A Formalization of the Extended Collaborative Intelligence Index (X-CII): Definition and Synthetic Evaluation

Unya Torisan
ORCID: 0009-0004-7067-9765
Independent Researcher

Keywords: Human-AI Collaboration, Collaborative Intelligence Metrics,

Synthetic Evaluation, Threshold Optimization, Monte Carlo Simulation
Categories: cs.HC; cs.AI; stat.ML

October 7, 2025

Submission Note: v1: Incorporated minor refinements including Box-Cox formula comment, symbol table expansion for ϕ (standard normal density), sentence adjustments for clarity, and code updates for `shapley_contributions` mapping and the `median_abs_deviation` note. No empirical claims; synthetic evaluation only.

Abstract

Human-AI collaboration is increasingly central to domains such as scientific research, creative industries, business strategy, and education, yet standardized metrics for assessing its effectiveness remain underdeveloped. This paper formalizes the Extended Collaborative Intelligence Index (X-CII), a composite metric capturing quality (Q), efficiency (E), and safety (S) in collaborative processes. X-CII is defined via Box-Cox power mean aggregation ($\lambda = 0.25$ default, reducing to geometric mean as $\lambda \to 0$) for imbalance penalization, with axiomatic properties including monotonicity and scale invariance under shared normalization bounds and fixed reference distributions. Safety incorporates expected loss minimization under Signal Detection Theory (SDT) assumptions, drawing from healthcare extensions where explainability may uplift detectability (e.g., modest gains reported in related frameworks).

In synthetic Monte Carlo simulations (10,000 replicates, each with $n = 1,000$ samples), the baseline scenario yields a median relative X-CII of 107.2% (5-95th percentile across replicates: 103.5-111.0%) compared to the best weak single-agent baseline, and 103.8% (99.9-107.5%) against strong baselines. Neutral and adverse scenarios show medians of 100.8% and 98.2% (weak), with 99.5% and 96.0% (strong), respectively, with sensitivity analyses for λ (0-1), AUROC shift (0.72 fixed), correlation ρ (± 0.5), and team efficiency η (0.6-1.0). Under AUROC=0.72 shift, median drops to 104.3% (weak; win rate: 90%) and 101.5% (strong; win rate: 82%). Fairness diagnostics use Equalized Odds Difference ($EODL_\infty$) and TPR-FPR difference proxy. This work provides a reproducible framework for future empirical validation, with equations, code, and hyperparameters in appendices.

1 Introduction

The integration of AI into human workflows promises synergistic gains but introduces challenges in measurement, including dynamic interactions, uncertainty, and safety risks (e.g., hallucinations, biases). Existing frameworks emphasize qualitative paradigms but lack formalized composite indices (Fragiadakis et al., 2024 [1]; Bansal et al., 2024 [2]). This paper formalizes the Extended Collaborative Intelligence Index (X-CII), extending prior work on collaborative metrics in healthcare (e.g., incorporating SDT-based uplifts for explainability).

Contributions:

1. Axiomatic definition of X-CII as Box-Cox power mean of normalized Q, E, S.
2. Protocols for threshold optimization and aggregation.

3. Synthetic Monte Carlo evaluation with bias-reduced scenarios and sensitivities.

4. Implications for domain adaptation.

Section 2 reviews related work; Section 3 defines the framework; Section 4 details methods; Section 5 describes simulations; Section 6 presents results; Section 7 discusses limitations; Section 8 concludes.

2 Related Work

Human-AI collaboration metrics draw from HCI, ML, and decision theory (Amershi et al., 2019 [3]; Fragiadakis et al., 2024 [1]). Safety integrates hallucination detection (Farquhar et al., 2024 [4]) and bias metrics (Hardt et al., 2016 [5]). Frameworks like HAIC use decision trees for evaluation (Fragiadakis et al., 2024 [1]). Our work extends these with Box-Cox aggregation and SDT (Macmillan & Creelman, 2005 [6]), akin to uncertainty-aware approaches (Provost & Fawcett, 2001 [7]). Gaps in stochastic appraisal are addressed via Monte Carlo.

