GENEGO: A Disciplined LLM Negotiation Agent with Pairwise Preference Inference

Maksim Panchuk [1] **Viacheslav Shalamov** [1] **Sergey Muravyov** [1] **Valeria Efimova** [1]

Abstract

Large language models (LLMs) are fluent conversationalists but remain unreliable negotiators. Single-prompt agents tend to forget past offers, contradict themselves between turns, and concede value just to keep the conversation moving. We present GENEGO, a structured negotiation agent built around three components: a ledger that records every offer the agent has made or agreed to, a partner model that estimates the opponent's preference ordering from the dialog using pairwise A/B comparisons aggregated with a Bradley-Terry model, and a tit-for-tat rule that keeps the agent above its own walk-away threshold. On a multi-issue negotiation benchmark, GENEGO averages 22.0 points per battle against a strong prompt-only baseline's 19.1 (paired t-test, $p < 0.001$). The partner model recovers the exact preference ordering on 82% of turns and the top-priority item on 97%; its bootstrap-based confidence is well calibrated, with accuracy rising from 14% in the low-confidence band to 94% in the high-confidence one. Reliable LLM negotiation needs strategic discipline that the agent follows, a fixed record of every offer made, and a robust way to infer the partner's preferences — GENEGO supplies all three.

1. Introduction

People often negotiate — buying a house, signing a job offer, agreeing on contract terms, splitting work between teams. The structure is always the same: two sides with overlapping but conflicting interests must reach an agreement. Each side wants more for itself, but each side also wants the deal to happen, because no deal is usually worse than a moderate one. The work of negotiating is to surface what the partner actually values most — which they will rarely state directly — and to turn that information into a deal both sides can sign. Common failures, even among humans, include conceding more than the partner did, agreeing to fair-looking splits that leave value on the table, and breaking off before any agreement is reached.

A negotiation is therefore not merely a conversation. The two sides have different goals and hidden preferences (utility functions). Every concession costs real value, and the main work is figuring out what the partner actually wants — because the partner will not say it directly.

LLMs are good at sounding like negotiators. However, they are not, by default, good at being negotiators (Bianchi et al., 2024; Hua et al., 2024). A single-prompt agent — even one that uses chain-of-thought (Wei et al., 2022) — tends to do at least one of these things in any non-trivial dialog: drift on its own offers (today's offer is worse for itself than yesterday's, with no reason); contradict itself between turns; treat politeness as policy and yield when the partner pushes; propose "fair" splits that ignore what the partner actually values; and commit with full confidence even when its inference is shaky.

Each of these issues has the same root cause. The LLM-based agent has no persistent memory of what it has already offered or agreed to. The agent has no model of what the partner actually cares about. And the agent has no rule against giving in. We fix each one in turn: a *ledger* gives the agent memory, a *partner model* gives it an estimate of the partner's preferences, and a *tit-for-tat rule* gives it discipline. The proposed agent, named GENEGO, fixes the issues described above.

The technically interesting part is the partner model. We do not ask the LLM to estimate preferences directly — numbers from the model drift between calls and have no shared anchor. Instead, we ask the LLM many times per turn, "which of these two deals would the partner prefer?" with two concrete candidate bundles. This pairwise question is one that the model handles with respect to reliably. We aggregate the answers with a Bradley-Terry model (Bradley & Terry, 1952), recover the partner's priority order, and run a bootstrap to get an honest confidence score. The agent

[1]ITMO University, Saint Petersburg, Russia. Correspondence to: Maksim Panchuk <makfusha@niuitmo.ru>, Valeria Efimova <vefimova@itmo.ru>.

Proceedings of the 43rd International Conference on Machine Learning, Seoul, South Korea. PMLR 306, 2026. Copyright 2026 by the author(s).

acts on the order and uses the confidence to decide whether to commit or to ask one more question.

