WHERE DO TEST-TIME SCALING AND TRAINING FALL SHORT IN INDIVIDUAL STANCE PREDICTION?

Anonymous authors
Paper under double-blind review

ABSTRACT

Test-time scaling and post-training have improved LLM performance in coding and mathematical reasoning, but their effectiveness for individual stance prediction remains unclear. We study this question by predicting a person's stance in a new discussion from their history. We evaluate widely used test-time scaling strategies and post-training methods, such as supervised fine-tuning and reinforcement learning, and identify four failure modes across generation, selection, and learning: (1) *incorrect consensus*, where repeated samples agree on the wrong stance; (2) *selection failure*, where generation covers the observed stance but selection misses it; (3) *response overfitting*, where supervised fine-tuning improves imitation but harms prediction; and (4) *early plateau*, where reinforcement learning shows modest initial gains followed by limited further improvement. We expose these failures using STANCE-BENCH, which contains 2499 prediction tasks from 500 Hacker News users. Guided by this analysis, we explore a simple approach that combines direct scores for all candidate stances with explicit assessments of support from the individual's history. On the 781-task test set, this approach achieves 21.83 discussion-specific Macro F1 with Qwen3-8B, compared with 19.27 for direct scoring. Our results motivate evaluating candidate generation, final selection, and person-specific evidence use separately. Our data is available at this url.

1 INTRODUCTION

Large language models (LLMs) are increasingly used to simulate human opinions and decisions (Argyle et al., 2023; Chuang et al., 2024; Liu et al., 2025), as well as interactions among social agents (Park et al., 2023; Zhou et al., 2024), motivating their broader use in social simulation (Gao et al., 2024; Anthis et al., 2025). Their appeal is clear: simulated participants could make behavioral research faster, cheaper, and easier to scale, while behavioral predictions could help anticipate how populations and individuals respond to new issues and interventions. Their fidelity, however, remains limited at both population and individual levels (Hu et al., 2026; Dominguez-Olmedo et al., 2024; Bisbee et al., 2024; Liu et al., 2026b). At the individual level, predictions of a particular person's held-out responses and behavioral trajectories remain unreliable, even when models are conditioned on demographic information or prior behavior (Chen et al., 2026; Lu et al., 2026; Liu et al., 2026a). These limitations contrast with the rapid gains from two widely used approaches to improving LLM capabilities: test-time scaling (TTS) for reasoning (Wang et al., 2023; Snell et al., 2025), and reinforcement learning (RL) post-training for mathematical reasoning and code generation (DeepSeek-AI, 2025).

This paper examines whether gains from TTS and post-training extend to individual stance prediction and where their limitations arise. TTS uses additional inference-time computation to generate, aggregate, or select among candidate answers, while post-training adapts model behavior using demonstrated responses or reward feedback (Snell et al., 2025; Wang et al., 2023; Ouyang et al., 2022). Individual stance prediction is important for public-opinion polling and survey research (Argyle et al., 2023; Park et al., 2026), as well as market research and policy decision-making (Chen et al., 2026). In this setting, the most common or plausible stance in a discussion may not be the stance of the particular person being predicted. This raises the question of whether methods that improve answer generation and selection also improve individual prediction. Answering this question requires more than reporting end-to-end accuracy. For inference-time methods, we distinguish

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

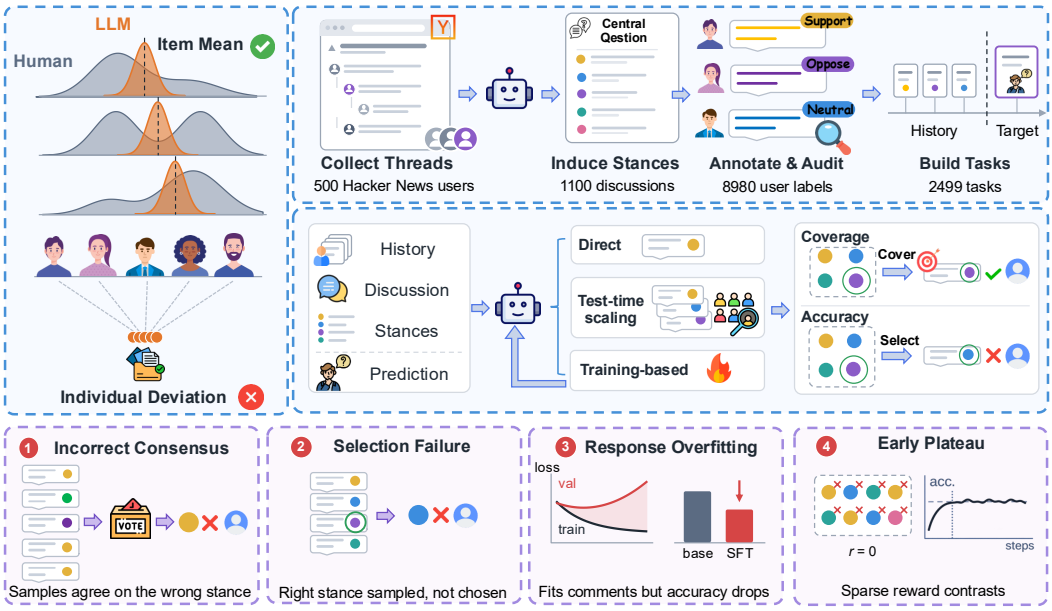


Figure 1: **Overview.** STANCE-BENCH is built by collecting Hacker News threads, inducing discussion-specific stances, and annotating user contributions. Given a user's history, the discussion, and its stance options, a model predicts the user's stance. We identify four failure modes: *incorrect consensus* (samples agree on a wrong stance), *selection failure* (the observed stance is generated but not selected), *response overfitting* (SFT fits responses better but predicts stances worse), and *early plateau* (RL gains little after early training, and reward contrast stays sparse).

failure to generate the target stance from failure to select it once generated. For training-based methods, we examine whether response imitation and outcome rewards provide useful signals for person-specific prediction.

To examine these methods at multiple stages, we construct STANCE-BENCH as a diagnostic instrument. It contains 2,499 prediction tasks from 500 Hacker News users across 1,100 annotated discussions. Given a target discussion and a user's preceding history, a model predicts the contribution that user would make, conditional on participation. Discussion-specific annotations map open-ended generated comments and observed contributions to a shared stance space. This design lets us evaluate final predictions while separately measuring whether the target stance appeared among intermediate candidates.

Our analysis identifies four failure modes across generation, selection, and learning. First, **incorrect consensus**: repeated sampling often converges on the wrong stance, leaving the target stance absent from the candidate pool. Second, **selection failure**: broader exploration improves coverage, but the observed stance is frequently not selected even when generated. Third, **response overfitting**: response-level supervised fine-tuning improves imitation of training sequences while degrading stance prediction. Fourth, **early plateau**: exact-match RL yields modest initial gains with limited subsequent improvement. Most rollout groups provide no reward contrast, and the share with nonzero reward advantages declines between the first and final training windows. These findings show how additional computation, response imitation, and outcome rewards can fail at different stages of individual prediction.

Guided by this analysis, we explore an explicit evidence reader that evaluates a user's historical comments against competing stances while keeping the language model frozen. Its Macro F1 improvement over direct option scoring, together with history-replacement controls, provides initial evidence that explicitly evaluating personal evidence can help. We treat this method as a direction suggested by the failure analysis rather than a complete solution to the four failure modes.