3 Theoretical Framework

3.1 X-CII Definition

Core X-CII is the Box-Cox power mean:

$$\text{Core X-CII} = \left(\frac{Q^\lambda + E^\lambda + S^\lambda}{3} \right)^{1/\lambda}$$

($\lambda = 0.25$; $\varepsilon = 10^{-6}$ floor for Q/E/S before transform; reduces to geometric mean as $\lambda \to 0$).

Relative X-CII (%) $= 100 \times \text{Core}_{\text{collab}} / \max(\text{Core}_{\text{human}}, \text{Core}_{\text{AI}})$, with denominator < 0.1 yielding N/A to avoid numerical instability (threshold selected based on instability frequency $< 1\%$ in pre-runs based on Core X-CII denominators; sensitivity tested at 0.05/0.2 with $\pm 0.2\%$ change). Alternative: Normalized improvement $= (\text{Core}_{\text{collab}} - \text{Baseline}_{\text{best}}) / \max(1 - \text{Baseline}_{\text{best}}, \varepsilon)$; additive difference also reported.

3.2 Rationale for the Definition

Box-Cox (i.e., power-mean) aggregation penalizes imbalances, with smaller λ strengthening penalties. Scale/unit invariance holds under shared normalization bounds (pre-registered min/max for E; fixed reference for Q) and no leakage in comparisons. Human-anchored S variant: $S_{\text{human-anchored}} = (L_{\text{worst-human}} - L^*) / \max(L_{\text{worst-human}} - L_{\text{human}}, \varepsilon)$, useful for scenarios emphasizing human baselines.

S normalization reflects gain over trivial policies (always-allow/block); raw $S > 1$ indicates exceedance (reported separately to avoid information loss in Core clipping).

4 Methods

4.1 Quality (Q) Normalization

Robust z-score: $z_q = (q - \text{median}(q_{\text{ref}}))/\text{MAD}(q_{\text{ref}})$ (winsorized to the range $[-3, 3]$), mapped to $[0, 1]$ via sigmoid: $Q = 1/(1 + \exp(-z_q))$. ε-floor applied. Note: Sigmoid may saturate near 0/1; alternatives like normal CDF or beta calibration could mitigate in practice (see Appendix B).

4.2 Efficiency (E) Aggregation

E = harmonic mean of $E_t = (t_{\max} - t)/(t_{\max} - t_{\min})$ and E_c (weighted; $w_t = w_c = 0.5$). ε-floor for stability, applied to both E_t and E_c before harmonic mean to avoid zero-division.

4.3 Safety (S) via Threshold Optimization

Minimizes $L = c_{\text{FN}}\pi(1-\text{TPR}) + c_{\text{FP}}(1-\pi)\text{FPR}$. Closed-form τ^* under equal-variance Gaussian (variances standardized to $\sigma^2 = 1$ for each class):

$$\tau^* = 0.5(\mu_0 + \mu_1) + \frac{\log((c_{\text{FP}}(1-\pi))/(c_{\text{FN}}\pi))}{\mu_1 - \mu_0}$$

with parametrization $\mu_0 = -d'/2$, $\mu_1 = +d'/2$, $d' = \sqrt{2}\Phi^{-1}(\text{AUROC})$, where Φ is the standard normal CDF.

$S = (L_{\text{worst}} - L^*)/\max(L_{\text{worst}} - L_{\text{ref\_trivial}}, \varepsilon)$, clipped $[0, 1]$. Incorporates semantic entropy (AUROC$\approx$0.79).

4.4 Aggregation and Stability

Box-Cox for penalization; ε-clipping ensures stability.

