Work-in-Progress: Exploring Locally Hosted Low-Cost LLMs for Qualitative Content Analysis of Engineering Curricula

Anthony Ha-Anh Pham, Texas A&M University

Anthony Ha-Anh Pham is a graduate student at Texas A&M, earning his bachelor's in Multidisciplinary Engineering in 2025 and working towards his masters degree in engineering technology. Alongside his thesis research, he has been doing research on the creation and implementation of generative AI-based tools in engineering education.

Dr. Lance Leon Allen White, Texas A&M University

Lance White is a recent Ph.D. from Texas A&M University in Interdisciplinary Engineering with a thrust in Engineering Education. He is working as a Lecturer for the Engineering Academic and Student Affairs unit at Texas A&M University in the College of Engineering. He teaches first-year engineering coursework and AI literacy courses available to all students. His research focuses in organizational change, faculty development, LLM assisted qual methods development, narrative inquiry, first-year engineering, and broadening participation.

Dr. Karan Watson P.E., Texas A&M University

Karan L. Watson, Ph.D., P.E., is currently a Regents Senior Professor of Electrical and Computer Engineering, having joined the faculty at Texas A&M University in 1983 as an Assistant Professor. She is also serving as the C0-Director of the Institute

Dr. Kristi J. Shryock, Texas A&M University

Dr. Kristi J. Shryock is the Frank and Jean Raymond Foundation Inc. Endowed Associate Professor in Multidisciplinary Engineering and Affiliated Faculty in Aerospace Engineering in the College of Engineering at Texas A&M University. She also serves as Director of the Craig and Galen Brown Engineering Honors Program. She received her BS, MS, and PhD from the College of Engineering at Texas A&M. Kristi works to improve the undergraduate engineering experience through evaluating preparation in areas, such as mathematics and physics, evaluating engineering identity and its impact on retention, incorporating non-traditional teaching methods into the classroom, and engaging her students with interactive methods.

Work-in-Progress: Exploring Locally Hosted Low-Cost LLMs for Qualitative Content Analysis of Engineering Curricula

Introduction

This Methods/Theory Work-in-Progress paper explores how locally hosted, low-cost large language models (LLMs) can accelerate qualitative curriculum content analysis through rapid, iterative codebook refinement. Engineering degree programs are built under layered constraints: accreditation requirements, institutional policies, disciplinary norms, transfer pathways, and local faculty priorities. In practice, this creates a tension that curriculum researchers and program leaders recognize immediately where programs must remain legible to external standards while also expressing a local identity through distinctive sequencing, electives, concentrations, and signature experiences [1]. In the lead investigator's doctoral work this tension became a methodological problem [2, 3]. The dataset consists of thousands of course records spanning multiple universities and disciplines, and answering even basic comparative questions required systematic qualitative coding of course descriptions across programs.

That dissertation produced a usable codebook through a human-in-the-loop AI assisted coding process [4] but exposed a scalability limit. Curriculum content analysis can be made rigorous [5-10], but it is slow, even when using AI as a supplement. The bottleneck is not only applying codes once; it is the repeated cycle of refining definitions, reconciling edge cases, and updating the codebook as the dataset grows. This paper is motivated by a practical need: a low-cost, repeatable way to accelerate codebook refinement and code application as curricular datasets expand, without requiring proprietary, internet-connected tooling. Existing work on LLM-assisted qualitative analysis motivates this approach, but it does not validate the workflow by repetition alone [11-23].

We explore a locally hosted LLM workflow designed specifically for this dataset condition: large volumes of short, structured text (course titles/descriptions) that are consistent in format but diverse in meaning. Rather than using an LLM as a "replacement coder," we treat the LLM as a codebook stress-testing engine: it iteratively critiques, revises, and re-applies code definitions, while documenting each change for auditability. Using a system in this manner would suggest that narrowly defined codes should converge more easily than ambiguous ones. However, divergence itself may be informative rather than a failure of the system. Conceptually, the study is guided by qualitative content analysis and human-in-the-loop AI: codebook revisions remain analytic decisions, and LLM outputs are treated as prompts for human review rather than findings [7-9, 18, 19].

Research Questions:

RQ1: To what extent can an iterative LLM-driven workflow reduce the human burden of codebook refinement, shifting humans from primary editors to supervisors?