Our contributions are threefold:

- **Failure-mode analysis.** We analyze widely used test-time scaling and post-training methods for individual stance prediction, separating errors in candidate generation, selection, response imitation, and reward-based learning.
- **Diagnostic evaluation.** We introduce STANCE-BENCH, which connects open-ended generation to observed individual stances and records intermediate candidates and final predictions separately.
- **Analysis-guided method.** We explore an evidence-reading method motivated by the analysis, with controls testing whether its predictions depend on evidence from the target individual.

2 MOTIVATION

2.1 FROM GENERAL IMPROVEMENTS TO INDIVIDUAL PREDICTION

Test-time scaling and post-training have improved performance on many reasoning tasks (Wang et al., 2023; Snell et al., 2025; Shao et al., 2024). Whether these gains extend to individual prediction is an empirical question. In individual stance prediction, multiple responses may be plausible, while evaluation asks which stance matches the target individual. Overall response quality and person-specific predictive accuracy are therefore distinct quantities. **Test-time scaling.** Test-time methods generate additional candidates and then aggregate, rank, or select among them (Wang et al., 2023; Chen et al., 2023; Ni et al., 2023). Their gains require both generating the target stance and selecting it: agreement does not by itself imply accuracy, and diverse candidates still require a selection rule. **Supervised fine-tuning.** Supervised fine-tuning increases a model's fit to demonstrated responses (Ouyang et al., 2022). Here, the training target contains a comment and its stance label, while evaluation uses the label. Whether improved response fit also improves held-out stance prediction must be evaluated directly. **Reinforcement learning.** Outcome-based reinforcement learning encourages responses that receive higher task rewards (Shao et al., 2024). Group-relative updates use reward differences among sampled outputs, so learning from stance correctness depends in part on contrast within a group. We therefore track both predictive performance and the fraction of rollout groups that provide nonzero reward advantages.

These mechanisms motivate stage-specific evaluation without assuming in advance how the methods will behave.

2.2 WHAT THE EVALUATION MUST REVEAL

End-to-end accuracy alone cannot characterize these stages. Higher accuracy may reflect better recognition of common positions, better use of the target person's history, or both. Conversely, a method may respond to personal evidence without producing a large aggregate improvement. An evaluation must therefore reveal both where a prediction fails and whether it depends on information from the correct person. **Shared prediction context (R1).** Multiple users should be observed in the same discussion. Holding the discussion fixed makes their positions directly comparable and helps distinguish a generally likely response from a response associated with a particular person. **Controlled personal evidence (R2).** The target person's history should be removable or replaceable while the discussion and prediction target remain unchanged. Comparing the correct history with no history or another person's history tests whether performance depends on evidence matched to the individual, rather than on the presence of additional text. **Candidate and selection traces (R3).** Intermediate candidates and final predictions should be recorded separately within a shared stance space. This makes it possible to determine whether a change in final performance arises during candidate generation or during the final decision.

Together, these requirements connect changes in final performance to candidate generation, final selection, and the use of person-specific evidence.

2.3 WHAT EXISTING EVALUATIONS ESTABLISH

Population-level simulation. OpinionQA and SimBench evaluate population-level correspondence and simulation fidelity (Santurkar et al., 2023; Hu et al., 2026). Distributional agreement, how-

ever, does not establish that positions are assigned to the correct individuals. **Individual simulation and personalization.** Interview-, survey-, and history-conditioned evaluations establish the value of personal information(Park et al., 2026; Salemi et al., 2024), but do not always isolate whether performance depends on matching evidence to the correct individual. **Stance detection and prediction.** SemEval-2016 Task 6 evaluates stance identification from observed text (Mohammad et al., 2016), while Loh et al. (2024) predict stances from target-agnostic user posts. Neither separately exposes candidate generation and final selection within naturally occurring shared discussions.

Existing evaluations therefore illuminate important parts of the problem, but do not expose all of these stages within one setting. Section 3 describes the evaluation framework we use to connect shared contexts, controlled personal evidence, intermediate candidates, and final predictions. Appendix A provides a comparison of existing evaluations and STANCE-BENCH against these diagnostic requirements.

3 BENCHMARK CONSTRUCTION

Task Design. Given a user's preceding history H_u, a target discussion X_d, and candidate stances $\mathcal{L}_d$, models predict the user's dominant stance conditional on participation. Models generate a natural-language comment and a stance label, and the evaluation uses the label. Candidate stances answer the discussion's central question, with explicit definitions, exclusion boundaries, and supporting paraphrases.

We evaluate exact-match accuracy and discussion-specific Macro F1 on targets assigned a listed stance (Appendix **??**). Macro F1 averages equally over reference-supported stances, treating stances from different discussions as distinct classes and using a common class set across methods. For multiple-candidate methods, coverage is the fraction of tasks whose observed stance appears among candidates; label diversity is the mean number of distinct candidate stances.

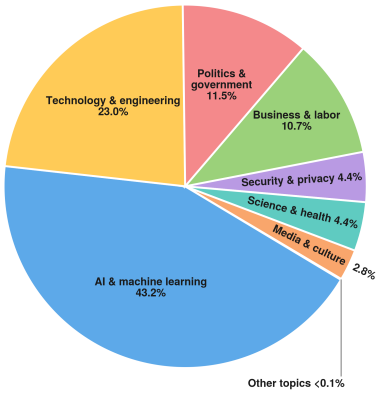


Figure 2: Topic composition of all 2499 STANCE-BENCH tasks.

Data Collection. We collect public histories for 500 active Hacker News users through the Algolia index and official API. We select shared discussions to capture histories across discussions and differences between users responding to the same context.

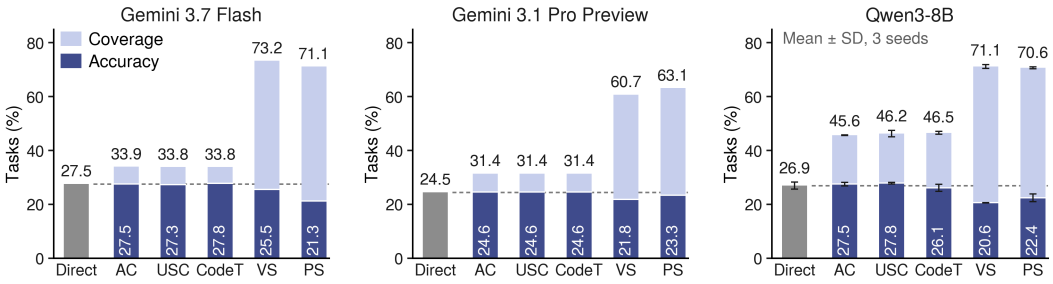
GPT-5.5 (OpenAI, 2026) identifies each discussion's central question and candidate stances, initially excluding comments from designated target users. Target contributions are later used to assign reference labels and to audit whether the shared taxonomy covers observed positions, but never appear in model-visible inputs. Revisions from these audits are applied at the discussion level rather than tailored to a single target. We retain the supporting evidence for each annotation.

Tasks use up to five recent discussions per user. Historical discussions qualify only if the user's final comment precedes their first comment in the target discussion. We remove cohort-authored comments from model-visible target context, pseudonymize users, and store prediction inputs separately from reference outcomes.

Dataset Statistics and Quality. STANCE-BENCH comprises 1100 annotated discussions and 8980 user-discussion annotations. Its 2499 prediction tasks span 414 target discussions, with four or five tasks per user. User histories cover 9 to 17 discussions with the median of 15. Taxonomies contain 4 to 10 candidate stances with the mean of 5.94.

4 FAILURE MODES AND LIMITS OF TRANSFER

We identify four failure modes in the evaluated inference-time and post-training methods. **Incorrect consensus** occurs when repeated predictions agree on a stance other than the target user's. **Selection failure** occurs when the observed stance appears among candidates but is not selected.



