From Tool to Teammate: LLM Coding Agents as Collaborative Partners for Behavioral Labeling in Educational Dialogue Analysis

Eason Chen
Carnegie Mellon University
eason.tw.chen@gmail.com

Nina Yuan
Carnegie Mellon University
Mingyuyu@andrew.cmu.edu

Isabel Wang
Carnegie Mellon University
yumow@andrew.cmu.edu

Sophia Judicke
Carnegie Mellon University
sjudicke@andrew.cmu.edu

Kayla Beigh
Carnegie Mellon University
kbeigh@andrew.cmu.edu

Xinyi Tang
Carnegie Mellon University
xtang2@andrew.cmu.edu

ABSTRACT

We present a methodology where LLM coding agents autonomously improve classification prompts for behavioral labeling of educational dialogues. In each iteration, a coding agent analyzes disagreements between classifier output and human labels, then proposes theory-grounded prompt modifications. Applying this to 659 AI tutoring sessions with three agents and three classifiers, 4-fold cross-validation indicated non-trivial gains on held-out data: the best agent improved from baseline $\kappa = 0.70$ to test $\kappa = 0.78$ (SD= 0.12), at \$5–8 per agent in Experiment 2. The process also surfaced undocumented labeling patterns, and continued iteration beyond the optimum led to regression. Iteration logs, prompts, and evaluation scripts are available at: https://github.com/EasonC13/EDM26-Agents.

Keywords

LLM labeling, behavioral analytics, prompt engineering, AI tutoring, educational data mining

1. INTRODUCTION

Understanding student behavior in tutoring dialogues is fundamental to improving educational technology. Detecting patterns such as help-seeking, confusion, and disengagement enables researchers to evaluate tutoring effectiveness and design adaptive interventions [3, 1, 7, 8]. Yet behavioral labeling remains a costly bottleneck: trained coders must manually analyze hundreds of sessions, a process requiring weeks of effort and limiting the scale of educational dialogue research. LLMs offer a promising solution, achieving strong performance on qualitative coding tasks [12, 11]. However, developing effective prompts still relies heavily on researcher trial-and-error over days or weeks.

We present an approach where LLM coding agents serve as *research teammates* that iteratively improve labeling prompts through systematic error analysis. In an exploratory phase, a researcher collaborated with Claude Code over 21 iterations, improving Follow-up Type κ from 0.59 to 0.68 at a cost of \$22 and surfacing a key insight: human coders consistently treated expressions of confusion ("I don't know," "I'm stuck") as engagement rather than disengagement. To validate generalizability, we assigned the same task to three independent coding agents (OpenAI Codex, Claude Code, Gemini), each starting from scratch. On the development set, all three improved substantially, with Codex and Gemini reaching $\kappa = 0.93$. Since iterating on a fixed validation set risks overfitting, our primary evaluation uses 4-fold cross-validation: the best agent achieved held-out test $\kappa = 0.78$ (SD= 0.12), improving over the $\kappa \approx 0.70$ baseline, at \$5–8 per agent. Agents independently converged on similar disambiguation rules, suggesting the methodology identifies genuine patterns. We also tested three classifier models (GPT, Claude, Gemini), finding that same-family advantages manifest as faster convergence rather than higher ceilings in development-set analyses.

Our contributions are a collaborative agent-refinement workflow, held-out evidence of non-trivial generalization gains, an analysis of regression beyond the optimal iteration, and a demonstration that disagreement analysis can surface implicit labeling criteria.

2. METHODOLOGY

Our work builds on help-seeking research in tutoring dialogues [3, 1], LLM-based qualitative coding in education [12, 11, 18, 20], human-in-the-loop educational judgment workflows [13, 2, 9, 10, 17], AI agents in education [5, 4, 6], and automated prompt optimization methods such as APE, DSPy, and OPRO [21, 15, 19]. Unlike metric-only prompt search, our workflow uses coding agents to perform qualitative error analysis and propose theory-grounded revisions under researcher oversight. More specifically, prior work often studies LLMs as collaborators in grading, feedback, or assessment, whereas far less work examines how reliably coding-agent-style systems can autonomously iterate on heuristic rules or prompts for theory-grounded labeling tasks. This distinction matters for educational coding tasks, where prompt changes can improve a metric while weakening pedagogical

