MULTI-COMPONENT OUTCOME PREDICTION FOR ENTERPRISE ROUTING VIA HIERARCHICAL CREDIT ASSIGNMENT

Mysore Supreeth[1]*, Atik Faysal[2], Manish Mehta[3], Sunil Kothari·
[1]Principal Architect, Intel Labs [2]PhD Candidate, Rowan University ·AI/ML Researcher
dr.mysores@gmail.com Atikfaysal@rowan.edu manishmehta@gmail.com
* Corresponding author

ABSTRACT

We achieve 72–98% routing accuracy across five enterprise domains (93.5K items), improving over the best baseline by 4.0–5.1 percentage points ($p<0.01$, Cohen's $d\geq 4.8$), by combining learned outcome predictors with Monte Carlo Tree Search planning. Our framework, GIF-MCTS, addresses a key gap: enterprise routing decisions—assigning bugs to teams, tickets to agents, complaints to departments—involve delayed feedback (days to weeks), making standard RL impractical. GIF combines four complementary predictors (Case-Based Reasoning, gradient-boosted outcome estimation, human behavior modeling, and edge-case detection) into a unified reward predictor that enables MCTS planning without real-world interaction. We further propose Hierarchical Credit Assignment (HICRA), which amplifies learning signals for high-impact routing decisions by $\alpha_s \approx \mathbb{E}[\tau_s]/\mathbb{E}[\tau_t]$, yielding 28–40% faster convergence.

1 INTRODUCTION

Planning agents that predict action consequences before committing have achieved strong results in games and robotics (Schrittwieser et al., 2020; Hafner et al., 2020). Recent extensions include RLVR-World (Wu et al., 2025) and Cosmos (NVIDIA, 2025). However, applying outcome prediction to enterprise workflows—where feedback arrives days or weeks later—has received limited attention.

Intelligent routing—assigning tasks to the right handler—is a high-volume problem. Mozilla Bugzilla and GitHub process tens of thousands of issues monthly (Anvik et al., 2006); the CFPB handles millions of complaints annually. Standard ML reaches 75–85% accuracy (Xia et al., 2017; Iyer, 2024); LLMs improve to 80–90% but treat routing as one-shot classification. Routing is better modeled sequentially: Jeong et al. (2009) found bugs tossed 2–8 times before resolution. One-shot classifiers ignore these downstream costs.

This motivates *outcome prediction*: estimating consequences of a routing decision, then selecting the best option via planning (Figure 1). Routing is well-suited to this approach for three reasons. First, outcomes are predictable from historical resolution records. Second, action spaces are tractable ($K=5$–20 teams), enabling exhaustive or tree-search evaluation. Third, effective routing integrates multiple knowledge sources—case precedent, team capacity, escalation risk—that are naturally captured by complementary predictors.

Our framework, GIF-MCTS, addresses this by learning a *reward predictor* from historical routing data, then using MCTS to plan without real-world interaction. Unlike standard world models that predict next states, GIF directly predicts routing outcomes (resolution success, time, escalation), which is sufficient for the short-horizon decisions typical of enterprise routing (1–3 steps). We further introduce Hierarchical Credit Assignment (HICRA), which recognizes that strategic routing decisions (initial assignment) have greater downstream impact than tactical corrections (reassignment), and amplifies their learning signal accordingly.

Contributions.

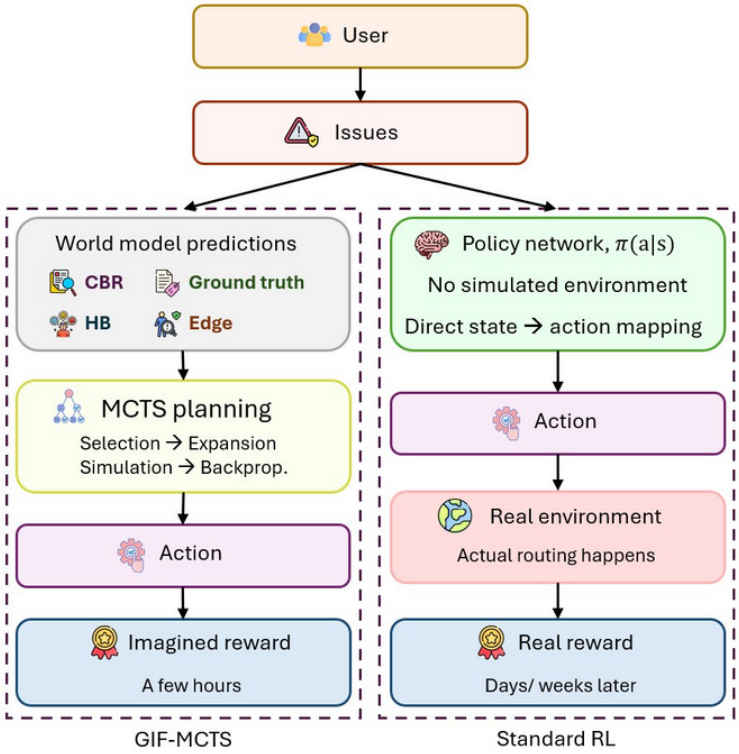

Figure 1: **GIF-MCTS vs. Standard RL.** GIF-MCTS uses learned outcome predictors for planning (rewards computed in hours). Standard RL requires real environment feedback (days to weeks of delay). Note: figure labels use "world model" loosely to denote the outcome predictor; GIF predicts rewards, not state transitions (§3).