RQ2: To what extent can a small, locally hosted open model adopt and apply a curriculum codebook with reliable agreement against a human baseline?

RQ3: Can convergence and divergence patterns be leveraged as signals about curricula, especially for identifying course structures that do not fit an existing codebook and warrant new codes or analytic attention?

Contributions:
This work-in-progress contributes: 1) An auditable, locally hosted pipeline for iterative codebook refinement and code application on curriculum datasets. 2) Empirical evidence through agreement trends across iterations that support the feasibility of low-cost local models for tasks of this nature. And 3) an analytic framing in which "out-of-codebook" course clusters can be treated as potential indicators of unique curricular structure, emergent topics, or misfit between inherited codebooks and evolving programs.

Data Context and Why This Dataset Drives the Design
The data, codebook, and evaluation baseline originate from dissertation work [2, 3] that compiled course records from engineering degree programs across three engineering disciplines (mechanical, electrical, and civil). Courses were coded using a structured codebook developed for content analysis of curricular emphasis. This provides three assets that make an iterative system both possible and meaningful: a large corpus of structed short text, an existing codebook, and a human-in-the-loop QA baseline to be used for evaluation. These assets were limited to just one discipline (mechanical engineering) to avoid additional complexities of slight variations of codes with the same name to reflect the nuances of each discipline. The short text contains 498 course entries containing institution and course names, course codes, and course descriptions. This text has been restricted to entries that have all four pieces of information. The codebook consists of 40 entries, each containing the code name and description. The human standard is the short text with the assigned codes as well.

The dataset condition is critical to the workflow practicality. Compared to interview or open-ended reflection data, course descriptions are typically bounded and structurally similar, reducing some sources of interpretive uncertainty. At the same time, the size of the dataset is such that manual iteration becomes quickly prohibitive [5]. In short: the texts are simple enough to automate partially on their own, but the scale is large enough that systematic and reliable automation is highly desirable.

Pipeline Overview: Iterative Codebook Refinement and Application (Local LLM)
This workflow is designed as a repeatable loop [24, 25] over codes rather than a one-shot labeling pass. The pipeline receives three inputs: 1) An unlabeled set of courses, 2) the current codebook with code names, definitions, and examples when available, and 3) a human-in-the-loop baseline standard subset for evaluation. The pipeline operationalizes codebook refinement into three stages executed sequentially for one code at a time:

Stage 1 (Criticism): The LLM critiques the current code definition using a) the code name/definition and b) examples of courses labeled positive and negative. On later iterations, it also sees false positives/false negatives from the prior cycle. The LLM outputs specific programs and proposed revisions with rationale.
Stage 2 (Review): A separate LLM instance reviews the proposed changes, either approving them or proposing an alternative revision, with justification. Using separate instances reduces cross-stage anchoring and helps keep each step focused.
Stage 3 (Application): A third LLM instance applies the revised code to a test subset. The pipeline then computes agreement against the human baseline using inter-rater reliability style metrics [26] such as Cohen's kappa [27]. A second agreement statistic, Gwet's AC1 [28-30], is

also reported to reduce sensitivity to prevalence effects [31]. These agreement statistics are used as diagnostic indicators for codebook refinement, not as evidence that interpretive validity has been automated [32]

The loop continues until either agreement reaches a predefined threshold (including the extreme case of perfect agreement) or the iteration cap is met. After each iteration, the pipeline saves a structured log: the proposed changes, the accepted revision, the application results, the updated agreement scores, and the rationale, following audit-trail and living-codebook practices for making analytical decisions inspectable [33, 34]. Humans can intervene after any iteration, but the design goal is that intervention will eventually become rare, allowing humans to supervise and adjudicate only the cases that merit attention.

In this work-in-progress paper, when we refer to "training" we are speaking of the iterative refinement of definitions and prompts, not gradient-based fine-tuning of the model weights themselves. The model itself remains fixed where the codebook becomes more operational.

Model Selection: Why a Small Local Qwen Variant

The practical constraint here is local feasibility. This work targets a configuration that can run on commonly available hardware with GPU memory that is limited when compared to commercially hosted LLM systems. Within that constraint, several models were compared with a local inference stack, and a Qwen 2.5 family 7B [35] parameter model was chosen based on consistently produced structure and instruction-following outputs in the criticism/review stages while remaining compact and locally deployable.

