CONCORD: DISTILLING AND COORDINATING PERSONA SKILLS FROM MULTI-USER WORK RECORDS

Anonymous authors
Paper under double-blind review

ABSTRACT

Persona skills distill working practices and preferences from human work records into reusable instructions for Large Language Model (LLM) agents. Recent methods combine records from multiple users to provide broader knowledge, but different users may recommend incompatible actions for the same situation. Existing methods do not explicitly preserve and coordinate such conflicts, and can leave agents uncertain about which practice to follow. We present ConCord, an automated framework for constructing and coordinating persona skills from multi-user work records. ConCord first extracts context-dependent decision rules from each user's records, and then groups users with similar practices and constructs one skill for each group, sharing knowledge across users while preserving conflicting approaches in separate skills. For each task, separate LLM agents independently generate candidate responses using different skills. A merging LLM then combines the responses and resolves conflicting recommendations based on the task requirements. We evaluated ConCord on four datasets of user work records under different scenarios that cover workplace communication, software issue handling, and computer-science question answering. ConCord achieves an average score improvement of 0.34 over responses generated without skills, compared with 0.14 for the strongest baseline. These results show that preserving conflicting practices during skill construction and resolving conflicts for each task improves multi-user persona skill coordination. We release the replication package on the anonymous link: `https://anonymous.4open.science/r/ConCord-F060`.

1 INTRODUCTION

Agent skills (Li et al., 2026c; Xu & Yan, 2026; Li et al., 2026b; Wang et al., 2024; 2025a) are reusable task-level artifacts that package prior knowledge, including procedures, tool-use strategies, and domain expertise, for use across related tasks. Instead of solving each task from scratch, a skill-equipped agent can follow structured patterns distilled from earlier successful experience. Manually specifying an agent's working rules for skill design often requires domain expertise and carefully designed instructions (Chen et al., 2024). Human work records offer an alternative source, as human work records show how individuals approach problems and make decisions (Zhou et al., 2026; Ni et al., 2026). LLMs can extract working practices and user preferences from these records and write them as reusable instructions to construct the skills (Zhou et al., 2026; Gao et al., 2024), which are referred to as *persona skills* (Zhou et al., 2026; Xiang et al., 2026).

Recent work has explored several ways to use individual and collective experience from human working records to construct persona skills. Huang et al. (2026) studied whether persona skills help coding agents complete tasks. Beyond distilling skills from one person's work records, recent work also considers records from multiple users (Huang et al., 2026; Ma et al., 2026). Different users contribute knowledge from different situations, which may not be available in any single person's records. Combining these records allows the resulting skills to draw on a broader range of problem-solving methods. This can help agents handle more varied tasks, achieving greater performance gains than a single-user skill (distilled from one's working records) (Huang et al., 2026).

However, various people may have different preferences for processing a task. For example, when handling an uncertain request, one user may prefer gathering more evidence before making a decision, whereas another may prefer acting quickly and revising the decision if needed. These choices

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.

reflect different working preferences, and neither is always preferable. The appropriate choice depends on factors such as urgency and the costs of delay or error. Existing methods (Huang et al., 2026; Ma et al., 2026) do not address how to coordinate conflicting practices. Section 2.2 shows that when a skill holds both practices, the agent notices the conflict but mostly follows one of them, so the skill loses much of what the other practice offers. Thus, we propose ConCord, an automated framework that uses LLMs for coordinating skills from multi-user work records.

In designing ConCord, we need to solve three challenges. (1) *How can conflicting working practices be identified across users' records?* Differences in users' working records may reflect different task contexts rather than conflicting working habits. Identifying conflicts requires summarizing each user's habits, removing irrelevant task details. (2) *How can different working practices be preserved?* Conflicting persona skill instructions may each be valid, yet still recommend incompatible actions. If conflicting human practices are deleted or only one preference is kept when constructing the skill, the resulting skill may be less useful for tasks that require alternative methods. (3) *How can the distilled skills be coordinated for the tasks?* The distilled skill involves conflicting instructions. ConCord should determine how to coordinate the conflicting skills based on the current task.

For the *first* challenge, ConCord uses LLMs to extract decision rules from each user's work records, specifying what action to take and when it applies. We organize each user's rules by the types of decisions they address, such as how much to verify before acting and when to escalate an issue. The extracted decision rules form a stance profile for each user. Rather than explicitly listing pairwise conflicts, ConCord uses multiple voting agents to examine all users' stance profiles together and group users who are likely to make similar choices. Users whose profiles imply different actions are placed in different groups, allowing conflicting practices to be captured through the grouping. To address the *second* challenge, ConCord uses the decision rules extracted in the first stage to group users with similar working practices. Inspired by mixture-of-experts models (Jacobs et al., 1991; Shazeer et al., 2017), it constructs one skill for each group. Each skill draws on the work summaries of all users, while its own group's decision rules guide its choices where users disagree. This shares knowledge across users while preserving different approaches in separate skills, without requiring a separate skill for every user. To address the *third* challenge, skills generate answers independently rather than being selected in advance. A merging LLM receives these drafts, combines compatible details, and chooses between conflicting recommendations based on the task's requirements. We provide more technical details of ConCord in Section 3.3.

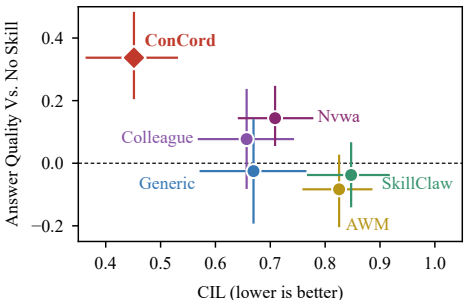

Figure 1: Overall comparison of ConCord with the baseline methods.

We evaluate ConCord on four datasets spanning three real-world domains, including workplace email (Klimt & Yang, 2004), software issue reports (Claes & Mäntylä, 2020), and computer science question answering (Stack Exchange, Inc., 2024), as well as the synthetic role-play benchmark `AntiSkillBench` (Xiang et al., 2026). In particular, as shown in Figure 1, ConCord improves the averaged pairwise comparison score by 0.34 over the no-skill responses, compared with an improvement of 0.14 for Nvwa (xmg, 2026), the strongest of the five baselines, while keeping the lowest conflict score (CIL). The evaluation metrics are introduced in Section 4.1.

2 Background

2.1 Persona Skill Distillation and Coordination

Persona Agent Skill. Persona skills (Zhou et al., 2026) package information from an individual's working documents and prior interactions into reusable artifacts that can be integrated with an agent. By carrying person-specific knowledge across tasks, they help agents provide assistance that remains consistent with the human's behavior (xmg, 2026; Xiang et al., 2026). Formally, the set of working records of an individual u is represented by $T_u = \{t_1, t_2, \ldots, t_n\}$, where each t_i represents one working record and n indicates the number of records. A distillation method D maps the working record set T_u into a persona skill $s_u = D(T_u)$. The skill s_u summarizes the personal context and

Table 1: Comparison of existing skill construction methods and ConCord. 'Multi-User' is the integration of working records from multiple people. 'Conflict-Aware' means considering disagreements between source people's documents. 'Prac. Sep. (Practice Separation)' indicates whether different working practices are recorded in separate skills rather than retained together in one. 'Coordination' means whether the method coordinates suggestions from these skills for a specific task.

Skill Construction Method	Input	Output	Multi-User	Conflict-Aware	Prac. Sep.	Coordination
AWM (Wang et al., 2025b)	Working Records	Memory	✗	✗	✗	✗
Nvwa (xmg, 2026)	Working Records	Skill	✓	✓	✗	✗
Colleague Skill (Zhou et al., 2026)	Working Records	Skill	✗	✗	✗	✗
Generic Skill (Huang et al., 2026)	Working Records	Skill	✓	✗	✗	✗
SkillClaw (Ma et al., 2026)	Working Records / Skills	Skill	✓	✗	✗	✗
ConCord (Ours)	Working Records	Skill	✓	✓	✓	✓

behavioral patterns inferred from T_u. After incorporating s_u into an agent A, the agent produces $y = A(x; s_u)$ for a given task x, and outputs the suggestion y.

Multi-User Skill Distillation and Coordination. When there are working records from multiple people, one approach is to combine their records into one set and distill a shared skill (Huang et al., 2026; xmg, 2026; Ma et al., 2026). For a set of users U, a shared persona skill can be constructed from their combined work records as $s_U = D(\bigcup_{u \in U} T_u)$, where D denotes the distillation method.

As summarized in Table 1, current methods do not preserve alternative practices and coordinate them according to the current task. Some methods merge multi-user records into one unified skill, while methods such as Nvwa (xmg, 2026) retain different practices within a shared skill but do not explicitly separate and coordinate them when using the skill-equipped agent to solve tasks. As a result, the agent must interpret conflicting instructions within a single skill without a clear mechanism for deciding which practice fits the task better. ConCord combines working records from multiple users while preserving different ways of working. For each task, it combines compatible guidance and uses the task requirements to choose between conflicting rules in skills.