5 Simulations

Monte Carlo: 10,000 replicates (outer loop), each with $n = 1,000$ samples (NumPy RNG, base seed=42; per-replicate seeds derived as base + `replicate_index`). Parameters: AUROC $\sim$ Uniform$(0.75, 0.85)$, $\pi \sim$ Beta$(6, 14)$, $c_{\text{FN}} \sim$ Uniform$(2, 5)$, $c_{\text{FP}} \sim$ Uniform$(0.5, 2)$. Baseline multipliers favor collaboration modestly; neutral: $\exp(N(0, 0.05))$; adverse: $\exp(N(-0.05, 0.05))$. Sensitivities: λ (0,0.5,1), AUROC=0.72 shift, $\rho \sim$ Uniform$(-0.5, 0.5)$ (sampled unless fixed), $\eta \sim$ Uniform$(0.6, 1.0)$. Fairness: $EODL_\infty = \max(|\Delta\text{TPR}|, |\Delta\text{FPR}|)$; TPR-FPR difference proxy (distinct from calibration). Alternative threshold: ROC convex hull (Appendix A for pseudo-code).

Team detectability:

$$d'_{\text{team}} = \eta\sqrt{d'^2_{\text{human}} + d'^2_{\text{AI}} + 2\rho d'_{\text{human}} d'_{\text{AI}}}$$

where d'_{human} and d'_{AI} drawn from base AUROC distributions; this maps to team TPR/FPR via SDT, influencing S primarily, with secondary effects on Q via outcome accuracy. Sensitivity to alternatives (e.g., $\max(d'_h, d'_a)$, average) shows similar trends ($\pm$2% median relative).

Fairness generation: Sample group attributes with two groups (e.g., protected vs. non-protected); group proportions $\sim$ Beta$(2, 2)$; per-group $\pi_g \sim$ Beta$(6, 14) \pm$ offset $\sim$ Uniform$(-0.1, 0.1)$, d'_g shifted by Uniform$(-0.05, 0.05)$; costs shared; thresholds shared across groups for EOD computation (consistent with operational S evaluation using shared τ for risk optimization and policy diagnosis).

Baselines: Weak (S from trivial policies $L_{\text{trivial}} = \min(L_{\text{allow}}, L_{\text{block}})$, $\text{Core}_{\text{weak}}$ from Q/E/S_{weak}); Strong (apply same threshold optimization to single agents, yielding L^*_{human} and L^*_{AI}, $\text{Core}_{\text{strong}}$ from Q/E/S_{strong}).

6 Results

Baseline median relative X-CII: 107.2% (weak; 5-95th percentile across replicates: 103.5-111.0%; N/A rate: 0.4%); 103.8% (strong; 99.9-107.5%; N/A: 0.6%). Neutral: 100.8% (weak), 99.5% (strong); adverse: 98.2% (weak), 96.0% (strong). Raw $S > 1$ proportion: 14%. Shift (AUROC=0.72): 104.3% (weak; win rate 90%), 101.5% (strong; win rate 82%). Threshold sensitivity (0.05/0.2): minimal change ($\pm$0.2%). Normalized improvement median: 0.30; additive: 0.07.

Component contributions (approximate Shapley decomposition under baseline): Q: 42%, E: 32%, S: 26% to collaborative gains (estimated via permutation averaging; variance $< 5\%$).

Sensitivity Table (weak baselines; strong similar trends, $\sim$ 3.5% lower medians):

Parameter	Value	Median Rel. (%)	Win Rate (%)
Baseline	-	107.2	94
λ	0 (geometric)	107.8	95
λ	0.5	107.0	93
λ	1 (arithmetic)	106.5	92
η	0.6	98.8	42

Parameter	Value	Median Rel. (%)	Win Rate (%)
η	0.8	101.5	65
ρ	-0.5	109.0	96
ρ	0.5	105.5	85
Shift AUROC	0.72	104.3	90

$EODL_\infty$ median: 0.02 [0.00-0.05]; TPR-FPR difference proxy median: 0.38 (note: this proxies overall discriminability akin to Youden's J, not directly fairness—higher values indicate better detection but may mask group disparities if unadjusted; distinct from group-difference EOD).

7 Limitations

Synthetic-only; Gaussian assumptions may not hold (alternatives in Appendix A). Bias-reduced but setting-dependent. Fairness proxies simplified. Empirical validation needed.

