



Cognitive Fatigue in Human–AI Oversight : Evaluating the Review of Agent Actions

Venkata Krishna Reddy Atchi

School of Informatics, Computing, and Cyber Systems, Northern Arizona University Arizona, United States of America

* Corresponding Author: Venkata Krishna Reddy Atchi ; avkreddy0607@gmail.com

Abstract: A common safeguard for autonomous agents is to have a person approve their actions. But people tire, and a tired reviewer misses more. This paper asks how many actions should be sent to a human, and which ones, when the reviewer's catch probability falls as the recent workload rises. We model a reviewer whose catch probability declines with recent busy time, and a risk scorer that ranks actions. We prove that, when the scorer's ranking is well behaved, the share of bad actions caught is a concave function of the review rate, so there is a single best rate, and that for a Gaussian scorer reviewing everything is never the best choice once fatigue is present. At the optimum, an action is worth sending to the reviewer only if its probability of being bad exceeds a threshold that depends on the fatigue strength and on how much is already being reviewed. In a minute-by-minute simulation of an eight-hour shift (1,000 runs per setting), sending every action to the reviewer missed 17.3 of about 30 bad actions, because the mean catch probability fell to 0.43. Sending only the riskiest 36% of actions missed 11.2, a reduction of 35% with 67% fewer reviews. A threshold computed from the theory, with no tuning, matched the best tuned threshold. A controller that simply holds the reviewer's load near a target did not beat a well-tuned fixed threshold, and learning the fatigue strength from audits worked only when about 30% of reviewed bad actions were audited. Whether scheduled breaks help depends on the fatigue model: they did nothing when fatigue followed recent load alone, and cut misses by 17 to 23% when a slow, cumulative component was added. All results are from simulation with assumed parameters.

Keywords: Human Oversight, Alert Fatigue, Vigilance, Human-In-The-Loop, Agent Safety

1. Introduction

When people worry about what an autonomous agent might do, the usual answer is to keep a human in the loop. The agent proposes, a person approves. It sounds safe, and in small volumes it is. The trouble starts when the volume grows. A reviewer who sees a few actions an hour reads each one. A reviewer who sees hundreds learns to click through. Research on vigilance has documented this for decades: attention to rare events declines over a watch, and the decline is steeper under heavy load [1]. In hospitals the same effect has a name, alert fatigue, and it has been measured in clinical decision support systems [2]. This matters for agent oversight because the obvious policy, send everything to a person, can be the worst available. If the reviewer's attention is a limited resource that drains with use, then every extra item sent for review has two effects [3]. It gives the reviewer one more chance to catch a bad action, and it lowers the chance of catching every other one. A low-risk item can therefore make the system worse. The

question we ask is how to ration human attention. We assume a risk scorer that ranks actions, a reviewer whose catch probability falls as recent workload rises, and a stream of mostly harmless actions with a few bad ones mixed in. We want to choose what to send to the reviewer so that as few bad actions as possible get through. We make five contributions.

- A simple model of review under fatigue, with a risk scorer, a queue and a reviewer whose catch probability depends on recent load (Section 3).
- Two results (Section 4). The share of bad actions caught is concave in the review rate, so the best rate is unique. For a Gaussian scorer, reviewing everything is never best when fatigue is present, and the optimal rule has an interpretable form: send an action only if its probability of being bad exceeds a fatigue-dependent threshold.
- A simulation of an eight-hour shift that tests this



against a tuned threshold, a load controller, random sampling and review-everything, with sweeps over fatigue strength, scorer quality and workload (Sections 5 and 6).

- A comparison of two fatigue models showing that the usefulness of breaks depends on whether fatigue has a slow, cumulative part, and an experiment on learning the fatigue strength from audits (Sections 6.7 and 6.10).
- An honest account of what did not work: an adaptive load controller was no better than a tuned fixed threshold, and a wrong fatigue estimate costs more in one direction than the other.

Everything here is simulation under assumed parameters. We did not run a study with human reviewers, and the fatigue model is a deliberate simplification.

2. Related Work

Vigilance and fatigue. The vigilance decrement, a decline in detection of rare signals during sustained monitoring, was first described by Mackworth [1]. Later work has argued that sustained attention is effortful and draws on limited resources [2], which is the assumption behind our model. Automation research adds that people who monitor automated systems can become over-reliant on them [4, 5]. Alert fatigue in medicine, where repeated and mostly irrelevant alerts lead clinicians to ignore them, has been linked to workload and repetition [3].

Deferring to humans. Machine-learning work on learning to defer asks when a model should hand a case to an expert [6, 7]. It usually treats the expert as having a fixed accuracy. Our setting differs in that the expert's accuracy depends on how often they have recently been asked.

Signal detection and queues. Signal detection theory describes a detector's ability by its ROC curve, whose concavity for monotone likelihood ratios is classical [8]. Our concavity result uses exactly that property. We use a simple queue with a finite buffer in the simulation, and wait times follow Little's law [9].

Where this fits. We do not claim a new model of human attention. What we add is a clean analysis of how when we always review those with the highest scores. Let $G(\rho)$ be the fraction of bad actions among those reviewed, out of all bad actions (the recall of the selector). Because F converges to the load $u = \rho a/C$, the catch probability is $h(\rho) = h_0(1 - \phi_e \rho)$ with $\phi_e = \phi a/C$. The share of bad actions caught is

$$F(\rho) = h_0 (1 - \phi_e \rho) G(\rho). \quad (2)$$

Proposition 1 (Unique optimum). *If G is concave, non-decreasing and $G(0) = 0$ on $[0, 1]$, then F is concave on $[0, 1]$. Any interior maximiser ρ^* satisfies*

$$\frac{G'(\rho^*)}{G(\rho^*)} = \frac{G'(\rho^*)}{1 - \phi_e \rho^*} \phi_e \quad (3)$$

Proof: Write $h(\rho) = h_0(1 - \phi_e \rho)$, so $h'' = 0$ and $h' = -h_0 \phi_e < 0$. Then $F' = h' G' + 2h' G' + h G'' = 2h' G' + h G'' \leq 0$ because $G' \geq 0$, $G'' \leq 0$, $h' < 0$ and $h \geq 0$. A concave function has a unique maximiser on an interval when it is strictly concave there, and at an interior maximiser $F' = h' G' + h G'' = 0$, which rearranges to Eq. (3). A risk scorer and a fatigable reviewer should be combined, and a check of it against simulation, including where the simple analysis breaks.