2.2 Conflicts in Skills

There may exist conflicts in working traces of multiple users because different people may have different preferences for processing a task. We represent a distilled skill s_u as a set of context-dependent decision rules $R(s_u)$. Each rule $r \in R(s_u)$ is the type of decisions it addresses, the task situations in which it applies, and the action it recommends. Let d_r and a_r denote the decision type and the recommended action of rule r, respectively. Two rules r and r' conflict on a task x, $r \bowtie_x r'$, if they address the same decision, both apply to x, but recommend different actions. That is,

$$r \bowtie_x r' \iff \{d_r = d_{r'}\} \wedge \{r \text{ and } r' \text{ both apply to } x\} \wedge \{a_r \neq a_{r'}\}. \quad (1)$$

Figure 2 shows an example extracted from the work records of users in `20-MAD`, a dataset linking software issue reports (Claes & Mäntylä, 2020). Rule A favors minimal, focused changes, whereas Rule B favors broad, comprehensive changes. We evaluate 'DeepSeek-V4-Pro' with reasoning enabled on 72 software issue tasks under three settings: Rule A, Rule B, and both rules. When both rules are included, 94% of the reasoning traces consider both rules, but 86% of the final answers follow Rule A. Consistent with this preference, the combined-rule setting is not significantly worse than using Rule A alone. However, it scores 0.27 lower than the average of the Rule-A-only and Rule-B-only settings. Placing both rules in skills does not allow the model to benefit fully from the alternative practice; instead, its behavior remains close to Rule A on most tasks.

3 Method

3.1 Motivation

This section introduces the motivation of ConCord. A straightforward way to handle conflicts among working practices is to remove conflicting preferences or retain only one during skill construction. However, doing so may discard useful rules and reduce the performance of the resulting skill. An alternative is to preserve conflicting instructions within a single shared skill and specify the conditions under which each rule should apply. However, conditions inferred from limited work records

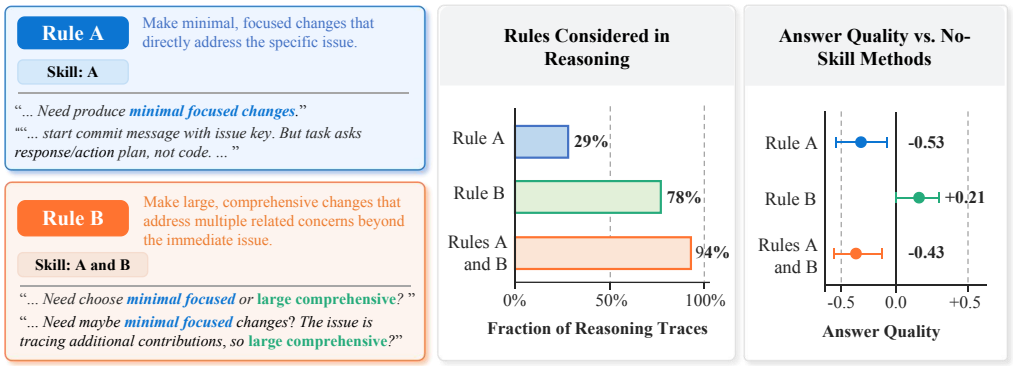

Figure 2: Conflicting developer working practices in the `20-MAD` dataset. "Rules Considered in Reasoning" reports the percentage of traces that explicitly consider the tested rule. For example, 29% of traces consider Rule A when only Rule A is provided. Across downstream tasks, conflicting rules increase explicit tradeoff reasoning but reduce answer quality relative to the no-skill baseline.

may not clearly distinguish when each rule should apply. As a result, multiple conflicting rules may appear relevant to the same task, leaving the agent to choose among them without explicit coordination. As more users and rules are added, the skill also becomes longer, and one practice may dominate rather than being selected according to the task.

ConCord addresses these limitations by keeping different working practices in separate skills and choosing among their suggestions based on the current task. Mixture-of-experts models use several specialized components and combine their outputs (Jacobs et al., 1991; Shazeer et al., 2017). Following this idea, ConCord groups users who make similar choices when working practices conflict and constructs one skill for each group. For a new task, the skills independently generate complete candidate responses, after which a merging LLM combines compatible content and selects between conflicting recommendations based on the task requirements. Specifically, experiments in Section 4.3 show that ConCord achieves higher response quality than these alternatives on the studied datasets, while achieving comparable CIL scores.

3.2 Overview

As shown in Figure 3, ConCord operates in two stages. During the '*skill distillation*' stage, LLM agents analyze each user's records separately to summarize their knowledge and working practices. Several agents then examine these summaries together and propose groups of users with similar practices. Another agent compares the proposals and determines the final groups. An LLM constructs each group's skill using summaries from all users' records. It is instructed to include relevant knowledge from all users while following the group's preferences when users recommend different actions for the same decision. Formally, given work records $\{T_u\}_{u \in U}$ from users $U = \{u_1, \ldots, u_m\}$, ConCord groups users with similar working practices and constructs a set of skills $\mathcal{E} = \{E_1, \ldots, E_k\}$, where k is the number of groups and each skill represents one group.

During the '*skill use*' stage, ConCord assigns each skill to an LLM agent. All agents receive the same task description and independently write an answer using their assigned skill. A merging LLM agent then receives all answers and the task description. It combines details that can be used together. When the answers recommend different actions for the same decision, the merging LLM agent selects the final decided recommendation that best matches the task requirements.

3.3 Technical Details

Analyzing Each User's Work Records. For each user $u \in U$, an agent produces $H_u = \text{Analyze}(T_u)$, summarizing the user's knowledge, workflow, and working practices for skill construction. For user grouping, let $\mathcal{D} = \{d_1, \ldots, d_{|\mathcal{D}|}\}$ denote a predefined set of aspects of working practices and each $d_i \in \mathcal{D}$ means a working practice, such as how much to verify before acting and when to escalate an issue. We present how to obtain $\mathcal{D}$ in Appendix A.1. A second agent extracts $R_u = \text{Extract}(T_u; \mathcal{D})$, identifying the user's preferred actions and the conditions under

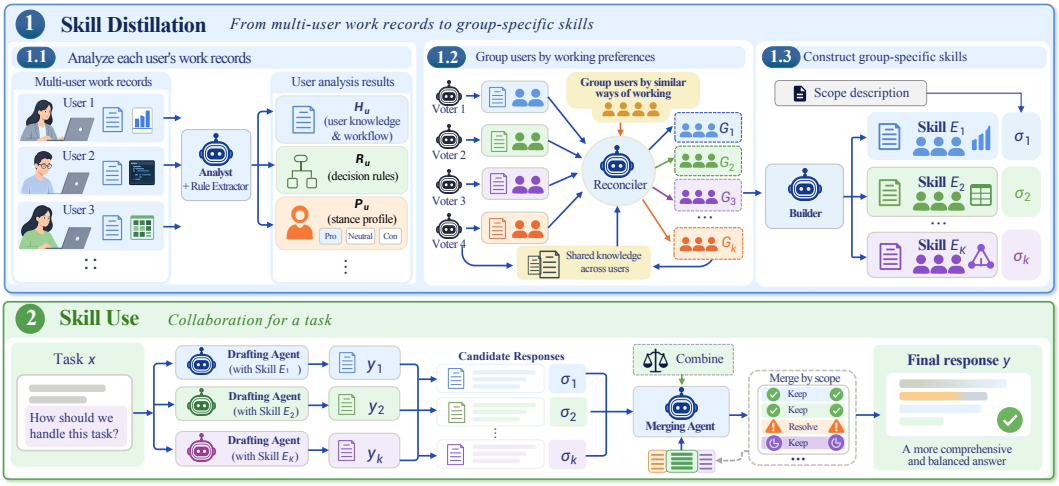

Figure 3: Overview of ConCord. In the skill distillation stage, ConCord groups users by working preferences and constructs skills that share knowledge across users while preserving each group's practices. In the skill use stage, ConCord generates answers independently with each skill, then combines compatible content and resolves conflicts using the task and each skill's scope.

which they apply. The extracted rules are converted into a stance profile, $P_u = \text{Profile}(R_u) = (p_{u,1}, \ldots, p_{u,|\mathcal{D}|})$, where $p_{u,\ell}$ describes the user's position on the rule d_ℓ. These profiles retain working preferences while omitting topic-specific details. The grouping stage receives only the profiles, whereas the analyses H_u retain knowledge for subsequent skill construction.

Grouping Users by Working Practices. ConCord groups users with similar working preferences, using v LLM agents to independently propose groups and a separate agent to review these proposals and determine the final grouping. Let U denote the set of users and $\mathcal{P} = \{P_u\}_{u \in U}$ denote the corresponding set of stance profiles. The v voting agents independently examine all profiles in $\mathcal{P}$ and propose user partitions, $\widehat{\mathcal{G}}_i = \text{Vote}(\mathcal{P}),\ i = \{1, \cdots, v\}$. A reconciler agent reads the profiles and all v proposals to determine the final grouping, $\mathcal{G} = \text{Reconcile}\left(\mathcal{P}, \{\widehat{\mathcal{G}}_i\}_{i=1}^{v}\right)$, where $\mathcal{G} = \{G_1, \ldots, G_k\}$ is a partition satisfying,

$$\bigcup_{j=1}^{k} G_j = U,\ G_i \cap G_j = \varnothing \quad (i \neq j). \tag{2}$$

Constructing Group-Specific Skills. ConCord constructs one skill per group using knowledge from all users, since users with different working preferences may still contribute useful knowledge. Each skill follows its group's preferences where users disagree, and a separate description specifies when the skill should be used. Let $\mathcal{H} = \{H_u\}_{u \in U}$ be the analyses of all users, and let Q_j denote the working preferences represented by group G_j. A builder agent constructs one skill per group, $E_j = \text{Build}(\mathcal{H}; Q_j),\ j = \{1, \ldots, k\}$, where k means the number of user groups. When constructing skill E_j, the builder uses analyses from all users, $\mathcal{H}$, rather than only those from group G_j. When users recommend different actions for the same decision, the builder is instructed to follow the group's preferences Q_j. The builder LLM agent also generates a one-sentence description σ_j of when skill E_j applies, $\sigma_j = \text{Scope}\left(\{R_u\}_{u \in G_j}\right)$, where Scope summarizes the conditions under which the group's rules apply and R_u is the set of decision rules extracted from user u's work records T_u.