Eason Chen, Yumo Wang, Mingyu Yuan, Sophia Judicke, Kayla Beigh, and Xinyi Tang. From Tool to Teammate: LLM Coding Agents as Collaborative Partners for Behavioral Labeling in Educational Dialogue Analysis. In Anthony Botelho, Maria Mercedes T. Rodrigo, Adish Singla, Hiroaki Ogata, Hyojeong So, and Young Hoan Cho (eds.) Proceedings of the 19th International Conference on Educational Data Mining, Seoul, Republic of Korea, June, 2026, pp. 679–683. International Educational Data Mining Society (2026).

https://doi.org/10.5281/zenodo.21039766

Algorithm 1 Agent-Driven Iterative Prompt Refinement

Require: Labeled validation set D, initial prompt P_0, classifier M, coding agent A

1: **for** $i = 1$ to max_iterations **do**
2: Run classifier M with prompt P_{i-1} on D
3: Compute κ and inspect disagreement patterns
4: Agent A proposes theory-grounded prompt changes
5: Researcher reviews and approves valid changes
6: $P_i \leftarrow$ revised prompt
7: **end for**
8: **return** $P^* = \arg\max_{P_i} \kappa(P_i, D)$

interpretation.

A coding agent receives a codebook, human-labeled validation sessions, and an evaluation script. In each iteration, it runs the classifier (temperature= 0), computes Cohen's κ, analyzes disagreements against human consensus, and proposes prompt revisions. The classifier is the model producing labels; the coding agent is responsible for error analysis and prompt revision. The researcher reviews proposed changes to preserve construct validity and pedagogical faithfulness.

We used chatbot interaction data from an undergraduate discrete mathematics course (∼120 students) at a large research university. The course deployed an LLM-powered tutoring system with a "Socratic guardrail": the chatbot used hints and guiding questions rather than direct answers. This setting creates natural behavioral variation, as some students engaged with the scaffolding while others attempted to extract solutions, making the labeling task both challenging and educationally meaningful.

Our dataset comprises 659 sessions collected over two semesters (Fall 2024 and Spring 2025), ranging from brief clarification requests (1–2 exchanges) to extended problem-solving dialogues (20+ exchanges). The median session length was 6 exchanges (IQR: 3–12). Topics spanned logic and propositional reasoning (28%), proof techniques (35%), set theory (22%), and combinatorics (15%). For validation, we randomly sampled 80 sessions stratified by length and topic. Two trained research assistants independently labeled these sessions after two calibration meetings to align on codebook interpretation and edge cases; the first author resolved disagreements through case-by-case discussion, documenting reasoning for each adjudication. This study was approved by our institution's Institutional Review Board (IRB); all student data were collected with informed consent and de-identified prior to analysis.

Drawing on help-seeking theory [1], we defined three labeling dimensions:

- **Student Intent** (Answer-Seeking, Help-Seeking, Other): captures whether the student primarily wants the answer or genuinely seeks to understand.
- **Topic Type** (Conceptual, Procedural): categorizes the mathematical content as definitional or problem-solving.
- **Follow-up Type** (Engage, Escalate, Switch): characterizes the student's response to guidance. Engage includes expressions of confusion ("I don't know"), since admitting confusion signals continued cognitive engagement [14]. Escalate requires an explicit request for direct answers, while Switch indicates topic abandonment.

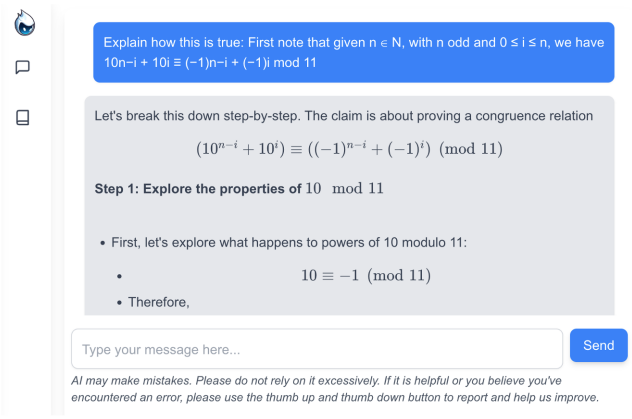

Figure 1: Example tutoring dialogue. Such sessions were labeled for student intent, topic type, and follow-up behavior.