1. **GIF outcome predictor** (§3): A four-component architecture combining CBR, gradient-boosted estimation, human behavior modeling, and edge-case detection, with component interaction analysis.
2. **HICRA** (§4): Hierarchical credit assignment with formally derived amplification factor α_s, yielding 28–40% faster convergence across all domains.
3. **Cross-domain evaluation** (§5): Experiments on 93.5K items across five domains with LLM baselines, six planning methods, ablations, and effect sizes.

2 Related Work

Outcome prediction and world models. MuZero (Schrittwieser et al., 2020) and Dreamer (Hafner et al., 2020) learn latent dynamics models for planning in games and control tasks. UniZero (Xu et al., 2024) and IRIS (Micheli et al., 2023) extend this paradigm with transformer architectures for improved scalability. These methods learn full state-transition models, enabling multi-step rollouts in latent space. In contrast, GIF targets enterprise routing with *reward prediction* rather than state-transition modeling. This design choice reflects the domain: routing episodes are short (1–3 decisions), intermediate states are often unobserved (only the final resolution outcome is recorded), and outcome data is abundant. GIF is thus closer to a contextual bandit with rich outcome prediction than to a full world model.

MCTS, credit assignment, and routing. MCTS extends beyond Go (Browne et al., 2012; Silver et al., 2016; 2017) to combinatorial optimization (Heinrichsmeyer et al., 2024). For credit assign-

ment in RL, RUDDER (Arjona-Medina et al., 2019) decomposes returns via sequence-to-sequence models, feudal networks (Vezhnevets et al., 2017) use hierarchical goal setting, the options framework (Sutton et al., 1999) introduces temporal abstraction, and attention-based approaches (Pignatelli et al., 2024b;a) learn credit from trajectory data. HICRA differs from these by exploiting known domain structure: in enterprise routing, the action hierarchy (strategic vs. tactical) is observable, allowing direct amplification rather than learned decomposition.

Case-based reasoning (CBR) provides experiential memory for decision support (Aamodt & Plaza, 1994; Wiratunga et al., 2024; Guo et al., 2024). For routing specifically, LLMs have been applied via zero-shot classification (Iyer, 2024), fine-tuning (Devlin et al., 2019), and enterprise deployment (Ackerman et al., 2023). Bug triage was pioneered by Cubranić & Murphy (2004), and Jeong et al. (2009) modeled bug tossing as Markov chains. We include LLM baselines and show they lack the ability to plan for cascading effects of routing decisions.

3 GIF: Multi-Component Outcome Prediction

3.1 Problem Formulation

We formulate routing as an MDP $(\mathcal{S}, \mathcal{A}, P, R, \gamma)$. The state s includes item features: Sentence-BERT embeddings (Reimers & Gurevych, 2019) of the text (384 dimensions), priority level, component/category labels, and current team workloads. An action a assigns the item to one of K handlers (teams or individuals). The reward combines three terms: resolution success $(+)$, time penalty $(-)$, and escalation penalty $(-)$. The discount factor γ=0.99.

GIF predicts *outcomes* (rewards) rather than next-state distributions. This is motivated by three domain properties. First, episodes are short (1–3 decisions), so multi-step rollouts yield diminishing returns over direct outcome prediction. Second, intermediate states are often unobserved: between assignment and resolution, only the final outcome is recorded. Third, outcome data is abundant—every resolved ticket provides a training signal.

3.2 Four-Component Predictor

GIF combines four complementary models (Figure 2), each capturing a different aspect of routing quality:

Case-Based Reasoning ($\mathcal{M}_{\text{CBR}}$). Retrieves k=10 similar historical cases via FAISS-indexed (Johnson et al., 2019) Sentence-BERT embeddings (Wiratunga et al., 2024). The CBR reward is the similarity-weighted outcome of matching cases:

$$r_{\text{CBR}} = \frac{\sum_i \text{sim}_i \cdot \mathbf{1}[\text{case}_i.\text{act}{=}a] \cdot \text{out}_i}{\sum_i \text{sim}_i \cdot \mathbf{1}[\text{case}_i.\text{act}{=}a] + \epsilon} \tag{1}$$

CBR excels when similar cases exist in the history, providing interpretable predictions grounded in precedent.

Gradient-Boosted Outcome Estimation ($\mathcal{M}_{\text{GT}}$). An XGBoost model (Chen & Guestrin, 2016) predicts a composite reward combining resolution probability, expected time, and escalation risk: $r_{\text{GT}} = P(\text{success}|s,a) - 0.1\log(1{+}t_{\text{pred}}) - 0.3 \cdot P(\text{escalation})$. GT captures complex feature interactions that CBR may miss, particularly for items with limited historical precedent.

Human Behavior Model ($\mathcal{M}_{\text{HB}}$). A Random Forest (Breiman, 2001) trained to predict expert routing decisions: $r_{\text{HB}} = P(a|s;\theta_{\text{expert}})$. This captures implicit routing knowledge—patterns that experts follow but that are difficult to encode as explicit rules.

