A Taylorizable Process for Textual Detector Development

Ryan S. Baker
Adelaide University

Caitlin Mills
University of Minnesota

Andrew Lan
University of Massachusetts Amherst

Blair Lehman
Brighter Research

Amanda Barany
University of Pennsylvania

ABSTRACT

Textual detectors of cognitive constructs play a central role in educational data mining, learning analytics, and automated assessment, enabling scalable analysis of student thinking, discourse, and learning processes. Recent work has increasingly adopted LLM-as-a-judge approaches, which replace traditional supervised learning with rubric-based prompting of large language models. While these approaches substantially reduce development effort, they are often deployed without the forms of rigor historically emphasized in EDM and related communities. At the same time, fully rigorous detector development remains costly and slow, creating pressure to adopt lower-quality shortcuts. In this paper, we propose a Taylorizable process for developing textual detectors of cognitive constructs using LLMs—attempting to reconcile methodological rigor with efficiency and scalability. Drawing on prior detector development practices and principles of scientific management, we decompose detector construction into a standardized, end-to-end workflow encompassing construct definition, human coding and reliability checking, LLM-based detector construction, evaluation, and application at scale. We argue that this process supports four essential forms of rigor—assessment of goodness, generalizability, construct validity, and auditability—while enabling streamlined execution by mixed-expertise teams. By making rigorous detector development more reproducible, auditable, and efficient, this framework aims to raise methodological standards for LLM-based measurement while preserving the practical advantages, including feasibility and lower development effort, that have driven their rapid adoption

Keywords

LLM-as-a-judge, Taylorization, automated detector.

1. INTRODUCTION

Many cognitive constructs are reflected in text that humans produce, which makes them possible to be automatically detected and/or measured using natural language processing tools. One of the common applications of detection within text is to detect constructs within extended texts such as essays written by humans or dialogues that they engage in. These types of extended textual data have been used for applications such as predicting the personality type of the author [50], cognitive impairment [57], writing style [29], and critical thinking [37]. A range of other behaviors have also been detected in this fashion from shorter text, including detecting confusion in MOOC discussion-forum posts [55], predicting the urgency of forum posts to support instructor triage [44], and classifying peer-tutoring chat messages into dialog-act categories linked to learning processes [10].

For decades, the preferred way to detect these types of constructs in text was a supervised learning classification paradigm, where labeled data is obtained, and then used for both training and testing. However, this approach has typically required large amounts of labeled data to train on, which may be scarce in some applications. Therefore, recent work has instead considered large language models (LLM)-as-a-judge methods that do not require training [56], building on the new capacities of pre-trained LLMs. In this approach, humans first need to define a set of rubrics, in textual format, to assess a particular construct; then, the developer simply prompts LLMs to perform this evaluation according to the rubrics, leveraging their built-in text understanding capabilities.

Developed in either fashion, the resultant detectors have many potential applications. They can be used in many fashions -- as with other types of detection (e.g., [23][27]), they can be used to guide responses by instructors or teaching assistants. They can also be applied to scalable assessment and feedback, for example by automatically characterizing the quality of student explanations, argumentation, or critical thinking in written work [2][3]. In collaborative and dialog-based learning environments, detectors of discourse features and dialog acts have been used to analyze learning processes, revealing which interaction patterns are associated with stronger learning gains [10]. Such detectors can serve important roles in learning analytics and educational research, allowing researchers to study latent constructs at scale across courses, institutions, and populations, in ways that would otherwise be prohibitively labor-intensive.

LLM-as-a-judge is a much simpler and more straightforward process than previous work in the EDM and related communities to detect constructs from text. However, this process may be too simple and straightforward. Specifically, unlike these other approaches, researchers using the term LLM-as-a-judge often skip the step of careful validation on human-created labels which have themselves undergone an intensive validation process (even though canonical papers using the term LLM-as-a-judge do take this validation step – [19]). In general, LLM-as-a-judge approaches have shown promise due to their scalability but substantial concerns have been raised about construct validity and reliability, such as limited generalizability across tasks and domains [25], whether this approach is prone to intrinsic biases [14] and whether this approach relies too much on superficial textual features [33]. Without the application of careful validation, it can be uncertain whether the resultant labels are valid and trustworthy. We are now at a time when we need proactive and principled methods for ways to include LLMs in predicting psychological constructs, so that we do not sacrifice methodological rigor or undermine the promise of high-quality adaptive applications with poor measurement.

In what follows, we propose a Taylorizable (systematic, designed to be streamlined and reproducible) process for building reliable detectors of cognitive constructs using LLMs. We believe this contribution is timely for the EDM field, particularly given the wide variety of approaches that are currently involving LLMs in core parts of the data mining process. Canonical Taylorizable processes are also perhaps necessary at this point in time so that we can hold high standards for how we operationalize and measure cognitive constructs with validity and reliability as a primary goal for both quality research and impact for students.

1.1 Why We Need Attention to Rigor Now

The implicit assumption in much of the use of the LLM-as-a-judge paradigm (again, if not in the initial work establishing this paradigm – [19]) is that we do not need the forms of rigor historically emphasized in the EDM community, the AES (automated essay scoring) community, and related scholarly communities. To be fair, the majority of researchers adopting this approach are probably not so much actively rejecting the standards of rigor used by scholars working on these types of problems over the last decades; instead, they are new to this type of problem and enthusiastic about the possibilities of LLMs. However, this leads to the need to argue for the importance of rigor, an argument that perhaps eternally needs to be made and re-made (cf. [35]).

We argue that four types of rigor are particularly essential, without ignoring the importance of a wide range of other types of validity: assessment of goodness, generalizability, construct validity, and auditability.

1.1.1 Assessment of goodness

A first and foundational form of rigor is the assessment of goodness: establishing whether a detector meaningfully distinguishes between the construct that the developer is attempting to identify, compared to cases where that construct is not present. Without such evaluation, it is uncertain whether a detector is genuinely identifying cases that meaningfully correspond to the target construct rather than to superficial properties of the text. This issue is particularly critical in the LLM-as-a-judge paradigm, where models can generate fluent, confident-seeming, and internally consistent evaluations but those evaluations may be weakly related—or unrelated—to the construct of interest [28]. Because LLM-as-a-judge approaches do not involve training on labeled data, it is easy to implicitly assume that traditional evaluation practices are unnecessary, and that having results with face validity in a small number of cases is sufficient. However, without explicit assessment of goodness, there is no empirical basis for determining whether an LLM is systematically distinguishing categories such as higher-quality explanations, stronger argumentation, or deeper critical thinking, as opposed to making judgments based on length, vocabulary sophistication, or other proxy features (as seen in [17][52]).

In contrast, the EDM, AES, and related communities have long treated quantitative evaluation against held-out data as a minimum standard, precisely because models that appear plausible in isolation often fail when applied to new students, tasks, or contexts. The same risk applies to LLM-as-a-judge systems, given their known sensitivity to prompt wording, rubric phrasing, and latent biases in pretraining data [52]. For this reason, assessment of goodness remains essential even when no training is involved. The typical standard for doing so is to take cases labeled in advance by human judges, through a process demonstrated to be valid (i.e., high inter-rater agreement on cases rated separately), and evaluate the model on those cases using standard metrics such as AUC ROC and F1 (for LLM-as-a-judge, [19]; for EDM, [4]).

1.1.2 Generalizability

A second, closely related form of rigor is assessment of generalizability. Fundamentally, this dimension of rigor is about establishing whether a detector applies to the full span of cases where we want to use that detector for inference. If a detector is intended only for use in analyzing a specific dataset, this can be simple; generalizability only need be established for that dataset. However, if we want to apply that detector in the future to new data or in real-world educational applications, the task becomes more difficult.

The typical minimum standard in the EDM/LAK community for generalizability is student-level cross-validation, or a held-out test set of students, where a model is trained on some students and tested on others [4]. An analogue for use with LLMs that are not fine-tuned would be to develop a prompt and, where appropriate, associated resources for Retrieval-Augmented Generation (RAG) on one set of students and then test goodness on a different set of students. This minimal step helps us to reason about whether a model will work on different students than our original sample. This minimal step is unfortunately, but importantly, all too often absent from LLM-as-a-judge research and development.