Expected benefits of a small local model.

- Cost control: no per-token fees, so scalable runs become feasible for unfunded or lightly funded projects.
- Governance/privacy posture: local inference avoids sending curricular text to external services innately [36].
- Reproducibility: fixed model snapshot and fixed inference parameters support repeatable iteration and auditable change logs [33, 34].

Expected limitations

- Boundary sensitivity: smaller models are more prone to confusing overlapping codes unless definitions contain explicit decision rules.
- Prompt/code definition sensitivity: minor wording differences can change outputs, increasing the importance of versioning.
- Nuance limitations: the model may "sound confident" in rationales even with evidence is weak, requiring safeguards.

These drawbacks are not treated as deal-breakers in this development. Instead, they are precisely what motivates iterative refinement and careful logging.

Preliminary Results

We report early performance here from two runs of the pipeline: one capped at five iterations and one capped at fifteen iterations. Across the codebook ($C = 40$ codes in the mechanical dataset subset), increased iterations produced a higher count of codes reaching perfect agreement and reduced the number of low-agreement codes.

Agreement trend summary (based on Cohen's kappa)

- Five-iteration cap: 15 codes reached perfect agreement with most remaining codes clustered in substantial or moderate [26, 27] agreement ranges, with a minority below 0.6
- Fifteen-iteration cap: 22 codes reached perfect agreement with every remaining code having at least fair agreement

These patterns support two preliminary findings. 1) iterative refinement can improve alignment for a substantial portion of codes, and 2) some codes resist convergence even with substantially more iterations. It is important to recognize here that the resisting codes are not necessarily "model failures". In many cases, resistance can reflect code definition overlap or genuine curricular heterogeneity where institutions may implement conceptually similar requirements, but through structurally different courses, particularly through electives and signature experiences.

Discussion: Why This Matters Beyond Speed

The most obvious value proposition is speed: when a dataset spans thousands of course records and expands over time, locally hosted automation can reduce the labor cost of re-applying and re-validating a codebook. But the more important implication is analytical: a system that converges easily on some codes and persistently diverges on others helps locate the conceptual boundaries of the codebook itself. In curriculum research, the most informative cases are often not the "typical" courses that align cleanly with common codes (e.g., calculus sequences). The informative cases are the outliers: courses that repeatedly fail to fit, or clusters that oscillate across multiple plausible codes. In a traditional workflow, these cases are expensive because they require repeated human adjudication. In the proposed workflow, they become a detectable signal. Persistent divergence can indicate at least three high-value analytic outcomes. ***Code definition ambiguity*** where the code may be conceptually meaningful but operationally underspecified; divergence suggests the definition needs an inclusion/exclusion rule or a decision hierarchy. ***Emergent topic areas*** where a cluster of courses may reflect new curricular priorities (e.g., AI/ML integration, humanitarian engineering pathways, cybersecurity emphases) that were not salient when the codebook was originally written. Divergence here is an argument for a new code or subcode. Finally, ***Institutional signature structures*** showcasing how some programs embed distinctive experiences (studio sequences, project-based thread courses, community-engaged pathways) that are not well represented in a legacy codebook. Flagging these misfits can support program leaders who want to articulate what makes their curriculum distinctive, rather than forcing everything into standardized bins. The workflow preserves the possibility of unexpected qualitative insight by routing repeated disagreement into human review rather than forcing the nearest existing label; analysts can then reread the original descriptions, compare rationale logs, and decide whether the case calls for a rule change, a new code, or a documented curricular outlier [32-34]

Under this framing, the pipeline does not merely "speed up coding." It helps distinguish between where the codebook needs tightening and where the curriculum is doing something genuinely different. That distinction matters for program assessment as it aids to avoid treating uniqueness as noise and instead surfaces it as a feature worthy of interpretation.

Limitations and Next Steps

This work has several limitations. Course descriptions vary in specificity; agreement with a human baseline is not perfect; and small local models are currently not comparable to frontier models on nuanced boundary judgments. Future work includes expanding evaluation across the remainder of the available dataset; reporting per-code performance alongside "ambiguity indicators"; implementing a label that triggers human review and supports principled codebook expansion; and comparing multiple local model sizes (e.g., Qwen 2.5 3B vs 7B) to quantify the cost–performance trade space.