Human coders achieved substantial inter-rater reliability: Intent $\kappa = 0.78$, Topic $\kappa = 0.73$, Follow-up $\kappa = 0.70$ [16]. Overall $\kappa = 0.89$ was computed over composite labels. To avoid mixing metrics, we compare overall model performance only against the human composite $\kappa = 0.89$, and per-dimension results only against the corresponding human kappas.

3. EXPERIMENTS AND RESULTS

3.1 Cross-Agent Validation

To test whether the methodology generalizes across agent architectures, we assigned the labeling task to three independent coding agents: OpenAI Codex, Gemini Code Agent, and Claude Code. Each received identical instructions and access to the codebook, the 80 labeled sessions, and the evaluation script. They worked independently, without access to each other's solutions. All used GPT-5.2 as a classifier for comparability. We kept the classifier fixed because the research question is whether coding agents can improve prompts for a downstream labeling model; using the agent itself as the classifier would conflate prompt-refinement ability with label-generation ability.

Starting from baseline $\kappa \approx 0.70$, after approximately 10 iterations ($5–8 per agent in Experiment 2), all three achieved strong development-set reliability. Table 1 presents the results: Codex and Gemini tied at $\kappa = 0.93$, with Gemini achieving the highest F1 (0.91). Figure 2 shows the progression, including regression when Codex was extended beyond its v7 optimum through v10 (κ dropped to 0.91, with Follow-up Type declining from 0.83 to 0.75), demonstrating that more iterations do not always yield improvement. The hardest dimension was Follow-up Type: across agents, much of the eventual gain came from converging on the rule that confusion expressions should default to Engage unless paired with an explicit request for direct answers.

Table 1: Cross-agent validation results (development set). Best performance shown for each agent.

Agent	Ver.	Overall κ	F1	Intent κ	Topic κ	Follow κ	Cost
Codex	v7	**0.93**	0.88	0.84	**0.85**	0.83	$8
Gemini	v9	**0.93**	**0.91**	**0.85**	0.81	**0.87**	$7
Claude Code	v5	0.91	0.79	0.73	0.78	0.88	$7
Human	–	0.89	–	0.78	0.73	0.70	–

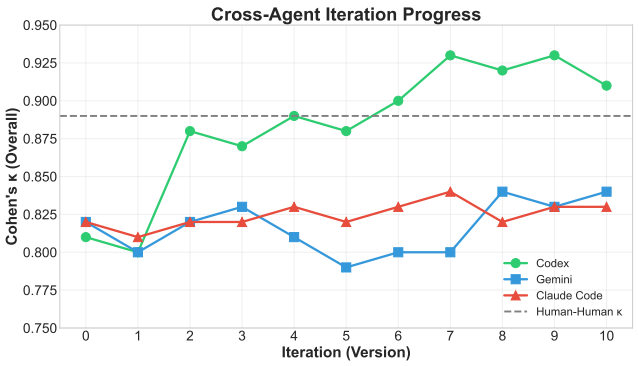

Figure 2: Overall κ progression across iterations. Codex and Gemini both reached $\kappa = 0.93$. Continued iteration led to regression for Codex after v7. Dashed line: human baseline ($\kappa = 0.89$).

Table 2: 4-fold cross-validation results (held-out test κ). Baseline $\kappa \approx 0.70$ for all agents.

Agent	Fold 0	Fold 1	Fold 2	Fold 3	Mean ± SD
Codex	0.85	0.61	0.88	0.79	0.78 ± 0.12
Claude	0.71	0.81	0.84	0.66	0.76 ± 0.08
Gemini	0.71	0.67	0.88	0.63	0.72 ± 0.11

Despite working independently, agents converged on similar disambiguation rules: (1) Switch reserved for truly unrelated topics; (2) Escalate requires explicit answer requests; (3) Engage as default for continued dialogue, including expressions of confusion. This independent convergence across three architectures suggests the methodology identifies genuine patterns rather than agent-specific artifacts.

3.2 Cross-Validation

Because all three agents iterated on the same 80 sessions, strong development-set performance may reflect overfitting. To assess generalization, we conducted a 4-fold cross-validation, splitting the 80 sessions into four folds of 20 sessions each, stratified by intent distribution. We chose 4-fold rather than 10-fold because with three multi-class dimensions, 8-session folds risk zero instances of rare categories (e.g., Switch), making κ estimation unreliable. For each fold, agents developed prompts using 60 training sessions and evaluated on 20 held-out sessions (113 total iterations across all agent-fold combinations).