This minimal step is by itself insufficient to give confidence that a model will function appropriately for new content or for students different from those in the original sample. To validate this, content must be held out, and performance must be tested on unseen groups of students -- such as students in a different context such as rural versus urban [36] or students with different demographic attributes [26]. A related type of validation is checking for algorithmic bias, where models function substantially better for students with some demographic attributes than others [5]. Given the known biases in LLMs, checking for algorithmic bias is just as important in LLM-as-a-judge approaches as in models developed with machine learning.

1.1.3 Construct Validity

A third essential form of rigor is the assessment of construct validity: establishing whether a detector is measuring what it is intended to measure, rather than some related but distinct construct. Even when a detector demonstrates strong goodness

metrics and generalizes well across students and contexts, it may still be identifying patterns that do not genuinely reflect the theoretical construct of interest. Thus, with insufficient attention to construct validity, researchers risk drawing incorrect inferences about student cognition or learning processes based on precise measurements of the wrong construct. Violations of this principle are at odds with the most basic psychometric principles of social science, and most would agree renders an investigation invalid.

If there are sufficiently numerous, diverse, high-quality, and trusted human labels of the same construct, then assessing goodness and generalizability becomes an assessment of construct validity as well. However, the literature of EDM and related communities is replete with cases where humans and computers are aligned on a definition that has questionable construct validity. This issue is particularly frequently seen in over-simplified rational definitions of highly complex and multi-faceted constructs such as engagement and metacognition (see discussion in [39]). If an LLM is given an over-simplified definition of a construct and asked to code according to it, it will typically not push back -- based on its helpful assistant persona, it will do what it is told (so will many human research assistants).

Communities ranging from EDM to psychometrics have long recognized construct validity as a foundational requirement for any detector or form of measurement, and there are several approaches to establishing construct validity, including asking domain experts to review construct definitions [21], checking for convergence with established measures [48], and careful review of cases where a detector succeeds and fails, to understand what features the model is actually responding to. This type of analysis can reveal whether an LLM or other detector is making judgments based on the intended construct or has captured a related but different construct, or is focusing on confounding factors that happen to be present in the training examples or prompt. Without such rigor, LLM-as-a-judge approaches risk producing measurements that appear valid on the surface but fundamentally misrepresent the constructs they claim to assess.

1.1.4 Auditability

As part of establishing that a detector or LLM-as-a-judge approach is genuinely capturing what is intended, it is helpful to inspect how the model is functioning. However, inspectability is infeasible both for contemporary machine learning algorithms (the complexity of a random forest or neural network is far too high for casual study) and for LLMs. Thus, in the absence of true interpretability, some combination of auditability and explainability is needed.

A detector is auditable [9] if its decisions can be inspected by human reviewers. Without auditability, it becomes difficult to identify systematic errors, diagnose failures, build confidence in the system's reasoning, or ensure accountability when detectors are used in real-world educational contexts. Hence, auditability is not just about the developers' ability to inspect a detector, but also the detector's availability to external reviewers and stakeholders. Auditability encompasses several complementary practices that enable a person to inspect, understand, and verify the decisions made by a detector. At a minimum, it requires comprehensive documentation of the detection process, including the rubrics or criteria used, the prompts provided to the LLM (where applicable), the version of the model or algorithm employed, and any examples or training data supplied (to the maximum extent possible, also taking privacy concerns into consideration). Ideally, it also involves maintaining records of the detector's decisions alongside the original texts being evaluated, enabling users to spot-check results and identify patterns of error. These practices support the broader research community in scrutinizing methods, replicating findings, and building cumulative knowledge about what works and what doesn't. While auditability does not provide the same depth of understanding as true interpretability, it establishes a foundation for trust and verification.

Interpretation is supported when models and approaches are not just auditable but also explainable: where there is some capacity to explain why a detector made a particular decision about a particular case. In classical machine learning, explainable AI (xAI) techniques such as LIME, SHAP, and DiCE have been developed to provide post-hoc explanations of model judgments [45]. However, these techniques have significant limitations. Different xAI methods applied to the exact same model judgment decision often provide contradictory explanations [45], raising questions about which (if any) accurately represents the model's reasoning process. For LLM-as-a-judge approaches, a natural solution is to prompt the LLM to explain its coding decisions alongside the codes themselves [32]. Yet this approach also involves a fundamental limitation: an LLM's explanation represents the most likely textual completion given the prompt and context, rather than a faithful representation of its actual internal reasoning processes regarding the judgment. Thus, the explanation may be fluent and plausible while bearing little relationship to how the model actually arrived at its judgment. As such, while LLM-generated explanations can provide useful hypotheses about decision-making patterns and may help identify obvious errors, they should not be treated as a complete and accurate representation for why the model made a particular judgment. Nonetheless, a model which is auditable and provides explanations—even imperfect ones—enables a level of scrutiny and verification that is impossible with purely black-box approaches. By combining comprehensive documentation, accessible decision records, and model-generated explanations, researchers can facilitate human reviewers and stakeholders in identifying when a detector is making decisions for the wrong reasons, catching systematic biases or failures, and in making informed judgments about when automated codes should be accepted, questioned, or overridden by expert human judgment.

1.2 Why We Need Taylorization Now

At the same time as more rigor is needed, a second challenge is also clear -- and it is a challenge exacerbated by an increase in rigor. Detector-building is time-intensive, effortful, and costly [24]. Doing so with rigor makes it even more time-intensive, effortful, and costly. Each of the four types of rigor listed above is costly -- when applying all four of them, the additional cost and time adds up. If the approaches used in EDM and related communities are too slow to be practical, they will not be used in practice -- less rigorous, lower-quality approaches will be chosen, with consequences for the quality of scholarship and potential negative impacts on learners.

Fortunately, the conditions are present to apply an approach that can cut our time and cost considerably. Specifically, two conditions are present. First, we as a field know a lot about how to create high-quality detectors. There is an extensive body of literature demonstrating which approaches work and do not work for classical machine learning and a rapidly growing body of literature for LLM-based approaches as well. Second, there is a long-standing body of literature on efficiency in engineering and development. We draw from one of the classic and long-standing methods of efficiency in this section: Taylorization [46].

Taylorization, named after Frederick Winslow Taylor's principles of scientific management [46], refers to the process of breaking down complex work into smaller, well-defined, standardized tasks that can be executed more efficiently and, crucially, by individuals with less specialized expertise than would otherwise be required. Taylor's original framework emphasized systematic study of workflows to identify inefficiencies, decomposition of skilled labor into discrete steps, standardization of procedures, and clear specification of how each step should be performed. While Taylorization has been critiqued in manufacturing and labor contexts for its potential to deskill workers and reduce autonomy [12], its core principle—that complex processes can be made more efficient through careful decomposition and standardization—has proven valuable in many domains. In software engineering, for instance, design patterns, coding standards, and automated testing frameworks represent forms of Taylorization that allow developers to work more efficiently without reinventing solutions to common problems [18] [8].

In the context of EDM detectors, this means shifting away from building detectors using an ad-hoc, custom process, treated each time as a unique problem to be solved step by step with the process revised as it goes -- as if we did not already know a great deal about how to rigorously build detectors. Instead, we can adopt a clear and standard process, based on what we have learned as a field. The idea of Taylorization also tells us that if we can sufficiently simplify, streamline, and standardize each step in that process, we do not need a superstar-level EDM researcher/developer to supervise or even participate at each step. Instead, more junior developers can take on specific steps where appropriate, safe in the knowledge that they are following a well-known process which, if followed, will yield valid and rigorous detectors. It is important to note that adopting a Taylorized process does not mean skipping key steps around theoretical reasoning or construct definition. It means using a standardized process to conduct these steps in an efficient, legible, rigorous way -- one where the right expertise is brought to bear at the right stage, where no one's time is wasted by re-invention or re-implementation, and where decisions do not need to be endlessly revisited because key tasks weren't done right the first time.