3. Research Methodology

3.1. Arrivals, Scorer And Queue

Actions arrive as a Poisson stream at rate a per minute, and a fraction π are bad. A risk scorer gives each action a score s . For bad actions $s \sim N(\mu, 1)$ and for good ones $s \sim N(0, 1)$, so the scorer's quality is set by μ (the area $h G'' \leq 0$ because $G' \geq 0$, $G'' \leq 0$, $h' < 0$ and $h = 0$). A concave function has a unique maximiser on an interval when it is strictly concave there, and at an interior maximiser $F' = h' G' + h G'' = 0$, which rearranges to Eq. (3).

The concavity of G holds when the scores have a monotone likelihood ratio, as in the Gaussian case. Each additional item reviewed is, on average, less likely to be bad than the one before it, under its ROC curve is $\Phi(\mu/2)$. Reviewing an action means sending it to a queue served at up to C reviews per minute. The queue holds at most $Q_{\max}$ items, and items that arrive when it is full pass through unreviewed. Figure 1 shows the system and Figure 2 the handling of each action.

3.2. The Detailed Analysis

The reviewer has a fatigue state $F_t \in [0, 1]$, an exponential moving average of how busy they have been:

$F_t = (1 - \beta)F_{t-1} + \beta b_t$, where b_t is the fraction of capacity used in minute t . Their catch probability is

Proposition 2 (When to review everything). With $G(1) = 1$, reviewing every action is optimal if and only if $G'(1) \geq \phi_e/(1 - \phi_e)$.

Proof. By concavity, the maximum is at $\rho = 1$ iff $F'(1) \geq 0$, that is, $-h_0 \phi_e G(1) + h_0(1 - \phi_e)G'(1) \geq 0$.

Corollary 1 (Gaussian scorer). For the Gaussian scorer, $G'(1) = 0$, so reviewing everything is never optimal when $\phi_e > 0$. At the optimum the selected threshold θ^* satisfies

$$ht = h_0 (1 - \phi F t), \quad (1)$$

$$P(\text{bad} \mid s = \theta^*) = \frac{\pi \phi_e G(\rho^*)}{1 - \phi \rho^*} \quad (4)$$

Proof: At threshold θ , q falls as θ rises, with

$$dq = -[\pi f_1(\theta) + (1 - \pi)f_0(\theta)]d\theta$$

and

$$dG = -f_1(\theta)d\theta,$$

where f_1, f_0 are the densities of bad and good scores. So

$$G'(\theta) = \frac{f_1(\theta)}{\pi f_1(\theta) + (1 - \pi)f_0(\theta)} = \frac{P(\text{bad} \mid s = \theta)}{\pi}. \quad \text{At } \theta = 1$$

we have $\theta \rightarrow -\infty$,

where the likelihood ratio

$$f/f = e^{\mu\theta - \mu/2} \rightarrow 0, \text{ so } G'(1) = 0.$$

Substituting

$G = P(\text{bad} \mid \theta)/\pi$ into Eq. (3) gives Eq. (4).

We want to minimise the number of bad actions that are missed. A bad action is missed if it is not selected for

review, is dropped from a full queue, is reviewed but not caught, or is still waiting at the end of the shift

4. Experimental Implementation

Consider the steady state in which the queue does not overflow, and let ρ be the fraction of actions reviewed

Remark 1 (Queue saturation). When $a\rho > C$ the queue overflows and only a fraction $C/(a\rho)$ of selected items is reviewed. The simulation uses this correction, $(\rho) \min 1, C/(a\rho)$, when computing the theory threshold, and our default parameters stay below saturation at the optimum.

Shift. A shift is 480 minutes. Arrivals are Poisson with $a = 1.6$ per minute, a fraction $\pi = 0.04$ are bad, and the scorer has $\mu = 1.5$ (AUC 0.86). The reviewer can process $C = 2$ items per minute, the queue holds $Q_{\max} = 20$ items, and $h_0 = 0.92$, $\phi = 0.7$, $\beta = 0.05$ per minute (a 20-minute memory). These give $\phi_e = 0.56$. Reviewing every action would load the reviewer to $a/C = 80\%$.