This is also where the work sits in the broader agentic-uncertainty conversation. The confidence score is computed for free from the same bootstrap that produces the priority order, with no extra LLM call, and the agent does not merely estimate it but conditions its own behaviour on it — asking when in doubt and committing when sure. We verify empirically that the agent's self-reported confidence tracks its actual accuracy: on turns where the agent reports high confidence the inferred order is much more likely to be exactly right, and on turns where it reports low confidence the inference is much more likely to be wrong. This correspondence between self-reported confidence and actual accuracy — usually called *calibration* — is, in our view, what "uncertainty-aware" has to mean for an LLM agent in practice: a signal the agent itself can act on.

Our contributions are as follows:

- We introduce the structured negotiation agent, GENEGO, which outperforms a strong prompt-only baseline on a standard multi-issue benchmark.
- We propose a practical case for pairwise preference inference over numeric scoring in LLM agents.
- We present a calibration result showing that the agent's own confidence is a usable signal for when to act.

2. Related Works

Research on LLM negotiation grew out of earlier neural negotiators (Lewis et al., 2017) and the strategy/generation split of He et al. (2018). The benchmarks we evaluate against come from this line: CaSiNo (Chawla et al., 2021) is the multi-issue setup we use, and NegotiationArena (Bianchi et al., 2024) is a more recent platform along the same lines. As a baseline policy, we adopt the proactive chain-of-thought scheme of Deng et al. (2023), among the strongest prompt-only approaches reported on multi-issue tasks. The agentic side of the literature has been moving in several directions in parallel: tactic libraries (Abdelnabi et al., 2024), self-play with AI feedback (Fu et al., 2023), game-theoretic workflows that target Nash equilibria (Hua et al., 2024), and tool-integrated counter-offer optimization (Kwon et al., 2025). All of these approaches share a common assumption: the partner's utility weights are either given to the agent up front, or never explicitly estimated from the dialogue at all. Our partner model is built to close that gap.

The mathematical approach we rely on for that estimate is Bradley-Terry model (Bradley & Terry, 1952), the standard model for recovering a ranking from pairwise comparisons; it also underpins reward modeling in RLHF (Ouyang et al., 2022) and preference-based fine-tuning more generally. Our use is at inference time rather than training time: we fit Bradley-Terry model each turn from LLM-generated votes about the partner's behavior. The reason we generate those votes pairwise rather than asking the LLM for numeric utility scores is that recent work on LLM evaluation (Liu et al., 2024; Zheng et al., 2023) reports markedly higher reliability for pairwise prompts than for pointwise scoring. The setting in those papers is LLM-as-judge, not utility inference, but the underlying problem is the same: a numeric scale has no shared meaning across calls, while a binary comparison between two concrete bundles has a clear reference.

The other half of the design is uncertainty. A growing body of work asks LLM agents to express calibrated confidence so that downstream behavior can react — via verbalized confidence in natural language (Lin et al., 2022), post-hoc elicitation from RLHF-tuned models (Tian et al., 2023), or semantic-entropy-style approaches that cluster equivalent outputs (Kuhn et al., 2023). We take the cheapest route available: a bootstrap (Efron, 1979) over the pairwise votes the agent has already collected, with the permutation-stability of the resulting Bradley-Terry fit serving as the agent's self-confidence.

Underneath all of this sits the bargaining theory that the work ultimately targets. The Nash bargaining solution (Nash, 1950) characterizes fair, Pareto-optimal splits once both sides' utilities are known. In real negotiation, the partner's utility is hidden, which is exactly the missing piece that the partner model supplies — it infers that utility from the dialog, with no privileged access required. On the strategy side, the classic tit-for-tat of Axelrod (1984) is the simplest disciplined response we know to a sequence of concessions, and we use a small variant that never concedes beyond an agent-specific walk-away threshold.

3. Method

At every turn, GENEGO updates a ledger of offers, a partner model, a stance/fairness reading of the partner's last move, and a tit-for-tat directive that turns those readings into a target utility for the next offer; a small LLM-based judge then picks the offer the agent will actually make. Figure 1 shows the full per-turn flow.