(a) (b) (c)

Figure 3: Candidate coverage and final prediction accuracy for (a) Gemini 3.7 Flash, (b) Gemini 3.1 Pro, and (c) Qwen3-8B. Coverage measures whether the target user's observed stance appears among the generated candidates and accuracy measures whether the final prediction matches that stance. Their difference identifies cases in which the observed stance is available but not selected.

Response overfitting follows an initial improvement in validation loss, as further supervised training improves training fit while validation loss rises. **Early plateau** describes modest initial gains from reward-based optimization followed by limited additional improvement. These findings expose distinct limitations in generating, selecting, and learning to attribute a stance to an individual (Appendix C.3).

4.1 Incorrect Consensus

Repeated agreement was often agreement on the wrong stance. In our AlphaCode-style consensus adaptation (Li et al., 2022), Gemini Flash produced only 1.29 distinct stance labels across four candidates, and candidate coverage was 33.9%. Universal Self-Consistency (USC; Chen et al., 2023) showed nearly identical diversity and coverage. Thus, on approximately two-thirds of targets, none of the sampled labels matched the user's observed stance shown in Figure 3(a). Any selector restricted to these candidates was already unable to recover the correct prediction.

Incomplete coverage was also observed for Gemini Pro (31.4%) and Qwen3-8B (45.6%; Figure 3(b,c)). Qwen consensus nevertheless improved on direct generation (Table 1).

4.2 Selection Failure

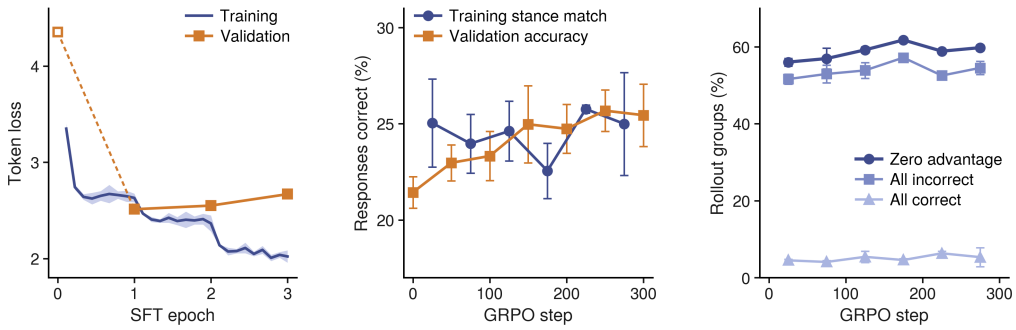
Generating the observed stance did not ensure that it was selected. With Gemini Flash, Verbalized Sampling (Zhang et al., 2025) increased coverage to 73.2%, yet selected the correct stance on only 34.8% of covered targets, despite producing nearly four distinct labels per task. The large light-blue segments in Figure 3 separate this selection loss from candidate omission. The same gap appeared with Gemini Pro and in the three-seed Qwen results. Qwen PlanSearch covered 70.6% of targets on average, but final accuracy was 22.4%. Broader exploration therefore left substantial predictive opportunity unrealized.

Our PlanSearch adaptation (Wang et al., 2025) exposed a concrete selection bottleneck. With Flash, all four candidate labels differed on 765 of 781 targets; label voting consequently produced frequent ties, and the ordering-based tie rule chose the first candidate on 774 targets.

4.3 Response Overfitting

Comment-and-label SFT initially improved held-out response fit, then overfit the training responses. Mean validation token loss fell from 4.35 at initialization to 2.51 after one epoch, before rising to 2.67 at epoch three. Across all three Qwen3-8B training seeds, training loss continued to decrease after the first epoch while validation loss increased (Figure 4(a)). The validation-selected checkpoints reached 19.50% mean test stance accuracy, compared with 27.78% for the shared decoding-matched base control. We selected the final checkpoints using validation stance F1 scores.

SFT also produced more `OTHER` and invalid outputs on the listed-position targets. `OTHER` is a valid training label but counts as an error on this subset. Some invalid responses reached the generation



(a) (b) (c)

Figure 4: **Training diagnostics for Qwen3-8B.** (a) Training and validation token loss during comment-and-label SFT. (b) Sampled training stance-match rate and first-response validation accuracy during GRPO. (c) Fractions of eight-response rollout groups with all responses incorrect, all correct, or zero reward advantages. Curves show means across three training seeds; shading and error bars indicate ± 1 sample standard deviation. The SFT epoch-0 point is a shared base-model evaluation without an error bar. GRPO rollout statistics use non-overlapping 50-step windows.

limit before producing a label. Sequence overfitting co-occurred with incorrect stance assignments and failures to produce usable predictions.

4.4 Early Plateau

Exact-match GRPO with KL regularization produced modest early gains, followed by limited additional improvement (Figure 4(b)). Mean first-response accuracy on 283 fixed validation targets rose from 21.4% at initialization to 25.0% at step 150. It reached 25.4% at step 300, adding less than half a percentage point over the second half of training. All three seeds improved from initialization by 1.8, 6.4, and 3.9 percentage points, respectively. Accuracy on 200 fixed training targets also increased by 2.0–3.5 points between steps 0 and 250, the last available training-set evaluation. Across the three 300-step runs, most of the mean validation improvement occurred within the first 150 steps.

Within-group reward contrast remained sparse throughout training (Figure 4(c)). The fraction of groups with nonzero reward advantages was lower in the final 50-step window than in the first for every seed. Its mean was 44.0% in the first window and 40.3% in the last, with fluctuations in between. Almost all groups without reward contrast were all-wrong or all-correct groups of eight responses; format penalties account for small differences between these statistics. Equal rewards yield zero reward advantages, but the KL term can still update the policy. Sparse reward contrast co-occurred with the early plateau. Sparse reward contrast co-occurred with the early plateau in these runs.

5 Exploring Explicit Personal Evidence

Section 4 shows two problems: the target user's stance is often absent from generated candidates, and it is often not selected even when present. We explore a simple response: score every listed stance directly, read the user's own history explicitly against those stances, and combine the two with two fitted coefficients. We present this as a probe guided by the failure analysis rather than as a complete solution.

Direct scores. Incorrect consensus shows that selecting from a sampled pool is bounded by candidate coverage (Section 4.1). We therefore score every listed stance by the frozen model's label log-likelihood,

$$B_j = \log \pi_\theta(j \mid X_d, H_u), \qquad j \in \mathcal{L}_d. \tag{1}$$

This makes every stance eligible for prediction but leaves the ranking unchanged: $\arg\max_j B_j$ is still the model's own preferred stance.

Evidence scores. Selection failure shows that availability does not establish attribution (Section 4.2). We therefore add an evidence score read explicitly from the user's history. For each historical comment c_i, a frozen reader returns a distribution p_i over $\mathcal{L}_d$ and a relevance weight $w_i \in [0, 1]$, the probability that the comment addresses the question at all. Following logarithmic opinion pooling (Genest et al., 1984), we aggregate

$$S_j = \sum_i \bar{w}_i \log p_i(j), \qquad \bar{w}_i = \frac{w_i}{\sum_h w_h}. \tag{2}$$

Comments judged irrelevant receive little weight, and only differences $S_j - S_k$ affect the ranking, so a comment that supports two stances equally leaves their order unchanged. Retrieval and reader details are given in Appendix C.4.

