully above zero: 0.008515 to 0.008752 for Decision Tree, 0.007902 to 0.008134 for Random Forest, and 0.008817 to 0.008979 for XGBoost. The Wilcoxon signed-rank tests had the same direction, although the two-sided p-values were 0.0625 because there were only five paired outer-run observations. With $n = 5$ paired observations, 0.0625 is the smallest attainable exact two-sided Wilcoxon p-value, so this value reflects the resolution limit of the test rather than a change in the direction of the result. We therefore interpret the statistics as evidence of a stable advantage in this benchmark design, not as proof that the exact gap would hold in every IoT dataset.

## 4.2 Reviewer sensitivity checks

The disjoint sensitivity analysis addressed the main independence concern by resampling both classes. Each of five outer seeds used a non-overlapping subset with 100,000 benign and 100,000 attack rows, followed by three inner train-validation-test splits. Table 4 shows that the ranking remained similar to the main experiment. XGBoost again had the highest default F1-score (0.982572 ± 0.000612), while Logistic Regression remained lower (0.974407 ± 0.000920). The mean outer-run F1 improvements over Logistic Regression were 0.008165 for XGBoost, 0.007374 for Decision Tree, and 0.007093 for Random Forest. Bootstrap confidence intervals remained above zero: 0.007926 to 0.008428 for XGBoost, 0.007189 to 0.007522 for Decision Tree, and 0.006754 to 0.007432 for Random Forest. The disjoint result does not replace a full external validation dataset, but it reduces the concern that the original result came only from reusing benign records.

We also checked whether the high feature-importance rankings for `Number` and `HTTPS` were driving the result. Dropping those two features caused only small F1 decreases: -0.000171 for Logistic Regression, -0.000397 for Decision Tree, -0.001110 for Random Forest, and -0.000344 for XGBoost. The tree-based models still outperformed Logistic Regression after the ablation, with XGBoost maintaining the highest F1-score (0.982228 ± 0.000724). This does not prove that the models generalize to every dataset, but it suggests that the main conclusion is not dependent only on those two features.

Variance diagnostics also clarified the role of the outer-run structure. All variance values here refer to default-threshold F1-score. In the original attack-resampling experiment, the outer-seed variance of the outer-run means was smaller than the mean inner-split variance for every model: Logistic Regression $1.71 \times 10^{-8}$ vs. $7.15 \times 10^{-8}$, Decision Tree $2.15 \times 10^{-8}$ vs. $5.76 \times 10^{-8}$, Random Forest $2.08 \times 10^{-8}$ vs. $1.44 \times 10^{-7}$, and XGBoost $9.19 \times 10^{-9}$ vs. $4.13 \times 10^{-8}$. This supports the concern that reusing benign rows limited between-run variation. In the disjoint sensitivity analysis, the corresponding outer-vs.-inner variances were closer: Logistic Regression $5.23 \times 10^{-7}$ vs. $5.57 \times 10^{-7}$, Decision Tree $4.03 \times 10^{-7}$ vs. $4.07 \times 10^{-7}$, Random Forest $3.29 \times 10^{-7}$ vs. $2.65 \times 10^{-7}$, and XGBoost $2.54 \times 10^{-7}$ vs. $2.20 \times 10^{-7}$. This makes the disjoint analysis a better check on between-sample stability than the original attack-resampling design.

## 4.3 Supplementary imbalanced validation

The supplementary imbalanced validation tested whether the main conclusion depended too much on balancing. Each outer seed used a stratified 100,000-row sample with 95,014 attack records and 4,986 benign records, followed by three inner train-validation-test trials. Table 6 shows that the ranking changed somewhat, but the main message stayed similar. XGBoost again had the highest default-threshold F1-score (0.990388 ± 0.000448), with Random Forest close behind (0.990141 ± 0.000602). Decision Tree stayed above Logistic

**Table 4: Reviewer-requested disjoint balanced sensitivity analysis. Each outer run used non-overlapping 100,000-row benign and 100,000-row attack subsets, with three inner splits per outer run. Values are mean ± SD across 15 trials per model.**

| Model | Accuracy | Precision | Recall | F1-score | Latency ($\mu$s/record) |
|---|---|---|---|---|---|
| Logistic Regression | 0.975029 | 0.999262 | 0.950760 | 0.974407 ± 0.000920 | 0.438 |
| Decision Tree | 0.982053 | 0.996863 | 0.967151 | 0.981781 ± 0.000798 | 0.388 |
| Random Forest | 0.981824 | 0.999314 | 0.964311 | 0.981500 ± 0.000687 | 1.630 |
| XGBoost | 0.982842 | 0.998262 | 0.967369 | 0.982572 ± 0.000612 | 0.436 |