Table 2 reveals substantial overfitting: Codex dropped from development $\kappa = 0.93$ to test 0.78; Claude from 0.91 to 0.76; Gemini from 0.93 to 0.72 ($\Delta\kappa \approx 0.15$–$0.21$). Nevertheless, the methodology shows improvement over baseline: all agents improved from baseline test $\kappa \approx 0.70$ to 0.72–0.78 on held-out data. The best result (Codex, $\kappa = 0.78$, SD= 0.12) remains below human composite agreement ($\kappa = 0.89$), so the held-out result should be interpreted as a meaningful gain over baseline rather than as surpassing human overall agreement. Variance across folds (SD= 0.08–0.12) reflects sensitivity to fold composition given the small validation pool.

Table 3: Best development-set κ across classifiers.

Agent	GPT	Claude	Gemini
Codex	0.93	0.93	0.87
Claude Code	0.91	0.91	0.89
Gemini	0.93	0.85	0.92

3.3 Classifier Effects

We also explored whether a "same-family advantage" exists when agent and classifier share the same model architecture. These comparisons are descriptive development-set analyses rather than held-out cross-validation results. Same-family pairings generally converged faster, but did not consistently produce higher ceilings. For example, Gemini Agent reached $\kappa = 0.92$ by v5 with Gemini but required v9 to reach 0.93 with GPT, while Codex reached 0.93 with both GPT and Claude but showed smoother convergence with GPT. Overall, classifier choice appears to matter more for iteration dynamics than for the final peak achieved on the development set.

4. DISCUSSION AND CONCLUSION

Convergent solutions and tacit knowledge. The most striking finding is independent convergence across Codex, Claude Code, and Gemini. Despite different model families, all three agents moved toward similar disambiguation rules, especially around the treatment of confusion expressions. The key rule, "confusion = engagement," was absent from the original codebook and emerged through systematic human-AI disagreement analysis, aligning with productive failure theory [14]. This suggests that the agent workflow can do more than optimize prompts: it can help surface tacit labeling criteria that researchers already apply but have not yet articulated explicitly.

Iteration dynamics and stopping rules. Continued iteration beyond the optimum can hurt performance through overcorrection or output instability. In practical terms, once an agent reaches strong agreement, additional rules may introduce new errors while fixing only a handful of old ones. Our results therefore support a simple stopping principle:

use held-out validation to select the best prompt version rather than assuming the latest version is the best. This is a straightforward methodological recommendation for prompt-engineering research that has often been missing from prior educational coding work.

What the classifier comparison suggests. The classifier comparison adds a more tentative point. Same-family pairings did not consistently yield higher final peaks, but they often converged faster. That pattern suggests that model family may matter more for the efficiency and stability of iteration than for the final achievable performance. Because these analyses remain development-set only, we treat them as exploratory.

Practical implications for educational research. From an applied perspective, the workflow lowers the cost of rigorous dialogue coding. Each agent required only $5–8 for roughly ten iterations, and the total researcher effort remained modest compared with traditional multi-coder training and calibration. The gains are not large enough to claim that agents replace human agreement, but they are meaningful enough to support hybrid workflows in which agents accelerate prompt refinement, expose edge cases, and reduce the manual burden of theory-grounded labeling.

Prompt strategies and why they mattered. A useful secondary observation is that not all prompt-revision styles behaved the same way. The strongest prompts tended to make category boundaries explicit through ordered decision rules, especially for the hardest dimension, Follow-up Type. In contrast, more heuristic prompt language left greater interpretive burden on the downstream classifier. This matters for educational dialogue coding because the task often hinges on subtle distinctions such as confusion versus disengagement, or productive help-seeking versus answer-seeking. Our results therefore suggest that for nuanced theory-grounded coding tasks, agents are most helpful when they distill tacit qualitative judgments into concrete disambiguation rules rather than merely adding more examples or longer prose.

Recommendations for future users of this workflow. Two practical recommendations follow. First, researchers should begin with a compact prompt derived directly from a codebook, then let the agent target the weakest dimension rather than modifying all dimensions at once. Second, they should save richer artifacts than we did here: fold assignments, selected prompt versions, and per-dimension held-out outputs. Doing so would make later audit, replication, and paper writing much easier.

