The Telemetry Level: Signal Hygiene and Multimodal Attention Tracking in Stealth Assessment

1. The Crisis of Dwell Time: Why Passive Metrics Fail in Educational Data Mining

1.1 The "Sandwich Problem" and the Illusion of Engagement

In the rapidly evolving landscape of educational technology, the paradigm of "Stealth Assessment" promises a shift away from high-stakes, discrete testing toward a model of continuous, unobtrusive measurement of learner competency.[1] This approach relies on the seamless collection of telemetry data to infer a student's knowledge state in real-time. However, this promise is fundamentally compromised by the fidelity of the data being collected. As we transition from discrete assessment events to continuous telemetry, we encounter a critical vulnerability: signal noise. The primary artifact of this noise is the "False Positive" engagement metric, most notoriously represented by the naive calculation of "Dwell Time".[2]

The foundational assumption of many early Learning Management Systems (LMS) and Intelligent Tutoring Systems (ITS) was that digital presence equals cognitive engagement. "Time-on-Task" became a standard proxy for effort and, inversely, for the difficulty of the material.[3] In a controlled laboratory setting, where a proctor ensures the student is seated and attending to the terminal, this proxy holds substantial validity. If a student is seated in front of a terminal for 20 minutes, they are likely working for 20 minutes. However, in the unconstrained, asynchronous environment of web-based remote learning, this assumption collapses entirely.

We define this failure mode as the "Sandwich Problem." This scenario describes a user who interacts with a problem set, leaves the browser tab active (either visible or backgrounded), engages in an off-task activity—such as making a sandwich, answering a text message, or browsing social media on a secondary device—and returns ten minutes later to submit an answer. To a standard telemetry collector that relies on simple timestamps, this session appears as a continuous block of engagement: a start time at 10:00:00, an end time at 10:10:00, and a derived metric of 600 seconds of deliberation.

The 'Sandwich Problem': Why Dwell Time Deceives BKT Engines

Figure 1 compares the 'Perceived Learning Session' logged by a standard LMS against the 'Actual User Activity.' While the system records 600 seconds of active learning, the true cognitive engagement is only 30 seconds. This discrepancy introduces massive noise into the Bayesian Knowledge Tracing model, confusing 'deliberation' with 'idleness.'

When this corrupted data point is fed into a Bayesian Knowledge Tracing (BKT) engine, the algorithm interprets the 600-second duration as evidence of significant cognitive effort or, depending on the specific model parameters, extreme difficulty with the concept.[3] If the student eventually submits the correct answer, the system might infer that the student "struggled but mastered" the content, updating the learning probability ($P(L)$) accordingly. In reality, the student may have mastered the content in the first 30 seconds and spent the subsequent 570 seconds entirely disengaged. This false positive distorts the "Slip" and "Guess" parameters of the BKT model, degrading its predictive power for future tasks and potentially leading to inappropriate pedagogical interventions.[4]

1.2 Theoretical Implications of Signal Noise in BKT

The presence of outlier response times—data points that deviate significantly from the norm due to non-cognitive factors—can distort statistical analyses and lead to incorrect interpretations of student capability.[5] In the context of "Stealth Assessment," where the

system attempts to infer knowledge state without explicit testing, this noise is particularly damaging because the system lacks the corrective feedback loop of a human tutor who would visually observe the disengagement.

Research indicates that off-task behavior is not merely a pause in learning but a distinct behavioral state that must be detected and filtered.[1] Models that fail to account for "forgetting" or time gaps between interactions consistently over-predict student performance.[4] The standard BKT model updates the probability of mastery based on binary correctness, but its sensitivity to the *rate* of learning is heavily influenced by the temporal density of interactions. When the time dimension is polluted by idle time, the model's estimation of the "Learn Rate" becomes unstable.

Furthermore, the issue is not merely one of inflated time but of "Detector Rot".[6] Over time, as a system collects more noisy data, the classifiers used to detect student states (e.g., "frustrated," "bored," "engaged") degrade in accuracy because the ground truth labels (derived from behavior) no longer correspond to the actual cognitive state of the user. To address this, we must move beyond the "black box" of dwell time and peer inside the interval. We need to answer not just *how long* a user was on a page, but *what they were doing* during that time. This requires a shift from measuring *duration* to measuring *dynamics*.

The solution lies in "Multimodal Attention Tracking." By leveraging granular telemetry—specifically Mouse Velocity, Scroll Depth Heatmaps, and Page Visibility—we can filter idle time from valid learning time. This report details the architectural framework for this signal hygiene, moving from the physics of mouse movement to the mathematics of Time-Dependent Bayesian Knowledge Tracing (TD-BKT), and finally to the implementation of real-time "Micro-Interventions" triggered by WebSocket infrastructure.

---

2. Multimodal Attention Tracking: The Physics of Engagement

To filter the "Sandwich" from the "Study Session," we must implement Multimodal Attention Tracking. This approach treats the user's peripheral interactions—specifically mouse movements and scroll behavior—as a continuous stream of cognitive metadata. The premise is that physical engagement with the input device is a high-fidelity proxy for cognitive engagement with the interface.