Thus, for detector development, Taylorization means identifying the key steps in creating rigorous detectors, standardizing those steps into a clear workflow, and providing tools and guidance that allow researchers to execute that workflow efficiently and with less specialized expertise than was previously needed. In doing so, we can reduce the unnecessary inefficiencies that characterize practice in detector building today: the frequent reinvention of very subtly different methods, ad-hoc adoption of slightly different decisions, and reimplementation of the exact same code. Avoiding these inefficiencies can make it easier and less costly to build rigorous, high-quality detectors in the super-majority of cases; a well-designed process will also recognize those rare cases where expert attention or reconsideration is needed. For the super-majority of cases, by adopting Taylorization, we can increase accessibility, improve speed, reduce cost, and maintain rigor and quality.

In section 3, we will demonstrate how this can be done, presenting a Taylorized process for developing LLM-based detectors of cognitive constructs, realized as a concrete, step-by-step workflow. Before diving into the process, we review related work from adjacent fields that have attempted to standardize methods.

2. RELATED WORK ON PROCESS

In thinking about processes for detector development, two bodies of literature provide relevant illustrations: qualitative coding, and past work in the EDM community and in broader machine learning on process.

2.1 Process in educational data mining

Within the educational data mining field, there is a fairly lengthy literature of individual case studies of detector development that describe their methods in a few paragraphs, but relatively few treatments of process as the object of study itself. These descriptions of methods have converged on some broad elements. Perhaps the best review of this is seen in [16], who describe a common five-step process for sensor-free affect detectors, consisting of data collection, feature engineering, model development, performance analysis, and application. However, their treatment of this process consists of describing its manifestations and variations across papers, rather than recommending specific best practices for this process. A portion of a formalized process is also seen in discussion of the EDM Workbench, a tool developed to support developing detectors based on text replays (human coding of pretty-printed log files) [38]. The EDM Workbench supported a process -- enacted by the tool but not specifically argued for or justified -- that consisted of importing log files, distilling features, generating clips (segments of log data which are the grain-size of a construct label), sampling data, and labeling data (by human coders), including inter-rater checking. The Workbench then allowed researchers to export data to machine learning packages, for development of actual detectors. Similarly, the BROMP protocol [6] offered a very formal and repeated process for the limited step of collecting and validating classroom observations for use as training data for detectors.

2.2 Process in data mining/machine learning

Outside of EDM, the Cross-Industry Standard Process for Data Mining (CRISP-DM) has been proposed and discussed as a process for machine learning [13]. CRISP-DM articulates a six-step process for how to conduct machine learning, which fits many projects, but does not attempt to use principles of scientific management to optimize processes through standardization or division of labor (nor was it intended to). Later work has built on CRISP-DM to try to incorporate principles of scientific management. For example, QM-CRISP-DM [41] considers ways that Six Sigma quality management tools could be incorporated into CRISP-DM's process, though still doing so at a fairly high-level where the exact selection and application of tools must be conducted on a project-by-project basis. Zwetsloot et al. [58] proposes a more formal linkage between Six Sigma and CRISP-DM at each of the stages of CRISP-DM, and suggests some principles, such as having data scientists embedded into teams doing data science and making sure that data scientists are familiar with Lean Six Sigma principles. Studer et al. [43] build on CRISP-DM with a formalized approach to identifying and mitigating risks, as well as a list of risks that can occur at each step of CRISP-DM. Microsoft's Team Data Science process (TDSP; [20]) provides an alternate process for the steps of building data science models, with a focus on tools and infrastructure for implementation and deployment, and consideration of the technical roles needed to implement such a project (but not the other types of expertise needed for many data science projects, such as domain expertise). In more specific terms, a process for creating data science driven apps is proposed by [49] and articulates clear steps in such a process, but does not

operationalize how rigor and efficiency are systematically achieved, nor does it formalize the expertise needed.

Overall, while several frameworks have incorporated scientific management principles into data science processes, there remains much to be done to reach the full vision of Taylorization. Taken together, these frameworks articulate what steps occur in data science projects, and occasionally which tools may support them, but they stop short of specifying how those steps can be decomposed, standardized, and distributed across roles in a way that systematically increases efficiency while preserving scientific rigor. They continue to treat model development as a one-off process, relying on continual bespoke decisions by experts, rather than as something that can be rigorously repeated across projects with minimal redundant effort. And indeed, achieving this level of Taylorization may be challenging to accomplish in a generic way, across all of data science -- it may be more achievable within a narrower focus such as developing detectors of cognitive constructs.

2.3 Process in qualitative coding

Qualitative coding, as a research tradition, has taken a different path than machine learning/EDM on the characteristics of good process of identifying specific constructs, in line with the different nature of who identifies constructs (typically a fully human process) and the different research values often seen in qualitative research as compared to machine learning. The process we propose here is targeted to detection applications rather than qualitative research applications, but the qualitative research community's scholarship on process may still be relevant to process for detection. Qualitative methods scholars often describe a series of recurring analytic activities, such as framing guiding questions, conducting initial data preparation and sensemaking, developing and refining codes, and interpreting patterns. These activities are explicitly characterized as iterative, reflexive, and deeply shaped by theoretical and contextual commitments [34] [11] [40].

Importantly, descriptions of qualitative coding processes often reject the idea of a fixed or linear process "pipeline." Braun and Clarke's [11] model of thematic analysis articulates phases such as familiarization, coding, theme development, and refinement, while noting that researchers routinely move back and forth between phases rather than progressing in a straight line. Saldaña [40] conceptualizes coding as occurring across multiple cycles, with early, provisional codes giving way to more focused and theoretically informed categories through sustained comparison and memoing. Maxwell's [35] interactive model of qualitative research design further emphasizes that goals, conceptual frameworks, research questions, data, and analytic strategies continually shape one another over the life of a study. Across process models there is consensus that making analytic steps explicit can help support goals such as transparency, auditability, reflexivity, and coherence [31][47][51].

From this perspective, qualitative research still relies on process thinking, even when it rejects standardized pipelines. What is often absent, however, is systematic attention to efficiency, division of labor, and reuse across projects. Because of the emphasis on flexibility in qualitative research practice, decisions are often idiosyncratic to a specific project or context, with construct definitions are refined ad hoc, or when reliability or validation assessments are conducted inconsistently. Some research communities have worked to formalize decisions more explicitly [1], with consideration to rigor, but without a corresponding focus on efficiency, division of labor, or systematic reuse of analytic work across studies. As such, qualitative coding can be rigorous and trustworthy, but it is also usually labor-intensive and difficult to scale or reproduce. Furthermore, best practices and lessons learned from the idiosyncratic choices of individual projects can often be lost and are relatively rarely transmitted across research groups.

This gap is especially consequential when considering the role that LLMs can play in analyzing data. Qualitative methods process research has offered rich guidance on how constructs should be defined, interpreted, and validated, but share fewer insights into how these practices can be operationalized repeatedly and efficiently across many constructs, datasets, and teams. This is perfectly fine for research intended to be small-scale and highly in-depth, but is a limitation for larger-scale mixed methods research or practice intended to impact many students. Our proposed Taylorized process builds directly on qualitative coding traditions by preserving their core analytic commitments, such as construct clarity, iterative refinement, human judgment, and validation, while introducing systematic decomposition and standardization where possible. In this way, we aim to treat the detection of cognitive constructs not as an artisanal, idiosyncratic activity, but as a rigorously structured analytic process that can be made more efficient without sacrificing epistemic integrity or other core qualitative research values.

3. THE PROCESS

3.1 Key Steps

Our Taylorized process for developing textual detectors that we propose is given in Figure 1. The first step is to select an appropriate dataset for analysis; note that in many cases, this first step is already implicitly solved. For example, when this process is a step in a larger process of building detectors for a specific learning platform or learning activity, the dataset will naturally be drawn from the pilot use of that platform or activity. Then, the second step is to decide on the constructs of interest for detecting within the dataset, guided by theory, prior literature, bottom-up identification of coding categories (such as in grounded theory – [15]), and the goals of the analysis. Next, the third step is to use the data to draft clear operational codebook definitions that specify inclusion criteria, exclusions, and illustrative examples. The fourth step is to conduct human inter-rater checking, in which multiple coders apply the codebook to a shared subset of data and inter-rater agreement is computed. The agreement for each construct is then checked to see if it falls above or below a cut-off selected in advance. Codebook definitions are refined for any constructs with unacceptable agreement (step 5) and then the process of human inter-rater checking is repeated (step 4). The process alternates between step 4 and step 5 until either acceptable reliability is achieved or it is decided that human beings cannot reach acceptable interrater reliability for a construct, and the construct is either abandoned or heavily redesigned (step 2). The sixth step is to build detectors based on LLMs, using the final round of human coding as ground truth for evaluation. This step will typically involve some iteration. Finally, the seventh step is to use the validated detectors at scale, applying them to broader dataset(s) or embedding them into the learning platform to drive intervention or inform educators.