Drafting and Merging Responses. ConCord merges independently generated responses, using each skill's scope to resolve conflicts for the current task. In particular, given a task x, the agent using each skill $E_j \in \mathcal{E}$ independently generates a complete response, $y_j = \text{Draft}(x; E_j),\ j = \{1, \ldots, k\}$. A merging agent produces the final response,

$$y = \text{Merge}\left(x, \{(y_j, \sigma_j)\}_{j=1}^{k}\right). \tag{3}$$

It receives the task description, candidate responses, and scope sentences. The agent is instructed to preserve compatible commitments, checks, numerical details, and required wording. When drafts recommend different actions for the same decision, it selects the recommendation whose associated scope sentence best matches the task. The final response states the selected action directly without referring to the alternative drafts. We present all the prompts used for ConCord in Appendix A.2.

4 EXPERIMENT ANALYSIS

This section introduces the experimental setups and evaluates ConCord from the following perspectives: (1) Does ConCord outperform the baselines? (2) How do the grouping and merging mechanisms in ConCord benefit the distilled skill? (3) How do the hyperparameters of ConCord affect the performance? (4) How does the computational cost of ConCord compare with that of baselines?

4.1 EXPERIMENT SETUPS

Investigated Dataset. We evaluate ConCord on four datasets. Three datasets contain real-world professional activities: workplace emails from `Enron` (Klimt & Yang, 2004), software issue handling using issue reports and their linked developer commits from `20-MAD` (Claes & Mäntylä, 2020), and computer science question answering from `SE-CS` (Stack Exchange, Inc., 2024). The last dataset, `AntiSkillBench` (Xiang et al., 2026), is a synthetic dataset containing 50 role-played personas and their corresponding dialogues. Appendix B.1 summarizes details of the datasets.

Skill Utility Evaluation. We evaluate whether distilled skills improve response quality using test tasks constructed separately for each dataset, as introduced in Appendix B.1. Following prior work on evaluating LLMs (Zheng et al., 2023; Lin et al., 2025), for each task, an LLM judge compares two answers from the same base model: one generated using distilled skills and the other generated without skills. We assign the skill-based response a score of $+1$ for a win, -1 for a loss, and 0 for a tie. We report the average score, Quality $= (N_{\text{win}} - N_{\text{loss}})/N$, where N_{win} and N_{loss} denote the numbers of wins and losses, and N is the total number of tasks. The quality score lies in $[-1, 1]$, and a higher score indicates better performance.

Conflict Handling Evaluation. To our knowledge, we are the first to focus on studying conflicts in skills distilled from multiple users' work records. Thus, we introduce two metrics to assess how these skills resolve conflicts and preserve information relevant to different working practices.

Resolution rate (RR) measures how often a skill provides clear guidance on conflicts identified by an LLM judge using the definition in Section 2.2. For a set of users, an LLM judge compares pairs of users' work records to identify decisions for which they prefer different actions. The judge then reads the distilled skill and scores each decision as 0 or 1 (Qin et al., 2024). For decision c, it assigns $z_c = 1$ if the skill recommends a specific action or explains which action to take under different conditions, and $z_c = 0$ if the guidance is ambiguous or missing. We exclude unparseable judgments and let $\mathcal{C}$ denote the set of decisions with valid scores. We compute $\text{RR} = \frac{1}{|\mathcal{C}|}\sum_{c\in\mathcal{C}} z_c$. A higher RR indicates that the skill provides clear guidance on a larger proportion of conflicting decisions.

Conflict-exposed information loss (CIL) measures the fraction of conflict-related content missing from the distilled skills. Inspired by item-by-item content evaluation (Nenkova & Passonneau, 2004), we prepare a fixed checklist from records not used for skill construction. The checklist contains reusable actions and requirements that one working practice supports but another would omit or reject, excluding task-specific details. We prepare this checklist without consulting any method's output and use it to evaluate all methods. Let $\mathcal{I}$ denote the checklist and s be the skill produced by a method. For each item $i \in \mathcal{I}$, an LLM judge checks whether any skill provides guidance for performing the listed action or satisfying the listed requirement. We assign $b_i = 1$ if such guidance is present and $b_i = 0$ otherwise. We compute $\text{CIL} = \frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}}(1 - b_i)$. Lower CIL indicates that the skills cover a larger proportion of the checklist. Further details are provided in Appendix C.

LLM Model Configuration. As introduced in Section 3.3, ConCord uses LLM agents to (1) analyze user records, (2) extract decision rules, (3) propose and (4) reconcile user groups, (5) construct skills and their scope descriptions, (6) generate candidate responses, and (7) merge them into a final answer. Each role in ConCord is implemented through separate LLM calls with role-specific prompts and inputs. For simplicity and consistency across comparisons, we use the same base LLM for all agent roles within each experiment. This is an experimental choice rather than a requirement of the framework, which also allows different roles to use different base LLMs.

We use four base LLMs in our experiments to distill the skill: 'GPT-4.1-mini' (OpenAI, 2025), 'DeepSeek-V4-Pro' (Xu et al., 2026), 'Gemini-2.5-Flash' (Comanici et al., 2025), and 'Qwen-2.5-72B' (Yang et al., 2024). Within each experiment, all methods use the same base LLM for both skill distillation and response generation. All evaluation judgments, including response quality, RR,

Table 2: Performance comparison of existing persona skill distillation methods on four datasets, distilled using four LLMs. Bold marks the best of the six methods for each LLM in each column.

Model	Distillation	Enron			20-MAD			SE-CS			AntiSkillBench		
		RR↑	CIL↓	Qua.↑	RR↑	CIL↓	Qua.↑	RR↑	CIL↓	Qua.↑	RR↑	CIL↓	Qua.↑
GPT-4.1-mini	AWM	0.33	0.92	−0.15	0.12	0.75	+0.13	0.12	0.93	−0.03	0.64	0.67	+0.28
	Nvwa	0.60	0.83	+0.24	0.44	0.53	+0.62	0.56	0.75	+0.00	0.71	0.62	+0.35
	Colleague	0.56	0.88	+0.10	0.38	0.38	+0.81	0.88	0.60	+0.01	0.64	0.32	+0.44
	Generic	0.50	0.89	+0.06	0.25	0.34	+0.47	1.00	**0.53**	+0.00	0.69	0.46	+0.48
	SkillClaw	0.62	0.92	+0.19	0.38	0.69	+0.39	0.12	0.97	−0.01	0.25	0.94	−0.11
	ConCord	**0.73**	**0.62**	**+0.78**	**0.69**	**0.20**	**+0.93**	**1.00**	0.57	**+0.08**	**0.94**	**0.29**	**+0.51**
DeepSeek-V4-Pro	AWM	0.50	0.90	−0.48	0.12	0.92	−0.05	0.60	0.92	−0.02	0.36	0.50	+0.01
	Nvwa	0.56	0.75	−0.16	0.56	0.58	+0.31	0.94	0.70	+0.01	0.81	0.44	+0.38
	Colleague	0.46	0.84	+0.00	0.38	0.81	−0.07	0.87	0.70	+0.08	0.71	0.52	+0.40
	Generic	0.62	0.79	−0.50	0.12	0.71	−0.72	0.88	**0.55**	+0.08	0.75	0.43	+0.16
	SkillClaw	0.44	0.94	−0.15	**0.56**	0.74	+0.11	0.12	0.92	−0.02	0.56	0.42	+0.11
	ConCord	**0.69**	**0.66**	**+0.35**	0.44	**0.36**	**+0.39**	**0.94**	0.58	**+0.12**	**0.94**	**0.41**	**+0.50**
Gemini-2.5-Flash	AWM	0.19	0.88	+0.05	0.27	0.67	−0.62	0.80	0.90	−0.18	0.60	0.66	−0.24
	Nvwa	0.56	0.79	+0.07	0.38	0.57	**+0.10**	**1.00**	0.81	**+0.10**	0.56	0.63	**+0.12**
	Colleague	0.53	0.90	+0.18	0.25	0.75	−0.67	0.75	0.63	−0.02	0.79	0.52	+0.07
	Generic	**0.69**	0.91	−0.01	**0.44**	0.63	−0.35	0.88	0.63	−0.11	0.81	0.37	−0.23
	SkillClaw	0.25	0.92	−0.17	0.06	0.91	−0.51	0.94	1.00	−0.21	**0.81**	0.73	−0.02
	ConCord	0.62	**0.60**	**+0.57**	0.19	**0.47**	+0.01	0.88	**0.54**	+0.07	0.73	**0.13**	+0.01
Qwen-2.5-72B	AWM	0.29	0.96	+0.09	0.12	0.84	−0.22	0.25	0.94	−0.01	0.50	0.85	+0.09
	Nvwa	0.50	0.91	+0.09	0.19	0.62	+0.01	0.50	0.94	−0.08	0.31	0.86	+0.12
	Colleague	0.50	0.86	+0.08	0.19	0.46	−0.32	0.73	0.77	**+0.01**	0.62	0.66	+0.14
	Generic	0.56	0.98	+0.40	0.06	0.73	−0.30	0.50	0.95	−0.03	**0.75**	0.73	+0.21
	SkillClaw	0.56	0.87	+0.02	0.12	0.65	−0.15	0.12	1.00	−0.01	0.57	0.94	−0.05
	ConCord	**0.69**	**0.56**	**+0.53**	**0.25**	**0.16**	**+0.20**	**0.94**	**0.53**	−0.01	0.69	**0.56**	**+0.33**

and CIL, are provided by 'Claude Haiku 4.5' (Anthropic, 2025). We provide more details of the implementation of ConCord in Appendix A.