2.1 Mouse Dynamics as a Proxy for Attention

Mouse cursor tracking is a powerful, low-cost method for predicting user attention and detecting mind-wandering.[7] Unlike eye-tracking, which requires specialized hardware or invasive webcam permissions, mouse tracking is native to the web browser and invisible to the

user. Research suggests a strong correlation between gaze position and cursor position, particularly during active information processing.[9] When a user is reading or engaging with content, the mouse often follows the eye, or at least exhibits "micro-movements" indicative of presence. Conversely, when a user zones out or leaves the device, the mouse becomes static or exhibits "ballistic" movements unrelated to the content.[11]

2.1.1 The "Reading Pattern" vs. "Idle State"

User behavior studies have identified distinct mouse signatures associated with reading. Some users trace lines of text horizontally, effectively using the cursor as a reading guide, while others engage in "vertical reading," moving the cursor down the margin as they scan.[12] These patterns are characterized by low, consistent velocity and frequent, small-scale directional changes. This behavior is distinct from "Navigation," which typically involves high-velocity, linear movements towards interactive elements like buttons or menus.

In contrast, "Idle Time" is physically distinct. It is not merely low velocity; it is the *absence* of velocity for a sustained threshold, often preceded by a high-velocity "exit" vector (moving the mouse away from the content area or parking it in a neutral zone).[13] The detection of "Mind Wandering" can also be inferred from cursor trajectories that deviate from the optimal path or exhibit high entropy without clicking.[11]

By tracking **Mouse Velocity** (pixels per millisecond) and **Acceleration**, we can define a heuristic for "Active Reading":

- **Active Reading:** Velocity > 0 and Velocity < Threshold (e.g., 50px/s) + Variance in Direction.
- **Navigation:** Velocity > Threshold (e.g., 500px/s).
- **Idle/Disengaged:** Velocity = 0 for > T seconds.

2.1.2 Technical Challenge: DPI and Resolution Independence

A critical challenge in measuring mouse velocity is the variability in hardware. A high-DPI gaming mouse moving one inch on a 4K screen generates significantly more "pixels" of movement than a standard mouse on a 1080p screen.[14] If our velocity metric is purely "pixels per second," a user with a sensitive mouse might trigger "Navigation" thresholds while simply reading.

To maintain signal hygiene, the system must normalize velocity. This can be achieved by calculating velocity relative to the viewport dimensions (e.g., % of viewport width per second) rather than raw pixels.[16] Furthermore, the system must account for the polling rate of the mouse. Gaming mice may report position 1000 times per second (1000Hz), while standard office mice report at 125Hz.[14] The telemetry engine must normalize these inputs to a standard time-base (e.g., milliseconds) to ensure comparable velocity metrics across devices.

Figure 2 illustrates simulated mouse velocity profiles. 'Active Reading' (Green) is characterized by low-amplitude, high-frequency micro-movements. 'Navigation' (Blue) shows high-amplitude spikes indicating ballistic movement. 'Idle' (Red) is a flatline at zero. The 'Stealth Assessment' hook filters out the Navigation spikes and Idle flatlines to calculate true 'Cognitive Dwell Time'.

Data sources: National Library of Medicine, Trymata, Mortoray

2.2 Scroll Depth Heatmaps: Verifying Content Consumption

While mouse velocity indicates *activity*, Scroll Depth indicates *coverage*. A user might move their mouse extensively but never scroll past the fold, indicating they have not consumed the instructional material. Conversely, a user who scrolls to the bottom instantly and clicks "Next" is likely "gaming" the system.[17]

By implementing **Scroll Depth Heatmaps**, we create a secondary validation layer. We divide the content area into logical "buckets" (e.g., 10% increments or semantic sections) and track

the cumulative dwell time within each bucket.[18] This granular tracking allows us to distinguish between "skimming" (passing through a bucket with high velocity) and "reading" (dwelling in a bucket with low scroll velocity).

Effective signal hygiene requires determining if the scroll speed is physiological. Human reading speed limits the rate at which content can be validly scrolled. If the scroll_velocity exceeds a "Reading Speed Threshold" (e.g., > 1000 pixels/second sustained), the system flags the behavior as "Skimming" rather than "Reading".[20] This distinction is vital for determining if a user has truly been exposed to the learning material or if they are merely searching for answers.

The integration of these two signals—Mouse Velocity (indicating presence/micro-attention) and Scroll Depth (indicating coverage)—forms the basis of our "Stealth Assessment" logic. It moves the metric from "Time on Page" to "Time on Content."

2.3 The Biological Basis: Microsaccades and Motor Planning

The theoretical underpinning for using mouse movement as a proxy for attention lies in the biological connection between the oculomotor system (eyes) and the manual motor system (hand). Research into "microsaccades"—tiny, involuntary eye movements that occur even during fixation—reveals that they are not random noise but are modulated by attention.[21]

Interestingly, parallel research in rodents has shown that motor planning involves coordinated movements of whiskers and eyes.[23] While humans do not have whiskers, the coordination between our primary sensory input (eyes) and our primary manipulation output (hands/mouse) is similarly coupled. When a user creates a motor plan to click on an object, the eye often moves to the target before the hand. More importantly for reading, the "micro-movements" of the mouse often mirror the saccadic jumps of the eye across text, a phenomenon leveraged by tools like "Mouse Tracking for Reading" (MoTR).[24]

MoTR studies have shown that mouse reading times correlate well with eye-tracking data, suggesting that the mouse can serve as a "poor man's eye tracker".[24] By analyzing the *quality* of these micro-movements—specifically their frequency and amplitude—we can infer the user's cognitive state. A "frozen" mouse suggests a decoupling of the eye-hand system, which often precedes a decoupling of attention (mind-wandering).