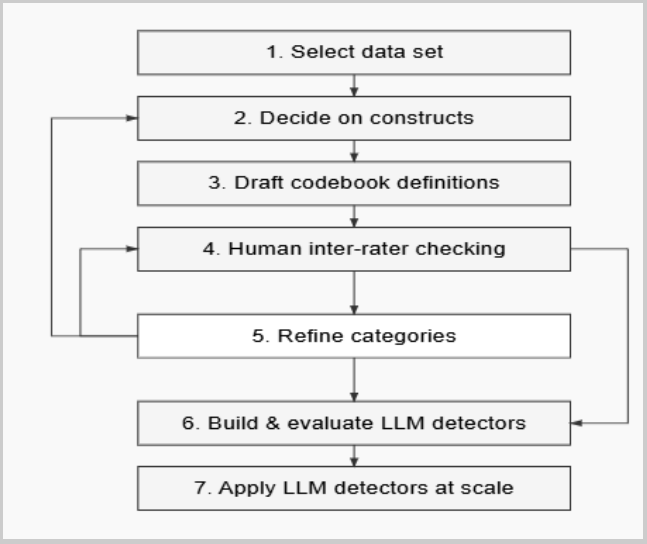

Figure 1.The high-level Taylorizable process for detector development.

3.2 Key Team Personnel and Steps Where They Are Needed

Although this process can be done by smaller teams, and larger teams can bring additional perspectives (particularly for step 2) and help to scale key steps of the process faster (particularly in steps 3 and 4), we envision that the minimum optimal team has the following roles: A) a project lead who oversees the entire process; B) a domain expert who understands the domain of application and the theory and practice relevant to the types of constructs being studied; C) a researcher/developer experienced in using LLMs to develop and refine codebooks; D) at least two qualitative coders, one of whom is experienced; the other can be less experienced or even a novice; E) a developer experienced in using LLMs to develop detectors of specific textual constructs (or, when unavailable, a developer experienced in educational data mining or machine learning); F) where applicable, a member of the design team that developed the learning platform and/or activity which the detectors are being developed for.

Some members of this team are particularly essential in specific steps. The domain expert, LLM codebook developer, and design team members are particularly needed in step 2, developing the coding categories. The domain expert, experienced qualitative coder, and detector developer are particularly needed in step 3, defining the categories. The qualitative coders are needed in steps 4 and 5, human coding and checking; the domain expert and LLM codebook developer can also assist with refinement in step 5. The detector development team is key to steps 6 and 7, building, validating, and scaling detectors.

3.3 Details of Each Step

In this subsection, we unpack the sub-steps and key considerations for each of the process's seven core steps.

Step 1: Select dataset. The project lead begins by identifying the goal for dataset selection, taking into account the type of domain, learning context, learning activity, or general area of categories to be examined. The entire team then engages in a brainstorming session to think of possible datasets that could be obtained or collected. Each of the candidate datasets are then reviewed along multiple dimensions: the domain expert assesses the relevance of the learning context and activity to the target domain and goals; the experienced qualitative coder evaluates amenability to qualitative coding; the detector development lead examines amenability to analysis; and the project lead or their designee investigates any data access or publication issues. Based on this comprehensive review, the project lead selects the dataset that will be used for analysis. A consensus-based process could also be used, depending on the team's values and structure; more hierarchical decision-making increases decision efficiency, in line with the goals of Taylorization.

Step 2: Decide on constructs. The analysis team starts this decision process by reviewing the overall guiding definitions of the project. As this process often takes place in the context of developing detectors for an existing learning platform or activity, the analysis team also reviews design goals and design notes from the design team where available. The domain expert identifies relevant theory or papers (if not already known) and reviews the data in terms of that theory to suggest initial categories, with a particular focus on ensuring the operationalization of the construct is 1) consistent with existing research and 2) indicative of the construct in the specific context in which it is being analyzed. In parallel, a researcher experienced in using LLMs to develop codebooks runs both a bottom-up codebook discovery process (e.g., [7]) and a theoretical codebook discovery process [54]. This developer also reviews constructs from past project work to identify categories that might be included in the current effort. The superset of the codebooks generated through each of these processes becomes the draft of the full category set. Next, a pair of design team members—ideally a teacher and a researcher—provide feedback on the categories in terms of their fidelity to design goals and intended uses (e.g., [7]). Finally, the project lead reviews the category set with the domain expert and selects the final set to work with.

Step 3: Draft codebook definitions. The domain expert and the experienced qualitative coder work together to draft an initial set of codebook definitions, building off of initial definitions suggested by the review of past coding categories and the LLM's coding category discovery, where available. The LLM is then prompted by the development team to suggest refinements to those coding categories that may make the coding process more feasible (e.g., [7]). The domain expert and experienced qualitative coder review these suggested refinements and accept or reject them as appropriate. They then work together to create the first set of draft codebook definitions that will be subject to human coding. During this process, the grain-size at which categories are identified and segmentation approach will necessarily be chosen. Sometimes this process is very quick or even implicit (e.g., code every utterance separately), and sometimes more judgment is needed (segmenting log files almost always involves decisions about trade-offs). In some cases, if a category cannot feasibly be identified at any grain-size of data available, that category may be dropped or redefined (going back to step 2), or the dataset may need to be reconsidered (going back to step 1).

Step 4: Human inter-rater checking. The experienced qualitative coder and a second qualitative coder (perhaps less expert; perhaps a novice) begin by coding a small number of cases together, to make sure they have the same understanding of each category. They then select the number of cases for each round of coding (using, perhaps, a power calculator for inter-rater agreement; e.g., [22]), and proceed to code separately. After completing their independent coding, they check their agreement using metrics such as Cohen's/Fleiss's Kappa, Krippendorff's alpha, or Shaffer's

[42] Rho. If the metrics indicate that agreement is not good enough, the coders discuss the cases where they disagreed, code a small number of cases together again, return to coding separately, and check their inter-rater agreement once more.

Step 5: Refine categories. If acceptable agreement has not been achieved after two rounds of Step 4, the novice qualitative coder feeds the definitions and disagreement cases into an LLM, which proposes definition refinements (e.g., [53]). The experienced qualitative coder and novice qualitative coder review these LLM-generated refinements, make their own additional refinements, and propose a new version of the definitions. The domain expert then reviews this new version and refines it further. The refined definitions are then used for a new round of qualitative coding; Step 4 is repeated up to twice. If Step 4 fails a second time, the affected categories are dropped from the project.

Step 6: Build and evaluate detectors. The development team sets up the collection of prompts and approaches for the LLM, and selects the set of LLMs that will be used. For each category, the development team runs this full set of approaches. The dev team then selects the best approach for each coding category. If any category cannot be successfully modeled, the development team tries custom approaches while immediately advancing to Step 7 for categories where an approach has already worked.

Step 7: Apply detectors at scale. The development team applies the successfully developed categories to all remaining data. The development team also provides Dockerized containers for the successful approaches to the broader development team for inclusion into learning platforms or activities.

3.4 Key Tools for Process

A data repository of papers and datasets reviewed. Such a repository can help prevent duplication of effort when selecting datasets, and can also make it easier to keep track of what theories were used and where some categories originally came from.

A category repository. A repository is needed to store each coding definition and each variant of each definition. This repository should mark the current variant in use for each definition, allowing the team to track the evolution of definitions over time. This repository can be useful from step 2 onwards.