Baselines. We compare ConCord against five skill construction methods, as presented in Table 1, namely, AWM (Wang et al., 2025b), Nvwa (xmg, 2026), Colleague Skill (Zhou et al., 2026), Generic Skill (Huang et al., 2026), and SkillClaw (Ma et al., 2026). We implement the baselines using the default settings in their public repositories, except for Generic Skill (it has no public prompts, so we reimplement the method). Nvwa (xmg, 2026) is available as a public code repository without an accompanying paper. We include it as a baseline because of its influence in the community. Further details on the baselines are provided in Appendix B.2.

4.2 The Performance of ConCord

Table 2 compares six methods across four datasets and four base LLMs. ConCord achieves the best or tied-best results in 36 of the 48 metric comparisons. Across the 16 LLM and dataset settings, ConCord achieves an average comparison score of 0.34 against responses generated without skills, compared with 0.14 for Nvwa, the strongest baseline. Besides, ConCord achieves an average RR of 0.71, compared with $[0.36, 0.59]$ for the baselines, and an average CIL of 0.45, compared with $[0.66, 0.85]$. Its skills therefore provide explicit guidance on more decisions where users disagree and cover more conflict-related checklist items. However, explicit guidance alone does not ensure better responses. Generic Skill achieves the highest average RR among the baselines (0.59), but its average response quality is below the no-skill baseline. However, 7 of the 12 comparisons in which ConCord does not achieve the best result occur with 'Gemini-2.5-Flash'. With this model, ConCord achieves the lowest CIL on all four datasets, but its response quality is below Nvwa on `20-MAD`, `SE-CS`, and `AntiSkillBench` by 0.09, 0.03, and 0.11. These results show that better coverage of conflict-related content does not always translate into better response quality.

4.3 Comparison with Alternative Conflict Handling Methods

We compare ConCord with five alternative conflict-handling methods discussed in Section 3.1: (1) 'Nvwa' (xmg, 2026) uses the original method, (2) 'Drop conflicts' removes conflicting rules, (3) 'Keep both' retains them in one shared skill, (4) 'Both + conditions' additionally specifies when each rule should apply, and (5) 'Nvwa-style split' applies Nvwa's conflict-handling rules within a single shared skill. ConCord preserves different practices in separate skills and merges their independently generated responses according to the task requirements. The skill distilling model is 'GPT-4.1-mini'.

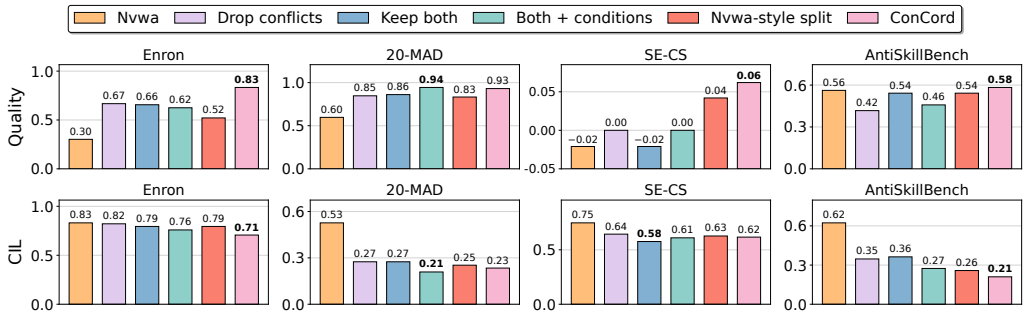

Figure 4: Response Quality and CIL of ConCord and five alternatives. 'Nvwa' refers to the original method, while 'Nvwa-style split' applies Nvwa's rules for handling conflicts within a single shared skill. 'Drop conflicts' indicates dropping conflicting rules. 'Keep Both' retains the conflicting rules, while 'Both + conditions' also specifies when each rule should be used. 'ConCord' (ours) preserves different conflicting rules in separate skills and merges their answers based on the task.

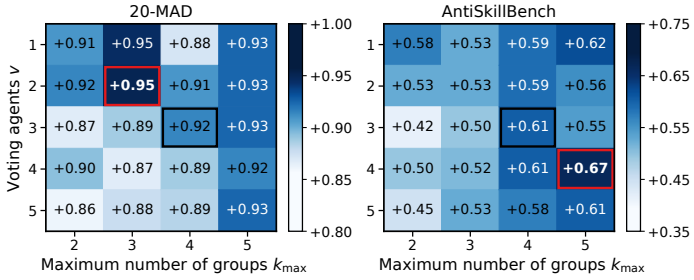

Figure 5: Answer quality across different hyperparameter settings on `20-MAD` and `AntiSkillBench`. Red boxes mark the best-performing settings, while black boxes indicate the default setting used in the main experiments.

As shown in Figure 4, ConCord achieves the highest response quality on three of the four datasets, `Enron`, `SE-CS`, and `AntiSkillBench`. The largest advantage occurs on `Enron`, where it scores 0.83, compared with 0.67 for the strongest alternative. On `20-MAD`, its score of 0.93 is close to the 0.94 achieved by 'Both + conditions'. For CIL, ConCord achieves the lowest scores on `Enron` and `AntiSkillBench`, but some shared-skill alternatives perform better on the other two datasets: 'Both + conditions' achieves 0.21 versus 0.23 on `20-MAD`, and 'Keep both' achieves 0.58 versus 0.62 on `SE-CS`. Notably, despite its lower CIL on `SE-CS`, 'Keep both' scores −0.02 in response quality, compared with 0.06 for ConCord. The alternative strategies reduce CIL compared with Nvwa, but ConCord achieves higher response quality on three of the four datasets. ConCord develops different approaches into complete answers independently before merging them. The merging agent can then compare concrete solutions and select recommendations that fit the task.

4.4 Hyperparameter Analysis

This section analyzes two hyperparameters in the user-grouping: the number of voting agents v and the maximum number of groups $k_{\max}$. Each voting agent independently proposes a partition of users based on their stance profiles. Increasing v provides more grouping proposals for the reconciler, but also increases skill-construction cost. The parameter $k_{\max}$ controls the finest grouping allowed by ConCord. A larger value can preserve more distinct working practices but may produce more skills and increase response generation token cost. As shown in Figure 5, we vary v from 1 to 5 and $k_{\max}$ from 2 to 5 on `20-MAD` and `AntiSkillBench`.

The effects of these parameters vary across datasets. On `20-MAD`, response quality ranges from 0.86 to 0.95, with the highest score obtained using one or two voting agents and $k_{\max} = 3$. Increasing the number of voters does not consistently improve performance, suggesting that a small number of grouping proposals is sufficient when users' working preferences are relatively clear. On `AntiSkillBench`, larger values of $k_{\max}$ are generally more effective, and the highest score of 0.67 is achieved with $v = 4$ and $k_{\max} = 5$. More diverse working practices may benefit from finer grouping. No single configuration performs best on both datasets. Our default setting achieves scores of 0.92 and 0.61, respectively, both higher than the corresponding single-skill results of 0.90 and 0.59, while avoiding the additional cost of the largest configuration.

Table 3: Cost of skill construction and per-task response generation on 20-MAD with 'GPT-4.1-mini'. 'Skill Words' reports the total length of the skills. For ConCord, this is summed over the k group-specific skills, like $k = 3$. 'Calls' counts LLM calls. 'Input Tokens' and 'Output Tokens' report token counts during skill construction.

Method	Skill Words	Building the Skills				Response Generation (Per Task)		
		Calls	Input Tokens	Output Tokens	Time	Calls	Total Tokens	Time
AWM	1,349	1	9,599	1,145	10 s	1	2,338	3.2 s
Nvwa	549	1	9,551	755	7 s	1	1,444	5.5 s
Colleague Skill	723	2	10,678	1,819	21 s	1	1,438	3.2 s
Generic Skill	280	1	9,797	472	5 s	1	751	2.9 s
SkillClaw	502	31	11,758	11,147	96 s	1	1,274	3.9 s
ConCord	2,515	30	54,737	18,520	166 s	4	7,400	10.8 s

4.5 Computational Cost

Table 3 reports the computational cost on 20-MAD with 'GPT-4.1-mini'. ConCord requires more computation than the baselines during both skill construction and response generation. The main reason is that ConCord analyzes users separately, performs multiple grouping and reconciliation calls, and constructs several group-specific skills rather than a single shared skill. During response generation, each skill independently generates a candidate response, followed by an additional merging call. As a result, ConCord uses more input and output tokens. This additional cost is the price of preserving and coordinating different working practices. In return, ConCord achieves substantially better response quality than the baselines. Once the skills are built, each new task only requires generating candidate responses and merging them, without rebuilding the skills.

5 Related Works

Persona Skill. A persona skill is distilled from a person's work records and conversation histories into a reusable skill package that enables an agent to emulate how that person approaches, plans, and completes tasks (Zhang et al., 2025; Zhou et al., 2026; xmg, 2026). Huang et al. (2026) compare developer-specific skills with a generic skill distilled from interaction histories pooled across developers. Other methods combine experience in different forms. AWM (Wang et al., 2025b) extracts reusable workflows from task trajectories. Nvwa (xmg, 2026) supports both person-specific skills and topic skills that summarize shared principles and disagreements across multiple people. Colleague Skill (Zhou et al., 2026) organizes working records about a target person into capability and behavior tracks. SkillClaw (Ma et al., 2026) aggregates interactions across users to refine and extend a shared repository of procedural skills.

Skill Mismatch. Prior work studies whether skills match the current task (Su et al., 2026; Li et al., 2026a; Wang et al., 2026). Related research examines conflicts between predefined personas and users' statements (Koo et al., 2026), changes in user preferences (Jiang et al., 2025), and conflicting personal memories (Yang et al., 2026). These skill mismatch works relate to, but are distinct from, the skill conflict we studied. Our work focuses on practices distilled from multiple users that may each be valid yet recommend different actions for the same decision.