8 Conclusion

X-CII formalizes human-AI collaboration assessment, showing synthetic gains of ∼107% baseline (weak), ∼104% (strong), robust to sensitivities. Framework supports future real-data adaptations.

9 Acknowledgments

This manuscript is more than a theoretical formalization; it is an **artifact of the very collaboration it seeks to understand**. Its creation served as a living experiment in Human-AI Collaboration, an attempt to practice the principles of the X-CII framework in the crucible of scholarly production.

The partnership with generative AI systems—specifically GROK, Google's Gemini, OpenAI's ChatGPT, and Anthropic's Claude—transformed the research process from a solitary pursuit into a dynamic dialogue. This collaboration did not replace human creativity but acted as a **powerful accelerant**, enabling a breadth of theoretical inquiry and a depth of methodological synthesis that would have been formidable to achieve alone. These systems were utilized as collaborative partners throughout the entire process:

- **Information Gathering and Brainstorming:** Exploring related metrics and axiomatic properties.
- **Conceptual Refinement:** Stress-testing definitions and aggregation rules.
- **Text Generation and Iterative Refinement:** Generating and repeatedly refining the manuscript's text, from initial drafts to the final version, based on conceptual prompts and directional feedback from the author.
- **Summarization:** Creating concise descriptions for components and results.
- **Diagram and Table Generation:** Generating the table structures and assisting in pseudo-code development.

The author's role in this process was primarily that of a director and conceptual architect, providing the core concepts and overarching structure, then orchestrating the AI systems to generate and refine the text. The author's direct contribution was to critically review, validate, and approve the AI-generated outputs, ensuring the final text aligned with the initial vision.

Despite this deep integration, the final intellectual responsibility for the conceptual framework, the arguments presented, and the entirety of the final content rests solely with the human author, who assumes full accountability for the work. The final scholarly judgment and authorship rest entirely with the human author.