Appendix A: AI Statement

We disclose that generative AI was used for this research. Claude was used to help generate boilerplate code, brainstorm ideas, and improve prose of the work-in-progress paper. Consensus was used to help find relevant papers during the initial literature review. We assert that we take full responsibility for the contents and results of our research and resulting paper and that we took the necessary steps to verify and adapt the AI results that were used in any part of the research or paper.

References

[1] ABET. "Criteria for Accrediting Engineering Programs." ABET. https://www.abet.org/accreditation/accreditation-criteria/criteria-for-accrediting-engineering-programs-2025-2026/ (accessed 2026).

[2] L. L. A. White, "Structures and Stories: How Institutional Identity Shapes Undergraduate Engineering Curricula," Ph.D. , Multidisciplinary Engineering, Texas A&M University, Oak Trust, 2025.

[3] L. White, D. Jaison, C. Stanley, T. Hammond, and K. Watson, "WIP: A Novel Methodology for Assessing Mechanical, Electrical, and Civil Engineering Programs with an Emphasis on Institutional Type," in *2023 IEEE Frontiers in Education Conference (FIE)*, 2023: IEEE, pp. 1–5.

[4] Z. O. Dunivin, "Scaling hermeneutics: a guide to qualitative coding with LLMs for reflexive content analysis," *EPJ Data Science,* vol. 14, no. 1, p. 28, 2025.

[5] M. Beresford *et al.*, "Coding qualitative data at scale: Guidance for large coder teams based on 18 studies," *International Journal of Qualitative Methods,* vol. 21, p. 16094069221075860, 2022.

[6] J. T. DeCuir-Gunby, P. L. Marshall, and A. W. McCulloch, "Developing and using a codebook for the analysis of interview data: An example from a professional development research project," *Field methods,* vol. 23, no. 2, pp. 136–155, 2011.

[7] K. M. MacQueen, E. McLellan, K. Kay, and B. Milstein, "Codebook development for team-based qualitative analysis," *Cam Journal,* vol. 10, no. 2, pp. 31–36, 1998.

[8] K. Krippendorff, *Content analysis: An introduction to its methodology*. Sage publications, 2018.

[9] H.-F. Hsieh and S. E. Shannon, "Three approaches to qualitative content analysis," *Qualitative health research,* vol. 15, no. 9, pp. 1277–1288, 2005.

[10] P. Mayring, "Qualitative content analysis: theoretical foundation, basic procedures and software solution," 2014.

[11] R. H. Tai *et al.*, "An Examination of the Use of Large Language Models to Aid Analysis of Textual Data," *International Journal of Qualitative Methods,* vol. 23, p. 16094069241231168, 2024, doi: 10.1177/16094069241231168.
[12] F. Gilardi, M. Alizadeh, and M. Kubli, "ChatGPT outperforms crowd workers for text-annotation tasks," *Proceedings of the National Academy of Sciences,* vol. 120, no. 30, p. e2305016120, 2023.
[13] L. Abraham, C. Arnal, and A. Marie, "Prompt selection matters: enhancing text annotations for social sciences with large language models," *Journal of Computational Social Science,* vol. 8, no. 3, p. 73, 2025.
[14] M. S. Deiner, V. Honcharov, J. Li, T. K. Mackey, T. C. Porco, and U. Sarkar, "Large language models can enable inductive thematic analysis of a social media corpus in a single prompt: human validation study," *JMIR infodemiology,* vol. 4, no. 1, p. e59641, 2024.
[15] S. Amani, L. White, T. Balart, K. Watson, and K. Shryock, "Applying Generative AI for Thematic Analysis: A Model for Researchers Exploring Qualitative Methods," *Available at SSRN 5128905,* 2025.
[16] D. L. Morgan, "Exploring the use of artificial intelligence for qualitative data analysis: The case of ChatGPT," *International journal of qualitative methods,* vol. 22, p. 16094069231211248, 2023, doi: https://doi.org/10.1177/16094069231211248.
[17] S. De Paoli, "Performing an Inductive Thematic Analysis of Semi-Structured Interviews With a Large Language Model: An Exploration and Provocation on the Limits of the Approach," *Social Science Computer Review,* vol. 42, no. 4, pp. 997–1019, 2024, doi: 10.1177/08944393231220483.
[18] J. L. Feuston and J. R. Brubaker, "Putting tools in their place: The role of time and perspective in human-AI collaboration for qualitative analysis," *Proceedings of the ACM on Human-Computer Interaction,* vol. 5, no. CSCW2, pp. 1–25, 2021, doi: https://doi.org/10.1145/3479856.
[19] S. A. Gebreegziabher, Z. Zhang, X. Tang, Y. Meng, E. L. Glassman, and T. J.-J. Li, "Patat: Human-ai collaborative qualitative coding with explainable interactive rule synthesis," in *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, 2023, pp. 1–19, doi: https://doi.org/10.1145/3544548.3581352.
[20] S. Lopez-Fierro and H. Nguyen, "Making Human-AI contributions transparent in qualitative coding," in *Proceedings of the 17th International Conference on Computer-Supported Collaborative Learning-CSCL 2024, pp. 3-10*, 2024: International Society of the Learning Sciences.
[21] D. C. Nguyen and C. Welch, "Generative artificial intelligence in qualitative data analysis: Analyzing—Or just chatting?," *Organizational Research Methods,* vol. 29, no. 1, pp. 3–39, 2026.
[22] K. Nguyen-Trung, "ChatGPT in thematic analysis: Can AI become a research assistant in qualitative research?," *Quality & Quantity,* vol. 59, no. 6, pp. 4945–4978, 2025, doi: 10.1007/s11135-025-02165-z.
[23] R. Sinha, I. Solola, H. Nguyen, H. Swanson, and L. Lawrence, "The role of generative AI in qualitative research: GPT-4's contributions to a grounded theory analysis," in *Proceedings of the 2024 Symposium on Learning, Design and Technology*, 2024, pp. 17–25.