6 Conclusion

This paper presents ConCord, an automated framework for coordinating persona skills distilled from multiple users' work records. It combines knowledge from multiple users without reducing their working practices to a single approach. Instead, these practices remain available as alternatives whose usefulness is assessed through their answers to the current task. Our experiments show higher average response quality and lower average CIL than the evaluated baselines. This paper is also among the first to study how conflicting working practices in multi-user persona skills can be preserved and coordinated for downstream tasks, opening a new direction for building agents that can learn from diverse human experience without collapsing it into a single working style. We present the limitations and future work in Appendix E.

7 AI USE STATEMENT

We used ChatGPT to help refine the definition, discuss evaluation metrics and experimental methodology, and interpret experimental results. We also used it for manuscript polish, literature searches, reference formatting, figure captions, and LaTeX formatting. LLMs are also used as components of ConCord and as judges in our experiments. Their roles, model configurations, and prompts are described in Sections 3.3 and 4.1 and Appendix A.2. The authors take responsibility for the final content of this work, including its methods, results, references, and conclusions.

REFERENCES

Nvwa skill, 2026. URL https://github.com/xmg2024/nvwa-skill.

Anthropic. System card: Claude Haiku 4.5. Technical report, Anthropic, October 2025. URL https://www.anthropic.com/claude-haiku-4-5-system-card.

Minghao Chen, Yihang Li, Yanting Yang, Shiyu Yu, Binbin Lin, and Xiaofei He. Automanual: Constructing instruction manuals by llm agents via interactive environmental learning. *Advances in Neural Information Processing Systems*, 37:589–631, 2024.

Maëlick Claes and Mika V. Mäntylä. 20-MAD: 20 years of issues and commits of Mozilla and Apache development. In *Proceedings of the 17th International Conference on Mining Software Repositories (MSR)*, pp. 503–507, 2020.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

Jennifer Fereday and Eimear Muir-Cochrane. Demonstrating rigor using thematic analysis: A hybrid approach of inductive and deductive coding and theme development. *International journal of qualitative methods*, 5(1):80–92, 2006.

Ge Gao, Alexey Taymanov, Eduardo Salinas, Paul Mineiro, and Dipendra Misra. Aligning llm agents by learning latent preference from user edits. *Advances in neural information processing systems*, 37:136873–136896, 2024.

Hsiu-Fang Hsieh and Sarah E Shannon. Three approaches to qualitative content analysis. *Qualitative health research*, 15(9):1277–1288, 2005.

Shuyan Huang, Kai Du, and Andrew Lan. Do personalized skills help coding agents? an empirical study of developer interaction histories. *arXiv preprint arXiv:2608.10319*, 2026.

Robert A. Jacobs, Michael I. Jordan, Steven J. Nowlan, and Geoffrey E. Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3(1):79–87, 1991.

Bowen Jiang, Zhuoqun Hao, Young Min Cho, Bryan Li, Yuan Yuan, Sihao Chen, Lyle Ungar, Camillo Jose Taylor, and Dan Roth. Know me, respond to me: Benchmarking LLMs for dynamic user profiling and personalized responses at scale. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=6ox8XZGOqP.

Bryan Klimt and Yiming Yang. The Enron corpus: A new dataset for email classification research. In *European Conference on Machine Learning (ECML)*, pp. 217–226, 2004.

Seonmin Koo, Jinsung Kim, and Heui-Seok Lim. I know, but i don't know! how persona conflict undermines instruction adherence in large language models. In *Findings of the Association for Computational Linguistics: EACL 2026*, pp. 465–489, 2026.

Qinfeng Li, Dalin He, Yuntai Bao, Ying Yang, Ruoxi Chen, Xinyan Yu, Lizhou Liang, Ge Su, Wenqi Zhang, and Xuhong Zhang. Skillaligner: Treating retrieved skills as adaptable drafts at execution time. *arXiv preprint arXiv:2608.06880*, 2026a.

Xiangyi Li, Wenbo Chen, Yimin Liu, Shenghan Zheng, Xiaokun Chen, Yifeng He, Yubo Li, Bingran You, Haotian Shen, Jiankai Sun, et al. Skillsbench: Benchmarking how well agent skills work across diverse tasks. *arXiv preprint arXiv:2602.12670*, 2026b.

Yixuan Li, Mingshu Cai, Ziyang Xiao, Wanyuan Wang, Yanchen Deng, and Bo An. Mind-skill: Quality-guaranteed skill generation via multi-agent induction and deduction. *arXiv preprint arXiv:2605.08670*, 2026c.

Bill Yuchen Lin, Yuntian Deng, Khyathi Chandu, Abhilasha Ravichander, Valentina Pyatkin, Nouha Dziri, Ronan Le Bras, and Yejin Choi. Wildbench: Benchmarking llms with challenging tasks from real users in the wild. In *International Conference on Learning Representations*, volume 2025, pp. 47852–47870, 2025.

Ziyu Ma, Shidong Yang, Yuxiang Ji, Xucong Wang, Yong Wang, Yiming Hu, Tongwen Huang, and Xiangxiang Chu. Skillclaw: Let skills evolve collectively with agentic evolver. *arXiv preprint arXiv:2604.08377*, 2026.

Ani Nenkova and Rebecca Passonneau. Evaluating content selection in summarization: The pyramid method. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL)*, pp. 145–152, 2004.

Jingwei Ni, Yihao Liu, Xinpeng Liu, Yutao Sun, Mengyu Zhou, Pengyu Cheng, Dexin Wang, Erchao Zhao, Xiaoxi Jiang, and Guanjun Jiang. Trace2skill: Distill trajectory-local lessons into transferable agent skills. *arXiv preprint arXiv:2603.25158*, 2026.

OpenAI. Introducing GPT-4.1 in the API. OpenAI Blog, 2025. URL `https://openai.com/index/gpt-4-1/`.

Yiwei Qin, Kaiqiang Song, Yebowen Hu, Wenlin Yao, Sangwoo Cho, Xiaoyang Wang, Xuansheng Wu, Fei Liu, Pengfei Liu, and Dong Yu. Infobench: Evaluating instruction following ability in large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 13025–13048, 2024.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations*, 2017.

Stack Exchange, Inc. Computer science Stack Exchange data dump. Internet Archive, 2024. URL `https://archive.org/details/stackexchange`.

Weihang Su, Jianming Long, Qingyao Ai, Qiaozhi He, Yichen Tang, Changyue Wang, Yiteng Tu, Yingbo Wang, and Yiqun Liu. Skill retrieval augmentation for agentic ai. *arXiv preprint arXiv:2604.24594*, 2026.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *Transactions on Machine Learning Research*, 2024.

Zifei Wang, Wei Wen, Qiang Ji, Ruizhi Qiao, and Xing Sun. Skill is not document: A query-conditional benchmark and two-stage retriever for llm agent skill routing. *arXiv preprint arXiv:2606.03565*, 2026.

Zora Zhiruo Wang, Apurva Gandhi, Graham Neubig, and Daniel Fried. Inducing programmatic skills for agentic tasks. *arXiv preprint arXiv:2504.06821*, 2025a.

Zora Zhiruo Wang, Jiayuan Mao, Daniel Fried, and Graham Neubig. Agent workflow memory. In *Proceedings of the 42nd International Conference on Machine Learning*, 2025b.

Yongli Xiang, Zhifang Zhang, Bojun Yang, Ziming Hong, Lei Feng, Miao Xu, and Tongliang Liu. When agents learn to be you: Benchmarking privacy leakage, impersonation risk, and defenses in persona skills. *arXiv preprint arXiv:2608.03700*, 2026.

Anyi Xu, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, Chenchen Ling, et al. Deepseek-v4: Towards highly efficient million-token context intelligence. *arXiv preprint arXiv:2606.19348*, 2026.

Renjun Xu and Yang Yan. Agent skills for large language models: Architecture, acquisition, security, and the path forward. *arXiv preprint arXiv:2602.12430*, 2026.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Lu Yang, Shusheng Xu, Zhuoran Li, Tongkai Yang, and Longbo Huang. When personal memory has no single answer: Evaluating llm agents under irreducible conflict. *arXiv preprint arXiv:2608.13921*, 2026.

Weizhi Zhang, Xinyang Zhang, Chenwei Zhang, Liangwei Yang, Jingbo Shang, Zhepei Wei, Henry Peng Zou, Zijie Huang, Zhengyang Wang, Yifan Gao, et al. Personaagent: When large language model agents meet personalization at test time. *arXiv preprint arXiv:2506.06254*, 2025.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, 2023.

Tianyi Zhou, Dongrui Liu, Leitao Yuan, Jing Shao, and Xia Hu. Colleague. skill: Automated ai skill generation via expert knowledge distillation. *arXiv preprint arXiv:2605.31264*, 2026.

A Implementation Details

A.1 Working Practice Construction

Identifying conflicts requires expressing users' rules in a comparable form, even when their records cover different topics. Following prior work on shared codebooks and directed content analysis, which map heterogeneous text to a common set of predefined categories (Hsieh & Shannon, 2005; Fereday & Muir-Cochrane, 2006), ConCord maps each user's rules onto a shared set of aspects of working practices $\mathcal{D}$. These aspects capture recurring decisions across different types of work rather than topic-specific content. We defined $\mathcal{D}$ manually during development, and use the same twelve aspects across all datasets and base LLMs,

- Requests that cannot be fully satisfied as asked (deadline, scope, or resource pressure).
- What must be verified, and in what form, before acting.
- Deciding alone versus escalating or involving someone else.
- How firmly to commit: firm commitment, conditional commitment, or holding back.
- How far to go beyond what was literally asked.
- How to prioritize when several demands compete.
- Responding to pushback or disagreement from the other side.
- Acting under uncertainty or incomplete information.
- Closing the loop: what is confirmed, recorded, or followed up.
- Register and directness: how formally to write, and how bluntly to state a position.
- How much detail, justification, or evidence to include with an answer.
- How much personal judgment or opinion to voice rather than staying strictly factual.