---

3. Heuristics for False Positive Filtering

To operationalize these physical signals, we must establish rigorous heuristics. The raw data stream is too noisy; it is filled with jitters, accidental bumps, and hardware anomalies. We need a logic layer to "clean" the signal before it reaches the BKT engine.

3.1 The 30-Second Idle Threshold

The "30-second rule" is a critical heuristic for distinguishing "Thinking" from "Leaving." While some web analytics literature suggests longer inactivity thresholds (e.g., 30 minutes for session timeouts [25]), the context of *micro-assessment* requires much tighter bounds. A student solving a math problem might pause to think, but for how long?

Research on "micro-stops" during reading suggests that pauses for cognitive processing (thinking about the content) rarely exceed a few seconds.[20] A pause of greater than 30 seconds typically indicates a break in attention—a "zone out" event or an off-task distraction.[11] Therefore, we implement a **"Learning Clock"** that is distinct from the **"Wall Clock."**

- **Wall Clock:** Continuous absolute time.
- **Learning Clock:** Accumulates only when:
  - Mouse Velocity > 0 (or micro-movements detected)
  - Page Visibility = Visible
  - Scroll Activity is recent (< 5s ago)
  - **Logic:** If Idle Time > 30s, the Learning Clock pauses. It resumes only upon re-engagement (e.g., mouse movement > threshold).

This heuristic aligns with research suggesting that inactivity thresholds should be dynamic and task-dependent, but for general web-based learning, 30 seconds serves as a robust upper bound for "active idle" time before it likely becomes "off-task" time.

3.2 Filtering "Ballistic" Navigation

Not all movement is good movement. As established in the "Physics of Engagement" section, high-velocity mouse movements often indicate a desire to exit or navigate away.

Heuristic: If Mouse Velocity > 600px/s (ballistic threshold), the time is categorized as "Navigational Overhead" rather than "Cognitive Processing".[7] This time should be excluded from the "Time-on-Step" metric fed into the BKT model. This filter prevents the system from rewarding a user who is frantically moving their mouse in frustration or boredom.

3.3 The Tab Visibility API

Modern browsers provide the Page Visibility API (document.visibilityState), which allows us to detect if the user has switched tabs or minimized the browser.[27] This is the coarsest but most effective filter for the "Sandwich Problem."

Heuristic: If document.visibilityState === 'hidden', the Learning Clock is **immediately** paused. No heuristic buffer is required. This hard filter eliminates the most obvious cases where the user is digitally absent. It also serves as a privacy-preserving measure, as we stop tracking

detailed telemetry when the user is not looking at our page.

---

4. Technical Implementation: The useStealthAssessment Hook

We now translate these heuristics into a concrete software architecture. We define a custom React hook, useStealthAssessment, written in TypeScript. This hook encapsulates the complexity of signal hygiene, exposing a clean "engagement" state to the frontend application while abstracting away the raw event listeners.

4.1 Architecture of the Hook

The hook manages three primary states:

1. **isIdle**: Boolean. True if no interaction > 30s.
2. **engagementScore**: Number (0-100). A composite score based on velocity variance and scroll depth.
3. **learningTime**: Number (ms). The "clean" accumulated time.

It relies on high-frequency event listeners (mousemove, scroll, visibilitychange) but must optimize performance to avoid blocking the main thread, which would degrade the user experience.

4.2 Performance Optimization: requestAnimationFrame vs. setInterval

Attaching logic directly to the mousemove event can fire hundreds of times per second, causing performance degradation and "jank" in the UI.[28] Furthermore, standard setInterval or setTimeout loops are not synchronized with the display's refresh rate, leading to potential timing artifacts.

To maintain 60fps performance and precise timing, we must use requestAnimationFrame (rAF) to throttle calculations.[29] The rAF loop ensures that our telemetry code runs exactly once per frame, synchronized with the browser's painting cycle.

The logic follows this pattern:

1. **Event Listener:** Captures clientX, clientY and simply updates a mutable ref (does not trigger a React re-render).
2. **rAF Loop:** Runs every frame (~16ms).
    - Reads the ref.
    - Calculates Euclidean distance moved since the last frame.

- Calculates instantaneous velocity: $V = \Delta d / \Delta t$ (using performance.now() for high-resolution timestamps).
- Updates a "Velocity Buffer" (array of last 60 frames).