[24] A. Madaan *et al.*, "Self-refine: Iterative refinement with self-feedback," *Advances in neural information processing systems,* vol. 36, pp. 46534–46594, 2023.
[25] N. Shinn, F. Cassano, A. Gopinath, K. Narasimhan, and S. Yao, "Reflexion: Language agents with verbal reinforcement learning," *Advances in neural information processing systems,* vol. 36, pp. 8634–8652, 2023.
[26] J. Landis, "The Measurement of Observer Agreement for Categorical Data," *Biometrics,* 1977, doi: https://doi.org/10.2307/2529310.
[27] J. Cohen, "A coefficient of agreement for nominal scales," *Educational and psychological measurement,* vol. 20, no. 1, pp. 37–46, 1960.
[28] K. L. Gwet, "Computing inter-rater reliability and its variance in the presence of high agreement," *British Journal of Mathematical and Statistical Psychology,* vol. 61, no. 1, pp. 29–48, 2008.
[29] N. Wongpakaran, T. Wongpakaran, D. Wedding, and K. L. Gwet, "A comparison of Cohen's Kappa and Gwet's AC1 when calculating inter-rater reliability coefficients: a study conducted with personality disorder samples," *BMC medical research methodology,* vol. 13, no. 1, p. 61, 2013.
[30] T. Byrt, J. Bishop, and J. B. Carlin, "Bias, prevalence and kappa," *Journal of clinical epidemiology,* vol. 46, no. 5, pp. 423–429, 1993.
[31] W. Vach and O. Gerke, "Gwet's AC1 is not a substitute for Cohen's kappa–A comparison of basic properties," *MethodsX,* vol. 10, p. 102212, 2023.
[32] C. O'Connor and H. Joffe, "Intercoder Reliability in Qualitative Research: Debates and Practical Guidelines," *International Journal of Qualitative Methods,* vol. 19, p. 160940691989922, 2020, doi: 10.1177/1609406919899220.
[33] B. L. Rodgers and K. V. Cowles, "The qualitative research audit trail: A complex collection of documentation," *Research in nursing & health,* vol. 16, no. 3, pp. 219–226, 1993.
[34] V. Reyes, E. Bogumil, and L. E. Welch, "The living codebook: Documenting the process of qualitative data analysis," *Sociological Methods & Research,* vol. 53, no. 1, pp. 89–120, 2024.
[35] Qwen_Team, "Qwen2.5 Technical Report," 2025. [Online]. Available: https://dx.doi.org/10.48550/arxiv.2412.15115
[36] E. Tabassi, "Artificial intelligence risk management framework (AI RMF 1.0)," 2023.