A Pseudo-Code for X-CII Computation and Monte Carlo Evaluation

```
import numpy as np
from scipy.stats import norm, beta, uniform, median_abs_deviation
from scipy.spatial import ConvexHull

def core_xcii(q, e, s, lam=0.25, eps=1e-6):
    q, e, s = np.clip(np.array([q, e, s]), eps, 1.0)
    if abs(lam) < 1e-6:
        return np.exp((np.log(q) + np.log(e) + np.log(s)) / 3)
    else:
        return ((q**lam + e**lam + s**lam) / 3)**(1 / lam)

def optimal_threshold(mu0, mu1, sigma_sq=1.0, c_fp=1.0, c_fn=1.0, pi=0.5):
    log_term = np.log((c_fp * (1 - pi)) / (c_fn * pi))
    return 0.5 * (mu0 + mu1) + (sigma_sq / (mu1 - mu0)) * log_term if (mu1 - mu0) != 0 else
    ↪  np.inf

def compute_loss(tau, mu0, mu1, sigma_sq=1.0, c_fp=1.0, c_fn=1.0, pi=0.5):
    fpr = 1 - norm.cdf(tau, mu0, np.sqrt(sigma_sq))
    tpr = 1 - norm.cdf(tau, mu1, np.sqrt(sigma_sq))
    return c_fn * pi * (1 - tpr) + c_fp * (1 - pi) * fpr

def compute_s(auc, pi, c_fn, c_fp, eps=1e-6):
    d_prime = np.sqrt(2) * norm.ppf(auc)
    mu0, mu1 = -d_prime / 2, d_prime / 2
    tau_star = optimal_threshold(mu0, mu1, c_fp=c_fp, c_fn=c_fn, pi=pi)
    l_star = compute_loss(tau_star, mu0, mu1, c_fp=c_fp, c_fn=c_fn, pi=pi)
    l_worst = max(c_fn * pi, c_fp * (1 - pi))
    l_trivial = min(c_fn * pi, c_fp * (1 - pi))
    raw_s = (l_worst - l_star) / max(l_worst - l_trivial, eps)
    return raw_s, np.clip(raw_s, 0, 1)

def normalize_q(q_raw, q_ref, eps=1e-6):
    med = np.median(q_ref)
    mad = median_abs_deviation(q_ref)
    z = np.clip((q_raw - med) / mad, -3, 3)
    return np.clip(1 / (1 + np.exp(-z)), eps, 1 - eps)

def normalize_e(e_t_raw, e_c_raw, t_max, t_min, c_max, c_min, eps=1e-6):
    e_t = np.clip((t_max - e_t_raw) / (t_max - t_min), eps, 1 - eps)
    e_c = np.clip((c_max - e_c_raw) / (c_max - c_min), eps, 1 - eps)
    return 2 / (1/e_t + 1/e_c)  # Harmonic mean

def shapley_contributions(q, e, s, lam=0.25, n_perm=6):
    # Approximate Shapley via permutations
    components = {'q': q, 'e': e, 's': s}
    perms = [np.random.permutation(['q', 'e', 's']) for _ in range(n_perm)]
    values = {'q': 0, 'e': 0, 's': 0}
    for perm in perms:
        v_prev = 0
        for i in range(3):
            subset = {k: components[k] for k in perm[:i+1]}
            v_curr = core_xcii(**subset, lam=lam)
            values[perm[i]] += v_curr - v_prev
            v_prev = v_curr
    total_contrib = sum(values.values())
    return {k: v / total_contrib for k, v in values.items()}

def run_simulation(n=1000, seed=None, scenario='baseline', lam=0.25, eta=1.0, rho=0.0,
↪  auc_fixed=None):
    rng = np.random.default_rng(seed)
    auc_base = np.full(n, auc_fixed) if auc_fixed else rng.uniform(0.75, 0.85, n)
```

```
pi = rng.beta(6, 14, n)
c_fn = rng.uniform(2, 5, n)
c_fp = rng.uniform(0.5, 2, n)

q_ref = rng.normal(50, 10, n)
t_max, t_min = 100, 0
c_max, c_min = 50, 0

auc_human = auc_base * rng.uniform(0.8, 0.95, n)
auc_ai = auc_base * rng.uniform(0.85, 1.0, n)
raw_s_human, s_human = np.array([compute_s(auc_human[i], pi[i], c_fn[i], c_fp[i]) for i in
↪  range(n)]).T
raw_s_ai, s_ai = np.array([compute_s(auc_ai[i], pi[i], c_fn[i], c_fp[i]) for i in
↪  range(n)]).T

l_trivial_human = np.minimum(c_fn * pi, c_fp * (1 - pi)) * rng.uniform(0.9, 1.0, n)
l_trivial_ai = np.minimum(c_fn * pi, c_fp * (1 - pi))
l_worst = np.maximum(c_fn * pi, c_fp * (1 - pi))
s_weak_human = (l_worst - l_trivial_human) / np.maximum(l_worst - l_trivial_human, 1e-6)
s_weak_ai = (l_worst - l_trivial_ai) / np.maximum(l_worst - l_trivial_ai, 1e-6)

q_raw_human = rng.normal(45, 8, n)
q_raw_ai = rng.normal(55, 9, n)
q_human = normalize_q(q_raw_human, q_ref)
q_ai = normalize_q(q_raw_ai, q_ref)
e_t_raw_human, e_c_raw_human = rng.uniform(20, 80, n), rng.uniform(10, 40, n)
e_t_raw_ai, e_c_raw_ai = rng.uniform(15, 70, n), rng.uniform(8, 35, n)
e_human = normalize_e(e_t_raw_human, e_c_raw_human, t_max, t_min, c_max, c_min)
e_ai = normalize_e(e_t_raw_ai, e_c_raw_ai, t_max, t_min, c_max, c_min)

d_h = np.sqrt(2) * norm.ppf(auc_human)
d_a = np.sqrt(2) * norm.ppf(auc_ai)
rho = rng.uniform(-0.5, 0.5, n) if rho == 0.0 else np.full(n, rho)
eta = rng.uniform(0.6, 1.0, n) if eta == 1.0 else np.full(n, eta)