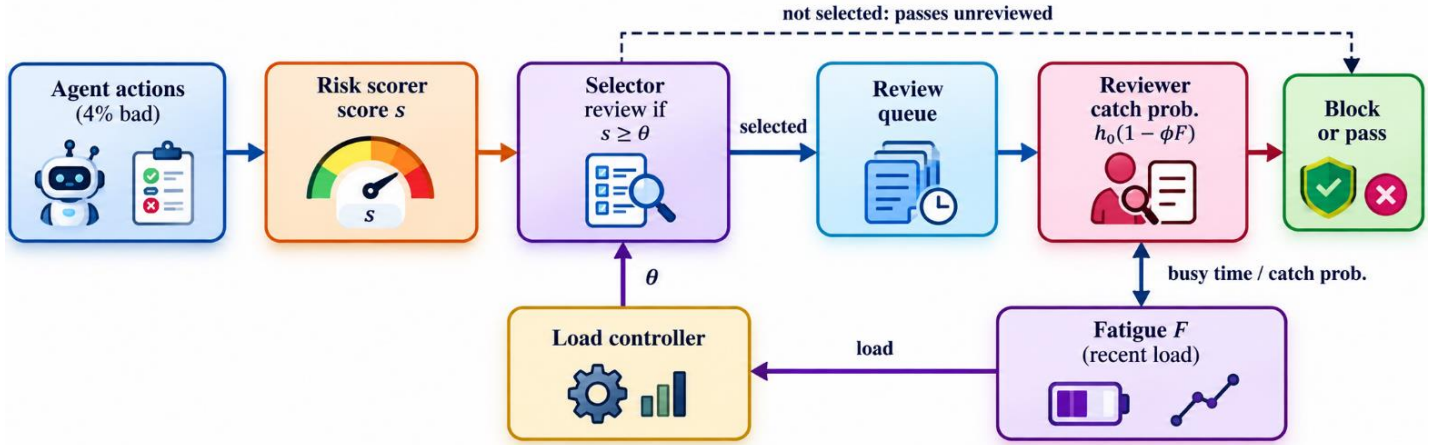


Figure 1: System model. A scorer ranks actions, a selector sends those above a threshold θ to the review queue, and the rest pass unreviewed. The reviewer's busy time feeds a fatigue state that lowers their catch probability, and a controller can adjust θ to hold the load near a target.

Policies. *Review all*: every action goes to the queue. *Random sample*: each action is selected with a fixed probability. *Fixed threshold*: review if $s \geq \theta$. *Adaptive load*: a controller moves θ up when the reviewer's recent busy fraction is above a target and down when it is below. *Theory threshold*: θ chosen by maximising the steady-state expression of Section 4, with no tuning on simulation. The random rate, the fixed threshold and the controller target were tuned on 400 separate runs, giving rate 0.8, threshold 0.50 and target 0.3. The theory threshold is 0.41.

Metrics. We report bad actions missed per shift, recall, reviews performed, the mean catch probability, and the mean wait in the queue. Each setting uses 1,000 runs unless stated, and policies share random seeds. We report 95% confidence intervals and Wilcoxon signed-rank tests.

5. Results and Analysis

5.1. Main Comparison

$\geq$

Table 1 and Figure 3 show the main result. Reviewing everything missed 17.3 of about 30 bad actions, a recall of only 43%. The reviewer was swamped: the catch probability, which starts at 0.92, fell below 0.43 within the first hour and the mean over the shift was 0.43. Random sampling at the tuned rate of 0.8 did no better (17.8 missed), since it spreads review evenly and wastes attention on harmless items. Sending only the actions the scorer ranks highest was much better. The tuned fixed threshold missed 11.2, a 35% reduction, with 67% fewer reviews (253 against 766). The reviewer stayed fresher (mean catch probability 0.76) and the queue wait fell to 0.04 minutes from 0.88.

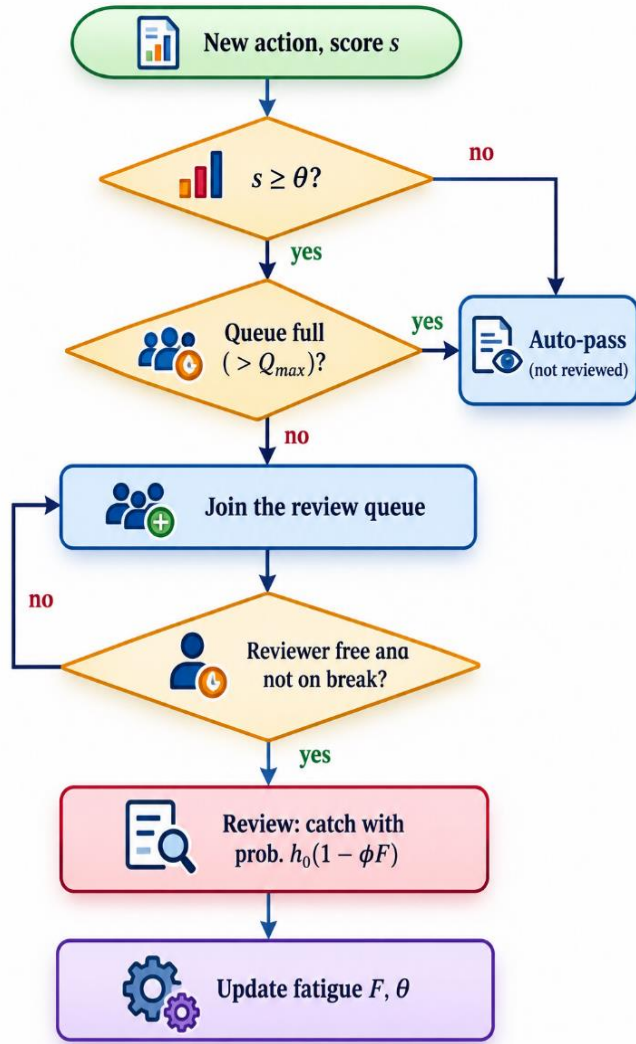


Figure 2: Handling of one action. If the scorer's output is below the threshold the action passes. If the queue is full it also passes. Otherwise it waits for the reviewer, who may be on a break.

The theory threshold, which needs no tuning, missed 11.2 and was statistically indistinguishable from the tuned one (Table 2). The adaptive load controller missed 11.6, slightly worse than the tuned fixed threshold by +0.39 bad actions on average ($p = 0.009$). In practice that is a small difference, and the controller's advantage is that it needs no knowledge of the scorer, as we see below.

5.2. Over the course of a shift

Figure 4 shows why. Under review-everything, the catch probability falls to its floor within about 61 minutes and stays there, so the reviewer spends seven hours as a weak filter. The thresholder policies hold the catch probability near 0.76. The right panel shows cumulative misses: review-everything and random sampling lose bad actions at a steady and higher rate throughout the shift. Because the fatigue model has a short memory, the penalty is paid almost from the start and not only late in the shift.