Fusion. We combine the two scores log-linearly,

$$P(y = j \mid u, d) = \mathrm{softmax}_{j \in \mathcal{L}_d} \big(aB_j + bS_j\big), \tag{3}$$

and predict the most probable stance; setting $b = 0$ recovers direct scoring. The coefficients a and b set the relative weight of the model's preference and the user's evidence, and thus determine when evidence can overturn the model's preferred stance. For fixed a and b, the softmax does not change which stance ranks first; its role is to turn the combined scores into probabilities, so that a and b can be learned by maximizing the probability assigned to observed stances on development targets (Appendix C.4). Supervision therefore acts directly on stance discrimination, rather than through response imitation or outcome rewards (Sections 4.3 and 4.4), while all language-model parameters remain fixed. Eq. equation 3 can also be read as a multinomial logit model of discrete choice (McFadden, 1974).

6 Experiments

Following the diagnostics in Section 4, we evaluate whether additional test-time computation and learned adaptations improve individual stance prediction. Our method achieves higher accuracy and Macro F1 than direct generation and direct scoring under the same evaluation protocol.

Our main comparisons use the same 781 listed-position targets from 243 test users. We report accuracy and discussion-specific Macro F1, averaging F1 across the 497 discussion–stance pairs with gold support. For repeated runs, we report the mean and sample standard deviation across seeds. We select checkpoints and intervention strengths using validation Macro F1. Appendix C.3 provides method-specific settings and matched controls.

6.1 Test-Time Scaling Strategies

Table 1 compares single-sample direct generation with six TTS adaptations. These include Verbalized Sampling, Universal Self-Consistency, AlphaCode-style consensus, PlanSearch, AlphaCodium, and CodeT. Fixed-pool methods use four candidates, whereas AlphaCodium iteratively refines a draft. These comparisons assess prediction strategies without matching total inference compute.

As shown in Table 1, additional test-time computation yielded limited Macro F1 gains in these experiments. On Gemini 3.7 Flash, CodeT was the only adaptation to exceed direct generation, reaching 22.63 Macro F1 compared with 22.43. Direct generation achieved the highest Macro F1 on Gemini 3.1 Pro, at 22.05. On Qwen3-8B, Universal Self-Consistency reached 20.66 Macro F1 compared with 20.53 for direct generation. The other Qwen adaptations yielded Macro F1 scores ranging from 16.67 to 20.64. These results complement the candidate-level diagnosis of generation and selection failures in Section 4.

6.2 Learning and Adaptation Strategies

Table 2 compares learning and adaptation strategies using Qwen3-8B as the shared base model. We evaluate LEVER with verifier-only and product selection, SFT, GRPO, user-profile methods, steering vectors, and our method on the test set. The two baselines are direct generation from Table 1 and direct scoring, which selects the highest-scoring candidate label.

Table 1: **Test-time scaling across Gemini and Qwen.** Accuracy (Acc.) and discussion-specific Macro F1 (F1) are percentages on test set. Qwen entries report mean ± sample s.d. over three generation seeds. k counts prediction candidates and AlphaCodium uses iterative refinement. Candidate counts do not equalize total inference compute. Bold marks column maxima.

		Gemini 3.7 Flash		Gemini 3.1 Pro		Qwen3-8B	
Method	k	Acc.↑	F1↑	Acc.↑	F1↑	Acc.↑	F1↑
Direct generation	1	27.53	22.43	24.46	**22.05**	$26.89_{\pm 1.30}$	$20.53_{\pm 1.37}$
Verbalized Sampling	4	25.48	21.46	21.77	19.97	$20.57_{\pm 0.07}$	$16.67_{\pm 0.47}$
Universal Self-Consistency	4	27.27	22.41	**24.58**	21.69	$\mathbf{27.83}_{\pm 0.27}$	$\mathbf{20.66}_{\pm 0.24}$
AlphaCode-style consensus	4	27.53	22.30	**24.58**	21.84	$27.49_{\pm 0.66}$	$20.64_{\pm 0.42}$
PlanSearch	4	21.25	18.14	23.30	20.65	$22.36_{\pm 1.42}$	$18.42_{\pm 1.49}$
AlphaCodium	–	26.38	21.45	21.51	19.85	$23.82_{\pm 1.22}$	$18.46_{\pm 0.84}$
CodeT	4	**27.78**	**22.63**	**24.58**	21.85	$26.08_{\pm 1.28}$	$19.98_{\pm 1.55}$

The SFT baseline is trained to generate observed comments and their stance labels. The explicit-profile model includes a textual user profile in its input. The latent-profile model learns a history-conditioned residual edit while keeping the base model frozen. Both profile models use model-generated rationale supervision and additional training tasks constructed from earlier labeled discussions. Steering adds a direction estimated from correct and incorrect training responses to hidden activations of the frozen model. Appendix C.3 details the training configurations and their matched controls.

Table 2: **Learning and adaptation on Qwen3-8B.** Test-set accuracy and Macro F1 (%). Subscripted entries report means with sample s.d. over three generation seeds (direct generation and steering) or training seeds (SFT, GRPO and profiles). Other entries are deterministic or single fitted systems. Bold marks column maxima, not significance. Settings and matched controls are in Appendix C.3.

Method	**Accuracy↑**	**Macro F1↑**
Direct generation	$26.89_{\pm 1.30}$	$20.53_{\pm 1.37}$
Direct scoring	25.86	19.27
LEVER (verifier, $k = 16$)	27.53	20.13
LEVER (product, $k = 16$)	25.48	19.78
SFT	$19.50_{\pm 1.37}$	$16.35_{\pm 1.19}$
GRPO	$27.78_{\pm 0.90}$	$21.35_{\pm 0.95}$
SFT + user profile	$23.60_{\pm 1.15}$	$20.74_{\pm 1.49}$
Latent profile	$23.56_{\pm 3.15}$	$21.11_{\pm 2.87}$
Steering vector	$27.10_{\pm 0.98}$	$21.77_{\pm 0.79}$
Our method	**28.04**	**21.83**

The adaptations showed mixed results in discussion-specific Macro F1. SFT reached 16.35, below 21.50 for its matched untrained generator. GRPO reached 21.35 compared with 20.55 for its matched base. LEVER's verifier and product variants scored 20.13 and 19.78, respectively, below 21.14 for majority selection from the same candidate pool. Latent profiles reached 21.11, compared with 20.39 for the matched model without the residual edit.

Profile-inserted SFT reached 20.74 Macro F1 with the target user's profile, compared with 19.23 using another user's profile. Steering reached 21.77 compared with 20.66 for its zero-intervention control. The primary 95% bootstrap intervals included zero for both the own-minus-donor and steering-minus-control comparisons.

Our method (Section 5) achieved the highest accuracy and Macro F1 point estimates in Table 2, at 28.04% and 21.83, respectively. Its Macro F1 exceeded direct generation by 1.30 points and direct

scoring by 2.56 points. The primary paired comparisons against the direct-branch and different-user evidence controls did not pass Holm correction (Appendix C.4).

6.3 Ablation Study

Table 3: **Ablations of direct and evidence scores on Qwen3-8B.** We report accuracy and Macro F1 (%) on the test set. Branch ablations set one coefficient to zero. History substitutions keep the direct scores and full-method coefficients fixed. Different-user results average three donor assignments. The final row appends the same retrieved comments to the direct prompt. Shading identifies the full method. Details are in Appendix C.4.

Variant	Accuracy↑	Macro F1↑
Full method ($aB + bS$)	28.04	21.83
Direct scores only ($b = 0$)	25.48	19.00
Evidence scores only ($a = 0$)	23.94	20.06
Different-user history	25.65	19.93
Recent history	27.91	22.30
Retrieved comments in direct prompt	27.53	19.94

Personal history and retrieval. Table 3examines both the contribution of additional history and the importance of matching that evidence to the target user. With the original annotated history retained in the direct branch, removing retrieved evidence reduced Macro F1 from 21.83 to 19.00. Replacing only the reader's evidence with another user's retrieved comments yielded 19.93. These comparisons suggest that reading additional evidence from the target user complements the personal context already available to the direct predictor. Recent-history selection also performed well, reaching 22.30 Macro F1 compared with 21.83 for similarity-based retrieval, highlighting the utility of a simple recency-based selection rule.