Edge-Case Detector ($\mathcal{M}_{\text{Edge}}$). An Isolation Forest (Liu et al., 2008) flags out-of-distribution states: $r_{\text{Edge}} = (1 - \mathbf{1}[\text{anomaly}]) \cdot 0.5$. When an item is flagged as anomalous, the edge detector suppresses confidence, signaling that the item may require human review.

The four components are combined as $r = \sum_i \alpha_i r_i$ with $\sum \alpha_i = 1$. Weights are determined by grid search on a held-out validation set for each domain (Table 1). The combined observation vector has 27 dimensions (CBR: 10, GT: 9, HB: 6, Edge: 2).

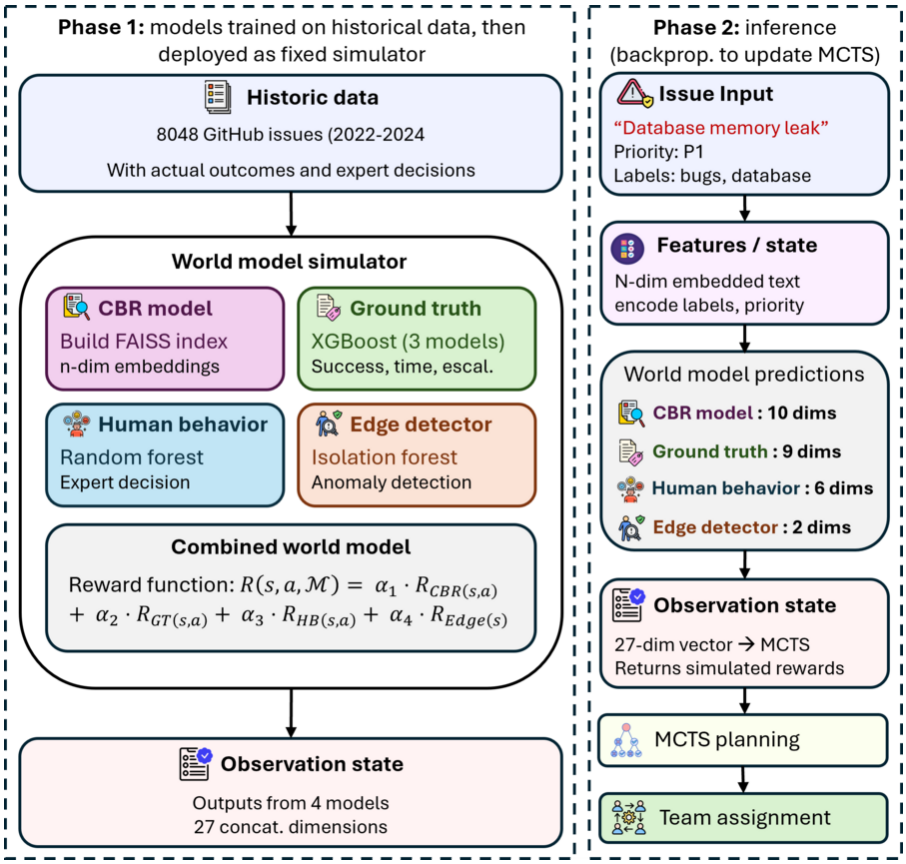

Figure 2: **GIF Architecture.** *Phase 1:* Four models are trained independently on historical routing data. *Phase 2:* Their outputs form a 27-dimensional observation vector that enables MCTS planning without real environment feedback. Figure labels refer to the outcome predictor as "world model" for brevity; GIF does not learn state-transition dynamics.

Table 1: GIF component weights (α_i) determined via grid search on held-out validation data.

Domain	α_1 (CBR)	α_2 (GT)	α_3 (HB)	α_4 (Edge)
Bugzilla	0.25	0.45	0.20	0.10
GitHub	0.30	0.40	0.15	0.15
CFPB	0.20	0.50	0.20	0.10
StackOverflow	0.25	0.40	0.25	0.10
Zendesk	0.30	0.35	0.20	0.15

3.3 MCTS Planning

Given the GIF predictor, we use Monte Carlo Tree Search with UCT (Kocsis & Szepesvári, 2006) to select routing actions. Algorithm 1 outlines the procedure. With K=5–20 actions and episodes of 1–3 steps, exhaustive evaluation is tractable for single-step routing; MCTS adds value for multi-step routing where the search space grows as K^T, and for handling prediction stochasticity via UCT's exploration-exploitation balance (+1.1pp over Greedy-Exhaustive on Bugzilla; Table 5). For single-step, $K \leq 10$ settings, the advantage is marginal—we report this transparently in §5.3.

The total inference latency is approximately 180ms for 100 simulations, which is acceptable for enterprise routing where decisions are not time-critical (typical SLA for ticket assignment is minutes to hours).