**Table 5: Feature-ablation check on the disjoint sensitivity design after dropping `Number` and `HTTPS`.**

| Model | Full-feature F1 | Ablated F1 | ΔF1 |
|---|---|---|---|
| Logistic Regression | 0.974407 | 0.974236 | -0.000171 |
| Decision Tree | 0.981781 | 0.981384 | -0.000397 |
| Random Forest | 0.981500 | 0.980390 | -0.001110 |
| XGBoost | 0.982572 | 0.982228 | -0.000344 |

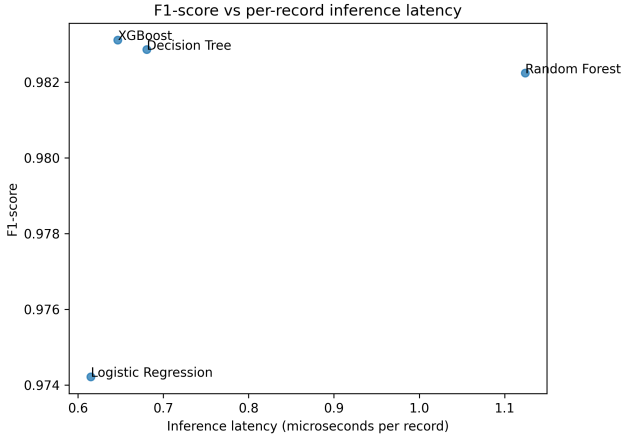


**Figure 2: Performance-versus-latency tradeoff using per-record inference latency. The x-axis is log-scaled so the lower-latency models remain legible.**

Regression, but the two ensemble methods separated more clearly from it than they did in the balanced experiment.

The imbalanced result is useful because it checks a different failure mode. In the balanced experiment, Decision Tree, Random Forest, and XGBoost were tightly grouped. In the imbalanced validation, XGBoost and Random Forest were more clearly ahead, and Logistic Regression again had lower recall. Compared with Logistic Regression, the mean outer-run F1 improvements were 0.016213 for XGBoost, 0.015966 for Random Forest, and 0.010198 for Decision Tree. The paired t-tests and bootstrap intervals again supported positive improvements over Logistic Regression. This does not replace testing on raw streaming traffic, but it does reduce the concern that the balanced results were only caused by artificial class proportions.

## 4.4 Deployment cost and operational behavior

The predictive ranking did not settle the cost question by itself. Table 7 and Figure 2 report both full-test-split inference time and per-record latency. The full balanced test split contained 314,211 rows, so the per-record latency is the clearest way to compare models. Logistic Regression was smallest and fastest per record, XGBoost was close behind with higher F1, and Random Forest was much larger and slower.

H2 was only partly supported. Tree-based models improved F1-score over Logistic Regression, but their costs differed sharply. Figure 4 shows the deployment footprint across models. Random Forest had the clearest cost penalty: it averaged 14.958 MB on disk and 1.124 $\mu$s per record on the balanced test split. Decision Tree was compact, but it did not beat XGBoost on F1. XGBoost was much smaller than Random Forest and had per-record latency close to Logistic Regression in this offline notebook benchmark. Because the timing was not measured on an edge device, this should be read as deployment-motivated evidence rather than a completed deployment benchmark.

The error counts, plotted in Figure 3, show why a small F1 gap can matter in security settings. Logistic Regression had the fewest false positives on average (108.00), but it missed the most attacks, with 7795.68 false negatives on the balanced test splits. Expressed per million attack flows, its false-negative rate corresponds to about 49,621 missed attacks. XGBoost missed about 32,078 attacks per million attack flows, Decision Tree about 31,993, and Random Forest about 34,590. In this benchmark, XGBoost therefore avoided roughly 17,543 missed attacks per million attack flows compared with Logistic Regression, although the exact operational impact would depend on the traffic volume and class mix in a real deployment.

Threshold tuning changed the ranking slightly. The optimized thresholds were 0.467 for Logistic Regression, 0.533 for Decision Tree, 0.316 for Random Forest, and 0.414 for XGBoost. At those thresholds, Random Forest had the highest optimized F1-score (0.983313), with XGBoost extremely close (0.983184). Because Random Forest was much heavier, the small tuned advantage would need to be justified by a