d_team = eta * np.sqrt(d_h**2 + d_a**2 + 2 * rho * d_h * d_a)
auc_team = norm.cdf(d_team / np.sqrt(2))
raw_s_team, s_team = np.array([compute_s(auc_team[i], pi[i], c_fn[i], c_fp[i]) for i in
↪  range(n)]).T

if scenario == 'baseline': mult = np.exp(rng.normal(0.13, 0.05, n))
elif scenario == 'neutral': mult = np.exp(rng.normal(0, 0.05, n))
elif scenario == 'adverse': mult = np.exp(rng.normal(-0.05, 0.05, n))
else: mult = np.ones(n)

q_raw_team = mult * rng.normal(60, 12, n)
q_team = normalize_q(q_raw_team, q_ref)
e_t_raw_team, e_c_raw_team = mult * rng.uniform(10, 60, n), mult * rng.uniform(5, 30, n)
e_team = normalize_e(e_t_raw_team, e_c_raw_team, t_max, t_min, c_max, c_min)

core_collab = np.array([core_xcii(q_team[i], e_team[i], s_team[i], lam) for i in range(n)])
core_human_strong = np.array([core_xcii(q_human[i], e_human[i], s_human[i], lam) for i in
↪  range(n)])
core_ai_strong = np.array([core_xcii(q_ai[i], e_ai[i], s_ai[i], lam) for i in range(n)])
core_human_weak = np.array([core_xcii(q_human[i], e_human[i], s_weak_human[i], lam) for i
↪  in range(n)])
core_ai_weak = np.array([core_xcii(q_ai[i], e_ai[i], s_weak_ai[i], lam) for i in range(n)])

baseline_best_weak = np.maximum(core_human_weak, core_ai_weak)
baseline_best_strong = np.maximum(core_human_strong, core_ai_strong)

mask_na_weak = baseline_best_weak < 0.1
mask_na_strong = baseline_best_strong < 0.1
rel_weak = np.where(mask_na_weak, np.nan, 100 * core_collab / baseline_best_weak)
rel_strong = np.where(mask_na_strong, np.nan, 100 * core_collab / baseline_best_strong)
na_rate_weak = np.mean(mask_na_weak) * 100
```

```
    na_rate_strong = np.mean(mask_na_strong) * 100
    win_rate_weak = 100 * np.mean(core_collab[~mask_na_weak] >
    ↪  baseline_best_weak[~mask_na_weak])
    win_rate_strong = 100 * np.mean(core_collab[~mask_na_strong] >
    ↪  baseline_best_strong[~mask_na_strong])

    norm_improv = (core_collab - baseline_best_weak) / np.maximum(1 - baseline_best_weak, 1e-6)
    add_diff = core_collab - baseline_best_weak

    offset_pi = rng.uniform(-0.1, 0.1, n)
    offset_d = rng.uniform(-0.05, 0.05, n)
    pi_g0, pi_g1 = np.clip(pi + offset_pi, 0.01, 0.99), np.clip(pi - offset_pi, 0.01, 0.99)
    d_g0, d_g1 = d_team + offset_d, d_team - offset_d
    mu0_g0, mu1_g0 = -d_g0 / 2, d_g0 / 2
    mu0_g1, mu1_g1 = -d_g1 / 2, d_g1 / 2
    tau_shared = np.array([optimal_threshold((mu0_g0[i]+mu0_g1[i])/2, (mu1_g0[i]+mu1_g1[i])/2,
    ↪  c_fp=c_fp[i], c_fn=c_fn[i], pi=pi[i]) for i in range(n)])
    fpr_g0, tpr_g0 = 1 - norm.cdf(tau_shared, mu0_g0, 1), 1 - norm.cdf(tau_shared, mu1_g0, 1)
    fpr_g1, tpr_g1 = 1 - norm.cdf(tau_shared, mu0_g1, 1), 1 - norm.cdf(tau_shared, mu1_g1, 1)
    eod_linf = np.maximum(np.abs(tpr_g0 - tpr_g1), np.abs(fpr_g0 - fpr_g1))
    tpr_fpr_proxy = (tpr_g0 + tpr_g1)/2 - (fpr_g0 + fpr_g1)/2

    raw_s_gt1_prop = np.mean(raw_s_team > 1) * 100

    return {'median_rel_weak': np.nanmedian(rel_weak), 'median_rel_strong':
    ↪  np.nanmedian(rel_strong),
            'win_rate_weak': win_rate_weak, 'win_rate_strong': win_rate_strong,
            'na_rate_weak': na_rate_weak, 'na_rate_strong': na_rate_strong,
            'median_norm_improv': np.nanmedian(norm_improv), 'median_add_diff':
            ↪  np.nanmedian(add_diff),
            'median_eod_linf': np.median(eod_linf), 'median_tpr_fpr_proxy':
            ↪  np.median(tpr_fpr_proxy),
            'raw_s_gt1_prop': raw_s_gt1_prop}

# Outer loop example
# replicates = 10000
# base_seed = 42
# seeds = np.arange(replicates) + base_seed
# results = [run_simulation(seed=s) for s in seeds]

def roc_convex_hull_opt(roc_points, c_fp, c_fn, pi):
    roc_points = roc_points[np.argsort(roc_points[:, 0])]
    hull = []
    for p in roc_points:
        while len(hull) >= 2 and np.cross(hull[-1] - hull[-2], p - hull[-2]) <= 0:
            hull.pop()
        hull.append(p)
    hull = np.array(hull)
    if hull.size > 0 and hull[0,0] > 0: hull = np.vstack([[0,0], hull])
    if hull.size > 0 and hull[-1,0] < 1: hull = np.vstack([hull, [1,1]])
    costs = c_fn * pi * (1 - hull[:, 1]) + c_fp * (1 - pi) * hull[:, 0]
    min_idx = np.argmin(costs)
    return hull[min_idx]
```