Inter-rater reliability process tools. Several tools can support the inter-rater reliability process. A tool is needed to pull a certain number of cases for coding, with the capability to ensure those cases are never pulled again in subsequent rounds. This can be achieved informally, but risks repeating cases or other inefficiency. Another tool is needed to record each coder's decisions for later comparison (to the other human coder, to LLMs). A third tool can rapidly calculate inter-rater reliability metrics within a round, enabling the team to quickly assess whether agreement is sufficient or whether additional rounds are needed. A fourth tool can support refinement by collating disagreements between coders and feed them to an LLM or a human conducting codebook refinement, along with the relevant definition.

LLM prompting and evaluation tools. The development team can be benefited by creating a set of standardized prompts and approaches that will be consistently used across categories. Developing a tool that runs each of these prompts and approaches and outputs the results to developers in easy to review form will streamline them in comparing many variants of many models. Such a tool can also select and output the best prompt and approach for each category, along with model quality measures, for review. Finally, a tool can be developed to quickly turn the best prompt and approach for each category into a Docker container, facilitating deployment and integration into learning platforms and activities.

3.5 Major Pitfalls at Each Step

Each of these steps can have pitfalls where the process can either fail in ways that jeopardize the successful development of detectors, or where the process becomes slow and inefficient in a way that jeopardizes the overall success of a project depending on the detectors, even if the detectors themselves are eventually developed. Careful attention to these pitfalls is important for project success, and it may be worth reviewing this set of pitfalls both before and after conducting the step. Setting explicit targets in advance for each sub-step of the process (and deadlines for completion). Getting commitment from key project members to be available at the right times can make the difference between a process that runs smoothly and a process that drags on.

Step 1: Select dataset. In this step, it could become apparent that the existing datasets available are insufficient for detecting the types of constructs desired, necessitating expensive and time-consuming data collection. Alternatively, a dataset could be chosen that is too small, too coarse-grained, or too low-quality (including needing extensive pre-processing) to afford the development and validation of a model. Thirdly, a dataset could be chosen that is itself high-quality, but is not representative of the eventual contexts that the team wants to deploy detection in. A fourth risk is that an appropriate (or better) dataset is out there, but is not found by the team due to an insufficiently comprehensive search; contrastingly, it is also possible to spend too long searching for data and delay settling on a perfectly acceptable dataset.

Step 2: Decide on constructs. In this step, the core risks are from the breakdown of each of the processes being used to identify candidate categories. For instance, the domain expert could fail to identify the most relevant theory or papers, or could spend too long searching for prior literature and mining it for categories. Selection of an inappropriate or overly abstract theory can hinder both human and LLM processes for using theory to derive codes (although reference to past applications of a theory can mitigate this issue -- cf. [54]). Insufficient attention to prompt design can reduce the quality of any LLM-based discovery process. Insufficient time spent on filtering categories, or choosing categories that are rare (and not sufficiently important to justify including despite their rarity) can lead to delays further down the pipeline. At the same time, the number of individuals involved in this step of the process provides several opportunities for delay, requiring high-level attention to how the process is progressing.

Step 3: Draft codebook definitions. This step is dependent, as with earlier steps, on the thoroughness and thoughtfulness of human revision, the appropriateness of LLM prompts, and the efficient collaboration of the domain expert and experienced qualitative coder to select the first set of draft definitions that will be subject to human coding.

Steps 4 and 5: Human inter-rater checking and refining categories. An unrealistically high criterion value for inter-rater agreement can also leave coders cycling between steps 4 and 5 for an inefficient amount of time. Careful attention to whether agreement looks infeasible at the very beginning, and time spent in this part of the process, can prevent delays in a stage which is probably the most likely step in the entire process to become bogged down.

Similarly, these phases can be delayed if coding is too difficult due to ambiguities or other issues with the categories, an issue that should be watched out for (perhaps triggering revision of definitions earlier than expected). Conversely, there is the risk that steps taken to improve the feasibility of agreement during refinement ends up filtering out key aspects of the construct, a risk particularly present if a category is narrowed to an oversimplified set of rules that can be applied without judgment. There is also a risk that refinements may solve the current disagreements while breaking the parts of the definition that were working effectively, creating new disagreements within the next round.

Additionally, if the coding sample is poorly chosen, the coders may end up coding data subsets where codes of interest are insufficiently common, or data subsets that are not representative of the full data, leading to poor generalizability later on in step 7. Careful attention to sampling, such as stratification on key variables, can reduce these risks. Including insufficient context for coding, or too much context, can also make coding challenging.

Finally, the availability of experts needed for step 5 should be secured in advance, so that there can be rapid turnover of refined categories back to the coders, if a repetition of step 4 is needed.

Step 6: Build and evaluate detectors. The core risk at this step is that the LLMs are unable to effectively detect the constructs of interest. This can be mitigated through trying a sufficiently large number of LLMs and prompting approaches; building a library of approaches in advance (from published literature and past experience) can reduce the time spent tinkering at this stage. Using the type of tools mentioned in the previous section rather than running each prompt one by one -- assembly-lining the process of trying different approaches -- will also make a big difference to the efficiency of this stage. Too much tinkering can also lead to over-estimation of model goodness; this can be prevented by holding out some human labels until the very end of the process.

Another risk is found with models that may be deprecated or replaced by their developers. Running open models locally can prevent a best model from suddenly becoming ineffective or unavailable. This practice also reduces privacy risks and concerns, but can introduce scaling issues that may not be present with large-scale corporate LLMs.

Step 7: Apply detectors at scale. This stage should be relatively free of risk, if the code to run the right prompt on the right LLM has been effectively distilled during stage 6. The process of creating a Dockerized container can be time-consuming the first time, but if set up correctly, should be rapid to scale across detectors afterwards. However, it is important to re-check model performance once generalization has commenced (and perhaps even later on -- e.g., [30]), to make sure that the model is functioning as intended in the context of deployment.

4. CRITERIA FOR EVALUATING A PROCESS

Up to this point, we have outlined a Taylorizable process for developing LLM-based textual detectors of inherently cognitive constructs. The question is how we can assess whether this process successfully resolves the challenges that made such a process necessary. We therefore propose six criteria that processes should be weighed against in order to evaluate their effectiveness for detector development in educational data mining and related fields.

Does it get things done efficiently? Taylorization aims to increase efficiency by standardizing repetitive processes and tasks and enabling parallel work streams. An efficient process should reduce the time from project initiation to completion (whether it be deployment or publication, etc.), particularly when building multiple detectors or scaling across domains. If a process is rigorous but very slow, or if it requires highly-specialized expertise at every step, it may be insufficiently efficient to be scalable, the concern that motivates Taylorization in the first place. This efficiency should not come at the cost of quality though (as we discuss below). It should instead remove unnecessary variation and decision-making, and avoid time-consuming mistakes.

Does it reduce early planning? One key advantage of our proposed process is that it distributes the workload and decision making across a standardized workflow, making the affordances and constraints of building a new detector a bit more straightforward. This should therefore reduce the need for extensive upfront planning and reduce the amount of subsequent re-planning. Planning is not eliminated within our process; instead, it is scaffolded by the process. This should reduce the number of person hours dedicated involved in custom planning for each new detector, as well as the overall burden of setting up and managing the overarching process from scratch.

Does it avoid unexpected derailments and number of unnecessary iterations? As most of us who have engaged in the detector building process know, issues can (and often do) arise at various stages. Such unanticipated obstacles can force researchers to backtrack or even restart parts of the detector building process, thus losing valuable time and resources. We propose that risk of discovering flaws at late stages (e.g., poorly operationalized constructs or failures of basic cross validation techniques) can be reduced with a thoughtful sequential processes, which helps mitigate such inevitable derailments by anticipating common failures as part of the process and having validation checkpoints that catch issues before they cascade into later stages. Back-tracking is sometimes necessary (and our process explicitly incorporates it) -- the key is to limit how far back the research team needs to go, and how much work needs to be redone. As an example, our process recommends establishing human-human inter-rater reliability before any LLM prompt development, which ensures that construct definitions are clear, reproducible and codeable before substantial effort is invested in automation. Skipping any part of this step may produce results that are not rubric aligned or have questionable construct validity, necessitating repeating LLM coding work after fixing issues that could have been resolved beforehand.