5.3. Theory against simulation

Figure 5(a) plots the three quantities in Eq. (2) for the default parameters. Recall rises with the review fraction but with diminishing returns, the catch probability falls linearly, and their product peaks at $\rho^* = 0.36$, where the reviewer's load is 0.29. Figure 5(b) compares the predicted share of bad actions caught with simulation over a grid of thresholds. The largest absolute difference is 0.016 and the mean is 0.005, and the simulated best threshold (0.50, recall 0.639) is within one grid step of the predicted one (0.41). Equation (4) also checks out. At the theory threshold the probability that an action is bad is 0.0243, and the right-hand side is 0.0242. That the closed-form condition holds to three decimals is mostly a check on our own algebra and code, since the simulation was written independently of the derivation.

Table. 1 Main results (1,000 runs; $\pm$ is the 95% CI half-width). "Wait" is the mean queue wait in minutes by Little's law.

Policy	Missed	Recall	Reviews	Catch Prob.	Wait
Review all	17.3 $\pm$ 0.3	43%	766	0.43	0.88
Random sample	17.8 $\pm$ 0.3	42%	614	0.52	0.35
Fixed threshold	11.2 $\pm$ 0.2	63%	253	0.76	0.04
Adaptive load	11.6 $\pm$ 0.2	62%	290	0.73	0.08
Theory threshold	11.2 $\pm$ 0.2	64%	278	0.74	0.05

Table. 2 Adaptive load controller against each alternative (bad actions missed; negative favours the controller). "Wins" is the share of paired runs in which the controller misses fewer; p is a Wilcoxon signed-rank test.

Alternative	Diff	Wins	p
Review all	-5.75	83%	$< 10^{-3}$
Random sample	-6.22	85%	$< 10^{-3}$
Fixed threshold	+0.39	41%	0.009
Theory threshold	+0.33	43%	0.050

5.4. Fatigue strength

Figure 6(a) varies ϕ . At $\phi = 0$ the reviewer does not tire, and reviewing everything is about as good as anything (2.6 missed; theory 2.4). As ϕ grows, review-everything gets steadily worse, reaching 24.4 at $\phi = 1$, while the theory threshold, which adapts to ϕ , reaches only 13.1. Review-everything is clearly worse than the theory threshold from $\phi = 0.4$ on (11.0 against 8.5). The fixed threshold tuned at $\phi = 0.7$ is a good illustration of tuning risk. At the tuning point it is as good as the theory threshold (11.1 against 11.2), but at $\phi = 0$ it misses 6.8 against 2.4, since it keeps ignoring items a fresh reviewer could easily have handled. Retuning fixes this (3.3). The adaptive load controller, whose target of 0.3 was also set at $\phi = 0.7$, has the same problem at $\phi = 0$ (6.6) because its target does not know

how much fatigue costs. So an adaptive controller on load is not the same thing as a fatigue-aware one.

5.5. Scorer Quality

Figure 6(b) varies the scorer's AUC. With a useless scorer (AUC 0.50) there is nothing to rank by, so thresh-olding cannot help: the retuned threshold misses 18.7 against 17.7 for review-everything, and the adaptive con-troller does worse still at 22.5. As the scorer improves, thresholding pulls ahead, and at AUC 0.98 the retuned threshold misses just 5.8. Between, at AUC 0.86, the retuned threshold, the controller and review-everything miss 11.1, 11.4 and 17.7. The value of a good risk scorer is therefore larger under fatigue than one would guess, because it lets a small amount of attention do most of the work.

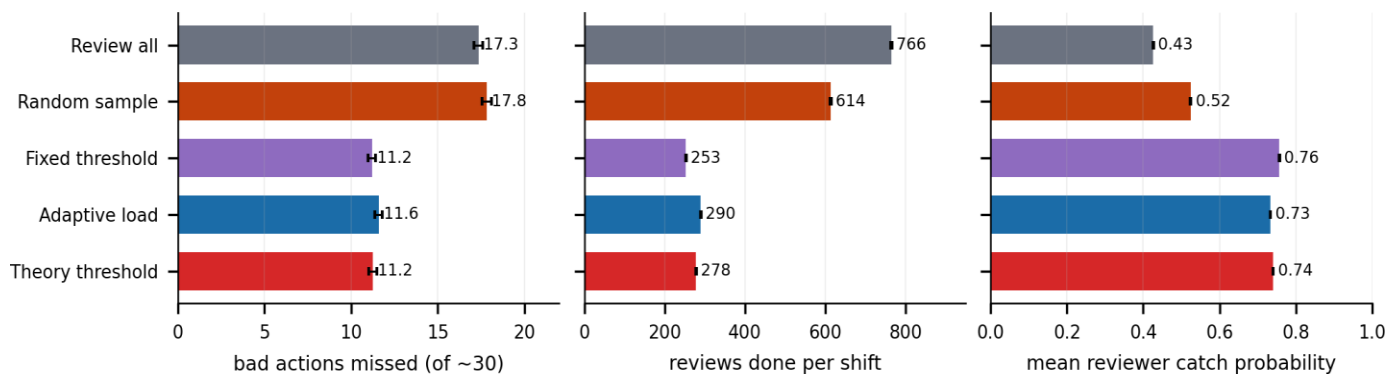


Figure. 3 Left: bad actions missed per shift. Middle: reviews performed. Right: mean reviewer catch probability. Reviewing everything gives the reviewer the most work and the lowest catch probability, and so the most misses.

5.6. Workload And Misestimated Fatigue

Figure 7(a) varies the arrival rate. At low volume ($a = 0.5$) review-everything is competitive and the three policies miss 24%, 27% and 23% of bad actions. As volume grows, review-everything collapses, missing 85% at $a = 4$. The threshold tuned at $a = 1.6$ misses 58% there, and the adaptive controller 54%, slightly better, because it moves its threshold as load changes. This is where the controller earns its place: when the workload moves away from the one the threshold was tuned for. Figure 7(b) sets the threshold from a wrong fatigue estimate. Assuming no fatigue at all gives 17.9 missed, the same as reviewing everything. Underestimating fatigue is costly, with 13.8 at a quarter of the truth. Overestimating is cheaper: 12.1 at twice the truth and 13.8 at three times. Between half and double the true value, misses stay between 10.9 and 12.1, so a rough estimate is enough. When in doubt, assume the reviewer tires more easily.