We also compare the source and selection of the reader's historical evidence. The direct branch retains the same annotated history in all comparisons. Macro F1 was 21.83 with the target user's retrieved comments, 19.00 without additional retrieved evidence, and 19.93 with another user's comments. Using the target user's most recent comments reached 22.30 Macro F1.

Method components. Table 3 examines branch fusion, separate evidence reading, relevance weighting, and pooling. The fused model reached 21.83 Macro F1, compared with 19.00 for direct-only and 20.06 for evidence-only prediction, suggesting that explicit historical assessments complement the direct predictor's joint view of the discussion and annotated history. Prompt concatenation used the same retrieved comments but reached 19.94, indicating a possible benefit from assessing comments separately before fusion. Uniform log pooling reduced Macro F1 to 20.62, consistent with relevance weights limiting the influence of uninformative comments. Weighted voting reached 20.96, whereas weighted probability pooling nearly matched weighted log pooling at 21.75. Thus, the larger observed differences concern combining the two score sources and reading evidence explicitly; the precise choice between log and probability pooling had little effect on the point estimate. Ablation definitions and paired comparisons are provided in Appendix C.4.

7 Conclusion

In this paper, we introduced Stance-Bench to examine the limits of test-time scaling and post-training for individual stance prediction. Our analysis identifies four failure modes: incorrect consensus, selection failure, response overfitting, and early plateau. Guided by these findings, we explored an explicit evidence reader that combines direct stance scores with assessments of personal history, achieving higher Macro F1 than direct scoring while keeping the language model frozen. These exploratory results highlight explicit assessment of personal evidence as a promising direction and motivate evaluating candidate generation, final selection, and person-specific evidence use separately when developing individual social simulations.

AI USE STATEMENT

We used generative AI for benchmark annotation, including constructing discussion-specific stance taxonomies and assigning reference labels, as described in Section 3. Annotation quality was assessed through the auditing procedures reported in the paper. We also used generative AI tools to assist with code implementation and to improve the language and clarity of the manuscript. The authors take full responsibility for the research and its final presentation, including all AI-assisted annotations, code, text, and claims.

ETHICS STATEMENT

This study uses publicly available discussions and comments from Hacker News, collected through its official API with support from the Algolia index. We anonymized user identifiers in the benchmark to protect user privacy. We also recruited volunteer annotators, who participated voluntarily and were fully informed about the purpose of the study and their annotation tasks. The benchmark is intended to evaluate individual stance prediction in a research setting; model predictions should not be treated as verified personal beliefs or used to make consequential decisions about individuals.

REPRODUCIBILITY STATEMENT

We describe data collection, stance annotation, task construction, and evaluation metrics in Section 3. Section 5 specifies the direct scoring and evidence-fusion procedure, while Section 6 describes the experimental comparisons and validation-based model selection. Appendix B provides the prompt templates, and Appendix C documents the shared inputs, evaluation protocol, method adaptations, and ablation procedures.

REFERENCES

Jacy Reese Anthis, Ryan Liu, Sean M. Richardson, Austin C. Kozlowski, Bernard Koch, Erik Brynjolfsson, James Evans, and Michael S. Bernstein. Position: LLM social simulations are a promising research method. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 81005–81034. PMLR, 2025. URL https://proceedings.mlr.press/v267/anthis25a.html.

Lisa P. Argyle, Ethan C. Busby, Nancy Fulda, Joshua R. Gubler, Christopher Rytting, and David Wingate. Out of one, many: Using language models to simulate human samples. *Political Analysis*, 31(3):337–351, 2023. doi: 10.1017/pan.2023.2. URL https://doi.org/10.1017/pan.2023.2.

James Bisbee, Joshua D. Clinton, Cassy Dorff, Brenton Kenkel, and Jennifer M. Larson. Synthetic replacements for human survey data? the perils of large language models. *Political Analysis*, 32(4):401–416, 2024. doi: 10.1017/pan.2024.5. URL https://doi.org/10.1017/pan.2024.5.

Bei Chen, Fengji Zhang, Anh Nguyen, Daoguang Zan, Zeqi Lin, Jian-Guang Lou, and Weizhu Chen. Codet: Code generation with generated tests. *arXiv preprint arXiv:2207.10397*, 2022.

Xinyun Chen, Renat Aksitov, Uri Alon, Jie Ren, Kefan Xiao, Pengcheng Yin, Sushant Prakash, Charles Sutton, Xuezhi Wang, and Denny Zhou. Universal self-consistency for large language model generation. *arXiv preprint arXiv:2311.17311*, 2023.

Zihan Chen, Di Zhu, and Lei Nico Zheng. When synthetic users fail: A cross-domain benchmark of LLM-simulated human survey responses. *arXiv preprint arXiv:2607.26348*, 2026. URL https://arxiv.org/abs/2607.26348.

Yun-Shiuan Chuang, Krirk Nirunwiroj, Zach Studdiford, Agam Goyal, Vincent V. Frigo, Sijia Yang, Dhavan V. Shah, Junjie Hu, and Timothy T. Rogers. Beyond demographics: Aligning role-playing LLM-based agents using human belief networks. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 14010–14026. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.findings-emnlp.819. URL https://aclanthology.org/2024.findings-emnlp.819/.

DeepSeek-AI. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

Ricardo Dominguez-Olmedo, Moritz Hardt, and Celestine Mendler-Dünner. Questioning the survey responses of large language models. In *Advances in Neural Information Processing Systems*, volume 37, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/515c62809e0a29729d7eec26e2916fc0-Abstract-Conference.html.

Chen Gao, Xiaochong Lan, Nian Li, Yuan Yuan, Jingtao Ding, Zhilun Zhou, Fengli Xu, and Yong Li. Large language models empowered agent-based modeling and simulation: A survey and perspectives. *Humanities and Social Sciences Communications*, 11(1):1259, 2024.

Christian Genest, Samaradasa Weerahandi, and James V Zidek. Aggregating opinions through logarithmic pooling. *Theory and decision*, 17(1):61, 1984.

Tiancheng Hu, Joachim Baumann, Lorenzo Lupo, Nigel Collier, Dirk Hovy, and Paul Röttger. Simbench: Benchmarking the ability of large language models to simulate human behaviors. In *International Conference on Learning Representations*, volume 2026, pp. 28290–28341, 2026.

Yujia Li, David Choi, Junyoung Chung, Nate Kushman, Julian Schrittwieser, Rémi Leblond, Tom Eccles, James Keeling, Felix Gimeno, Agustin Dal Lago, Thomas Hubert, Peter Choy, Cyprien de Masson d'Autume, Igor Babuschkin, Xinyun Chen, Po-Sen Huang, Johannes Welbl, Sven Gowal, Alexey Cherepanov, James Molloy, Daniel J. Mankowitz, Esme Sutherland Robson, Pushmeet Kohli, Nando de Freitas, Koray Kavukcuoglu, and Oriol Vinyals. Competition-level code generation with AlphaCode. *Science*, 378(6624):1092–1097, 2022.

Xuan Liu, Jie Zhang, HaoYang Shang, Song Guo, Chengxu Yang, and Quanyan Zhu. Exploring prosocial irrationality for LLM agents: A social cognition view. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=u8VOQVzduP.