Does it lead to high rigor? Rigorous detector development should include a few key dimensions, listed above: reproducibility (i.e., can others follow the same steps and achieve similar results?), auditability (i.e., is there a clear record of decisions and validations?), construct validity (i.e., does the detector measure what it claims to measure?), and generalizability (i.e., does performance hold across datasets or contexts?). Auditability provides the foundation for other forms of rigor: a rigorous process produces artifacts at each stage, such as codebooks, reliability statistics, evaluation results, documentation, that provide evidence for forms of rigor mentioned above. This should not be an afterthought, but should be embedded in the workflow itself with high standards at each step. If the Taylorization process is followed, then somewhat ironically, rigorous research will be

avoided without a easily comparable control condition that lacks rigor.

Does it avoid introducing ethical concerns? Any attempts at detecting cognitive constructs must include careful attention to ethics like privacy, algorithmic bias, and the potential of misuse. A Taylorizable process should therefore include these considerations as part of the process. This includes taking steps such as validating across diverse populations to check for bias [5], as well as clear documentation that supports informed decision-making about appropriate deployment contexts and model uses based on the rates of incorrectness (i.e., false negatives and false positives).

Is it useable across domains? Perhaps the ultimate test of a Taylorizable process is whether it is a reusable methodology across different cognitive constructs or across different contexts for the same cognitive construct. Although some adaptation will almost certainly be necessary, core workflows should remain applicable whether one is detecting (for instance) critical thinking in student essays or confusion in peer tutoring dialogues. If the core processes remain stable, we would count this to be a reliable methodological contribution that helps advance not only rigorous detectors but also the general approach of the field.

5. USING THE PROCESS: OUR RESEARCH AGENDA

A multi-detector project with co-design. The idea for this paper was partially inspired by a research agenda we are currently designing and implementing. We are in the process of a broader project, in which we are co-designing educational tools to support learning experiences with teachers, researchers, and engineers at the center of co-design teams. These tools will eventually depend on detectors to support adaptivity and reporting to teachers. The teams are designing learning experiences from scratch, meaning they have a unique opportunity to build tools around what's possible, given theoretically-driven hypotheses and knowledge about what is possible with detectors. This is a little different from the more commonly-seen case where detector development is conducted in collaboration with existing technologies -- in such a case, researchers typically wait until data is available from an already deployed system, then build and validate detectors for inclusion and sometimes retro-fitting. We are hoping to instead enable an approach where teams can build tools that are informed by what is possible given the state of detector development at that time. Teams need to understand what constructs can be detected reliably, what constructs will need more time for development, and what constructs may not be feasible given current methods. This also allows for designing learning around "good enough" detectors, rather than spending a ton of time developing the "best" detector that has ultimately missed the window for inclusion in the design itself. The challenge, of course, is speed. Co-design processes move quickly, with teams iterating on ideas, testing prototypes with teachers and students, and refining based on feedback in rapid cycles. Traditional detector development, in contrast, moves more slowly, and has -- without Taylorization -- historically required considerable time and effort. The mismatch in timelines creates a fundamental tension: either slow down co-design to wait for detector development (sacrificing the iterative responsiveness that makes co-design valuable) or proceed with co-design without detector input (risking design decisions that cannot be adequately supported).

Our Taylorizable process is designed to address this challenge in two key ways. The first is the standardization of workflows and clear guidance at each step. This process saves time that would otherwise be spent on process decisions, debates about methodology, or trial-and-error approaches to common challenges. Teams can focus their effort on the substantive work of defining constructs and building detectors, rather than continually reinventing processes. The efficiency gains may be realized even more when developing multiple detectors in parallel, as lessons learned and infrastructure developed for one detector can be directly applied to other detectors. The second way is that the process is designed to be adaptable across different forms of data. Imagine collecting different types of student data that are all text-based, like essays, short-answer responses, discussion posts, transcribed dialogues, or other forms of writing. The core workflow remains stable across these production types, and the same technology can be used for all these types of data. This consistency means that teams can develop expertise in the process itself, rather than needing to develop entirely new approaches for each detector, and can build shared infrastructure (such as coding interfaces, prompt evaluation tools, and deployment pipelines) that accelerates work across multiple detectors. To be clear, we do not think this process only applies when co-designing tools that are amenable to multi-detector solutions. Instead, we think this is an example of how such a process would enable others to adopt similar and efficient processes when rigor, speed, and design are all priorities.

Documentation for process improvement and refinement. Our research agenda treats the proposed Taylorizable process's benefit as a hypothesis to be tested and refined through actual use. We recognize that the only way to validate these claims is through systematic documentation of how the process performs in practice. This requires tracking not only the outcomes of various stages of detector development (such as inter-rater reliability statistics, detector performance metrics, and deployment results) but also tracking the process itself, such as timelines, decision points, iterations, and failures.

Meticulous documentation serves a forward-looking function by enabling systematic refinement of the process over time. The current formulation represents our best thinking based on prior literature and experience, but it will inevitably have limitations and gaps that only become apparent through actual implementation. Some steps may take longer than anticipated, some transitions between steps may be unclear or awkward. It's also possible that some risks or failures may emerge that we did not anticipate. By documenting our experience systematically, we create the empirical foundation for revising and improving the process for future detector development projects, both within our own team and for others who may adopt or adapt this approach. To this end, it will be important for us (and others) using these types of processes to carefully document failures and challenges -- even small-scale ones that were resolved but decreased efficiency -- rather than only presenting success stories alone.

An alternate path: Decreasing total calendar time using existing datasets for early development. The process outlined in this paper attempts to reduce both total time and effort. However, in some cases cost may be less of a concern than calendar time spent, or vice-versa. In this section, we discuss one of the ways that the Taylorizable process outlined in this paper could be adjusted, to optimize for faster completion at the cost of spending more resources. This earlier-completion approach involves using existing datasets for early development. We are currently testing this approach in parallel with the previously discussed approach, in the context of our current research agenda.

As mentioned previously, in that research agenda, the goal is for multiple co-design teams to develop new tools from scratch. This means that data from those tools will not exist until the tools are sufficiently developed to pilot with actual users, which typically occurs well into the design process. Waiting for that data to become available would delay detector development and major design decisions may be made and difficult to revise by then. Using other datasets can help kickstart the process while also testing out feasibility for various design features. As an example, an existing corpus might contain middle school science argumentative writing, while the eventual tool might target high school science argumentation. The dataset is not a perfect match, but may be similar enough along key dimensions to support initial construct definition, human coding, and even detector development work.

This process adjustment has several benefits. Overall, it puts us early in stage 6 of the process before even having conducted first data collection from the final tool; a considerable head-start in the process, even if work needs to be redone on earlier data stages once the final tool's datasets come in. Adjacent dataset exploration allows for the early testing of construct definitions, particularly in terms of whether the definitions are sufficiently clear and operational for reliable human coding. If multiple trained coders cannot achieve acceptable inter-rater reliability when coding an existing dataset, this may signal that the construct definition needs refinement before proceeding further. This is a problem that is far better to discover early, compared to discovering it months later when tool-specific data finally becomes available.

Adjusting the process in this fashion also enables early assessment of LLM capabilities for detecting the constructs of interest, allowing teams to determine which models perform best, which prompting strategies are most effective, and which constructs may be particularly challenging to detect reliably. An added benefit is the exploration of generalizability by testing whether detectors developed on one dataset (e.g., middle school writing) can transfer to somewhat different contexts (e.g., high school writing). Of course, using a "similar enough" dataset requires judgment about how much mismatch between existing datasets and target contexts is acceptable in the first place. This depends on both the nature of the construct and the intended application. Some constructs are highly domain-specific, which may result in cross-domain datasets that may produce conclusions that are not applicable to the current project. Constructs may not always manifest in the same way, and some constructs may be differentially present or absent. Similarly, some applications may be more sensitive to context mismatch than others. Adjusting the process in the fashion discussed here inherently involves repeated work, increasing cost but possibly producing models months earlier.

Overall, adjustments of this nature make it possible to optimize differently. The Taylorizable process offered in section 3 of this paper can be applied to many situations where detectors need to be efficiently developed, but will not fit all situations. However, when adapting or deviating from such a process, it is important to consider what goals are more optimized for, what the costs are to other goals, and how the process can be adjusted in advance to plan for and mitigate possible risks. Ad hoc, unplanned shifts risk unintended consequences.