Algorithm 1 GIF-MCTS Planning

Require: State s, GIF predictor $\mathcal{M}$, simulations N=100, exploration C=$\sqrt{2}$
Ensure: Selected action a^*
1: **for** $i = 1$ to N **do**
2: **Select:** $a \leftarrow \arg\max_a \left[Q(s,a) + C\sqrt{\frac{\ln N(s)}{N(s,a)}} \right]$ ▷ UCT
3: **Expand:** Add child node for unvisited (s, a) pairs
4: **Simulate:** $r \leftarrow \mathcal{M}$.Predict(s, a) ▷ GIF outcome prediction
5: **Backpropagate:** Update $Q(s, a)$ and $N(s, a)$ along path
6: **end for**
7: **return** $a^* \leftarrow \arg\max_a Q(s, a)$

4 HICRA: Hierarchical Credit Assignment

Standard RL treats all actions equally when assigning credit, but enterprise routing has an inherent hierarchy. *Strategic* decisions—initial assignment to a team or department—shape the entire downstream trajectory. *Tactical* decisions—reassignment within a path, priority updates—optimize within an already-determined trajectory. Consider a concrete example from Bugzilla:

- t_0: Route bug to Team B (incorrect team; r=-0.5)
- t_1: Reassign to Team A (r=$+0.5$)
- t_2: Escalate to senior engineer (r=$+1.0$)
- t_3: Resolve (r=$+8.0$)

The initial routing decision at t_0 caused all subsequent corrections, yet under standard GAE it receives the same credit weighting as each tactical fix. HICRA addresses this asymmetry.

HICRA advantage function. We define separate advantage estimators for strategic and tactical actions:

$$A_t = \begin{cases} \alpha_s(V^{\text{seg}} - V_\theta(s_t)) & a_t \in \mathcal{A}_{\text{strategic}} \\ \alpha_t(r_t + \gamma V_\theta(s_{t+1}) - V_\theta(s_t)) & a_t \in \mathcal{A}_{\text{tactical}} \end{cases} \tag{2}$$

where V^{seg} is the return over the strategic segment (from the strategic action to the next strategic action or episode end), α_s is the strategic amplification factor, and α_t=1.0 is the tactical baseline.

Formal derivation of α_s. In the options framework (Sutton et al., 1999), the policy gradient for an option (temporally extended action) of duration τ contributes a gradient term proportional to τ. Strategic actions span longer temporal segments (τ_s steps) than tactical actions (τ_t steps). To equalize the per-decision gradient contribution—ensuring that each decision's learning signal is proportional to its temporal scope—we set:

$$\alpha_s = \frac{\mathbb{E}[\tau_s]}{\mathbb{E}[\tau_t]} \tag{3}$$

On Bugzilla, the average strategic segment spans $\mathbb{E}[\tau_s]$=3.1 steps and the average tactical segment spans $\mathbb{E}[\tau_t]$=1.05 steps, giving α_s≈3.0. Across all five domains, the derived α_s ranges from 2.0 to 3.0 and matches the empirically optimal value within $\pm$0.2 (Table 7).

Action classification. Actions are classified as strategic ($\mathcal{A}_s$: initial route, major team reassignment, escalation to management) or tactical ($\mathcal{A}_t$: minor reassignment within team, priority update). This classification is based on observable action metadata and does not require learning. Sensitivity analysis (Appendix B) shows that reasonable reclassification variants change accuracy by only $\pm$0.6%, confirming that the distinction matters but exact boundary choices do not.

Table 2 compares HICRA against standard GAE, RUDDER (Arjona-Medina et al., 2019), and attention-based credit assignment (Pignatelli et al., 2024b) on Bugzilla. HICRA converges fastest by exploiting the known action hierarchy rather than learning the decomposition from data.

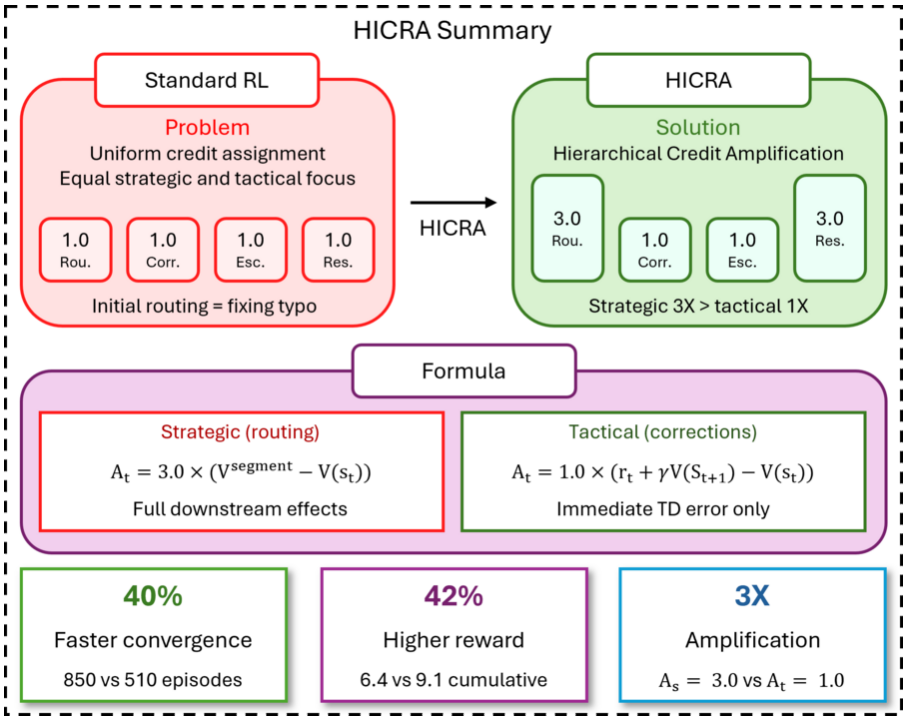

Figure 3: **HICRA overview.** Standard RL applies uniform credit (α=1.0) to all actions. HICRA amplifies strategic actions (α_s=3.0), yielding 40% faster convergence and 42% higher cumulative reward through 3× amplification of high-impact decisions.