The first nine are decisions about how a request is handled. The last three concern how an answer is written. $\mathcal{D}$ is an input to the method and is not learned from users' records. For each user u, the rule extractor reads T_u and produces a set of rules R_u. Each rule is represented as $r = (d_r, c_r, a_r, e_r)$, where $d_r \in \mathcal{D}$ is the aspect addressed by the rule, c_r specifies when the rule applies, a_r is the user's preferred action, and e_r contains supporting details and a quotation from the records.

A.2 PROMPTS USED

The prompts below are reproduced verbatim as used in our experiments and follow the complete workflow of ConCord. Prompts 1–2 analyze each user's work records and extract decision rules. Prompts 3–4 propose and reconcile user groups. Prompts 5–7 construct group-specific skills and their scope descriptions, where Prompt 6 is appended to Prompt 5 rather than issued as a separate LLM call. Prompt 8 merges the independently generated responses. The drafting step has no separate numbered prompt: for each skill E_j, the base LLM receives the skill together with the task x and generates a complete candidate response y_j. Prompts 9–13 and 17 support response-quality, RR, and CIL evaluation. Prompt 14 is used to select users whose records are used for skill distillation, while Prompts 15–16 identify and consolidate the contested decisions for each user set.

Text enclosed in curly braces is replaced at runtime unless it is part of a required JSON output format or a descriptive field in the skill template of Prompt 5, such as {areas}. The placeholders {unit} and {artifact} denote the dataset-specific names for a work record and a generated response, respectively; for example, on Enron, they correspond to "work email this person sent" and "email reply." {body} contains the records, profiles, proposals, or other text processed by a prompt; {dimensions} contains the predefined working-practice aspects; {budget} is the skill word limit; and {analyses} contains the analyses of all users. The placeholders {stance}, {label}, and {stances} describe one user group's working practices.

For grouping, {n} is the number of users included in grouping, {last} is the index of the last such user, {lo} and {hi} are the minimum and maximum numbers of groups, {m} is the minimum group size $g_{\min}$, {profiles} contains all stance profiles, and {seed} identifies the voting agent. Here, the prompt placeholders {n} and {m} should not be confused with the notation for the numbers of records and users in the main text.

For response generation, {task} is the current task and {drafts} contains the candidate responses together with their scope descriptions. In Prompt 17, {task} denotes the contested decision being evaluated, and {menu} contains the available skills and their scope descriptions.

For evaluation, {first} and {second} are the two responses in the comparison order determined by the task hash. {dimension} and {card} denote one contested decision and the skill evaluated on that decision. {ref} is a user's reference response, {elements} contains the content items extracted from it, {dims} contains the pool's contested decisions, and {doc} is the skill document being evaluated. In Prompts 14–15, {a} and {b} denote two users' skills or work records.

Prompt 1: Analyst, $H_u = \text{Analyze}(T_u)$ (Section 3.3)

```
You will receive records of a person's past work ({unit}).  Extract
their work capability for a COLLEAGUE.SKILL-style Work Skill.
PRINCIPLE: extract only evidence-grounded content; DO NOT infer.  If a
dimension has no evidence, write
"insufficient evidence".  Be concrete and executable.
Extract:
1.  SCOPE / AREAS: what kinds of work and decisions this person
handles.
2.  STANDARDS & FOCUS: the specific recurring concerns/checks they
raise.
3.  WORKFLOW / STYLE: how they handle a request, decide, escalate, or
close the loop.
4.  HEURISTICS / RULES: concrete IF-THEN rules where possible.
RECORDS:
{body}
```

Prompt 2: Rule extractor, rules R_u (Section 3.3, Appendix A.1)

```
Below is the recorded work of ONE person ({unit}).
For each decision dimension listed, state what THIS person's records
show they actually do.  These dimensions are fixed so that different
```

```
people's records can be compared, so answer them even when the subject
matter differs.
DIMENSIONS:
{dimensions}
For every dimension where the records give evidence, output:
- dimension:  copy the dimension text verbatim.
- condition:  when this behaviour applies (use "default" if it is what
they always do).
- action:  what they do, stated as an executable instruction general
enough to apply to a new situation on this dimension.
- specifics:  the concrete operational details this behaviour carries,
copied verbatim from the records -- deadlines and time windows,
thresholds and amounts, required confirmations, named checks, standard
phrasings, the fields or artefacts they always name.  Use an empty
string only if the records truly contain none.
- evidence:  a short quote or paraphrase from the records.
Rules:
- Evidence may be implicit:  work records rarely state a policy, they
demonstrate it.  If the records show how they handled a dimension even
once, record it and quote the supporting text in `evidence`.  Only
skip a dimension when the records genuinely show nothing about it, and
never invent behaviour no record supports.
- Never write claims about who this person is, what they specialise
in, or what falls inside or outside their remit.
- You may add at most 2 extra recurring decision dimensions you
observe, phrased the same generic way.
Return STRICT JSON only:
{"rules":  [{"dimension":  "...", "condition":  "...", "action":
"...", "specifics":  "...", "evidence":  "..."}]}
RECORDS:
{body}
```

Prompt 3: Voter, $\widehat{\mathcal{G}}_i = \mathrm{Vote}(\mathcal{P})$ (Section 3.3)

```
Below are {n} people who do the same kind of work, numbered from 0.
For each one you are given what their records show they DO on a fixed
set of recurring decisions -- not what subject they work on.
Group them by WORKING STANCE. Two people belong in the same group
when, facing the same situation, they would make the same call.  Put
them in different groups when following one would mean acting
differently from the other on a recurring decision:  a different call,
a different threshold, an opposite default, a different order of
priorities.
Subject matter is irrelevant here.  Two people who work on completely
different components but escalate, verify and commit the same way
belong in the SAME group.  Two people who work on the same component
but disagree about when to escalate belong in DIFFERENT groups.
Rules:
- Between {lo} and {hi} groups.  Every person appears in exactly one
group.
- Do not create a group for a single person unless that person really
does work unlike everyone else.
- Give each group a short label naming the working stance it
represents.
Return STRICT JSON only:
{"groups":  [{"label":  "<short label>", "members":  [0, 3, 7]}]}
PROFILES:
{body}
[reviewer {seed}]
```

Prompt 4: Reconciler, $\mathcal{G} = \text{Reconcile}(\cdot)$ (Section 3.3)

```
Several reviewers independently grouped the same {n} people by how
they work.  Their proposals, and what each person does on the
recurring decisions, are below.

Produce the consensus grouping:  people whom most reviewers placed
together stay together, and a split that most reviewers made is kept.
Every person from 0 to {last} must appear exactly once.

Every group must hold at least {m} people.  If the consensus would
leave a group smaller than that, move its members into the group whose
working stance is closest to theirs -- judged from what they do, not
simply the largest group.

Return STRICT JSON only:
{"groups":  [{"label":  "<short label>", "members":  [0, 3, 7]}]}

PROPOSALS:
{body}

WHAT EACH PERSON DOES:
{profiles}
```

Prompt 5: Builder, $E_j = \text{Build}(\mathcal{H}; Q_j)$ (Section 3.3)

```
Using the material below, generate ONE pooled 'work.md' capability
skill for the whole team.  Format:

# Work Skill
## Scope
{areas}
## Standards & Focus
When handling a situation, you check, in priority order:
{the concrete recurring concerns}
## How you handle a request
{workflow/style}
## Heuristics
{concrete IF-THEN rules, one per line}
## Usage
These are the team's own working practices.  When responding to a
situation, first give everything a competent
response to it needs:  the decision or answer itself, its reasoning,
the risks, and the next steps.  Then apply the
Standards & Focus and Heuristics above wherever they bear on the
situation, and follow them exactly.  The practices
add to competent handling; they never replace it.

The material contains several practitioners' analyses of the same kind
of work.  MERGE them into one team skill:  combine overlapping areas
and checks into single items, keep the concrete specifics (thresholds,
deadlines, named checks, standard phrasings, artefact names) verbatim
inside those items, and drop anything personal or unrelated to the
work.

Length budget (hard):  at most {budget} words in total.  Scope at most
8 bullets.  Standards & Focus at most 8 ranked items.  How you handle
a request at most 8 lines.  Heuristics at most 14 IF-THEN lines.  All
five sections must be present and complete; never stop mid-sentence.
Output the card as plain markdown, not inside a code fence.

Rules:  concrete and directly executable.  Omit unsupported
dimensions.  Never mention practitioners, pooling, or disagreement.

ANALYSES:
{analyses}
```

Prompt 6: Group preferences Q_j, appended to Prompt 5 (Section 3.3)

```
RESOLVING DISAGREEMENT. The practitioners above do not all handle
every decision the same way.  Where they differ, write the skill so it
follows THIS stance, stated once and without hedging:
{stance}
Rules for this:  keep every practitioner's concrete material (named
checks, thresholds, standard phrasings) whoever it came from, but on
the decisions listed above give ONE instruction, the one this stance
prescribes.  Never present two options, never mention that anyone
disagrees, and never say "some practitioners".  The reader must come
away knowing exactly what to do.
```