3. **Analysis Loop:** Runs every 1 second (via setInterval or a throttled counter in the rAF loop).
   - Analyzes the Velocity Buffer to determine the "Current State" (Reading, Navigating, Idle).
   - Checks the 30s threshold.
   - Updates React State (isIdle, learningTime) *only* if the status changes or a significant time delta has passed. This minimizes React re-renders, keeping the application performant.[31]

4.3 TypeScript Logic Flow

The following logic flow details the internal decision tree of the useStealthAssessment hook. It demonstrates how raw inputs are filtered through our heuristics to produce a "Clean Telemetry" output.

Logic Flow: The useStealthAssessment Algorithm

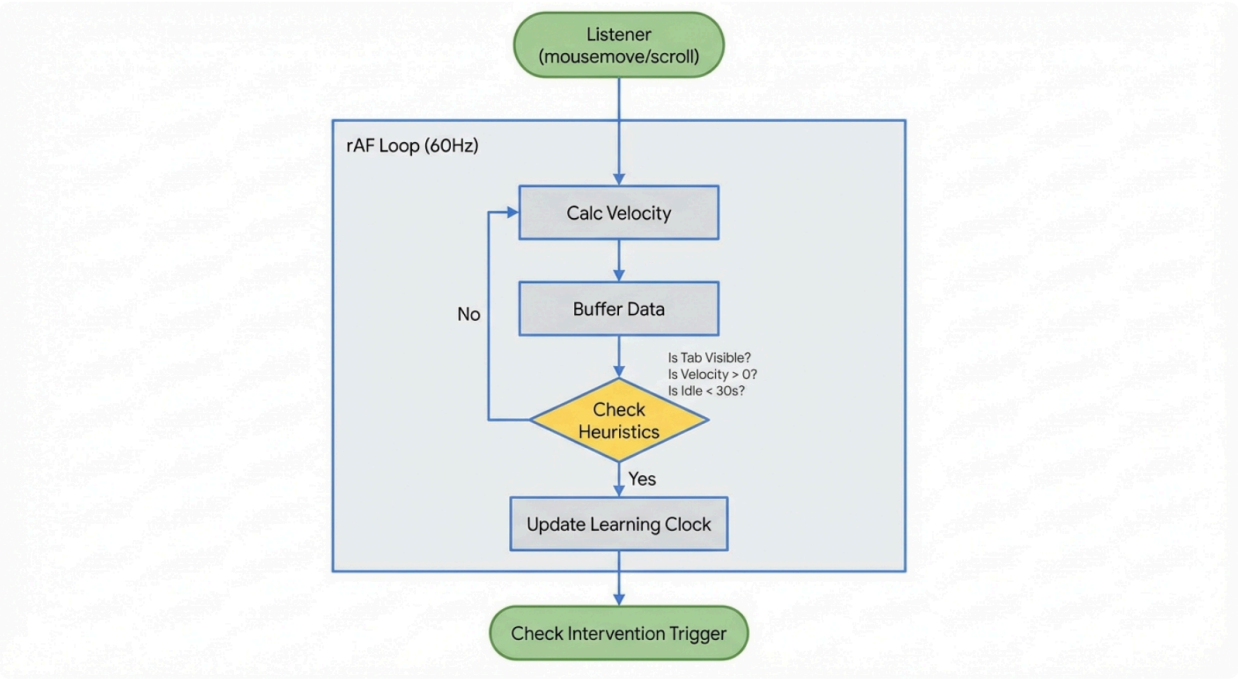

Figure 3 details the decision logic within the `useStealthAssessment` hook. Note the separation of high-frequency event capture (Green) from low-frequency state analysis (Blue). The 'Learning Clock' (Orange) only ticks when all 'Active Conditions' are met, effectively filtering out idle time and tab-switching.

4.4 Handling Scroll Buckets and Lazy Cleaning

To track "Scroll Depth Heatmaps," the hook requires knowledge of the document height and viewport height.

- **Bucket Size:** We define buckets as percentage integers (0, 10, 20... 100).
- **Tracking:** On the scroll event (throttled via rAF), we calculate currentScrollY / (docHeight - viewportHeight).
- **Storage:** We use a Set<number> data structure to store visited buckets. This ensures we track *unique* depth reached, preventing duplicate entries for the same section.[32]
- **Velocity Check:** Before adding a bucket to the Set, we check the scrollVelocity. If it exceeds the READING_SPEED_LIMIT (e.g., > 1000px/s), we do *not* mark the bucket as "read"—we mark it as "skipped."

The system also implements "Lazy Cleaning".[33] Rather than sending every scroll event to the server, we aggregate the Set of visited buckets locally and send a single summary payload (e.g., scroll_depth:) at the end of a session or at regular "heartbeat" intervals. This reduces network overhead and database load.

5. The Backend: Time-Dependent BKT and the Learning Clock

The frontend's responsibility is to send *clean* telemetry. The backend's responsibility is to consume this signal and update the learner model. We utilize a **Time-Dependent Bayesian Knowledge Tracing (TD-BKT)** approach [34], which explicitly models the passage of time as a factor in learning (and forgetting).

5.1 Integrating Capped Time into BKT Parameters

Standard BKT models do not natively account for "time spent." They primarily look at the sequence of correct/incorrect answers. However, sophisticated extensions incorporate response time as a parameter for estimating the Slip and Guess probabilities.[4]

When the useStealthAssessment hook sends a data point, it includes the learning_time_ms (the filtered, "clean" duration). The backend must treat this value as the "True Effort."