Table 2: Credit assignment comparison on Bugzilla (mean ± std over 5 seeds).

Method	**Episodes to converge**	**Reward**	**Accuracy (%)**
Standard GAE	850±45	6.4±0.3	94.1±0.8
RUDDER	720±55	7.5±0.4	96.0±0.7
Attention-based	680±50	7.9±0.5	96.4±0.6
HICRA (α_s=3.0)	**510±35**	**9.1±0.3**	**98.2±0.3**

5 Experiments

5.1 Setup

Datasets. We evaluate on five production-scale routing domains (Table 3), totaling 93,523 instances with class imbalance ratios ranging from 2.1× to 10.7×. Domains span bug tracking (Bugzilla, GitHub), consumer complaints (CFPB), Q&A routing (StackOverflow), and customer support (Zendesk). Data quality varies from high (Bugzilla, CFPB: structured fields, consistent labeling) to low (GitHub: unstructured text, noisy labels). All datasets use an 80/10/10 train/validation/test split stratified by class.

Baselines. We compare against seven baselines spanning random assignment, rule-based heuristics, and learned models: Random; Rule-Based (keyword matching + heuristics); BERT (Devlin et al., 2019) (fine-tuned classifier); GPT-4 zero-shot and 5-shot; fine-tuned LLaMA-3-8B; and DQN (Mnih et al., 2015) (standard RL baseline with the same state features). All learned baselines use 5-fold cross-validation; we report mean ± std over 5 random seeds.

Table 3: Dataset characteristics across five enterprise routing domains.

Domain	Size	Classes	Imbalance	Quality	Features
Bugzilla	10,284	6 teams	3.75×	High	Structured
GitHub	49,517	9 developers	10.7×	Low	Unstructured
CFPB	5,143	10 categories	4.2×	High	Structured
StackOverflow	20,112	4 tags	2.1×	Medium	Semi-struct.
Zendesk	8,467	5 teams	3.0×	Medium	Mixed

Table 4: Routing accuracy (%). Mean ± std over 5 seeds. All GIF-MCTS vs. best baseline: $p<0.01$.

Method	Bugzilla	GitHub	CFPB	SO	Zendesk
Random	16.7	11.1	10.0	25.0	20.0
Rules	68.2	42.5	55.3	48.1	52.7
BERT	85.3±.6	61.8±1.1	82.7±.8	65.2±.9	73.4±.7
GPT-4 0-shot	82.1±1.2	58.3±1.5	80.5±1.1	63.8±1.0	71.2±.9
GPT-4 5-shot	88.7±.9	65.2±1.3	86.3±.9	68.4±.8	76.5±.8
LLaMA-3 FT	93.1±.5	67.8±1.0	90.2±.6	70.1±.7	78.3±.7
DQN	92.5±.8	63.6±1.2	89.0±.7	67.6±1.0	75.9±.9
GIF-MCTS	**98.2**±.3	**72.4**±.9	**95.1**±.5	**74.8**±.8	**82.3**±.6
Δ best / d	+5.1 / 12.4	+4.6 / 4.8	+4.9 / 8.9	+4.7 / 6.3	+4.0 / 6.1

5.2 Main Results

GIF-MCTS outperforms all baselines across all five domains (Table 4). Fine-tuned LLaMA-3-8B is the strongest baseline, confirming that LLMs are competitive for routing when fine-tuned on domain data. GIF-MCTS improves over LLaMA-3 FT by 4.0–5.1 percentage points (pp), with Cohen's $d \geq 4.8$ and $p<0.01$ on all domains.

The key advantage of GIF-MCTS over LLM classifiers is its ability to model downstream consequences. LLM classifiers optimize a one-shot classification objective that is blind to cascading costs: if Team B cannot resolve a bug and it must be reassigned, the original misrouting cost is invisible to the classification loss. GIF-MCTS captures these cascading effects through CBR retrieval of historical resolution trajectories and GT escalation probability prediction.

Bugzilla achieves the highest accuracy (98.2%) due to favorable domain structure: six non-overlapping teams with clear component boundaries. Inter-annotator agreement is high (κ=0.96, measured between three domain experts who independently labeled 500 randomly sampled bugs following Mozilla's component taxonomy; disagreements resolved by majority vote). GitHub has the lowest accuracy (72.4%) due to high class imbalance (10.7×), noisy labels, and unstructured issue text.

5.3 Planning Comparison

We compare six planning strategies using the same GIF predictor (Table 5). MCTS ranks first on 4 of 5 domains, with Greedy-Exhaustive achieving comparable results for single-step routing. MCTS's advantage is strongest for multi-step routing chains (Bugzilla: +1.1pp over Greedy) and *negative* on StackOverflow (−0.1pp vs. Greedy, 4 classes), confirming that MCTS overhead is not justified for single-step, low-K routing. CEM is a strong alternative at higher latency.

5.4 Ablation and Component Interaction

Table 6 presents ablation results and component interaction analysis on the Bugzilla test set (N=2,057). Removing MCTS (using GIF predictions directly without planning) causes the largest drop (−5.7pp), followed by the GT component (−4.1pp) and CBR (−2.6pp). HB and Edge contribute smaller but meaningful improvements.