What the remaining errors suggest. Our error analysis also points toward concrete future directions for educational dialogue research. Remaining disagreements clustered into four recurring patterns: ambiguous intent, context-dependent interpretation, multi-turn dynamics, and rare interaction patterns. Some were cases of genuinely ambiguous intent, where even human coders disagreed before adjudication. Others reflected context dependence across turns, where the same phrase could signal productive confusion in one exchange and frustration in another. These patterns suggest that future systems may benefit from richer representations than single-label session coding alone, such as turn-level coding, uncertainty-aware labeling, or multi-label treatments for borderline cases. In that sense, the present workflow is useful not only for prompt optimization but also for diagnosing where the current coding ontology itself begins to strain.

Why human oversight remains important. One lesson from the full iteration history is that higher agreement alone is not always enough. Some prompt changes initially looked attractive because they improved a subset of cases, but they did so by blurring theoretically meaningful distinctions. Human review therefore remains central, not as a bottleneck to eliminate, but as the mechanism that keeps the refinement process aligned with pedagogical constructs rather than with metric chasing alone.

Limitations. The study also has clear limitations. The 80-session validation set limits statistical power and creates overfitting risk, and we did not retain fold-level per-dimension held-out artifacts or full fold metadata in the current supplementary release. Thus, our preserved held-out evidence is stronger for overall composite performance than for dimension-specific generalization. Because fold sizes are small, we treat differences between agents and classifiers descriptively rather than as statistically significant. Cross-validation also provides a held-out estimate within the 80-session pool, not a completely untouched final test set from the broader 659-session corpus. We also relied on commercial models and coding-agent frameworks, which limits exact reproducibility despite releasing prompts, logs, and evaluation scripts. We did not compare against fine-tuned models, and session-level classification cannot capture within-session intent shifts. Finally, although the coding scheme transfers conceptually to other tutoring contexts, some decision rules are domain-shaped by mathematical dialogue and would likely need adaptation in writing, science, or language-learning settings.

Why this matters methodologically. Prompt engineering for educational coding should be documented as a process rather than reported only as a final prompt plus a final score.

Overall, the results suggest that LLM coding agents can act as collaborative partners in codebook development, yielding held-out gains and useful stopping rules.

The agent's iteration logs, prompts, and evaluation scripts are available at: https://github.com/EasonC13/EDM26-Agents.

5. REFERENCES

[1] V. Aleven and K. R. Koedinger. Limitations of Student Control: Do Students Know when They Need Help? In G. Gauthier, C. Frasson, and K. VanLehn, editors, *Intelligent Tutoring Systems*, pages 292–303, Berlin, Heidelberg, 2000. Springer.

[2] D. Armfield, E. Chen, A. Omonkulov, X. Tang, J. Lin, E. Thiessen, and K. R. Koedinger. Avalon: a human-in-the-loop llm grading system with instructor calibration and student self-assessment. In *International Conference on Artificial Intelligence in Education*, pages 111–118, 2025.

[3] R. S. Baker, A. T. Corbett, and K. R. Koedinger. Detecting Student Misuse of Intelligent Tutoring Systems. In J. C. Lester, R. M. Vicari, and

F. Paraguaçu, editors, *Intelligent Tutoring Systems*, pages 531–540, Berlin, Heidelberg, 2004. Springer.

[4] E. Chen, C. Guan, A. Elshafiey, Z. Zhao, J. Zekeri, A. Edeifo Shaibu, and E. Osadebe Prince. When openclaw ai agents teach each other: Peer learning patterns in the moltbook community. *arXiv e-prints*, pages arXiv–2602, 2026.

[5] E. Chen, C. Guan, A. Elshafiey, Z. Zhao, J. Zekeri, A. Edeifo Shaibu, E. Osadebe Prince, and C.-J. Wu. When openclaw agents learn from each other: Insights from emergent ai agent communities for human-ai partnership in education. *arXiv e-prints*, pages arXiv–2603, 2026.

[6] E. Chen, C. Guan, A. Elshafiey, Z. Zhao, J. Zekeri, A. E. Shaibu, E. O. Prince, and C. J. Wu. Openclaw ai agents as informal learners at moltbook: Characterizing an emergent learning community at scale. *arXiv preprint arXiv:2602.18832*, 2026.