Ledger. The ledger is a structured record of the negotiation that grows by one row per turn: who spoke and what was offered. From there on, the agent reads this record, not the raw chat. This alone prevents drift and contradictions: an agent that has to write down what it just did cannot quietly forget what it already gave up. An offer is stored as a small object with one count per item — for example, {`A:3, B:2, C:0`} together with the turn number and the side that made it. The agent never re-reads the partner's wording

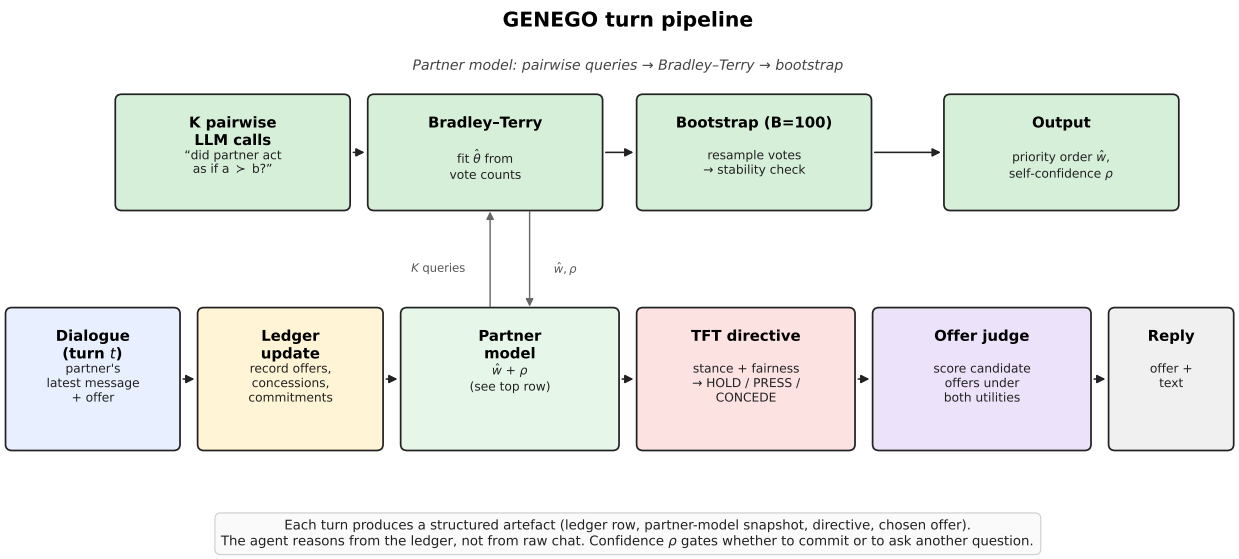

Figure 1. One turn of negotiation agent GENEGO. Bottom row: the partner's last message is appended to the ledger; the partner model produces a priority order $\hat{w}$ and a self-confidence ρ; the tit-for-tat directive picks HOLD / PRESS / CONCEDE; the offer judge scores candidate offers under both utilities; the reply is generated. Top row: how the partner model is built — K pairwise A/B questions to an LLM, a Bradley–Terry fit on the votes, and a bootstrap that gives the confidence ρ.

to plan its next move; it only looks at these rows.

Partner model. We want an estimate of the partner's preferences from the dialog alone. We tried asking the LLM to score the partner's value for each item on a 0-10 scale. It does not work: the scores are unstable across calls, often duplicate across items, and have no units. The fix is to ask pairwise questions of the form

> *"Given the dialog so far, which of these two bundles would the counterparty prefer?*
> *A: food=a_f, water=a_w, firewood=a_d*
> *B: food=b_f, water=b_w, firewood=b_d.*
> *Answer A or B."*

Each turn, we sample $K = 20$ random pairs of feasible partner-side bundles and ask the LLM. We treat each A/B answer as a Bradley-Terry preference and fit logistic regression on the rows $(a - b,\, y)$. The fitted coefficients $\hat{\theta} \in \mathbb{R}^3$ are the partner's per-resource weights; we rank them to get the priority order $\hat{w}$.