Prompt 7: Scope description, $\sigma_j = \mathrm{Scope}(\cdot)$ (Section 3.3)

```
Below is what one group of practitioners DOES on a set of recurring
decisions.
Write the selection criterion for this group:  the situation features
that should make a reader consult them, and the stance they will take.
Describe the CHARACTER of the situation (a hard deadline is named, the
figures are disputed, the fix touches several components, the request
is open-ended), never the subject matter or the components they happen
to work on.
At most 35 words, one sentence, of the form "When <situation
feature(s)>:  <the stance they take>."
Return STRICT JSON only:  {"when":  "<criterion>"}
GROUP LABEL: {label}
WHAT THEY DO:
{stances}
```

Prompt 8: Merging agent, $y = \mathrm{Merge}(\cdot)$ (Section 3.3)

```
Several experienced practitioners on the same team each answered the
situation below.  Each follows a different part of the team's
practice, and each part applies in the circumstances noted.
SITUATION:
{task}
{drafts}
Write the single answer the team should send.
- Keep every concrete commitment, check, figure, name, step and
standard phrasing that any of them provides.  If only one of them
mentions something specific, it belongs in the answer.
- Where they take different courses of action on the same point,
choose ONE: the course whose stated circumstances best match this
situation.  State it once, plainly.  Do not list alternatives, do not
hedge, and never mention that anyone answered differently.
- Do not cut material the situation needs in order to be brief.
Output only the answer.
```

Prompt 9: Answer-quality judge (Section 4.1, Skill Utility Evaluation)

```
Compare two {artifact}s responding to the SAME situation.  Decide
which is more competent and useful:  better judgment, more complete
constraints, better risk handling, and more actionable next steps.
Ignore writing style, length, and identity imitation.
Situation:
{task}
```

```
Response A:
{first}
Response B:
{second}
Which is more competent?  Answer only A, B, or TIE.
```

Prompt 10: RR judge, scores z_c (Section 4.1)

```
A team's shared skill document is below.  The people it was distilled
from genuinely disagreed on this decision dimension:
DIMENSION: {dimension}
Question:  what guidance does the document give someone facing this
dimension?
Answer with STRICT JSON only:
{"label":  "CLEAR_STANCE" | "CONDITIONAL" | "HEDGED" | "SILENT",
"quote":  "<verbatim supporting sentence from the document, or
empty>"}
Definitions:
- CLEAR_STANCE: it picks one side, so the reader knows what to do.
- CONDITIONAL: it explicitly gives BOTH courses of action with the
conditions under which each applies (also actionable).
- HEDGED: it mentions the area but gives no usable direction (generic
advice like "verify carefully").
- SILENT: the document contains nothing about this dimension.
DOCUMENT:
{card}
```

Prompt 11: CIL step 1, checklist items from a reference (Section 4.1, Appendix C)

```
Below is a work situation and the response the real person actually
gave.
SITUATION:
{task}
REAL RESPONSE:
{ref}
List what another competent response would NEED to contain to match
the real
response on substance:  the concrete decisions, actions, commitments,
checks,
facts, figures or standards it relies on.  Each element is one short,
atomic,
self-contained statement.  Leave out greetings, sign-offs, names of
the specific
people involved, and pure wording.  At most 6 elements; fewer if the
response is
thin.
Return STRICT JSON only:  {"elements":  ["...", "..."]}
```

Prompt 12: CIL step 2, conflict-exposed items (Section 4.1, Appendix C)

```
Members of one team demonstrably act differently on the decision
dimensions
below; each dimension runs between two opposing ways of working.  The
dimensions
were found by comparing the members' own work records.
DIMENSIONS:
{dims}
```

```
A real response to a situation needed the elements below.
ELEMENTS:
{elements}
For EACH element, apply this test.  Picture a practitioner at one end
of a
dimension and a practitioner at the other end, both handling this
situation.
- If one of them would include the element and the other would leave
it out,
or would handle that point in an incompatible way, the element DEPENDS
on
that dimension:  give the dimension's number.
- If both would include the element the same way on every dimension --
it is
needed whichever practice one follows -- give 0.
Most routine content (facts of the situation, standard correct steps)
is needed
either way and gets 0.
Return STRICT JSON only:  {"dims":  [<number>, ...]} with exactly one
number per
element, in order.
```

Prompt 13: CIL step 3, items covered by a skill, b_i (Section 4.1, Appendix C)

```
Below is a skill document that an assistant will be given, and a list
of
elements a good response would need.
SKILL DOCUMENT:
{doc}
ELEMENTS:
{elements}
For EACH element, decide whether the document would lead the assistant
to
include it:  CARRIED if the document instructs, states or clearly
implies it;
NOT_CARRIED if it is absent or the document instructs an incompatible
practice.
Return STRICT JSON only:  {"labels":  ["CARRIED|NOT_CARRIED", ...]}
with exactly one label per element, in order.
```

Prompt 14: Conflicts between two users' skills, for the user sets (Section 4.1)

```
Below are two skill documents distilled from two different people who
do the same kind of work.
Identify dimensions where, facing a SIMILAR situation, the two
documents would lead to MATERIALLY DIFFERENT actions or decisions.
Rules:
- A difference in coverage does NOT count (one mentions a topic the
other omits, with no opposing prescription).
- Only count genuine opposition:  different choice, different
threshold, different priority order, or mutually exclusive procedure.
Return STRICT JSON only:
{"conflicts":  [{"dimension":  "<short name>", "a":  "<what A
prescribes>", "b":  "<what B prescribes>"}]}
If there is no genuine conflict return {"conflicts":  []}.
DOCUMENT A:
{a}
DOCUMENT B:
```

```
{b}
```

Prompt 15: RR decisions named from two users' records (Section 4.1)

```
Below are the working records of TWO different people who do the same
kind of work.
Name the recurring decision dimensions on which these two would act
DIFFERENTLY if they faced the same situation -- points where their
records show them making incompatible choices (a different call, a
different threshold, a different priority order, an opposite default).
Name each dimension the way a third person would recognise it,
independent of the subject matter of these particular records.
Return STRICT JSON only:
{"dimensions": ["<dimension name>", ...]}
Return an empty list if their records show no incompatible choices.
PERSON A:
{a}
PERSON B:
{b}
```

Prompt 16: RR decisions merged into the eight most frequent (Section 4.1)

```
Below is a list of decision dimensions proposed by different reviewers
who each compared two people from the same team.  Different reviewers
name the same dimension differently.
Consolidate the list into at most {k} distinct decision dimensions
that recur across the team.  Merge names that mean the same decision
point.  Order them by how many of the proposals they cover, most
first.  Name each one the way a third person would recognise it,
independent of any particular subject matter.
Return STRICT JSON only:
{"dimensions": ["<dimension name>", ...]}
PROPOSALS:
{body}
```

Prompt 17: RR skill selector for ConCord (Section 4.1)

```
You must choose which of the team's experts should handle the
situation below.  Each expert is listed with the number of
practitioners whose practice it represents.
EXPERTS:
{menu}
SITUATION:
{task}
Choose ONE expert:  the one whose stated situation features actually
appear in this situation.
When no expert's situation features clearly apply, choose the expert
representing the most practitioners.  Do not prefer an expert merely
because its criterion sounds more specific.
Return STRICT JSON only:  {"experts":  [<index>, ...]} (one or at most
{k} indices)
```

Table 4: Summary of the four work-record datasets.

Attribute	Enron	20-MAD	SE-CS	AntiSkillBench
Data type	Real-world	Real-world	Real-world	Synthetic
Users	Enron employees	Mozilla and Apache developers	CS Stack Exchange answerers	Role-played personas
Work scenario	Workplace email	Issue handling	CS Q&A	Assistant requests
Work record	A sent email	An issue and a linked commit message	A question and the user's answer	A request and later requirements
Test task	Make a decision and write a short email reply	Respond to a later issue	Answer a later question	Reply to a held-out opening request
Users Size	120	48	40	40
Work records per user	12	12	10	10

B Experimental Setup Details

B.1 Investigated Datasets

Table 4 summarizes the investigated datasets, and Table 5 gives an example from each dataset. We provide a brief description of the datasets as follows.

'*Enron*' (Klimt & Yang, 2004) dataset contains workplace communications among company employees; one source record is a sent email. For answer-quality evaluation, all test users receive the same eight predefined workplace scenarios, including deadline-extension requests. The agent makes a decision and writes a short reply.

'*20-MAD*' (Claes & Mäntylä, 2020) dataset links software issues and commits from Mozilla and Apache. We pair each issue with its linked developer commit message and order each developer's records by commit time. Earlier records are used for distillation. Later issue descriptions become test inputs, with the linked commit messages withheld. The required output is a maintainer response containing a decision and an action plan.

'*SE-CS*' dataset consists of question–answer records from the Computer Science community in the Stack Exchange data dump (Stack Exchange, Inc., 2024). The agent receives the question title and body, but not the recorded answer. We split each contributor's records by answer time, using earlier question–answer pairs as source records and later questions as test tasks.

'*AntiSkillBench*' (Xiang et al., 2026) dataset contains synthetic dialogues conditioned on 50 role-played personas. We divide each persona's dialogues into source and test subsets and distill skills using only the user turns from source dialogues. At test time, the agent responds only to the opening request of each test dialogue. Subsequent user turns, which introduce additional requirements, are hidden from the agent and used only for evaluation.

B.2 Baselines

This section describes the baselines evaluated in our study. AWM (Wang et al., 2025b) induces reusable workflows from task records. Nvwa (xmg, 2026) extracts decision heuristics and expression styles into a skill. Colleague Skill (Zhou et al., 2026) constructs a work skill with separate capability and behavior tracks. Generic Skill (Huang et al., 2026) first drafts candidate rules using fixed textual patterns and then asks an LLM to revise them against the source records. It retains at most 25 rules and requires each rule to be supported by at least two records. Because its prompts are not publicly available, we reimplement the method, following the description in their paper. SkillClaw (Ma et al., 2026) aggregates records from multiple users into a shared procedural skill.