Logic for Parameter Adjustment:

- **Fast & Correct:** If learning_time_ms is very short (e.g., < 5s) and the answer is **Correct**, this implies high fluency. We might marginally increase the probability of Guess ($P(G)$) in the model if the time is *suspiciously* fast (sub-cognitive), or simply treat it as a high-mastery event.
- **Moderate & Correct:** If learning_time_ms is moderate (e.g., 30s-2m) and **Correct**, we increase the probability of Learn ($P(L)$). This "Goldilocks" zone indicates productive engagement and deliberation.
- **Long & Incorrect:** If learning_time_ms is long (e.g., > 5m, even after filtering) and **Incorrect**, the probability of Slip ($P(S)$) is unlikely. A student who thinks for 5 minutes and gets it wrong likely has a genuine knowledge gap, not a careless error.

Crucially, because we have filtered out the "Sandwich Time" on the client side, the backend rarely sees massive, noisy outliers (e.g., 600s durations). However, if an outlier does slip through (e.g., user reading continuously without moving the mouse), we apply a backend **Winsorization** or **Capping** heuristic.[37] Any duration > 3 standard deviations from the mean for that specific problem is capped at the 3σ value. This robust statistical treatment prevents a single anomalous session from wrecking the student's mastery model.

5.2 The "Forgetting" Factor

Standard BKT assumes knowledge state is constant between steps unless a learning event

occurs. However, time gaps induce forgetting. Using our high-fidelity telemetry, if we detect a "Session Break" (e.g., the user closed the tab or was idle for > 1 hour, detected via the timestamp delta between sessions), we apply a decay function to the prior mastery probability $P(L_{t-1})$ before processing the next step.[4]

$$P(L_{decayed}) = P(L_{t-1}) \times (1 - \delta)^{\Delta t}$$

Where δ is the forgetting rate and Δt is the filtered idle time. This ensures the model adapts to the user's "cold start" upon returning, preventing the system from presenting overly difficult material immediately after a long break.

---

6. Proactive Intervention: The "Micro-Intervention" Protocol

Cleaning the data is defensive. We can also be offensive. If the system detects signal degradation (disengagement), it should not just pause the clock—it should try to re-engage the learner. This is the domain of **Proactive Micro-Interventions**.

6.1 WebSocket Architecture for Real-Time Triggers

To enable real-time interventions, we move beyond the request-response cycle of HTTP and utilize **WebSockets** (specifically the Socket.io or standard ws protocol) to establish a full-duplex communication channel between the telemetry engine and the client.[39] This allows the backend to "push" an intervention to the frontend the moment the heuristic conditions are met, rather than waiting for the user to click something.

The Protocol Flow:

1. **Frontend:** The useStealthAssessment hook detects isIdle = true OR a cognitive_load_state = 'low' (based on ballistic mouse movement).
2. **Frontend:** It emits a TELEMETRY_UPDATE message via WebSocket containing the velocity profile, current idle duration, and scroll position.
3. **Backend (AI Engine):** The server analyzes the stream. If the probability_of_dropout exceeds a defined Threshold, it triggers an intervention logic.
4. **Backend:** The server emits a TRIGGER_INTERVENTION message to the specific client ID.
5. **Frontend:** Upon receiving the message, the client application pauses the main interface and spawns a **Modal Quiz** or **AI Chat Prompt**.

6.2 JSON Message Schema: Adhering to Standards

To ensure robustness and interoperability, we define a strict JSON schema for these

WebSocket messages, drawing inspiration from IMS Global's LTI 1.3 Proctoring standards.[41] While LTI is typically for LMS integration, its schema structure for "Start Assessment" and "End Assessment" provides a solid template for our "Start Intervention" messages.

Client -> Server (Telemetry Update)

JSON

```
{
  "type": "TELEMETRY_UPDATE",
  "payload": {
    "session_id": "sess_12345",
    "timestamp": 1678886400000,
    "metrics": {
      "mouse_velocity_avg": 0,
      "idle_duration_ms": 35000,
      "tab_visible": true,
      "scroll_depth_percent": 45
    },
    "state": "IDLE_WARNING"
  }
}
```

Server -> Client (Intervention Trigger)

JSON

```
{
  "type": "TRIGGER_INTERVENTION",
  "payload": {
    "intervention_id": "int_987",
    "priority": "HIGH",
    "ui_component": "MODAL_QUIZ",
    "content": {
      "title": "Quick Check-in",
      "body": "It looks like you've been paused for a while. Want to try a quick practice question to refresh?",
```

```
    "action_primary": "Start Quiz",
    "action_secondary": "I'm just reading"
  },
  "timeout_ms": 15000
 }
}
```