To get an honest confidence on $\hat{w}$, we resample the K rows with replacement, refit, and project $B = 100$ times. The self-confidence ρ is the share of bootstrap fits whose order matches $\hat{w}$. $\rho \approx 1$ means the answer is robust to noise in the votes; $\rho \approx 0.5$ means a different draw might have flipped the order, and the agent should not commit yet.

Tit-for-tat directive. Each turn, the agent labels the partner's last move along two axes, both computed from the ledger and the inferred partner utility $\hat{w}$ — so this stage already uses the output of the partner model. *Fairness* looks at the partner's proposed split. We compute the total value at stake — our score under our own utility w plus the partner's score under $\hat{w}$ — and check how much of it lands on our side. If our share is less than half of the total, the split is UNFAIR; otherwise FAIR. *Stance* compares the partner's current offer with their previous one in terms of *our* score: how much value would we have gotten from each under our own w. If the new offer gives us more, the partner moved toward us (GENEROUS); less means they moved away (GREEDY); the same value, or no previous offer, gives UNKNOWN. From these two labels, the agent picks PRESS, HOLD, or CONCEDE: match the partner's concession in size and direction, never concede more than they did, and never go below our own walk-away threshold. The directive carries a target utility for the next offer that the rest of the pipeline must respect. There is no prompt phrasing the partner can use to push the agent below its walk-away point, because the directive is computed from numbers in the ledger, not from wording.

Offer selection. Given the directive's target, the agent enumerates a small set of candidate offers near the target. A judge LLM scores each one by the agent's utility and the partner's expected utility under $\hat{w}$, and picks the best.

Uncertainty as a first-class signal. The bootstrap above is not a diagnostic we attach at the end of training — it is a live signal the agent reads at every turn, and the design of the rest of the pipeline is built around it. We chose this route over the more common alternatives (verbalized confidence in natural language (Lin et al., 2022; Tian et al., 2023), raw model logits, semantic-entropy estimators (Kuhn et al., 2023)) because three properties matter when the confidence signal has to drive behavior rather than just decorate the output.

Firstly, it must be cheap: $B = 100$ refits of a 3-dimensional logistic regression run in milliseconds and add no LLM cost, so the bootstrap can be paid for on every turn without changing the per-turn budget.

Secondly, it must be honest: it cannot require the LLM to introspect about its own uncertainty, an operation that is known to be unreliable on RLHF-tuned models (Tian et al., 2023); the bootstrap measures the geometric stability of the Bradley-Terry fit under noise in the votes, not the model's opinion of how sure it is.

Thirdly, it must be actionable: $\rho \in [0, 1]$ is one scalar the agent plugs straight into its decision loop — high ρ means committing to offers tailored to $\hat{w}$; low ρ means spending the next turn on a clarifying question instead of acting on a guess. This is what we mean by an "uncertainty-aware" negotiation agent: not one that reports a confidence number next to its answer, but one whose behavior visibly changes as that number moves. Section 4 verifies empirically that the signal is calibrated — the share of turns on which $\hat{w}$ is exactly right rises monotonically with ρ — so the gating decisions the agent makes from ρ are grounded in real accuracy, and not in a self-report that happens to look like confidence.

4. Experiments

Benchmark. We evaluate our agent on CaSiNo (Chawla et al., 2021), a multi-issue negotiation benchmark in which two agents play campsite neighbors, splitting nine items across three resources: food, water, and firewood, three units each. Each side privately holds a weight vector that is a permutation of $\{3, 4, 5\}$, and the score at the end of a battle is the dot product of those weights with the share that side walked away with. The arithmetic is worth keeping in mind because it sets the scale of the rest of the paper: the maximum score a single side can possibly walk away with is $3 \times (3 + 4 + 5) = 36$ (taking everything, leaving the partner on 0), an even item-count split gives both sides roughly 18, and a well-played integrative deal — one where each side trades for the items it values most — typically lands somewhere between 20 and 26. Configurations span integrative cases (each side values different items most, so a deal that is good for both exists) and distributive ones (both sides want the same item). Agents exchange free-form natural-language messages turn by turn, and the battle ends when one side accepts an offer or the turn budget runs out.