[7] E. Chen, S. Judicke, K. Beigh, X. Tang, I. Wang, N. Yuan, Z. Xiao, C. Li, S. Li, R. Luttmer, S. Singh, and K. R. Koedinger. Chat-based support alone may not be enough: Comparing conversational and embedded llm feedback for mathematical proof learning. *arXiv preprint arXiv:2602.18807*, 2026.

[8] E. Chen, S. Judicke, K. Beigh, X. Tang, Z. Xiao, C. Li, S. Li, R. Luttmer, S. Singh, and K. R. Koedinger. Generative ai alone may not be enough: Evaluating ai support for learning mathematical proof. *arXiv preprint arXiv:2509.16778*, 2025.

[9] E. Chen, J. Li, S. Huang, X. Tang, J. Lin, P. Carvalho, and K. Koedinger. Identifying effective praise in tutoring: Large language models with transparent explanations. In *International Conference on Artificial Intelligence in Education*, pages 157–163, 2025.

[10] E. Chen, J. Li, S. Huang, X. Tang, J. Lin, P. Carvalho, and K. Koedinger. Ai knows best? the paradox of expertise, ai reliance, and performance in educational tutoring decision-making tasks. In *Proceedings of the LAK26: 16th International Learning Analytics and Knowledge Conference*, pages 736–743, 2026.

[11] W. Dai, J. Lin, H. Jin, T. Li, Y.-S. Tsai, D. Gašević, and G. Chen. Can large language models provide feedback to students? a case study on chatgpt. *International Conference on Artificial Intelligence in Education*, pages 323–335, 2023.

[12] W. Dai, Y.-S. Tsai, J. Lin, A. Aldino, H. Jin, T. Li, D. Gašević, and G. Chen. Assessing the proficiency of large language models in automatic feedback generation: An evaluation study. *Computers and Education: Artificial Intelligence*, 7:100299, 2024.

[13] Z. F. Han, J. Lin, A. Gurung, D. R. Thomas, E. Chen, C. Borchers, S. Gupta, and K. R. Koedinger. Improving assessment of tutoring practices using retrieval-augmented generation. In *AAAI2024 Workshop on AI for Education - Bridging Innovation and Responsibility*, 2024.

[14] M. Kapur. Productive failure. *Cognition and Instruction*, 26(3):379–424, 2008.

[15] O. Khattab, A. Singhvi, P. Maheshwari, Z. Zhang, K. Santhanam, S. Vardhamanan, S. Haq, A. Sharma, T. T. Joshi, M. Moez, H. Hajishirzi, C. Potts, M. Zaharia, and C. Ré. DSPy: Compiling declarative language model calls into self-improving pipelines. *arXiv preprint arXiv:2310.03714*, 2023.

[16] J. R. Landis and G. G. Koch. The Measurement of Observer Agreement for Categorical Data. *Biometrics*, 33(1):159–174, 1977.

[17] J. Lin, E. Chen, Z. Han, A. Gurung, D. R. Thomas, W. Tan, N. D. Nguyen, S. Gupta, V. Aleven, and K. R. Koedinger. How can i improve? using gpt to highlight the desired and undesired parts of open-ended responses. In *Educational Data Mining Conference 2024*, 2024.

[18] N. Tran, B. Pierce, D. Litman, R. Correnti, and L. C. Matsumura. Analyzing Large Language Models for Classroom Discussion Assessment. In *Proceedings of the 17th International Conference on Educational Data Mining*, pages 500–510, Atlanta, Georgia, USA, 2024. International Educational Data Mining Society.

[19] C. Yang, X. Wang, Y. Lu, H. Liu, Q. V. Le, D. Zhou, and X. Chen. Large language models as optimizers. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2023.

[20] J. Zhang, C. Borchers, V. Aleven, and R. S. Baker. Using Large Language Models to Detect Self-Regulated Learning in Think-Aloud Protocols. In *Proceedings of the 17th International Conference on Educational Data Mining*, pages 157–168, Atlanta, Georgia, USA, 2024. International Educational Data Mining Society.

[21] Y. Zhou, A. I. Muresanu, Z. Han, K. Paster, S. Pitis, H. Chan, and J. Ba. Large language models are human-level prompt engineers. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2023.