**Table 6: Supplementary imbalanced validation on stratified 100,000-row samples. Values are mean ± SD across 15 trials per model. The samples preserve the original cleaned-data class distribution.**

| Model | Accuracy | Precision | Recall | Default F1 | Optimized F1 |
|---|---|---|---|---|---|
| Logistic Regression | 0.952160 | 0.999946 | 0.949701 | 0.974175 ± 0.001032 | 0.985031 |
| Decision Tree | 0.970724 | 0.998657 | 0.970493 | 0.984373 ± 0.000975 | 0.988555 |
| Random Forest | 0.981280 | 0.990931 | 0.989354 | 0.990141 ± 0.000602 | 0.990105 |
| XGBoost | 0.981764 | 0.991985 | 0.988797 | 0.990388 ± 0.000448 | 0.990425 |

**Table 7: Operational summary across 25 balanced trials. Batch inference time is measured on the full 314,211-row test split. Per-record latency normalizes that value to microseconds per record. FP and FN are false positives and false negatives averaged across test splits; Opt. F1 is threshold-optimized F1 using the validation-selected threshold.**

| Model | Train time (s) | Batch inf. (s) | $\mu$s/record | Size (MB) | FP | FN | Opt. F1 |
|---|---|---|---|---|---|---|---|
| Logistic Regression | 8.775 ± 0.510 | 0.193237 | 0.615 | 0.005 | 108.00 | 7795.68 | 0.974205 |
| Decision Tree | 12.471 ± 0.242 | 0.213913 | 0.681 | 0.066 | 278.00 | 5026.32 | 0.982852 |
| Random Forest | 27.693 ± 0.596 | 0.353203 | 1.124 | 14.958 | 51.24 | 5434.36 | 0.983313 |
| XGBoost | 7.410 ± 0.269 | 0.203198 | 0.647 | 0.263 | 185.64 | 5039.64 | 0.983184 |

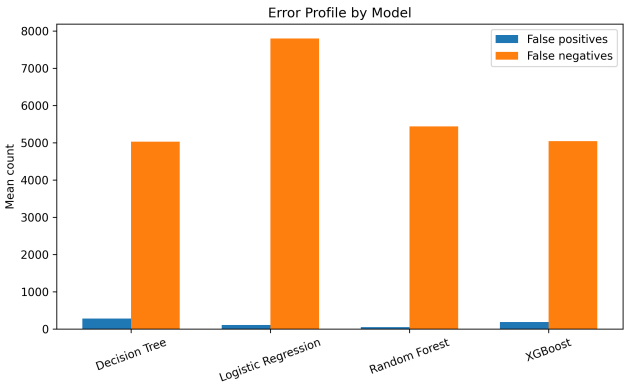


**Figure 3: Error profile by model. Bars show mean false-positive and false-negative counts across the 25 balanced trials.**

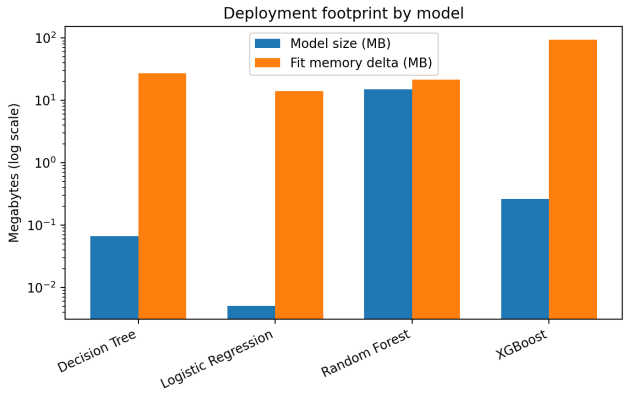


**Figure 4: Deployment footprint by model. The y-axis is log-scaled to show both small serialized models and larger memory deltas.**

deployment-specific cost-benefit analysis rather than by F1-score alone.

## 4.5 Feature importance and limits

The feature-importance results should be interpreted cautiously. In the Random Forest model, `Number` and `HTTPS` ranked highly in both impurity and permutation importance. That may reflect meaningful traffic-structure information, but it may also reflect dataset-specific shortcuts. For that reason, we do not use feature importance as causal evidence. The ablation analysis in Table 5 provides a more direct check: after dropping `Number` and `HTTPS`, all models lost some F1, but the losses were small and the tree-based models still outperformed Logistic Regression. This reduces, but does not eliminate, the concern that the result depends on those two features.

The study has several limits. First, it uses one benchmark dataset and a binary task, so it does not establish performance on every IoT environment, attack family, or multiclass formulation. The near-ceiling scores in prior CICIoT2023 work and in this paper suggest that binary detection is easier than attack-family or 33-class classification. Second, the main outer runs are attack-resampling runs rather than fully independent datasets because benign rows are reused; the disjoint sensitivity analysis addresses this limitation only at a smaller 100,000-per-class scale. Third, the supplementary imbalanced validation and disjoint sensitivity checks use stratified samples rather than a live traffic stream. Fourth,

timing and memory measurements come from notebook execution on a Windows machine, not from a dedicated edge device or resource-capped gateway. These limits do not invalidate the comparison, but they define its scope: the results are best read as a controlled, offline benchmark of model-family behavior, not as a full deployment certification.