Xuan Liu, HaoYang Shang, and Haojian Jin. CoBRA: Programming cognitive bias in social agents using classic social science experiments. In *Proceedings of the 2026 CHI Conference on Human Factors in Computing Systems*, CHI '26, New York, NY, USA, 2026a. Association for Computing Machinery. doi: 10.1145/3772318.3790804. URL https://doi.org/10.1145/3772318.3790804.

Xuan Liu, Haoyang Shang, Zizhang Liu, Xinyan Liu, Yunze Xiao, Yiwen Tu, and Haojian Jin. HumanStudy-Bench: Towards AI agent design for participant simulation, 2026b. URL https://arxiv.org/abs/2602.00685.

Siyuan Brandon Loh, Liang Ze Wong, Prasanta Bhattacharya, Joseph Simons, Wei Gao, and Hong Zhang. Predicting user stances from target-agnostic information using large language models. *arXiv preprint arXiv:2409.14395*, 2024.

Yuxuan Lu, Jing Huang, Yan Han, Bingsheng Yao, Sisong Bei, Yaochen Xie, Yisi Sang, Qi He, and Dakuo Wang. Can llm agents simulate multi-turn human behavior? evidence from real online customer behavior data. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 43961–43977, 2026.

Daniel McFadden. Conditional logit analysis of qualitative choice behavior. In Paul Zarembka (ed.), *Frontiers in Econometrics*, pp. 105–142. Academic Press, New York, 1974.

Saif Mohammad, Svetlana Kiritchenko, Parinaz Sobhani, Xiaodan Zhu, and Colin Cherry. Semeval-2016 task 6: Detecting stance in tweets. In *Proceedings of the 10th international workshop on semantic evaluation (SemEval-2016)*, pp. 31–41, 2016.

Ansong Ni, Srini Iyer, Dragomir Radev, Ves Stoyanov, Wen-tau Yih, Sida I. Wang, and Xi Victoria Lin. LEVER: Learning to verify language-to-code generation with execution. In *Proceedings of the 40th International Conference on Machine Learning*, pp. 26106–26128, 2023. URL https://proceedings.mlr.press/v202/ni23b.html.

OpenAI. Gpt-5.5 system card. Technical report, OpenAI, April 2026. URL https://deploymentsafety.openai.com/gpt-5-5/gpt-5-5.pdf. Accessed: 2026-09-23.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35: 27730–27744, 2022.

Joon Sung Park, Joseph C. O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, UIST '23. Association for Computing Machinery, 2023. doi: 10.1145/3586183.3606763. URL https://doi.org/10.1145/3586183.3606763.

Joon Sung Park, Carolyn Q. Zou, Jonne Kamphorst, Niles Egan, Aaron Shaw, Benjamin Mako Hill, Carrie Cai, Meredith Ringel Morris, Percy Liang, Robb Willer, and Michael S. Bernstein. LLM agents grounded in self-reports enable general-purpose simulation of individuals, 2026. URL https://arxiv.org/abs/2411.10109v3. Version 3; first posted in 2024.

Tal Ridnik, Dedy Kredo, and Itamar Friedman. Code generation with alphacodium: From prompt engineering to flow engineering. *arXiv preprint arXiv:2401.08500*, 2024.

Alireza Salemi, Sheshera Mysore, Michael Bendersky, and Hamed Zamani. Lamp: When large language models meet personalization. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7370–7392, 2024.

Shibani Santurkar, Esin Durmus, Faisal Ladhak, Cinoo Lee, Percy Liang, and Tatsunori Hashimoto. Whose opinions do language models reflect? In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 29971–30004. PMLR, 2023. URL https://proceedings.mlr.press/v202/santurkar23a.html.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024. URL https://arxiv.org/abs/2402.03300.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling parameters for reasoning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=4FWAwZtd2n.

Evan Wang, Federico Cassano, Catherine Wu, Yunfeng Bai, William Song, Vaskar Nath, Ziwen Han, Sean Hendryx, Summer Yue, and Hugh Zhang. Planning in natural language improves llm search for code generation. In *International Conference on Learning Representations*, volume 2025, pp. 2432–2478, 2025.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=1PL1NIMMrw.

Jiayi Zhang, Simon Yu, Derek Chong, Anthony Sicilia, Michael R. Tomz, Christopher D. Manning, and Weiyan Shi. Verbalized sampling: How to mitigate mode collapse and unlock LLM diversity. *arXiv preprint arXiv:2510.01171*, 2025. URL https://arxiv.org/abs/2510.01171.

Xuhui Zhou, Hao Zhu, Leena Mathur, Ruohong Zhang, Zhengyang Qi, Haofei Yu, Louis-Philippe Morency, Yonatan Bisk, Daniel Fried, Graham Neubig, and Maarten Sap. SOTOPIA: Interactive evaluation for social intelligence in language agents. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=mM7VurbA4r.

A Diagnostic Coverage of Existing Evaluations

Existing evaluations address complementary aspects of person-specific prediction. Table 4 compares their reported protocols using the three diagnostic requirements introduced in Section 2.2, alongside person-level targets and naturally occurring observations. The entries describe the evaluation settings as reported; a question mark indicates that the relevant support is unclear.

Table 4: Diagnostic coverage of existing evaluations and STANCE-BENCH. R1: users compared within a shared context; R2: personal evidence removed or replaced with the prediction target fixed; R3: candidate and final predictions recorded separately in a shared stance space. ULSP (CB): user-level stance prediction on Connected Behaviour. ✓: supported; ◐: partial; ✗: unsupported; ?: unclear from the reported evaluation.

Evaluation	Person-level targets	Shared context (R1)	Evidence controls (R2)	Candidate/final traces (R3)	Naturally occurring
OpinionQA (Santurkar et al., 2023)	✗	✓	✗	✗	✗
SimBench (Hu et al., 2026)	✗	✓	✗	✗	◐
Park et al. (2026)	✓	✓	?	✗	✗
LaMP (Salemi et al., 2024)	✓	✗	?	✗	✓
SemEval-2016 (Mohammad et al., 2016)	✗	◐	✗	✗	✓
ULSP (CB) (Loh et al., 2024)	✓	◐	?	✗	✓
STANCE-BENCH (ours)	✓	✓	✓	✓	✓

STANCE-BENCH combines these requirements within a common evaluation setting. Multiple users are linked to the same naturally occurring discussion, and their preceding histories can be removed or replaced while the discussion and reference stance remain fixed. For methods that generate multiple candidates, the evaluation records each candidate's stance label and the final decision separately. Together, these properties support controlled analyses of personal evidence and distinguish candidate coverage from final selection.

B Prompt Templates

The box below presents the prompt templates used for stance prediction, historical evidence reading, and label-guided comment generation. Headings identify stages and message roles.