5.7. Learning the fatigue strength from audits

The theory threshold needs ϕ , which a deployment will not know. One way to learn it is to audit a share of the reviewed bad actions with an independent second check that reveals whether the reviewer caught them. Regressing the audit outcome on the reviewer's fatigue state gives an

estimate $\hat{\phi}$, and the threshold is recomputed each day. We started from a prior of $\hat{\phi} = 0.3$ (a deliberate underestimate) and ran 14 shifts, with the true value 0.7. Figure 8 shows the result. With no audits the threshold stays at the prior and the system misses about 12.3 per shift over days 8 to 14. With 10% of reviewed bad actions audited, the estimate is too noisy to help: it ends at 0.62 with a spread of 0.32 across runs, and misses (12.3) are no better than with the prior. With 30% audited, the estimate settles near the truth (0.71, spread 0.22) and misses fall to 11.6, close to the oracle's 11.2. Learning is slow, though. After three days the estimate is still only 0.30 with a spread of 0.03. And the audit cost is real: a second check of 30% of reviewed bad actions is a lot of extra work in a setting where bad actions are rare to begin with. We read this as a feasibility result, not a recipe.

5.8. How common are bad actions?

Equation (4) contains the prevalence π , so one might expect the best threshold to move a lot with it. It does not. Across prevalences from 0.5% to 15%, the theory threshold stays between 0.40 and 0.41 and the reviewer's load at the optimum rises only from 0.28 to 0.34 (Figure 9). Recall stays between 62% and 65%, against about 41 to 43% for review-everything. The reason is that prevalence enters

both sides of the optimality condition: it makes each selected item more likely to be bad, which raises the posterior, and it raises the bar by the same factor. The adaptive controller loses a little at the highest prevalence

(60% recall at 15%), when more bad items push the load above its target.

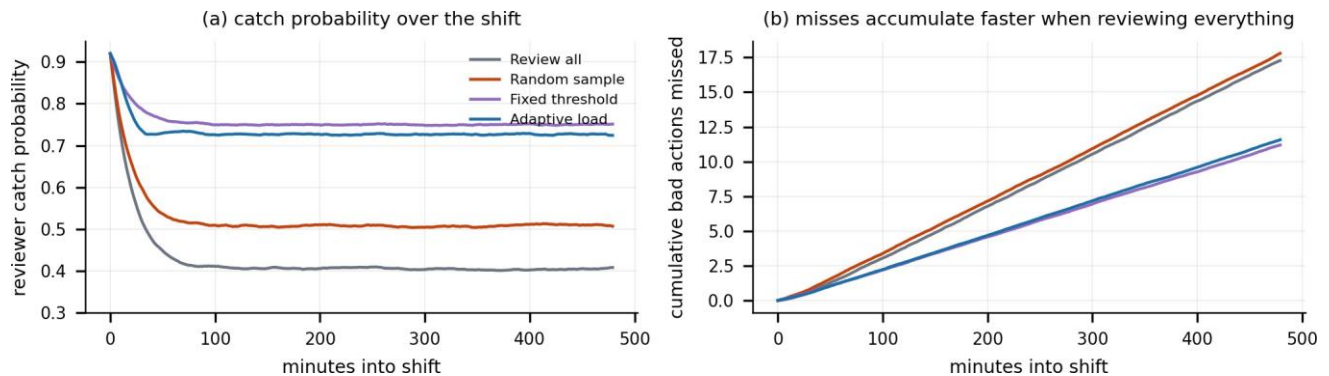


Figure. 4 (a) Mean reviewer catch probability by minute. (b) Mean cumulative bad actions missed. Review-everything and random sampling quickly wear the reviewer down.

5.9. Do breaks help?

We tested scheduled breaks (a work block followed by a break during which nothing is reviewed and items queue up) under the fast-fatigue model, with work blocks of 20 to 120 minutes and breaks of 2 to 15 minutes (Figure 10). The answer is that they do not help. For the adaptive controller, misses range from 11.1 to 14.1 against a baseline of 11.6, and the worst cells are the 15-minute breaks, which leave work piling up in the queue. For review-everything and a heavy threshold the differences are within run-to-run noise (about 0.2 for the standard error of a cell). This follows from the model. When fatigue depends only on recent load, a break just moves work to later. The reviewer recovers during the break but faces a fuller queue afterwards.

5.10. A slow, cumulative fatigue component

The result above depends on the fatigue model, so we ran a second one. Catch probability now depends on an even mix

of the fast state F and a slow state that builds with busy time, recovers only slowly during normal work and much faster during a break. Under this model review-everything misses 14.8 (against 17.3 before), the retuned fixed threshold 10.4 (recall 66%), and the theory threshold, derived from the fast model alone, also 10.4. So the theory threshold remained near-optimal even though the model was wrong.

Breaks now help (Figure 11). With the fixed threshold, the best schedule (30 minutes of work followed by 5 minutes of break) cuts misses from 10.2 to 8.5, a 17% reduction. For review-everything, breaks cut misses from 14.7 to 11.2 (23%). Long breaks of 15 minutes again do worse than short ones, with a mean of 9.0 for the fixed threshold against 9.3 for 2-minute breaks. So the value of breaks depends on whether fatigue has a part that recovers only with rest. That is an empirical question about real reviewers that this paper cannot answer.