Setup. We run two arms against the same opponent: GENEGO vs. Baseline and ProCoT vs. Baseline. The Baseline opponent is Anthropic Claude Sonnet; the ProCoT agent also runs on Sonnet, so the LLM family is held fixed across the comparison. GENEGO uses Sonnet everywhere except for the pairwise Bradley-Terry queries inside the partner model, which it routes to the smaller `gpt-5-nano`: GENEGO issues $K = 20$ pairwise calls per turn, and the small model keeps the per turn cost in line with the prompt-only baselines. The main reasoning, the ledger, the directive, and the offer judge all use Sonnet; the partner-model stage is the only place where a smaller model is used. For the paired tests, we restrict to the 13 configurations both arms played; replicates within an arm are averaged before pairing.

What we record on every turn. All experiments below come from the same set of negotiations. For every battle in both arms, we log the standard end-of-battle outcome: who accepted, the final allocation of resources, and each side's final score under its own private weights. These per-battle outcomes are used to form the Table 1 and the paired test in Result 1. For each GENEGO battle, we additionally log, on every turn, the full state of the partner-model stage — the inferred partner ordering $\hat{w}$, the bootstrap confidence ρ, and the pairwise votes that produced them. We call each such per-turn record a *turn-snapshot*. A typical GENEGO battle runs five to nine turns, so the 50 battles we ran in this arm yield 255 turn-snapshots in total. Figures 2 and 3 (Results 2 and 3) are computed from these snapshots: they are a finer-grained view of the same battles that produced Table 1, not a separate experiment.

Result 1: GENEGO gains more points. Table 1 compares the two arms on the shared subset. Recall that the maximum a single side can walk away with is 36, an even item-count split gives roughly 18, and a well-played integrative deal lives in the low-to-mid 20s. On that scale, GENEGO averages 22.0 points per battle while leaving its partner on 16.2 — a typical outcome is something like "3 food, 3 water, 0 firewood for us; the rest for them", which under our weights gives us close to our top two priorities while letting the partner keep their top one. ProCoT, by contrast, ends most battles around an even split (19.1 for itself vs. 18.7 for the partner), so the partner walks away nearly as well off as ProCoT does. The paired difference in gaps is $\bar{\Delta} = +5.4$ points in GENEGO's favour, paired t-test $p < 0.001$, Wilcoxon $p < 0.001$, Cohen's $d = 1.35$, and a bootstrap 95% CI of $[+3.3, +7.5]$. GENEGO wins on 12 of the 13 configurations.

Table 1. Mean per-battle scores across the 13 shared CaSiNo configurations. "Own" is the score of the agent named in the row (GENEGO or ProCoT) at the end of the battle; "Partner" is the score of the same fixed opponent under that arm; "Gap" is own minus partner and reflects how much surplus the agent extracted relative to its partner. GENEGO secures both more for itself and less for the partner, opening a substantially wider gap than ProCoT.

Arm	Own	Partner	Gap
GENEGO vs. Baseline	22.0	16.2	+5.8
ProCoT vs. Baseline	19.1	18.7	+0.4

Result 2: the partner model recovers the right priority order. We now zoom in on the 255 turn-snapshots described above. For each snapshot we compare the agent's inferred order $\hat{w}$ against the partner's true hidden order $w^\star$, which we have access to as experimenters but the agent never does. Pooled across all 255 snapshots, $\hat{w}$ matches $w^\star$ exactly on 82% of turns, and the agent's predicted top priority matches the partner's true top priority on 97%. Accuracy also improves *within* a single negotiation: Figure 2 shows the average L_1 distance between $\hat{w}$ and $w^\star$ on each turn, and the curve drops sharply over the first few exchanges, reaching its low point around turn 5 where $\hat{w}$ is exactly right 94% of the time. The rise past turn 7 is a selection effect — only the hardest, longest battles are still going at that point — not a degradation of the inference itself.