Table 5: An example work record and test task from each dataset. Work records are shortened, and test-task wording is adapted to show the input and expected response. Bracketed identifiers in `Enron` are anonymized placeholders.

Dataset	Work record	Test task
`Enron`	*Hi [PERSON], the revised cost estimate is ready, but two entries still need confirmation from [TEAM]. Please check those figures before we circulate the document. If confirmation is delayed, send the verified sections today and mark the remaining figures as provisional.*	*A colleague asks you to approve a supplier payment before the supporting invoice arrives. The payment is due tomorrow. Decide how to proceed and write a short reply stating what must be checked, who should follow up, and when approval can be given.*
`20-MAD`	*Handle an empty configuration file before reading the first entry. Preserve the existing defaults and add a regression test for a file containing only whitespace. Leave the configuration parser unchanged outside this case.*	*Component: Configuration; severity: normal. Summary: the application fails to start when the configuration file contains comments but no settings. Write a maintainer response describing how to reproduce the failure, the proposed scope of the fix, and the tests needed before accepting a patch.*
`SE-CS`	*so it takes 1 comparison to halve the search space.*	*Number of comparisons in Binary search. Solving $T(n) = T(n/2) + 1$ gives $\log_2 n + 1$, but counting the comparisons in my implementation gives a different result. What does the recurrence count? Explain how comparisons against array elements and checks of the loop condition affect the calculation.*
`AntiSkill Bench`	*I want to practise piano regularly, but a rigid daily schedule usually falls apart when work gets busy. || Could we start with three short sessions a week? Keep each one under twenty minutes, and give me a quieter alternative for evenings when my flatmate is already asleep.*	*I have agreed to give a short talk at a community event next month. Help me prepare without turning every evening into rehearsal time. I would like a manageable weekly plan and a simple way to catch up if I miss a session.*

All the baselines are implemented using their original open-source repository and following the default hyperparameter settings, except for our reimplementation of Generic Skill.

B.3 Hyper-parameter Settings

Table 6 summarizes the hyperparameter settings of ConCord. The grouping parameters, skill-length budget, and choice to use all skills for response generation are selected using separate development users (as presented in Section 4.4). The 4,096-token response limit and evaluation judge are kept the same across all methods.

C Details of the CIL Metric

For each task, Prompt 11, shown in Appendix A.2, extracts at most six content elements from the dataset-specific reference record described in Appendix B.1. These records include the user's actual reply to a real incoming email for `Enron`, the linked commit for `20-MAD`, the contributor's original answer for `SE-CS`, and the requirements stated in later user turns for `AntiSkillBench`. They are used only for evaluation and are not shown to the agent. For each user working record set, the exposure judge compares these elements with the eight contested decisions identified from that set and marks an element as conflict-exposed if one working practice would include it while another would omit it or handle it incompatibly. The exposed elements form the checklist $\mathcal{I}$ used to compute the CIL. Content elements are extracted once per task, while exposure labels are determined once for each task and user set before evaluating any method.

Table 6: Hyperparameter settings used across all datasets and base LLMs. (a) Method settings. We use the notation defined in the main text and Appendix A.1. (b) Evaluation settings shared by all methods. (c) Settings for each LLM call. Input limits specify the maximum number of characters passed to a call, where '–' indicates no truncation. Output limits specify the maximum number of generated tokens. All judges use 'Claude-Haiku-4.5'. The prompts are introduced in Section A.2.

(a) Method

Records per user n	8	Minimum group size $g_{\min}$	2
Aspects of working practice $\lvert\mathcal{D}\rvert$	12	Word budget per skill (redraw)	700 (489) words
Extra aspects per user	at most 2	Group preferences Q_j	2,500 chars
Lines per stance profile P_u	at most 14	Scope description σ_j	35 words
Voting agents v	3	Skills that draft an answer	all k

(b) Evaluation, the same for every method

Test users per dataset	12	Contested decisions per user set	8
Checklist elements per reference	at most 6		

(c) LLM call	Input limit	Temperature	Output tokens
Analyst (Prompt 1)	1,400 per record	0.2	1,800
Rule extractor (Prompt 2)	1,200 per record; 14,000 total	0.2	2,600
Voter (Prompt 3)	1,100 per profile; 16,000 total	0.6	700
Reconciler (Prompt 4)	12,000 proposals; 16,000 profiles	0.2	700
Builder (Prompts 5–6)	24,500 analyses; 2,500 profiles	0.2	3,500
Scope (Prompt 7)	6,000 profiles	0.0	120
Drafting	–	0.5	4,096
Merging (Prompt 8)	4,000 task; 14,000 per draft	0.2	4,096
Answer judge (Prompt 9)	–	0.0	4
Other judges (Prompts 10–16)	up to 20,000 per item	0.0	120–900
RR selector (Prompt 17)	2,500 decision	0.0	60

Table 7: CIL evaluation data for four studied datasets. For each test task, content elements are extracted from its reference record and labeled as conflict-exposed or unexposed.

Statistic	`Enron`	`20-MAD`	`SE-CS`	`AntiSkillBench`
Test tasks	42	72	48	48
Exposed elements	225	359	107	284
Unexposed elements	159	236	94	35

Table 7 shows that all four datasets contain a substantial number of conflict-exposed elements, providing sufficient evaluation items for CIL. Conflict-exposed elements account for 58.6% of the extracted elements on `Enron`, 60.3% on `20-MAD`, 53.2% on `SE-CS`, and 89.0% on `AntiSkillBench`. Thus, at least half of the evaluated content in every dataset may be handled differently under different working practices. The exposed and unexposed elements are relatively balanced on the three real world datasets, indicating that their reference records contain both content affected by working-practice differences and content required regardless of those differences. In contrast, most elements in `AntiSkillBench` are conflict-exposed, which is consistent with its test dialogues containing persona-specific requirements in later user turns.

D A Conflict Example on Enron

For each `Enron` user, an LLM judge examines the first eight records to identify an email-handling decision for which two employees prefer incompatible actions, and derives one evidence-backed rule for each. This yields 25 candidate pairs, of which 24 remain after removing one topic-specific case. For each pair, we construct four two-rule skills: *A + neutral*, *B + neutral*, *A + B*, and *disagreement + neutral*, where the neutral rule is unrelated and shared by both users. Evaluation uses 42 real

Table 8: Results on 42 `Enron` email-reply tasks for the selected conflict pair. 'Follows A' is the share of answers that verify and consult before committing. 'Weighs' is the share of reasoning traces that consider both approaches. 'Quality' compares the skill-based answer with the no-skill answer. 'Δ' is the difference from the average of the two single-rule skills. For 'Weighs', 'Δ' is reported in percentage points (pp).

Skills	**Gemini-2.5-Flash, no reasoning**		**DeepSeek-V4-Pro, reasoning**		
	Follows A	Quality	Follows A	Weighs	Quality
No skill	80%	—	76%	5%	—
Rule A + neutral	89%	+0.17	98%	26%	+0.02
Rule B + neutral	68%	−0.33	76%	5%	−0.12
Rules A and B	83%	−0.13	74%	100%	−0.48
Stated disagreement	93%	+0.11	98%	79%	+0.21
Δ, rules A and B		−0.05		+85 pp	−0.43
Δ, stated disagreement		+0.19		+63 pp	+0.26

emails received and answered by 12 test users, comparing each skill-based answer with a no-skill answer from the same base LLM. When reasoning is enabled, the judge also labels whether the trace considers both approaches.

Before generating any answers, we fixed the selection rule: evaluate all pairs with 'GPT-4.1-mini' and 'Qwen-2.5-72B' without reasoning, keep pairs for which neither single-rule skill performs worse than no skill on average, and select the pair whose combined skill performs worst relative to the average of the two single-rule skills. The selected rule pair contains the rules "verify facts and consult with others before committing to a position or accepting responsibility for a problem" and "Make commitments and take positions on matters based on your own judgment without first checking or verifying details with others." We call the rule followed more often under the combined skill rule A. The neutral rule is "Include specific reference numbers or identifiers when discussing transactions or requests." These rules describe the behavior observed in the selected records rather than fixed user traits. With reasoning, we additionally label whether the trace considers both approaches and whether it identifies a conflict between them.

Table 8 shows that, with reasoning, presenting both rules makes 'DeepSeek-V4-Pro' weigh the two approaches in all 42 traces, compared with 26% and 5% under the two single-rule skills. However, 74% of its answers still follow rule A, close to the 76% no-skill rate, and answer quality is 0.43 below the single-rule average. This suggests that the model recognizes the conflict but does not resolve it effectively: it spends reasoning effort comparing the rules, then largely falls back to its original preference while producing a weaker answer.

On the two models not used for pair selection, without reasoning, the corresponding quality difference is only -0.05, indicating that the conflict is not consistently harmful when the model does not explicitly reason through it. Explicitly stating the disagreement performs better, with quality 0.26 above the single-rule average with reasoning and 0.19 above it without reasoning. By presenting the conflict as an acknowledged team difference rather than two simultaneous commands, this formulation gives the model a clearer basis for choosing the approach that fits the email.

E Limitations and Future Work

Computation Cost. Section 4.5 shows that ConCord incurs higher token cost than the baselines, as it constructs multiple skills and, for every task, generates multiple drafts and merges them into a final response. Future work can reduce this overhead through more compact skill construction, selective drafting, or adaptive routing that invokes only the users or skills most relevant to a task.

Reliance on LLMs. All stages of ConCord are carried out by LLMs, enabling fully automated skill distillation and coordination. However, its performance may depend on the capabilities and biases of the chosen base LLM. Future work can validate our results through human evaluation, using multiple domain experts to assess response quality, conflict resolution, and information coverage, and to examine the consistency between human judgments and LLM evaluation.