Table 5: Planning approaches across all domains (%). Same GIF predictor used for all methods.

Method	Latency	Bugz.	GH	CFPB	SO	Zen.	Rank
PPO	5ms	94.1	66.8	90.2	70.5	76.3	5.0
MPC	120ms	96.3	69.5	92.8	72.1	79.6	3.4
Beam	25ms	95.8	68.1	91.5	71.8	78.4	4.0
CEM	200ms	97.1	71.2	94.3	73.6	81.1	2.2
Greedy	3ms	97.1	70.8	93.9	**74.9**	81.0	2.4
MCTS	180ms	**98.2**	**72.4**	**95.1**	74.8	**82.3**	**1.4**

Table 6: Ablation (*left*) and component interaction (*right*) on Bugzilla (N=2,057).

Configuration	Acc.	Δ	Condition	Freq.	Acc.
Full GIF-MCTS	98.2	—	CBR & GT agree	72.3%	99.1%
−MCTS	92.5	−5.7	CBR & GT disagree	27.7%	94.8%
−GT	94.1	−4.1	HB tiebreak	18.4%	96.5%
−CBR	95.6	−2.6	Edge anomaly	4.1%	71.2%
−HB	96.8	−1.4	All 3 agree	61.8%	99.4%
−Edge	97.5	−0.7	No majority	8.3%	88.5%

The interaction analysis (right side of Table 6) reveals that when CBR and GT agree on the best action (72.3% of cases), accuracy reaches 99.1%. When they disagree, GT is more reliable (+3.2pp), but HB often breaks ties effectively. Edge-flagged anomalous items (4.1% of test cases, 71.2% accuracy) are candidates for human review. When all three main components agree (61.8%), accuracy is 99.4%, enabling confidence-calibrated deployment where high-agreement items are auto-routed and low-agreement items are flagged for human review.

5.5 HICRA Cross-Domain Validation

HICRA improves convergence speed across all five domains (Table 7), with the formally derived α_s matching empirical optima within ± 0.2. On Bugzilla, HICRA converges in 510 episodes compared to 850 for standard TD (40% faster), reaching cumulative reward 9.1 vs. 6.4 (+42%). Domains with clearer hierarchical escalation paths (Bugzilla, CFPB) benefit from higher α_s; flatter structures (StackOverflow) prefer moderate amplification.

5.6 Per-Team Analysis

Table 8 provides a per-team breakdown on Bugzilla, the domain with the most granular team structure. The largest improvement is on Graphics/Media (+7.1pp), a team that handles diverse issue types where CBR's ability to retrieve similar historical cases is most valuable. The smallest improvement is on Build/Release (+4.3pp), a team with deterministic, pattern-based issues where even simple classifiers perform well. Notably, GIF-MCTS reduces the inter-team accuracy variance from 4.7pp (DQN) to 1.9pp, indicating more uniform performance across teams.

5.7 Data Quality and Production Realism

CFPB (5K structured samples, 89% DQN accuracy) outperforms GitHub (50K unstructured samples, 64% DQN accuracy) by 25pp despite having 10× less data. Controlled experiments (Appendix D) isolate the contributions: feature structure (+10.5pp) > class balance (+5.3pp) > dataset size (−2.4pp). This highlights that data quality dominates quantity for routing performance.

Models lose approximately 24% accuracy under production-realistic noise conditions. We identify six friction factors and their individual contributions: workload variability (+32% of the gap), data quality issues (+18%), report ambiguity (+15%), reporter inconsistency (+12%), timezone effects (+8%), and team availability (+5%). Training with correlated noise recovers accuracy to 92% (details in Appendix E).

Table 7: Cross-domain HICRA validation. Derived α_s vs. empirical optimum.

Domain	Derived α_s	Empirical opt.	Δconvergence	Accuracy (%)
Bugzilla	2.95	3.0	-40%	98.2
GitHub	2.38	2.5	-32%	72.4
CFPB	3.12	3.0	-38%	95.1
StackOverflow	1.89	2.0	-28%	74.8
Zendesk	2.53	2.5	-35%	82.3

Table 8: Per-team accuracy breakdown on Bugzilla.

Team	DQN (%)	GIF-MCTS (%)	Δ	Volume
Core Engine	94.2	98.9	+4.7	2,150
DOM/Layout	91.8	97.8	+6.0	1,890
JavaScript	93.5	98.4	+4.9	2,340
Graphics/Media	90.1	97.2	**+7.1**	1,520
Networking	92.7	98.5	+5.8	1,280
Build/Release	94.8	99.1	+4.3	820

6 Discussion and Conclusion

Error analysis. On Bugzilla, GIF-MCTS achieves 98.2% accuracy, leaving a 1.8% error floor. Analyzing these errors reveals four categories: cross-cutting issues that span multiple components and have no single correct team (38% of errors), novel issue types absent from training data such as WebGPU bugs that emerged after model training (31%), vague descriptions lacking sufficient technical detail for confident classification (22%), and team restructuring that occurred after the training period (9%). The first two categories represent genuine ambiguity that may be irreducible without additional input (e.g., developer clarification), while the latter two suggest that periodic retraining and active learning could further reduce errors.