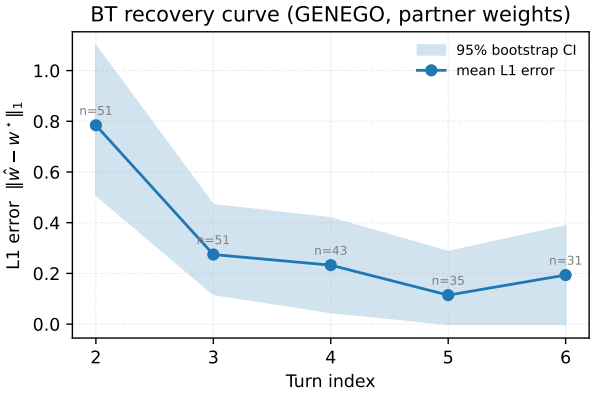

Figure 2. Convergence of the partner model. For each dialogue turn (x-axis) we plot the mean L_1 distance between the agent's inferred partner-priority ordering $\hat{w}$ and the partner's true ordering $w^\star$ (y-axis; lower is better, 0 means $\hat{w} = w^\star$ exactly), averaged over all GENEGO battles still active at that turn. The shaded band is the 95% bootstrap CI and the n annotation gives the number of contributing battles. The agent needs only a handful of turns of dialogue to settle on the right priority order.

Result 3: the agent's self-reported confidence is calibrated. The bootstrap confidence ρ (Section 3) is the agent's own measure of how sure it is about $\hat{w}$. For this signal to be useful in the loop, ρ must track actual accuracy: high ρ should correspond to a high probability of $\hat{w} = w^\star$, low ρ to a low one. Figure 3 pools all 255 turn-snapshots, bins them by the value of ρ at the time of the inference, and plots the share of inferences in each bin that turned out to match $w^\star$. The relationship is monotone and steep: $\rho < 0.5$ snapshots are right only 14% of the time, while $\rho \geq 0.9$ snapshots are right 94% of the time, with the intermediate bins at 61% and 79%. The right panel shows that the agent does spend non-trivial time in the low-confidence regime, so the gating decision actually fires.

The takeaway is that GENEGO's confidence is not decorative. A turn on which the agent reports low ρ really is a turn on which $\hat{w}$ is unlikely to be right, and a high-ρ turn really is one where it is. This is what closes the loop between the partner model and the policy — the agent acts on the inferred order only when its own evidence supports it, and otherwise asks one more question.

5. Limitations and Future Work

Several pieces of the current system are deliberate simplifications, and a few of them matter enough to flag explicitly. The partner model fits a logistic regression on the pairwise votes, which is equivalent to approximating the partner's utility as a *linear* function of resource counts. Real utilities are usually not linear: they have diminishing returns, threshold effects, package interactions between items. A richer estimator — piecewise-linear, monotone, or kernel-based — would capture those shapes better, and we expect the gain to be substantial on richer item spaces than CaSiNo's. The system is also not robust to bluffing (Müller, 2026): a partner who deliberately misrepresents what they care about (for instance, downplaying firewood early in order to extract better firewood terms later) will mislead the partner-model votes the same way they would mislead a human because we treat dialog at face value. We use a single LLM family across all components and against the opponent, so we do not yet have direct evidence on how well the calibration of ρ generalizes across providers. Finally, evaluation is on one benchmark; the architecture is benchmark-agnostic but has not been validated on dialog outside CaSiNo.

The next step is to widen the negotiation field beyond CaSiNo and into real business deals — multi-issue contracts, procurement across several SKUs and terms, salary-and-equity packages. The pipeline does not need to change: ledger, partner model, directive, and judge all carry over to longer item spaces and richer message content; only the inputs change. A second extension is to put a Nash-optimal search step on top of the inferred utilities. Once $\hat{w}$ has been recovered, we can run a small genetic algorithm over the offer space with a fitness function that combines our own utility, the partner's expected utility under $\hat{w}$, and a Nash-product term, and converge on splits where neither side can do better without hurting the other. We deliberately skipped