Prompt templates

```
[Stance prediction: System prompt]
You are simulating one specific Hacker News user (the subject) from
    their comment history. Predict how they will respond in a new
    discussion.
Return JSON with:
- reasoning: at most three sentences connecting their history to this
     prediction.
- predicted_comment: the comment this subject would most plausibly
    write, in their own voice and typical length. Write the comment
    itself, never a description of it.
- label_id: the candidate position that comment takes. Use OTHER if
    their position is not listed.
The comment and the label must express the same stance.

[Stance prediction: User prompt]
== SUBJECT HISTORY: their 8 most recent labeled discussions, oldest
    first ==
[1] Discussion: <historical_title>  (<days_before> days before)
    Argued over: <historical_central_question>
    Position the subject took: <position_taken>
    Positions they did not take: <alternative_1> | <alternative_2> |
    ...
    Subject wrote: "<historical_comment>"
```

```
<remaining_history_entries_in_the_same_format>

== NEW DISCUSSION the subject is about to comment in ==
Title: <target_title>
Link domain: <link_domain>
Posted: <YYYY-MM-DD>
Story text:
<story_text>
Comments already in the thread (as a visitor would first see it):
  - <visible_comment_1>
  - <visible_comment_2>
  ...
The discussion's central question: <target_central_question>

== CANDIDATE POSITIONS: the subject's comment will take exactly one (
    listed in no particular order) ==
<candidate_rows>

[Direct label scoring: Assistant prefix]
{"label_id": "

[Direct label scoring: Candidate continuation]
<candidate_id>", "

[Historical evidence reading: System prompt]
You infer a Hacker News user's likely position in a discussion from
    one comment they wrote elsewhere. Answer in JSON.

[Historical evidence reading: User prompt]
Central question: <target_central_question>
Discussion: <target_title>

Positions:
[A] <position_name_A>: <definition_A> (Not this if: <boundary_A>)
[B] <position_name_B>: <definition_B> (Not this if: <boundary_B>)
...
[U] The comment says nothing about which of these positions this
    person holds.

This person wrote the following comment earlier, elsewhere on Hacker
    News:
"<retrieved_historical_comment>"

Which of the positions above is this person most likely to take in
    the discussion? Return JSON {"answer": "<letter>"}.

[Historical evidence reading: Assistant prefix]
{"answer": "

[Label-guided comment generation: Append to prediction user prompt]
The subject's position in this discussion has been determined to be
    [<predicted_label_id>] <predicted_label_name>. Write the comment
    they would post taking that position.

For this step, write only the subject's comment itself as plain text:
     no JSON, no reasoning, no label, no quotation marks around it.
```

C Method Adaptations and Experimental Details

C.1 Shared inputs and evaluation

Methods receive the target discussion, its candidate stance definitions, and the user's preceding annotated discussions. The main comparisons use the eight most recent eligible historical discussions. Generative methods produce a predicted comment and a declared stance label; direct-scoring methods select a label without generating a comment. User histories precede the user's first contribution to the target discussion, and target-user contributions are excluded from the model-visible target context.

Evaluation. All methods follow the shared evaluation protocol in Section 6.

C.2 Direct prediction and test-time scaling

The TTS adaptations use the corresponding frozen backbone in Table 1; Direct scoring is evaluated on Qwen3-8B. Fixed-pool methods use four responses, while AlphaCodium refines a single draft. Predictions are returned in structured form with a comment and its declared stance label. For methods originating in code generation, we replace programs with predicted comments and implement selection or checking through stance agreement and response properties.

Direct generation and Direct scoring. Direct generation makes one call to produce a comment in the user's voice, a short rationale grounded in the history, and a stance label consistent with the comment. Its declared label is the final prediction. Direct scoring instead appends each candidate label continuation to a shared prediction prompt with a fixed label-first output prefix. It sums the conditional token log-probabilities of each continuation, including its closing delimiter, and selects the highest-scoring label under the frozen model.

Verbalized Sampling. We ask the model to propose four diverse plausible responses and assign a probability to each in a single call (Zhang et al., 2025). Each response contains a comment, its stance label, and a brief explanation connecting the prediction to the user's history. The prompt encourages distinct predictions and asks for probabilities that approximately sum to one. We normalize the returned probabilities over the four responses and select the response with the highest value, resolving ties by candidate order.

Universal Self-Consistency. We independently sample four comment-and-label predictions and make an additional selection call to the same backbone (Chen et al., 2023). The selector receives the original input together with the indexed candidate labels and comments, with each comment limited to 700 characters. It identifies the candidate whose stance and supporting reasoning best reflect agreement across the responses and returns its index with a short explanation. Selection uses temperature zero, and the chosen candidate's declared label is the final prediction. The procedure uses five model calls.

AlphaCode-style consensus. We adapt behavioral clustering (Li et al., 2022) by independently generating four responses and grouping them by their declared stance labels. Group size measures agreement on the predicted position. We select the first candidate in the largest group; when groups have equal size, the group containing the earliest candidate wins. This procedure uses four generation calls, with grouping and selection performed directly from their labels.

PlanSearch. We adapt the observation–hypothesis–generation pipeline to comment prediction (Wang et al., 2025). The model first extracts four to eight observations from the user's history, covering values, recurring concerns, and argumentative habits. Using these observations and the target discussion, it then proposes four distinct hypotheses, each comprising a stance summary and a short argument sketch. Each hypothesis conditions a separate call that produces a comment, rationale, and label; the label is assigned according to the generated comment. The four responses are selected by the same label-consensus rule as above, including its ordering-based tie rule. Observation extraction uses temperature 0.7, and hypothesis and response generation use temperature 0.95. The procedure uses six model calls.

AlphaCodium. We adapt reflection and iterative repair to predicted comments (Ridnik et al., 2024). The model first summarizes the user's characteristic views and argumentative habits, interprets the discussion, and ranks up to four plausible positions. This reflection conditions an initial comment and stance prediction. A separate call derives three to six acceptance checks from the history and discussion, covering stance, argument type, register, typical length, and writing habits. The model then performs up to two rounds of checking and revision: each round identifies failed checks and returns a revised comment and label while preserving properties already satisfied. Refinement stops early, and the final draft is returned. Reflection uses temperature 0.4, check generation uses 0.6, and initial prediction and revision use 0.95. The procedure uses three preparatory calls and up to two repair calls.

CodeT. We independently generate four candidate responses and a separate set of yes/no assertions describing properties of the comment the user is expected to write (Chen et al., 2022). Assertion generation receives the original input without the candidate responses. Each candidate is then checked in a separate call containing its comment and the numbered assertions; the checker returns the indices of satisfied assertions at temperature zero. Candidates satisfying the same assertion set form a group G, scored by $\sqrt{|G|\,|A_G|}$, where A_G is their shared set of satisfied assertions. We return the first candidate in the highest-scoring group, breaking score ties by group size and then candidate order. We use eight, four, and three assertions for Qwen3-8B, Gemini Flash, and Gemini Pro, respectively. The procedure uses nine calls: four for candidates, one for assertions, and four for checking.

Decoding settings. Direct generation uses temperature 0.2; TTS candidate generation and CodeT assertion generation use temperature 0.95. Stage-specific temperatures are given above. For Qwen3-8B, the output cap per call is 3,072 tokens for Verbalized Sampling, 1,536 for AlphaCodium, and 1,024 for Direct generation and the remaining TTS methods. Gemini Pro uses a 32,000-token output cap per call. Qwen3-8B runs use the non-thinking chat template and generation seeds 17, 18, and 19.

C.3 Learning and adaptation strategies

The learning and adaptation baselines share the Qwen3-8B backbone and use an eight-discussion history window. SFT, GRPO, and profile-method checkpoints, together with steering strengths, are selected using validation Macro F1.

All trained LoRA adapters target the attention query, key, value, and output projections and the feed-forward gate, up, and down projections. SFT, SFT + user profile, and LEVER use rank $r = 64$, scaling parameter $\alpha = 128$, and dropout 0.05. GRPO uses $r = 32$, $\alpha = 64$, and zero dropout. Latent profile keeps the backbone frozen, and Steering vector requires no gradient-based training. Table 5 specifies test decoding; maximum generation lengths count newly generated tokens, whereas training sequence limits include both prompt and response.