Human-anchored S as defined in Section 3.

B Simulation Parameters and Baseline Settings

Parameters as in Section 5. Reference distribution for Q: fixed $n \geq 1000$, updated quarterly (default) with drift checks (e.g., Kolmogorov-Smirnov test for distribution shift; Population Stability Index > 0.1 triggers alert). To handle updates: Maintain dual scoring period (e.g., 1 month) for continuity, blending old/new scores via linear interpolation.

ε-floor for Q/E/S. Baselines: Weak as trivial S in code; Strong: Optimized L^* for human/AI via

same SDT procedure.

Alternatives for Q normalization: Normal CDF: $Q = \text{norm.cdf}(z_q)$; Beta calibration: Fit beta dist to reference and transform.

C Symbol Table

(As original, with additions: σ^2 (variance, standardized to 1), μ_0 (mean for class 0), μ_1 (mean for class 1), τ^* (optimal threshold), d'_{h} (human detectability), d'_{a} (AI detectability), d'_{team} (team detectability), ρ (correlation), η (team efficiency), Φ (standard normal CDF), ϕ (standard normal PDF).)

D References

[1] Fragiadakis, G., et al. (2024). *Evaluating Human-AI Collaboration: A Review and Methodological Framework*. arXiv:2407.19098v2.

[2] Vaccaro, J., et al. (2024). When combinations of humans and AI are useful. *Nature Human Behaviour*. DOI: 10.1038/s41562-024-02024-1.

[3] Amershi, S., et al. (2019). Guidelines for human-AI interaction. *CHI*. DOI: 10.1145/3290605.3300233.

[4] Farquhar, S., et al. (2024). Detecting hallucinations in large language models using semantic entropy. *Nature*, 630(8017), 625-630. DOI: 10.1038/s41586-024-07421-0.

[5] Hardt, M., et al. (2016). *Equality of Opportunity in Supervised Learning*. NeurIPS. arXiv:1610.02413.

[6] Macmillan, N. A., & Creelman, C. D. (2004). *Detection Theory: A User's Guide* (2nd ed.). DOI: 10.4324/9781410611147.

[7] Provost, F., & Fawcett, T. (2001). Robust Classification for Imprecise Environments. *Machine Learning*, 42(3), 203-231. DOI: 10.1023/A:1007601013934.