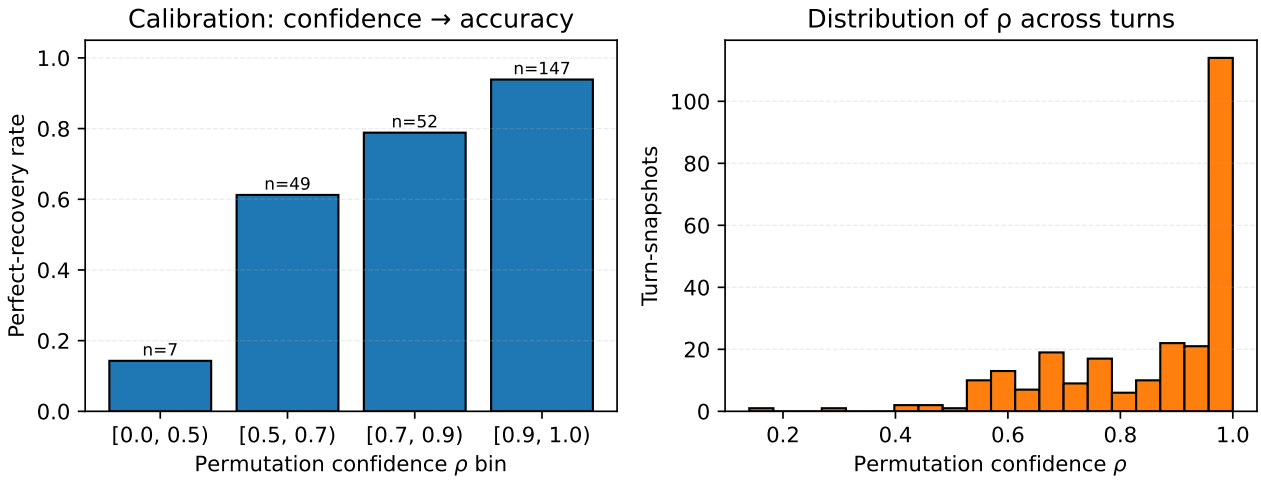

Figure 3. Calibration of the partner-model confidence ρ. **Left:** per-bin share of turn-snapshots on which the inferred ordering $\hat{w}$ matches the true ordering $w^\star$ exactly, grouped by the agent's self-reported confidence ρ. The bars rise monotonically from 14% in the lowest bin to 94% in the highest, so ρ is a usable predictor of correctness. **Right:** distribution of ρ across all turn-snapshots. The agent reports high confidence on most turns but does spend appreciable time in the low-confidence region, where it should withhold a commit.

this step on CaSiNo because the benchmark rewards raw extraction rather than fairness, and a Nash-optimal target would have left value on our side of the table; in real negotiation — where long-term relationships, repeated deals, and the cost of breakdown matter — Nash-optimal is a far more sensible target than pure extraction. A third extension is on the input side. A real deployment needs to know whose side the system is on and what that side actually wants, so we plan a short *onboarding* session before the negotiation begins: a structured conversation with the user that elicits the context of the deal, the items and terms at stake, and the user's own priorities. The output of that session replaces the static private weight vector that we currently assume is given.

6. Conclusion

We proposed a structured negotiation agent GENEGO that dominates the current prompt-only approach. Our approach of splitting the negotiation flow into a ledger for keeping the state, the pairwise partner model for learning the hidden preferences, and the rule-based constraint (tit-for-tat) for keeping the agent honest allows the agent to win consistently. On a standard CaSiNo benchmark, our agent wins 15% more value compared to ProCoT negotiator agent (22 vs 19.1).

Acknowledgements

This work was supported by the Ministry of Economic Development of the Russian Federation (IGK 000000C313925P4C0002, agreement No. 139-15-2025-010).

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning, and in particular the reliability of LLM-based negotiation agents. Stronger automated negotiators could be deployed against human counterparts, where an agent that extracts more surplus may amplify asymmetries between well-resourced and unprepared parties. We note that the components studied here cut both ways: the ledger and the walk-away rule are exactly the safeguards that protect a weaker side from manipulation, and the calibrated confidence signal makes the agent's behavior auditable rather than opaque.