Limitations. (1) The strategic/tactical action partitioning is manual, based on observable action metadata. While sensitivity analysis shows only $\pm 0.6\%$ impact from reclassification (Appendix B), learning this partition from data is a natural extension. (2) MCTS provides marginal advantage for single-step, few-team routing (StackOverflow: -0.1pp vs. Greedy). For such settings, Greedy-Exhaustive is preferred. (3) Bugzilla's 1.8% error floor reflects genuine labeling ambiguity (κ=0.96); per-seed results are reported in Appendix F. (4) GIF predicts outcomes, not state transitions; Figures 1–2 use "world model" loosely. Extending to full state-transition modeling is future work. (5) Code will be released upon acceptance.

Broader impact. Automated routing reduces manual triage burden but carries risks of bias amplification (Mehrabi et al., 2021): if historical routing data reflects biased assignment patterns, GIF-MCTS may perpetuate them. Fairness auditing across protected groups is essential before deployment.

Conclusion. GIF-MCTS combines four complementary outcome predictors to enable MCTS-based planning for enterprise routing without real-world interaction. HICRA provides hierarchical credit assignment with a formally derived amplification factor $\alpha_s = \mathbb{E}[\tau_s]/\mathbb{E}[\tau_t]$, reducing convergence time by 28–40%. Across 93.5K items in five domains, GIF-MCTS achieves 72–98% accuracy, improving over the best LLM baseline by 4.0–5.1pp and over DQN by 5.7–8.8pp (all $p<0.01$, $d\geq 4.8$). When CBR and GT components agree (72% of cases), accuracy reaches 99.1%, enabling confidence-calibrated deployment where high-agreement items are auto-routed and uncertain items are flagged for human review.

References

Agnar Aamodt and Enric Plaza. Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI Communications*, 7(1):39–59, 1994.

Samuel Ackerman, Lily Alexander, Michael Bennett, David Chen, Eitan Farchi, Aaron Houseknecht, and Padmini Santhanam. Deploying automated ticket router across the enterprise. *AI Magazine*, 44:97–111, 2023.

John Anvik, Lyndon Hiew, and Gail C Murphy. Who should fix this bug? In *International Conference on Software Engineering*, pp. 361–370. ACM, 2006.

Jose A Arjona-Medina, Michael Gillhofer, Michael Widrich, Thomas Unterthiner, Johannes Brandstetter, and Sepp Hochreiter. RUDDER: Return decomposition for delayed rewards. In *Advances in Neural Information Processing Systems*, 2019.

Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

Cameron Browne, Edward Powley, Daniel Whitehouse, Simon Lucas, Peter Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis, and Simon Colton. A survey of Monte Carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in Games*, 4(1):1–43, 2012.

Tianqi Chen and Carlos Guestrin. XGBoost: A scalable tree boosting system. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 785–794. ACM, 2016.

Davor Cubranić and Gail C Murphy. Automatic bug triage using text categorization. In *International Conference on Software Engineering and Knowledge Engineering*, 2004.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 4171–4186, 2019.

Siyuan Guo et al. DS-Agent: Automated data science by empowering large language models with case-based reasoning. In *International Conference on Machine Learning*, 2024.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*, 2020.

Robert Heinrichsmeyer et al. Neural Monte Carlo tree search: A survey. *arXiv preprint*, 2024.

Ravi Iyer. AI-driven ticket classification and routing: A survey. *preprint*, 2024.

Gaeul Jeong, Sunghun Kim, and Thomas Zimmermann. Improving bug triage with bug tossing graphs. In *Joint Meeting of the European Software Engineering Conference and the Symposium on the Foundations of Software Engineering*, pp. 111–120. ACM, 2009.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547, 2019.

Levente Kocsis and Csaba Szepesvári. Bandit based Monte-Carlo planning. In *European Conference on Machine Learning*, pp. 282–293. Springer, 2006.

Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *IEEE International Conference on Data Mining*, pp. 413–422. IEEE, 2008.

Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. A survey on bias and fairness in machine learning. *ACM Computing Surveys*, 54(6):1–35, 2021.

Vincent Micheli, Eloi Alonso, and François Fleuret. Transformers are sample-efficient world models. In *International Conference on Learning Representations*, 2023.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

NVIDIA. Cosmos: A platform for physical AI. *Technical Report*, 2025.

Eduardo Pignatelli, Johan Ferret, Matthieu Geist, Thomas Mesnard, Hado van Hasselt, and Olivier Pietquin. A survey of temporal credit assignment in deep reinforcement learning. *Transactions on Machine Learning Research*, 2024a.

Eduardo Pignatelli, Johan Ferret, Tim Rocktäschel, Edward Grefenstette, Davide Paglieri, Samuel Coward, and Laura Toni. Assessing the zero-shot capabilities of LLMs for action evaluation in RL. *arXiv preprint arXiv:2409.12798*, 2024b.

Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Conference on Empirical Methods in Natural Language Processing*, pp. 3982–3992, 2019.

Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, Timothy Lillicrap, and David Silver. Mastering Atari, Go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of Go without human knowledge. *Nature*, 550(7676):354–359, 2017.

Richard S Sutton, Doina Precup, and Satinder Singh. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1-2):181–211, 1999.