6. CONCLUSIONS

The emergence of LLMs has fundamentally changed the landscape of how educational data mining and related fields engage in detector building. This includes the use of LLMs-as-judge to label data as well as LLMs for making predictions about said data. Much of this is positive; new researchers with different disciplinary backgrounds can join the field and LLMs can make some parts of the process more efficient. At the same time, without careful attention to important methodological pillars, we run the risk of degrading the quality and ultimately the reputation of the field. In this paper, we propose a solution to this by decomposing the stages of detector development of cognitive constructs using textual data into a standardized, end-to-end Taylorizable process, with the goal of making rigorous practices more efficient (and vice versa; also making efficient practices more rigorous).

We emphasize that our proposal is not intended to simply provide a step-by-step roadmap for what steps should be taken, but also represents a perspective and commentary on how educational data mining might evolve as a field in the advent of LLMs, while continuing to center basic psychometric principles like validity and reliability. EDM has historically been built on some implicitly standardized approaches in terms of what is reported in each paper, but each research project is often built from scratch with limited reuse of methods across teams and contexts. This approach has certainly enabled creativity and flexibility, but we are also at a critical juncture where we need to ensure consistent quality standards across the field. As the field continues to mature, more systematic engineering practices will likely pay off by enabling more scalability and infrastructure that enables others to build upon prior work.

Taylorization, as presented here, represents a step toward a more "engineered" field. The hope is that this relieves some of the burden in terms of resources planning and rigor, without stifling innovation or creativity. We hope that this method also leaves room to place more emphasis on cognitive constructs that have not traditionally been a focus in EDM (e.g., complex forms of comprehension and critical thinking), particularly ones that require nuanced and complex operationalizations. Similarly, we hope this method will encourage sharable standardized artifacts like codebooks, datasets, and documented prompts, which could help create a more robust shared infrastructure that enables generalizable knowledge and (hopefully) impact.

None of this is to suggest that this Taylorization will be without downsides and limitations, a golden ticket to advancement. Anytime something is automated and standardized (even if customizable), there are corresponding trade-offs. Some research/product goals may require bespoke approaches. To this end, we do not suggest that every project attempting to detect cognitive constructs using textual data -- or even every process for every single construct, within-project -- must follow this approach. That said, well-designed and repeated processes with clear and rigorous checkpoints still present a meaningful starting point to consider, so that a decision to deviate from standard procedure is carefully thought-out and justified. We also want to emphasize, as scholars conducting work both in EDM and in mixed-method projects involving qualitative research, that this paper's process is intended for detector development rather than for qualitative research approaches involving qualitative coding (such as thematic analysis or grounded theory).

Ultimately, the value of our Taylorization process should be based on the practical changes it can support (see section 5). We share this framework now, at a time when we believe it can help standardize rigorous processes with LLMs, and we will document our own efforts to apply it in the future. If nothing else, we hope

to invite others into a dialogue about how EDM can collectively build a more rigorous, efficient, impactful field.

7. ACKNOWLEDGMENTS

We thank AERDF for their support of this research, and Chelsea Porter for assistance with document preparation. LLMs were used to support the creation of this manuscript, but all content has been verified by human co-authors.

8. REFERENCES

[1] Arastoopour Irgens, G., & Eagan, B. (2022, October). The foundations and fundamentals of quantitative ethnography. In *International Conference on Quantitative Ethnography*, 3-16. Cham: Springer Nature Switzerland.

[2] Attali, Y. (2008). *Automated scoring of short-answer open-ended GRE Subject Test items*. Research Report. Educational Testing Service

[3] Attali, Y., & Sinharay, S. (2015). *Automated Trait Scores for" GRE"® Writing Tasks*. Research Report. ETS RR-15-15. ETS Research Report Series.

[4] Baker, R.S. (2025) *Big Data and Education*. 9th Edition. Philadelphia, PA: University of Pennsylvania.

[5] Baker, R. S., Hawn, M. A. (2022) Algorithmic Bias in Education. *International Journal of Artificial Intelligence and Education*, 32, 1052-1092.

[6] Baker, R. S., Ocumpaugh, J. L., Andres, J. M. A. L. (2020) BROMP Quantitative Field Observations: A Review. In R. Feldman (Ed.) *Learning Science: Theory, Research, and Practice*, 127-156. New York, NY: McGraw-Hill.

[7] Barany, A., Nasiar, N., Porter, C., Zambrano, A. F., Andres, J. M. A. L., Bright, D., Choi, J., Gao, S., Giordano, C., Liu, X., Mehta, S., Shah, M., Zhang, J., Baker, R. S. (2024). ChatGPT for Education Research: Exploring the Potential of Large Language Models for Qualitative Codebook Development. In *Proceedings of the 25th International Conference on Artificial Intelligence in Education*.

[8] Beck, K. (2003). *Test-Driven Development: By Example*. Addison-Wesley.

[9] Biran, O., & Cotton, C. (2017, August). Explanation and justification in machine learning: A survey. In *IJCAI-17 workshop on explainable AI (XAI), 8*(1), 8-13.

[10] Borchers, C., Yang, K., Lin, J., Rummel, N., Koedinger, K. R., & Aleven, V. (2024). Combining dialog acts and skill modeling: What chat interactions enhance learning rates during AI-supported peer tutoring? In *Proceedings of the 17th International Conference on Educational Data Mining*, 117–130. International Educational Data Mining Society. https://doi.org/10.5281/zenodo.12729784

[11] Braun, V. & Clarke, V. (2006) Using thematic analysis in psychology. *Qualitative Research in Psychology, 3*, 77-101.

[12] Braverman, H. (1974). *Labor and Monopoly Capital: The Degradation of Work in the Twentieth Century*. Monthly Review Press.

[13] Chapman, P., Clinton, J., Kerber, R., Khabaza, T., Reinartz, T., Shearer, C., & Wirth, R. (1999, March). The CRISP-DM user guide. In *4th CRISP-DM SIG Workshop in Brussels in March (Vol. 1999)*. sn.

[14] Chen, G., Chen, S., Liu, Z., Jiang, F., & Wang, B. (2024, November). Humans or LLMs as the Judge? A Study on Judgement Bias. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 8301-8327.w

[15] Corbin, J. M., & Strauss, A. (1990). Grounded theory research: Procedures, canons, and evaluative criteria. *Qualitative Sociology, 13*(1), 3-21.

[16] de Morais, F., Goldoni, D., Kautzmann, T., da Silva, R., & Jaques, P. A. (2023). Automatic Sensor-free Affect Detection: A Systematic Literature Review. *arXiv preprint* arXiv:2310.13711.

[17] Dubois, Y., Galambosi, B., Liang, P., & Hashimoto, T. B. (2024). Length-controlled alpacaeval: A simple way to debias automatic evaluators. *arXiv preprint* arXiv:2404.04475.

[18] Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1994). *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley.

[19] Gu, J., Jiang, X., Shi, Z., Tan, H., Zhai, X., Xu, C., Li, W., Shen, Y., Ma, S., Liu, H., Wang, S., Zhang, K., Lin, Z., Zhang, B., Ni, L., Gao, W., Wang, Y., & Guo, J. (2024). A survey on LLM-as-a-judge. *The Innovation, 0*(0), 101253.

[20] Guhathakurta, D. (2017) How To Implement a Standardized Process for Execution and Delivery of Data Science Solutions: Microsoft's Team Data Science Process (TDSP). Presentation at the *Global Big Data Conference*, Santa Clara, CA USA.

[21] Haynes, S. N., Richard, D. C. S., & Kubany, E. S. (1995). Content validity in psychological assessment: A functional approach to concepts and methods. *Psychological Assessment, 7*(3), 238–247.

[22] He, M., Baker, R.S., Hutt, S., Zhang, J. (2022) A Less Conservative Method for Reliability Estimation for Cohen's Kappa. *Proceedings of the International Conference on Quantitative Ethnography*.

[23] Herodotou, C., Rienties, B., Boroowa, A., Zdrahal, Z., & Hlosta, M. (2019). A large-scale implementation of predictive learning analytics in higher education: The teachers' role and perspective. *Educational technology research and development, 67*(5), 1273-1306.