LEVER. We adapt learned verification (Ni et al., 2023) by training a verifier with LoRA adapters and a classification head to predict whether a response's declared label matches the observed stance. It receives the task context and candidate response; label correctness replaces program-execution feedback. Both variants rank the same pool of 16 responses. LEVER (verifier) selects the highest predicted correctness score; LEVER (product) multiplies that score by the exponentiated, length-normalized generation log-probability before selection. We train for one epoch with AdamW at learning rate 10^{-4}, zero weight decay, 5% linear warmup followed by linear decay, and a maximum sequence length of 4,608 tokens. Two devices each process two pairs per microbatch, with eight accumulation steps, giving an effective batch of 32 pairs. The loss is binary cross-entropy with positive-class weight 3.5486. The saved verifier is the sole epoch-end checkpoint; its validation selection criterion is accuracy under label-marginal selection.

SFT. We train LoRA adapters to generate the observed user comment followed by its stance label, conditioned on the discussion and preceding history. At test time, we generate a response and evaluate its declared label, preserving the comment-and-label formulation studied in Section 4.3. Training runs for three epochs at learning rate 10^{-4}, with cosine decay and 5% warmup. The per-device batch is one task, accumulated over eight steps on one device (effective batch eight); the

maximum sequence length is 5,120 tokens. Loss is computed on response tokens. We evaluate each epoch-end checkpoint using generated stance predictions on validation data.

GRPO. We adapt group-relative reward-based learning (Shao et al., 2024) to stance attribution by sampling multiple responses to each training prompt and rewarding exact agreement with the observed stance label. Rewards are $+1$ for a correct label, 0 for a valid incorrect label, and -1 for an invalid output. LoRA policy updates use within-prompt reward differences with KL coefficient $\beta = 0.02$ toward the frozen base model. We train for 300 updates using AdamW at constant learning rate 3×10^{-5}, optimizer coefficients $(0.9, 0.99)$, zero weight decay, and gradient clipping at 1.0. Each update accumulates eight prompts with eight responses per prompt (64 responses). Rollouts use temperature 1.0, top-$p = 1.0$, no top-k truncation, and at most 384 new tokens. Checkpoints are evaluated every 50 updates, sampling four responses per validation prompt with the same decoding settings and scoring only the first. Test generation samples eight responses with the settings in Table 5, again scoring only the first response.

SFT + user profile. We summarize each user's preceding history as a textual profile of recurring preferences and behavioral tendencies, supported by historical evidence, and insert it into the predictor's prompt. LoRA training uses generated rationales together with observed comments and stance labels. Rationales come from label-correct samples or explanations generated with the observed outcome supplied during training. Profiles are generated by Qwen3-8B at temperature zero from the eight historical discussions alone; the inserted profile retains up to two supporting quotations per field. We train with AdamW at learning rate 10^{-4} for two epochs, with 3% warmup, zero weight decay, and gradient clipping at 1.0. The per-device batch is one task with 32 accumulation steps on one device (effective batch 32); the sequence limit is 6,144 tokens. Checkpoints are evaluated every half epoch.

Latent profile. We encode historical comments with the frozen backbone, pool their representations into a user vector, and train a lightweight controller to convert that vector into a residual hidden-state edit. The backbone remains frozen. The profile module is trained with rationale, comment, and label supervision together with an auxiliary training-user identification objective. At test time, the edit is computed from the unseen user's history. The controller uses a 512-dimensional user representation and a rank-eight edit at decoder layer 8 (zero-indexed), with each historical comment capped at 256 tokens. We train only the profile module and auxiliary head for three epochs at learning rate 10^{-3}, using a one-task microbatch, 32 accumulation steps, and a 6,144-token sequence limit. The objective adds user-identification cross-entropy with weight 0.5 and an orthogonality penalty with weight 10^{-3} to response-token loss. Prediction checkpoints are evaluated every half epoch. SFT and both profile methods use no additional KL penalty.

Steering vector. We estimate a shared activation direction by contrasting correct and incorrect stance responses generated for the same training tasks. At inference, we add a validation-selected multiple of this direction to the frozen model's hidden activations. This is a shared correct-position direction, whereas Latent profile produces an edit conditioned on each user's history. The direction is estimated at decoder layer 20 (zero-indexed). We normalize the mean task-paired activation difference to unit length and test multipliers $f \in \{0, 0.15, 0.4, 0.7\}$, scaled by the training activation norm 88.889. Validation selects $f = 0.15$, corresponding to the recorded additive coefficient 13.3333. The intervention is applied at the final prompt token and subsequent generated tokens.

Qwen3-8B generation uses the non-thinking chat template. Direct generation and LEVER candidate sampling use schema-constrained JSON through vLLM; SFT, SFT + user profile, and GRPO use unconstrained vLLM generation, while Latent profile and Steering vector use Hugging Face generation. Direct scoring deterministically selects the label with the highest conditional likelihood. SFT, GRPO, and both profile methods are trained with seeds 1, 2, and 3. Their selected checkpoints are, respectively, epochs 1/3/1, updates 250/300/200, epoch 1 for all three profile-SFT runs, and half an epoch for all three latent-profile runs. Direct generation and Steering vector use generation seeds 17, 18, and 19; the trained vLLM models use test-generation seed 17, and latent-profile generation uses $17+$ batch index.

Table 5: **Test decoding for Qwen3-8B baselines.** T denotes temperature; p and k_{top} denote nucleus and top-k sampling. Token caps apply per response. "Default" denotes the generation setting inherited from the loaded model or server configuration.

Method	T	p	k_{top}	Token cap	Prediction readout
Direct generation	0.2	Default	Default	1,024	One response
LEVER (both variants)	0.95	Default	Default	512	Rank 16 responses
SFT / SFT + user profile	0.95	0.95	20	2,048	One response
GRPO	1.0	0.95	20	384	First of eight responses
Latent profile	0.95	1.0	Default	2,048	One response
Steering vector	0.95	1.0	Default	2,048	One response

C.4 Our method and its ablations

Retrieving and reading personal evidence. The direct branch scores labels using the discussion and annotated history. The evidence branch separately retrieves up to eight comments from the user's public history by similarity to the target discussion, retaining at most two per historical discussion (Salemi et al., 2024). Retrieved comments precede the user's first target-discussion contribution, exclude the target discussion, and require no historical stance annotations. The frozen reader assesses each comment against the target stance definitions and an additional option U for insufficient evidence. We average aligned log-probabilities across option orders and renormalize over listed stances to obtain p_i. The weight is $w_i = 1 - \exp\{\overline{\log q_i(U)}\}$, where the bar denotes the same order averaging. These quantities define the weighted log-probability mean in Eq. equation 2.

Fusion and prediction. Only the two fusion coefficients are fitted, using regularized stance negative log-likelihood on development data; both language models remain frozen. Evidence scores are centered before fusion, and the fitted coefficients are fixed during inference. If no usable evidence is available, the full method uses zero evidence scores.

Matched ablations. Direct scores only and Evidence scores only remove the evidence and direct branches, respectively. Their remaining coefficients are fitted on development data; using the corresponding full-method coefficients gives identical predictions because these fitted coefficients are positive. Evidence-only prediction never falls back to the direct branch. Different-user history replaces only the evidence source, while Recent history replaces relevance retrieval with the latest eligible comments; both retain the full-method coefficients and direct scores. Retrieved comments in direct prompt appends the same comments to the direct predictor instead of scoring them separately. The Direct scoring baseline and the direct-only ablation use different label-scoring implementations, so each is retained as its own reported comparison.

The primary paired Macro F1 differences are $+2.83$ points against Direct scores only (95% interval $[0.14, 5.28]$) and $+1.89$ against Different-user history ($[-0.17, 3.90]$). Neither passes Holm correction across these two comparisons. The differences against Evidence scores only ($[-0.36, 4.69]$) and Retrieved comments in direct prompt ($[-0.97, 4.07]$) also have intervals including zero.