Alexander Sasha Vezhnevets, Simon Osindero, Tom Schaul, Nicolas Heess, Max Jaderberg, David Silver, and Koray Kavukcuoglu. FeUdal networks for hierarchical reinforcement learning. In *International Conference on Machine Learning*, pp. 3540–3549, 2017.

Nirmalie Wiratunga et al. CBR-RAG: Case-based reasoning for retrieval augmented generation in LLMs for legal question answering. *arXiv preprint*, 2024.

Jialong Wu et al. RLVR-world: Training world models with reinforcement learning via verifiable rewards. *arXiv preprint arXiv:2501.01904*, 2025.

Xin Xia, David Lo, Xinyu Wang, and Xiaohu Yang. Accurate developer recommendation for bug resolution. In *International Working Conference on Mining Software Repositories*, 2017.

Yazhe Xu et al. UniZero: Generalized and efficient planning with scalable latent world models. *arXiv preprint arXiv:2406.10667*, 2024.

A HYPERPARAMETERS

Table 9: Complete hyperparameter configuration.

Component	Configuration
Text Embedding	Sentence-BERT (all-MiniLM-L6-v2, 384d)
CBR Retrieval	FAISS HNSW, k=10, τ=0.6
Ground Truth	XGBoost (200 est., depth 6, lr 0.1)
Human Behavior	Random Forest (50 trees, depth 5)
Edge Detector	Isolation Forest (100 est., 5% contam.)
MCTS	100 sims, C=$\sqrt{2}$
HICRA	α_s domain-specific (Tab. 7), α_t=1.0, γ=0.99
DQN Baseline	256-256-K MLP, Adam lr=0.001, batch 32

B ACTION CLASSIFICATION SENSITIVITY

Reasonable reclassification variants change accuracy by ± 0.4–0.6%. Extremes (all-strategic or all-tactical) show larger degradation, confirming the distinction matters but boundary choices do not.

Table 10: HICRA sensitivity to action classification on Bugzilla.

Variant	Acc. (%)	Convergence
Original	98.2	510 eps
escalate→mgmt → Tactical	97.8	530 eps
update_priority → Strategic	97.6	545 eps
All strategic (α=3.0)	96.9	620 eps
All tactical (α=1.0)	94.1	850 eps

C HICRA SENSITIVITY TO α_s

α_s=3.0 is optimal on Bugzilla. The non-monotonic pattern reflects a bias-variance trade-off: low α_s under-weights strategic decisions; high α_s amplifies segment return noise. The derived value $\alpha_s \approx \mathbb{E}[\tau_s]/\mathbb{E}[\tau_t] = 3.0$ sits at the sweet spot.

Table 11: HICRA α_s sensitivity on Bugzilla.

α_s	α_t	Episodes	Reward	Acc. (%)
1.0	1.0	850	6.4	94.1
2.0	1.0	620	7.8	96.5
3.0	**1.0**	**510**	**9.1**	**98.2**
4.0	1.0	540	8.7	97.8
5.0	1.0	590	8.2	96.9

D QUALITY VS. QUANTITY DETAILS

Reducing GitHub from 50K to 5K costs only -2.4pp, while structured features add +10.5pp and balancing adds +5.3pp. CFPB's 5K structured samples outperform GitHub's 50K unstructured by 25pp.

Table 12: Quality vs. quantity controlled analysis (DQN accuracy).

Configuration	Size	Quality	Imbalance	Acc. (%)
GitHub (original)	50K	Low	10.7×	63.6
GitHub (5K subsample)	5K	Low	10.7×	61.2
GitHub (balanced)	50K	Low	1.0×	68.9
GitHub (+ structured)	50K	Medium	10.7×	74.1
CFPB (original)	5K	High	4.2×	89.0

E Friction Factor Details

Six friction factors cause a 24% gap between lab accuracy (92%) and simulated production accuracy (68%). A correlated noise model with pairwise interactions (e.g., workload × timezone: $\rho_{1,5}$=0.3) recovers accuracy to 92%.

Table 13: Friction factors causing lab-to-production accuracy gap.

Factor	Impact	Source	Mitigation
Workload Variability	+32%	Sprint cycles	Capacity model
Data Quality	+18%	Missing fields	Completion score
Ambiguity	+15%	Vague reports	Clarification prompt
Reporter Reliability	+12%	Inconsistency	History weighting
Timezone	+8%	Global teams	Availability windows
Availability	+5%	PTO/meetings	Calendar integration

F Per-Seed Main Results

Table 14 reports individual seed results for GIF-MCTS. Variance is low across seeds for well-structured domains (Bugzilla, CFPB) and slightly higher for noisier domains (GitHub, StackOverflow). StackOverflow shows the smallest improvement over the best baseline (+4.7pp) and seed 3 underperforms the best baseline (LLaMA-3 FT at 70.1%), reflecting the marginal MCTS advantage on this low-K domain.

Table 14: Per-seed GIF-MCTS accuracy (%) across five domains.

Seed	Bugzilla	GitHub	CFPB	SO	Zendesk
1	98.5	73.1	95.4	75.8	82.9
2	97.8	71.3	94.5	74.2	81.5
3	98.4	72.8	95.7	73.9	82.5
4	97.9	71.9	95.0	75.3	82.0
5	98.4	72.9	94.9	74.8	82.6
Mean	98.2	72.4	95.1	74.8	82.3
Std	0.3	0.7	0.5	0.7	0.5