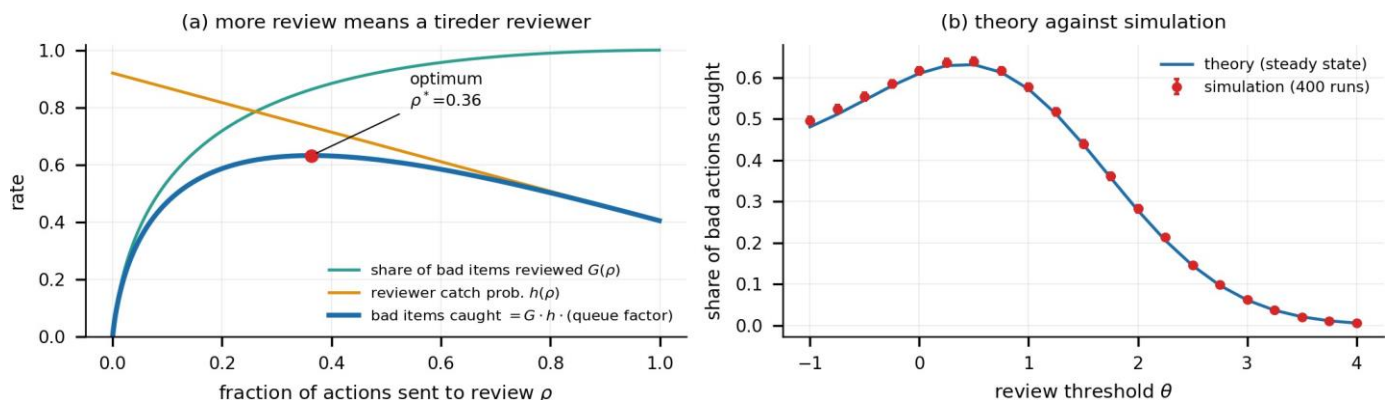


Figure. 5 (a) Recall of bad actions, reviewer catch probability, and their product against the review fraction. There is a single peak well short of reviewing everything. (b) Predicted share of bad actions caught against simulation across thresholds.

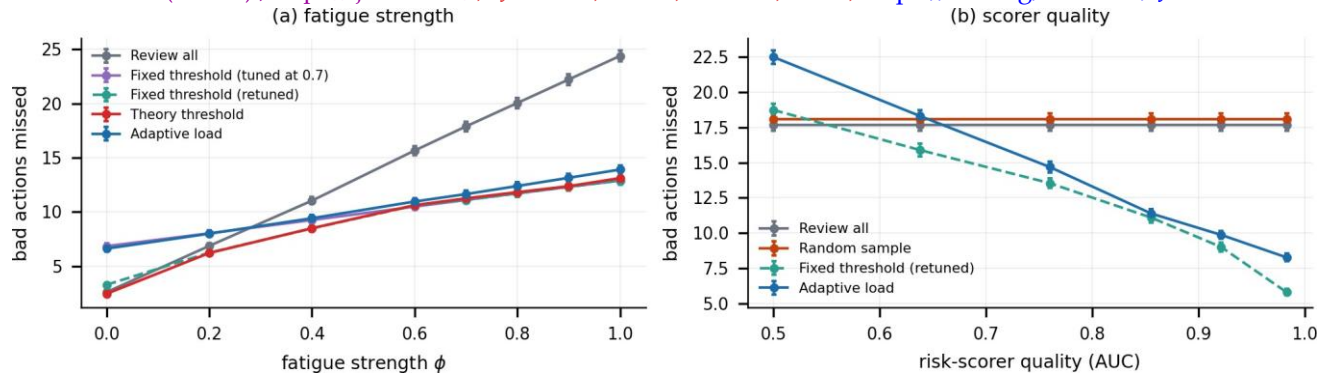


Figure. 6 (a) Bad actions missed as fatigue strength ϕ grows. Policies tuned at $\phi = 0.7$ lose out at low ϕ . (b) Bad actions missed as scorer quality improves. Thresholding helps only when the scorer carries information.

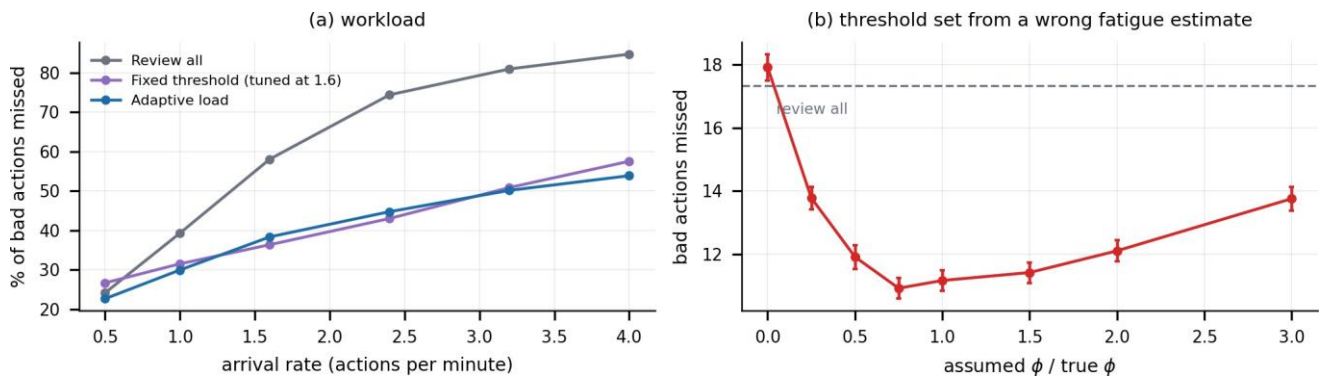


Figure. 7 (a) Share of bad actions missed as the arrival rate grows. (b) Bad actions missed when the theory threshold is computed from an incorrect fatigue strength. Assuming too little fatigue is the more costly error