System Architecture: The Proactive Intervention Loop

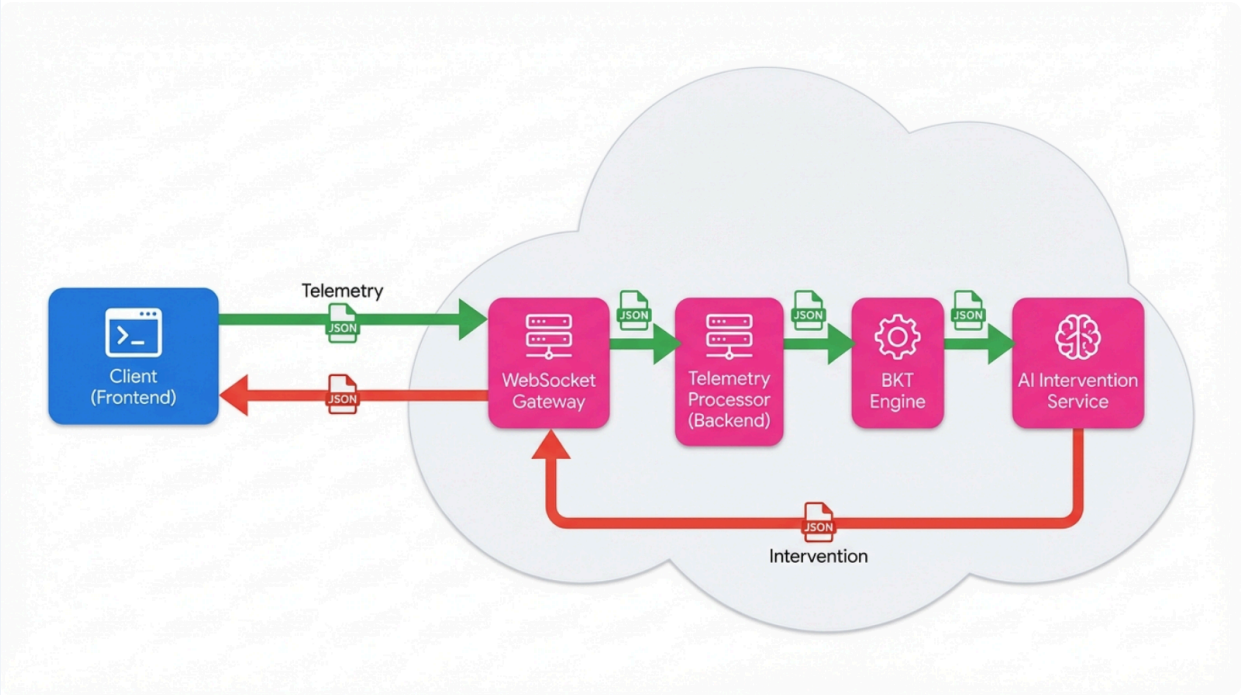

Figure 4 depicts the real-time intervention loop. 1. The Client pushes telemetry via WebSocket. 2. The Server processes this against the BKT model. 3. The 'Intervention Engine' detects disengagement. 4. The Server pushes a 'TRIGGER_INTERVENTION' payload back to the client, spawning the AI Chat Modal.

6.3 The "Perceived Cognitive" State & AI Chat

The intervention itself must be subtle. We do not want to annoy a user who is genuinely thinking. The "Micro-Intervention" uses the AI Chat to "check in".[42]

Logic:

- **Total Idle:** If Mouse Velocity == 0 for > 45s, the prompt is a generic "Are you still there?" modal. This confirms presence.
- **Stuck/Thinking:** If Mouse Velocity > 0 (user is present) but Scroll Velocity == 0 (user is

staying on one section) for > 60s, the system infers "Struggle" or "Confusion." The prompt is a gentle "Need a hint?" chat bubble.[43] This implies the user is present but potentially blocked.

This distinction is only possible because of our multimodal telemetry. A simple timeout cannot distinguish "Stuck" from "Sandwich." Mouse dynamics can.

---

7. Future Directions and Ethical Considerations

7.1 Privacy and Optical Telemetry

While high-fidelity mouse tracking offers immense value for assessment, it raises privacy concerns. Mouse movements can inadvertently reveal demographic information or even emotional states.[11] It is imperative that this telemetry is **anonymized** and **aggregated**. The raw coordinate data should be processed on the client side (edge computing) into abstract metrics (e.g., "engagement_score: 85") whenever possible, sending only the derived metadata to the server to minimize privacy risk.[44] Furthermore, users must be explicitly informed that "activity" is being tracked to aid their learning, distinguishing this from ad-tech surveillance.

7.2 The Next Frontier: Gaze Estimation via Cursor

Emerging research suggests that as users become accustomed to "mouse-heavy" interfaces, the correlation between gaze and cursor tightens.[10] Future iterations of this system could use "Cursor-Gaze Alignment" models to infer not just *if* a user is reading, but *which specific words* they are struggling with, enabling word-level interventions without the need for invasive webcam eye-tracking.[10] This would allow the BKT model to track mastery at the *concept* level within a single paragraph, identifying exactly where understanding breaks down.

Conclusion

The transition from "Time-on-Task" to "Multimodal Attention Tracking" represents a fundamental upgrade in the signal hygiene of educational technology. By implementing the useStealthAssessment hook, leveraging mouse velocity and scroll heatmaps, and filtering data through rigorous heuristics like the 30-second rule and tab visibility, we can effectively eliminate the "Sandwich Problem."