## 5 Conclusion

Across five attack-resampling outer runs and 25 balanced trials per model, all three tree-based classifiers outperformed Logistic Regression on binary CICIoT2023 intrusion detection. The original design reused benign rows across outer runs, so the main paired tests should be interpreted cautiously. A disjoint sensitivity analysis that resampled both benign and attack rows preserved the same general pattern, with XGBoost achieving the highest default-threshold F1-score and Logistic Regression remaining lower. A feature-ablation check dropping `Number` and `HTTPS` caused only small F1 decreases and did not remove the tree-based advantage.

The practical lesson is not that one model is always best. XGBoost was the strongest overall option in this controlled benchmark because it combined the best default F1-score with a much smaller footprint and lower latency than Random Forest. Random Forest was competitive, especially after threshold tuning, but its size and inference time make it less attractive for constrained settings unless the extra cost is acceptable. Decision Tree was compact and nonlinear, while Logistic Regression remained the smallest and simplest baseline. For IoT intrusion detection, small F1 improvements should be judged alongside missed-attack counts, per-record latency, model size, and the hardware where the model would actually run.

## Code and Data Availability

The reproducibility materials are included with this revision package: executed Python notebook, exported result tables, reviewer sensitivity tables, figures, and metadata file. The original CICIoT2023 dataset is publicly available from the Canadian Institute for Cybersecurity [5]. A public repository can be added after review if needed.

## References

[1] Kayode S. Adewole, Andreas Jacobsson, and Paul Davidsson. 2025. Intrusion Detection Framework for Internet of Things with Rule Induction for Model Explanation. *Sensors* 25, 6 (2025), 1845. https://doi.org/10.3390/s25061845

[2] Salah A. Almahaqeri, Mohammed Hashem Almourish, Adel A. Nasser, Abed Saif Ahmed Alghawli, Amani A. K. Elsayed, et al. 2026. An Optimized Gradient Boosting Framework for IoT Intrusion Detection: A Comprehensive Evaluation on the CICIoT2023 Dataset. *Scientific Reports* 16 (2026), 16909. https://doi.org/10.1038/s41598-026-47399-5

[3] Zainab Alwaisi et al. 2024. Securing Constrained IoT Systems: A Lightweight Machine Learning Approach for Anomaly Detection and Prevention. *Internet of Things* 28 (2024), 101398. https://doi.org/10.1016/j.iot.2024.101398

[4] Mohammed Berhili et al. 2024. Intrusion Detection Systems in IoT Based on Machine Learning: A State of the Art. *Procedia Computer Science* 251 (2024), 99–107. https://doi.org/10.1016/j.procs.2024.11.089

[5] Canadian Institute for Cybersecurity, University of New Brunswick. 2026. IoT Dataset 2023. https://www.unb.ca/cic/datasets/iotdataset-2023.html Accessed 3 July 2026.

[6] Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16)*. Association for Computing Machinery, New York, NY, USA, 785–794. https://doi.org/10.1145/2939672.2939785

[7] Eric Gyamfi and Anca Jurcut. 2022. Intrusion Detection in Internet of Things Systems: A Review on Design Approaches Leveraging Multi-Access Edge Computing, Machine Learning, and Datasets. *Sensors* 22, 10 (2022), 3744. https://doi.org/10.3390/s22103744

[8] Brunel Rolack Kikissagbe and Meddi Adda. 2024. Machine Learning-Based Intrusion Detection Methods in IoT Systems: A Comprehensive Review. *Electronics* 13, 18 (2024), 3601. https://doi.org/10.3390/electronics13183601

[9] Euclides Carlos Pinto Neto et al. 2023. CICIoT2023: A Real-Time Dataset and Benchmark for Large-Scale Attacks in IoT Environment. *Sensors* 23, 13 (2023), 5941. https://doi.org/10.3390/s23135941

[10] Souradip Roy, Juan Li, Bong-Jin Choi, and Yan Bai. 2022. A Lightweight Supervised Intrusion Detection Mechanism for IoT Networks. *Future Generation Computer Systems* 127 (2022), 276–285. https://doi.org/10.1016/j.future.2021.09.027

[11] İsa Yiltirak and Ali Öztürk. 2025. Comparison of Performances of Machine Learning Algorithms in Detection of Internet of Things Attacks. *Journal of Intelligent Systems: Theory and Applications* 8, 2 (2025), 130–140. https://doi.org/10.38016/jista.1635809