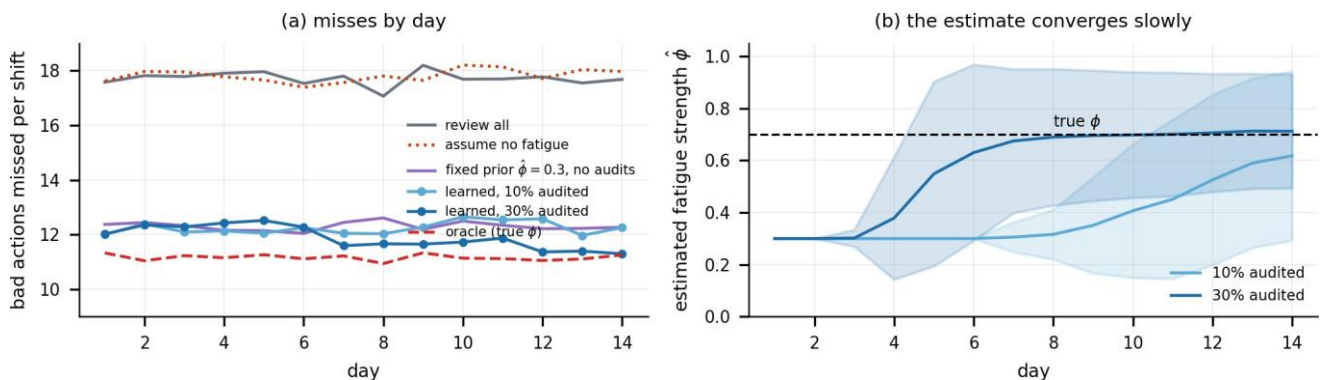


Figure. 8 Learning the fatigue strength from audits over 14 shifts (300 runs per day). (a) Bad actions missed per shift. (b) Estimated fatigue strength (mean and one standard deviation across runs). Light auditing is too noisy to help; heavy auditing approaches the oracle.

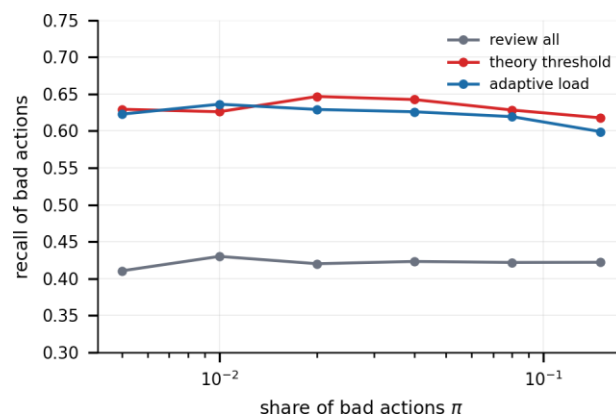


Figure. 9 Recall of bad actions as the share of bad actions varies. The optimal threshold hardly moves, so the theory threshold is robust to prevalence.

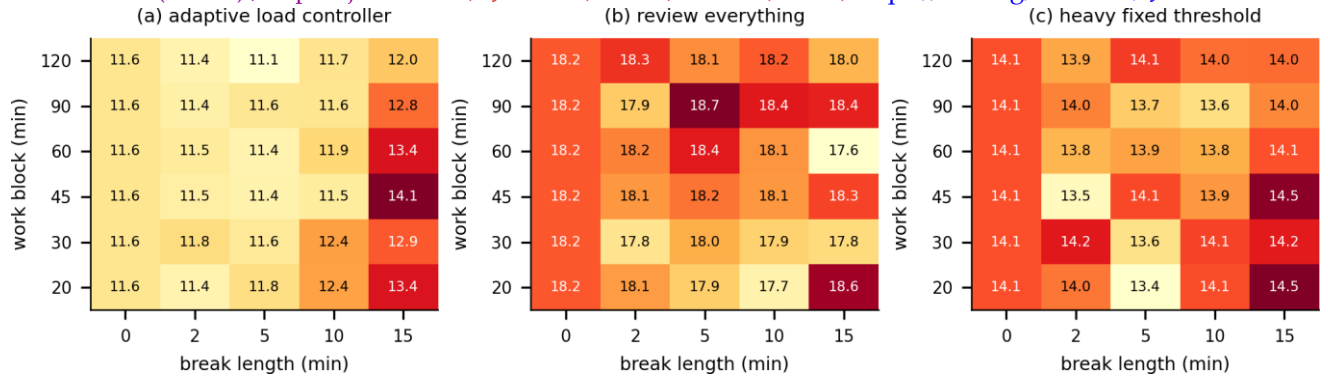


Figure. 10 Bad actions missed under scheduled breaks, in the fast-fatigue model. Rows are work-block lengths and columns are break lengths (0 means no breaks). Differences are within noise except for long breaks, which let the queue build.

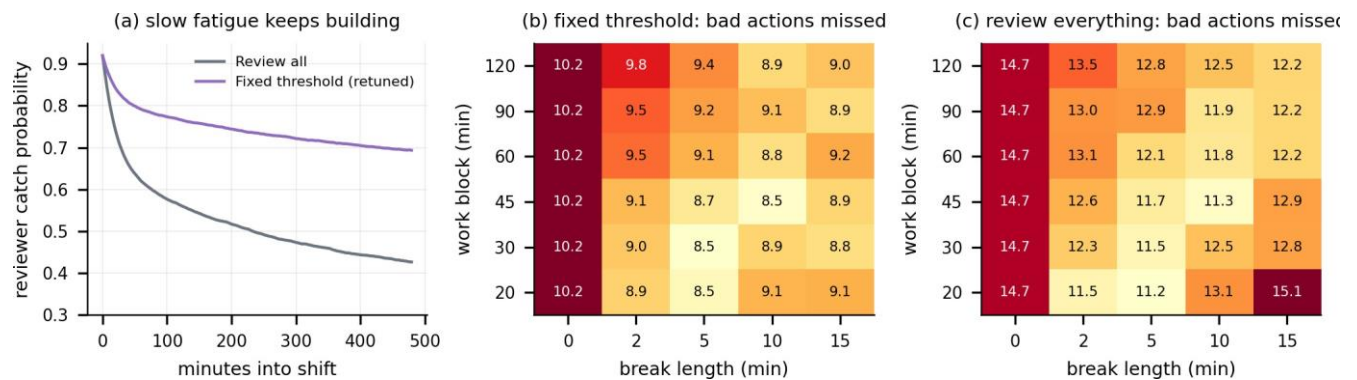


Figure. 11 Model with slow, cumulative fatigue. (a) Catch probability keeps falling through the shift. (b, c) Bad actions missed under break schedules: short breaks every 30 to 60 minutes help, long breaks do not.