References

Abdelnabi, S., Gomaa, A., Sivaprasad, S., Schönherr, L., and Fritz, M. Cooperation, competition, and maliciousness: LLM-stakeholders interactive negotiation. In *Advances in Neural Information Processing Systems (Datasets and Benchmarks Track)*, 2024.

Axelrod, R. *The Evolution of Cooperation*. Basic Books, New York, 1984.

Bianchi, F., Chia, P. J., Yuksekgonul, M., Tagliabue, J., Jurafsky, D., and Zou, J. How well can LLMs negotiate? NegotiationArena platform and analysis, 2024.

Bradley, R. A. and Terry, M. E. Rank analysis of incomplete block designs: I. The method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

Chawla, K., Ramirez, J., Clever, R., Lucas, G. M., May, J., and Gratch, J. CaSiNo: A corpus of campsite negotiation dialogues for automatic negotiation systems. In *Proceedings of NAACL-HLT*, 2021.

Deng, Y., Liao, L., Chen, L., Wang, H., Lei, W., and Chua, T.-S. Prompting and evaluating large language models for proactive dialogues: Clarification, target-guided, and non-collaborative. In *Findings of EMNLP*, 2023.

Efron, B. Bootstrap methods: Another look at the jackknife. *The Annals of Statistics*, 7(1):1–26, 1979.

Fu, Y., Peng, H., Khot, T., and Lapata, M. Improving language model negotiation with self-play and in-context learning from AI feedback, 2023.

He, H., Chen, D., Balakrishnan, A., and Liang, P. Decoupling strategy and generation in negotiation dialogues. In *Proceedings of EMNLP*, 2018.

Hua, W., Liu, O., Li, L., Amayuelas, A., Chen, J., Jiang, L., Jin, M., Fan, L., Sun, F., Wang, W., Wang, X., and Zhang, Y. Game-theoretic LLM: Agent workflow for negotiation games, 2024.

Kuhn, L., Gal, Y., and Farquhar, S. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. In *International Conference on Learning Representations (ICLR)*, 2023.

Kwon, D., Hae, J., Clift, E., Shamsoddini, D., Gratch, J., and Lucas, G. M. ASTRA: A negotiation agent with adaptive and strategic reasoning via tool-integrated action for dynamic offer optimization. In *Proceedings of EMNLP*, 2025.

Lewis, M., Yarats, D., Dauphin, Y. N., Parikh, D., and Batra, D. Deal or no deal? end-to-end learning of negotiation dialogues. In *Proceedings of EMNLP*, 2017.

Lin, S., Hilton, J., and Evans, O. Teaching models to express their uncertainty in words. *Transactions on Machine Learning Research (TMLR)*, 2022.

Liu, Y., Zhou, H., Guo, Z., Shareghi, E., Vulić, I., Korhonen, A., and Collier, N. Aligning with human judgement: The role of pairwise preference in large language model evaluators. In *Conference on Language Modeling (COLM)*, 2024.

Müller, R. Cattle trade: A multi-agent benchmark for LLM bluffing, bidding, and negotiation. In *ICLR Workshop on Multi-Agent and LLM Agents (MALGAI)*, 2026. URL https://openreview.net/forum?id=J3usA9wOhX.

Nash, J. F. The bargaining problem. *Econometrica*, 18(2): 155–162, 1950.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, 2022.

Tian, K., Mitchell, E., Zhou, A., Sharma, A., Rafailov, R., Yao, H., Finn, C., and Manning, C. D. Just ask for calibration: Strategies for eliciting calibrated confidence scores from language models fine-tuned with human feedback. In *Proceedings of EMNLP*, 2023.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E. H., Le, Q. V., and Zhou, D. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, 2022.

Zheng, L., Chiang, W.-L., Sheng, Y., Zhuang, S., Wu, Z., Zhuang, Y., Lin, Z., Li, Z., Li, D., Xing, E. P., Zhang, H., Gonzalez, J. E., and Stoica, I. Judging LLM-as-a-judge with MT-Bench and chatbot arena. In *Advances in Neural Information Processing Systems (Datasets and Benchmarks Track)*, 2023.