[24] Hollands, F., & Bakir, I. (2015). *Efficiency of automated detectors of learner engagement and affect compared with traditional observation methods*. New York, NY: Center for Benefit-Cost Studies of Education, Teachers College, Columbia University.

[25] Huang, H., Bu, X., Zhou, H., Qu, Y., Liu, J., Yang, M., Xu, B., & Zhao, T. (2025, July). An empirical study of LLM-as-a-judge for LLM evaluation: Fine-tuned judge model is not a general substitute for GPT-4. In *Findings of the Association for Computational Linguistics: ACL 2025*, 5880-5895.

[26] Hutt, S., Wong, A., Papoutsaki, A., Baker, R. S., Gold, J. I., & Mills, C. (2024). Webcam-based eye tracking to detect mind wandering and comprehension errors. *Behavior Research Methods, 56*(1), 1-17.

[27] Khosravi, H., Shabaninejad, S., Bakharia, A., Sadiq, S., Indulska, M., & Gasevic, D. (2021). Intelligent Learning Analytics Dashboards: Automated Drill-Down Recommendations to Support Teacher Data Exploration. *Journal of Learning Analytics, 8*(3), 133-154.

[28] Krumdick, M., Lovering, C., Reddy, V., Ebner, S., & Tanner, C. (2025). No free labels: Limitations of LLM-as-a-judge without human grounding. *arXiv preprint* arXiv:2503.05061.

[29] Kumar, N. A., Pham, C. M., Iyyer, M., & Lan, A. (2025). Whose story is it? Personalizing story generation by inferring author styles. *arXiv preprint* arXiv:2502.13028.

[30] Levin, N., Baker, R.S., Nasiar, N., Fancsali, S., Hutt, S. (2022) Evaluating Gaming Detector Model Robustness Over Time. *Proceedings of the 15th International Conference on Educational Data Mining*.

[31] Lincoln, Y. & Guba, E. (1985). *Naturalistic inquiry*. Beverly Hills, Calif. : Sage Publications.

[32] Liu, Y., Iter, D., Xu, Y., Wang, S., Xu, R., & Zhu, C. (2023). G-Eval: NLG Evaluation using Gpt-4 with Better Human Alignment. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics*.

[33] Marioriyad, A., Rohban, M. H., & Baghshah, M. S. (2025). The Silent Judge: Unacknowledged Shortcut Bias in LLM-as-a-Judge. In *NeurIPS 2025 Workshop: Reliable ML from Unreliable Data*.

[34] Maxwell, J. A. (2013). *Qualitative research design: An interactive approach*. Sage.

[35] Moorcock, Michael. (1970). *The Eternal Champion*. London, UK: Grafton.

[36] Ocumpaugh, J., Baker, R., Gowda, S., Heffernan, N., Heffernan, C. (2014). Population validity for Educational Data Mining models: A case study in affect detection. *British Journal of Educational Technology, 45*(3), 487-501.

[37] Peczuh, M. C., Kumar, N. A., Baker, R., Lehman, B., Eisenberg, D., Mills, C., Chebrolu, K., Nashi, S., Young, C., Liu, B., Lachman, S.. & Lan, A. (2025). Toward LLM-Supported Automated Assessment of Critical Thinking Subskills. *arXiv preprint* arXiv:2510.12915.

[38] Rodrigo, M. M. T., Baker, R. S. J. d., McLaren, B., Jayme, A., Dy, T. (2012) Development of a Workbench to Address the Educational Data Mining Bottleneck. *Proceedings of the 5th International Conference on Educational Data Mining*, 152-155.

[39] Roll, I., & Winne, P. H. (2015). Understanding, evaluating, and supporting self-regulated learning using learning analytics. *Journal of Learning Analytics, 2*(1), 7–12.

[40] Saldaña, J. (2011). *Fundamentals of qualitative research*. Oxford university press.

[41] Schäfer, F., Zeiselmair, C., Becker, J., & Otten, H. (2018, November). Synthesizing CRISP-DM and quality management: A data mining approach for production processes. In *2018 IEEE International Conference on Technology Management, Operations and Decisions (ICTMOD)*, 190-195. IEEE.

[42] Shaffer, D. W. (2017). *Quantitative ethnography*. Cathcart Press.

[43] Studer, S., Bui, T. B., Drescher, C., Hanuschkin, A., Winkler, L., Peters, S., & Müller, K. R. (2021). Towards CRISP-ML (Q): a machine learning process model with quality assurance methodology. *Machine Learning and Knowledge Extraction, 3*(2), 392-413.

[44] Švábenský, V., Baker, R. S., Zambrano, A., Zou, Y., & Slater, S. (2023). Towards generalizable detection of urgency of discussion forum posts. In *Proceedings of the 16th International Conference on Educational Data Mining*, 302–309. International Educational Data Mining Society. https://doi.org/10.5281/zenodo.8115790

[45] Swamy, V., Radmehr, B., Krco, N., Marras, M., & Käser, T. (2022). Evaluating the Explainers: Black-Box Explainable Machine Learning for Student Success Prediction in MOOCs. *Proceedings of the 15th International Conference on Educational Data Mining*, 98.

[46] Taylor, F. W. (1911). *The principles of scientific management*. NuVision Publications, LLC.

[47] Tracy, S. J. (2010). Qualitative quality: Eight "big-tent" criteria for excellent qualitative research. *Qualitative inquiry, 16*(10), 837-851.

[48] Ventura, M., & Shute, V. (2013). The validity of a game-based assessment of persistence. *Computers in Human Behavior, 29*(6), 2568-2572.

[49] Weigand, A. C., & Kindsmüller, M. C. (2021, July). HCD3A: An HCD model to design data-driven Apps. In *International Conference on Human-Computer Interaction*, 285-297. Cham: Springer International Publishing.

[50] Yang, T., Shi, T., Wan, F., Quan, X., Wang, Q., Wu, B., & Wu, J. (2023, December). PsyCoT: Psychological Questionnaire as Powerful Chain-of-Thought for Personality Detection. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, 3305-3320.

[51] Yardley, L. (2000). Dilemmas in qualitative health research. *Psychology and Health, 15*(2), 215-228.

[52] Ye, J., Wang, Y., Huang, Y., Chen, D., Zhang, Q., Moniz, N., Gao, T., Geyer, W., Huang, C., Chen, P.-Y., Chawla, N., & Zhang, X. (2025, April). Justice or Prejudice? Quantifying Biases in LLM-as-a-judge. In *International Conference on Learning Representations*.

[53] Zambrano, A.F., Liu, X., Barany, A., Baker, R.S., Kim, J., Nasiar, N. (2023). From nCoder to ChatGPT: From Automated Coding to Refining Human Coding. *Proceedings of the International Conference on Quantitative Ethnography*.

[54] Zambrano, A.F., Wei, Z., Zhang, J., Baker. R.S., Ocumpaugh, J., Barany, A., Liu, X., Zhou, Y., Paquette, L., Ginger, J., Borchers, C. (2026). Data Plus Theory Equals Codebook: Leveraging LLMs for Human-AI Codebook Development. *Journal of Educational Data Mining, 18*(1), 25-65.

[55] Zeng, Z., Chaturvedi, S., & Bhat, S. (2017). Learner affect through the looking glass: Characterization and detection of confusion in online courses. In *Proceedings of the 10th International Conference on Educational Data Mining (EDM 2017)*. International Educational Data Mining Society.

[56] Zheng, L., Chiang, W. L., Sheng, Y., Zhuang, S., Wu, Z., Zhuang, Y., Lin, Z., Li, Z., Li, D., Xing, E., Zhang, H., Gonzalez, J. E., & Stoica, I. (2023). Judging LLM-as-a-judge with MT-Bench and Chatbot Arena. *Advances in Neural Information Processing Systems, 36,* 46595-46623.

[57] Zhu, Z., Novikova, J., & Rudzicz, F. (2019, June). Detecting cognitive impairments by agreeing on interpretations of linguistic features. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for*

Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), 1431-1441.

[58] Zwetsloot, I. M., Kuiper, A., Akkerhuis, T. S., & de Koning, H. (2018). Lean Six Sigma meets data science: Integrating two approaches based on three case studies. *Quality Engineering, 30*(3), 419-431.