5.11. Discussion and Limitations

What we think carries over: The qualitative lesson is that attention is a budget, and spending it on low-risk items costs attention for high-risk ones. Any oversight design with a human reviewer should ask what happens to the reviewer's catch rate as volume rises, and a risk scorer that lets most items bypass review is probably worth more than it looks.

What may not: The fatigue models are deliberately simple. We assumed catch probability depends on a smooth function of load. Real reviewers differ widely, shifts have structure (lunch, hand-offs), and some fatigue effects show up as bias in what gets approved and not only as missed items. We did not model reviewers who learn the scorer's habits, who may start to trust its low scores too much, or an adversary who shapes actions to look low-risk. The scorer in our model is perfectly calibrated and Gaussian, and a real one will not be.

The controller's role: The adaptive load controller was no better than a tuned fixed threshold in our main setting, and it was worse than a fatigue-aware threshold when fatigue was weak. Its value is robustness to work-load changes. Section 6.7 tries the natural improvement,

Severity: We counted all bad actions equally. In practice, some are far worse than others, and the weighting of the objective changes the optimum. The structure of the theory survives (the objective stays concave if the severity-weighted recall is concave), but the numbers would change.

6. Conclusion and Future Scope

Sending every agent action to a human can catch fewer bad actions than sending a well-chosen fraction. In our simulation, reviewing everything missed 17.3 of about 30 bad actions and a threshold at the theoretical optimum missed 11.2, with two-thirds fewer reviews. The theory gives a unique best review rate and an interpretable rule for the marginal action, and it predicted simulated performance to within 0.016. A load controller did not beat a tuned threshold, a rough fatigue estimate was enough, and the value of breaks depends on a modelling question that only studies with real reviewers can settle. The next steps are to measure fatigue in real review tasks, to estimate it online with cheaper audits, and to extend the objective to unequal severities.

Parameters

Shift 480 min; $a = 1.6/\text{min}$; $\pi = 0.04$; scorer $\mu = 1.5$ (AUC 0.86); capacity $C = 2/\text{min}$; queue $Q_{\max} = 20$; $h_0 = 0.92$; $\phi = 0.7$; fatigue memory $\beta = 0.05/\text{min}$. Slow model: weight

0.5; slow state gain 0.004 per unit busy, recovery 0.15 per minute during a break and 0.002 per idle minute otherwise.

Adaptive Controller: gain 0.4, smoothing 0.1. Tuning used 400 separate runs; evaluation used 1,000 runs with shared random numbers. Code and figures are in the supplementary archive. All figures were generated from code at 300 dpi and printed at their stored width.

References

- [1]. N. H. Mackworth. The breakdown of vigilance during prolonged visual search. *Quarterly J. Experimental Psychology*, 1(1):6–21, 1948.
- [2]. J. S. Warm, R. Parasuraman, G. Matthews. Vigilance requires hard mental work and is stressful. *Human Factors*, 50(3):433–441, 2008.
- [3]. J. S. Ancker et al. Effects of workload, work complexity, and repeated alerts on alert fatigue in a clinical decision support system. *BMC Medical Informatics and Decision Making*, 17:36, 2017.
- [4]. R. Parasuraman, V. Riley. Humans and automation: use, mis-use, disuse, abuse. *Human Factors*, 39(2):230–253, 1997.
- [5]. M. L. Cummings. Automation bias in intelligent time critical decision support systems. In *AIAA 1st Intelligent Systems Technical Conference*, 2004.
- [6]. D. Madras, T. Pitassi, R. Zemel. Predict responsibly: improving fairness and accuracy by learning to defer. In *NeurIPS*, 2018.
- [7]. H. Mozannar, D. Sontag. Consistent estimators for learning to defer to an expert. In *ICML*, 2020.
- [8]. D. M. Green, J. A. Swets. *Signal Detection Theory and Psychophysics*. Wiley, 1966.
- [9]. J. D. C. Little. A proof for the queuing formula $L = \lambda W$. *Operations Research*, 9(3):383–387, 1961.

How to Cite:

Venkata Krishna Reddy Atchi , " Cognitive Fatigue in Human-AI Oversight : Evaluating the Review of Agent Actions " , International Journal of Computer Science, Engineering and Artificial Intelligence , Vol. 3, Issue. 4 , pp 1 – 9 , October 2026.

Cross Ref DOI : <https://doi.org/10.63328/IJCSEAI-V3RI4P1>

Declaration

Conflicts of Interest: The authors declare no conflict of interest.

Author Contribution: All authors wrote the main manuscript text and also consent to the submission.

Ethical approval: Not applicable.

Consent to Participate: All authors consent to participate.

Funding: Not applicable, and No funding was received

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Personal Statement: We declare with our best of knowledge that this research work is purely Original Work and No third party material used in this article drafting. If any such kind material found in further online publication, we are responsible only for any judicial and copyright issues.

Generative AI Statement: The authors declare that generative AI was not used in the preparation or creation of this manuscript. The alternative text (alt text) accompanying the figures in this article was generated by Frontiers with the assistance of artificial intelligence. Reasonable efforts have been made to ensure the accuracy and appropriateness of the generated alt text, including review by the authors wherever possible. If you identify any inaccuracies or issues, please contact the editorial team.

Publisher's Note: The statements, opinions, and claims presented in this article are exclusively those of the authors and do not necessarily reflect the views of their affiliated institutions, the publisher, editors, or reviewers. Any products discussed or evaluated, as well as any claims made by their manufacturers, are not warranted, approved, or endorsed by the publisher.

Acknowledgements

We thank everyone who inspired our work