We move from a system that asks, "How long was the tab open?" to a system that asks, "How deeply was the learner engaged?" This cleaner telemetry feeds more accurate BKT models, enables proactive AI interventions via WebSockets, and ultimately creates a responsive, intelligent learning environment that understands the learner's state in real-time. The risk of

signal noise is high, but the engineering solutions—rooted in granular, multimodal telemetry—are available and effective.

Works cited

1. Detecting Students' Off-Task Behavior in Intelligent Tutoring Systems with Machine Learning Techniques - Purdue e-Pubs, accessed January 22, 2026, https://docs.lib.purdue.edu/ccpubs/396/
2. Automatic Detection of Off-Task Behaviors in Intelligent Tutoring Systems with Machine Learning Techniques - ResearchGate, accessed January 22, 2026, https://www.researchgate.net/publication/224610749_Automatic_Detection_of_Off-Task_Behaviors_in_Intelligent_Tutoring_Systems_with_Machine_Learning_Techniques
3. Deep Learning vs. Bayesian Knowledge Tracing: Student Models for Interventions - ERIC, accessed January 22, 2026, https://files.eric.ed.gov/fulltext/EJ1195512.pdf
4. Does Time Matter? Modeling the Effect of Time in Bayesian Knowledge Tracing - Educational Data Mining, accessed January 22, 2026, http://educationaldatamining.org/EDM2011/wp-content/uploads/proc/edm2011_paper5_full_Qiu.pdf
5. The hazards of dealing with response time outliers - Frontiers, accessed January 22, 2026, https://www.frontiersin.org/journals/psychology/articles/10.3389/fpsyg.2023.1220281/full
6. Knowledge Tracing Over Time: A Longitudinal Analysis - Educational Data Mining, accessed January 22, 2026, https://educationaldatamining.org/EDM2023/proceedings/2023.EDM-short-papers.28/index.html
7. The Attentive Cursor Dataset - Frontiers, accessed January 22, 2026, https://www.frontiersin.org/journals/human-neuroscience/articles/10.3389/fnhum.2020.565664/full
8. Mouse tracking performance: A new approach to analyzing continuous mouse tracking data - PMC - PubMed Central, accessed January 22, 2026, https://pmc.ncbi.nlm.nih.gov/articles/PMC11289036/
9. Detecting Mouse Movement with Repeated Visit Patterns for Retrieving Noticed Knowledge Components on Web Pages | Request PDF - ResearchGate, accessed January 22, 2026, https://www.researchgate.net/publication/220242287_Detecting_Mouse_Movement_with_Repeated_Visit_Patterns_for_Retrieving_Noticed_Knowledge_Components_on_Web_Pages
10. Mouse-Guided Gaze: Semi-Supervised Learning of Intention-Aware Representations for Reading Detection - arXiv, accessed January 22, 2026, https://arxiv.org/html/2509.19574v1
11. Using Computer Mouse Tracking to Predict Mind Wandering - CEUR-WS.org, accessed January 22, 2026, https://ceur-ws.org/Vol-2265/paper10.pdf
12. Mouse movement patterns and user frustration - Trymata, accessed January 22,

2026, https://trymata.com/blog/mouse-movement-patterns-and-user-frustration/
13. (PDF) Wandering Minds, Wandering Mice: Computer mouse tracking as a method to detect mind wandering - ResearchGate, accessed January 22, 2026, https://www.researchgate.net/publication/342396577_Wandering_Minds_Wandering_Mice_Computer_mouse_tracking_as_a_method_to_detect_mind_wandering
14. what's the difference between dpi and pointer speed : r/MouseReview - Reddit, accessed January 22, 2026, https://www.reddit.com/r/MouseReview/comments/1adqhym/whats_the_difference_between_dpi_and_pointer_speed/
15. Understanding Mouse Setting: DPI, IPS, Acceleration, and LOD - Glorious, accessed January 22, 2026, https://www.gloriousgaming.com/blogs/resources/mouse-setting-dpi-ips-acc-lod
16. Calculating mouse velocity with accuracy - Stack Overflow, accessed January 22, 2026, https://stackoverflow.com/questions/17516101/calculating-mouse-velocity-with-accuracy
17. Repairing Disengagement With Non-InvasiveInterventions, accessed January 22, 2026, https://all.cs.umass.edu/pubs/2007/arroyo_fjdmfbmw_AIED2007.pdf
18. Heatmaps - Datadog Docs, accessed January 22, 2026, https://docs.datadoghq.com/session_replay/heatmaps/
19. How-to: Create a scroll depth heat map in Google Sheets to see how far readers are scrolling down your page – Chartbeat Help & FAQ, accessed January 22, 2026, https://help.chartbeat.com/hc/en-us/articles/360026549573-How-to-Create-a-scroll-depth-heat-map-in-Google-Sheets-to-see-how-far-readers-are-scrolling-down-your-page
20. Similarities and Differences Between Eye and Mouse Dynamics During Web Pages Exploration - PMC - NIH, accessed January 22, 2026, https://pmc.ncbi.nlm.nih.gov/articles/PMC8024563/
21. Micro and regular saccades across the lifespan during a visual search of “Where's Waldo” puzzles - PMC - PubMed Central, accessed January 22, 2026, https://pmc.ncbi.nlm.nih.gov/articles/PMC5851597/
22. Modulation of microsaccade rate by task difficulty revealed through between- and within-trial comparisons - Journal of Vision, accessed January 22, 2026, https://jov.arvojournals.org/article.aspx?articleid=2213284
23. Coordination between Eye Movement and Whisking in Head-Fixed Mice Navigating a Plus Maze - PubMed Central, accessed January 22, 2026, https://pmc.ncbi.nlm.nih.gov/articles/PMC9426778/
24. Mouse Tracking for Reading (MoTR): A New Naturalistic Incremental Processing Measurement Tool - Research Collection, accessed January 22, 2026, https://www.research-collection.ethz.ch/bitstream/handle/20.500.11850/653693/motr_manuscript.pdf?sequence=2
25. User Session Identification Based on Strong Regularities in Inter-activity Time,

accessed January 22, 2026, https://www.researchgate.net/publication/310823830_User_Session_Identification_Based_on_Strong_Regularities_in_Inter-activity_Time
26. Automated Disengagement Tracking Within an Intelligent Tutoring System - Frontiers, accessed January 22, 2026, https://www.frontiersin.org/journals/artificial-intelligence/articles/10.3389/frai.2020.595627/full
27. custom-react-hooks/use-idle - NPM, accessed January 22, 2026, https://www.npmjs.com/package/%40custom-react-hooks%2Fuse-idle
28. Tracking Scroll Position With React Hooks - DEV Community, accessed January 22, 2026, https://dev.to/n8tb1t/tracking-scroll-position-with-react-hooks-3bbj
29. Why is requestAnimationFrame better than setInterval or setTimeout - Stack Overflow, accessed January 22, 2026, https://stackoverflow.com/questions/38709923/why-is-requestanimationframe-better-than-setinterval-or-settimeout
30. Performant animations with requestAnimationFrame() and React hooks - Artiom Mozgovoy, accessed January 22, 2026, https://layonez.medium.com/performant-animations-with-requestanimationframe-and-react-hooks-99a32c5c9fbf
31. useMousePosition, a React hook that tracks the mouse cursor position within the page • Josh W. Comeau, accessed January 22, 2026, https://www.joshwcomeau.com/snippets/react-hooks/use-mouse-position/
32. react-scroll-tracker - NPM, accessed January 22, 2026, https://www.npmjs.com/package/react-scroll-tracker?activeTab=readme
33. Reviewing 8 JavaScript Heatmaps In LightningChart JS, accessed January 22, 2026, https://lightningchart.com/blog/javascript-heatmaps/
34. Time-dependant Bayesian knowledge tracing—Robots that model user skills over time - NIH, accessed January 22, 2026, https://pmc.ncbi.nlm.nih.gov/articles/PMC10925631/
35. Time-dependant Bayesian knowledge tracing—Robots that model user skills over time - Frontiers, accessed January 22, 2026, https://www.frontiersin.org/journals/robotics-and-ai/articles/10.3389/frobt.2023.1249241/full
36. Time-dependant Bayesian knowledge tracing—Robots that model user skills over time, accessed January 22, 2026, https://www.researchgate.net/publication/378899395_Time-dependant_Bayesian_knowledge_tracing-Robots_that_model_user_skills_over_time
37. The hazards of dealing with response time outliers - PMC - PubMed Central, accessed January 22, 2026, https://pmc.ncbi.nlm.nih.gov/articles/PMC10484222/
38. Why Detecting Outliers is Crucial for Accurate Data Analysis?, accessed January 22, 2026, https://www.dasca.org/world-of-data-science/article/why-detecting-outliers-is-crucial-for-accurate-data-analysis
39. Integrating WebSocket to enhance the EdTech project with real-time data - Mysoly, accessed January 22, 2026,

https://mysoly.nl/integrating-websocket-to-enhance-the-edtech-project-with-real-time-data/
40. WebSockets Unlocked: Mastering the Art of Real-Time Communication - DEV Community, accessed January 22, 2026, https://dev.to/raunakgurud09/websockets-unlocked-mastering-the-art-of-real-time-communication-2lnj
41. 1EdTech Proctoring Services Specification | IMS Global Learning Consortium, accessed January 22, 2026, https://www.imsglobal.org/spec/proctoring/v1p0
42. Using Intelligent Tutoring Systems to Enhance Student Engagement - ResearchGate, accessed January 22, 2026, https://www.researchgate.net/publication/392659841_Using_Intelligent_Tutoring_Systems_to_Enhance_Student_Engagement
43. Extending the Hint Factory for the Assistance Dilemma: A Novel, Data-driven HelpNeed Predictor for Proactive Problem-solving Hel, accessed January 22, 2026, https://jedm.educationaldatamining.org/index.php/JEDM/article/download/450/126
44. Mouse movements reveal your behaviour - University of Luxembourg, accessed January 22, 2026, https://www.uni.lu/en/news/mouse-movements-reveal-your-behaviour/