What is a protest anyway? Codebook conceptualization is still a first-order concern in LLM-era classification

Andrew Halterman
Michigan State University
halterm3@msu.edu

Katherine A. Keith
Williams College
kak5@williams.edu

Abstract

Generative large language models (LLMs) are now used extensively for text classification in computational social science (CSS). In this work, we focus on the steps before and after LLM prompting: *conceptualization* of the categories to classify and using LLM predictions in downstream *statistical inference*. We argue these steps have been overlooked in much of LLM-era CSS and LLMs can tempt analysts to skip conceptualization altogether. For example, a political scientist classifying "protest" with LLMs may never be forced to craft a definition: unlike human annotators who would ask clarifying questions, an LLM can silently accept an underspecified concept to classify and return plausible-looking labels. Using simulations, we show that conceptualization failures induce downstream inferential bias that cannot be corrected solely by a more accurate LLM or post-hoc bias correction methods. We conclude by reminding CSS analysts that conceptualization is still a first-order concern in the LLM-era and provide concrete advice for pursuing low-cost, unbiased, low-variance downstream estimates.

1 Introduction

Text classification—labeling documents with predefined (often complex) concepts—is a fundamental task in computational social science (CSS). "Traditional" text classification using supervised models is well understood and widely used as a social science methodology (D'Orazio et al., 2014; Wilkerson and Casas, 2017; Grimmer et al., 2022; Barberá et al., 2021). However, approaches to text classification have fundamentally shifted in the era of generative large language models (LLMs).

Analysts can now use LLMs for "zero shot" classification: prompting an LLM with the task description, the target document, and instructions to generate a class label. This approach to classification has led to a proliferation of optimistic claims that one can replace human annotators with LLMs (Gilardi et al., 2023; Ziems et al., 2024; Törnberg, 2024, *inter alia*).

LLMs clearly reduce analyst effort and annotation costs, potentially broadening access to text-based CSS methods. However, there are tradeoffs to this "faster and cheaper" approach. First, previous work has found the performance of zero-shot LLMs is limited for more "complex" CSS tasks (Thalken et al., 2023; Bamman et al., 2024; Halterman and Keith, 2025). Second, other work has emphasized the potential for statistical bias if one uses *only* LLM-generated labels in downstream statistical inference (Egami et al., 2023; Gligoric et al., 2025; Baumann et al., 2025).

Complementing this prior work, we highlight the important but often overlooked step prior to LLM prompting: creating *codebooks* that define the semantic classes of interest and describe annotation procedures. To set up the claims we will subsequently make, we limit our discussion to text classification tasks that satisfy two assumptions.

Assumption 1. Existence of a gold standard label for each document. Given a *codebook*—a fully specified definition and set of instructions C—there exists a true, gold standard label Y_i for each document X_i. This assumption excludes purely subjective or interpretive classification tasks, and rules out truly ambiguous documents.[1] It does *not*, however, rule out the possibility of different gold standard labels existing for the same document under different codebooks: there can exist a C' such that $Y_i|X_i, C \neq Y_i|X_i, C'$.

Assumption 2. Existence of an expert annotator with oracle scoring. Following previous work on bias correction (Angelopoulos et al., 2023; Egami et al., 2023), we assume there exists an "expert" annotator who can annotate a document with its

[1]Note, we acknowledge there are many important subjective classification tasks in NLP. Our excluding them is not a normative judgment of their importance, but rather a distinction needed for our technical claims.

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 2043–2059

gold standard label: $Y_i|X_i, C = \hat{Y}_i^{\text{expert}}|X_i, C = f_{\text{expert}}(X_i, C)$. Note, though, by Assumption 1, that there can exist a C' such that $f_{\text{expert}}(X_i, C) \neq f_{\text{expert}}(X_i, C')$.

Limiting ourselves to text classification tasks that follow these two assumptions, we spend the remainder of this paper expanding on and providing evidence for the following three main claims:

Claim 1. Annotation errors can be decomposed into conceptualization and scoring errors. Annotation errors, including errors in LLM-generated labels, result either from *conceptualization* errors—resulting from incomplete[2] class definitions in a codebook—or from *scoring* errors—annotator mistakes from incorrectly applying class definitions.[3] This decomposition draws on the social science measurement literature (Adcock and Collier, 2001; Fariss et al., 2020) and contemporaneous work in computer science (Wallach et al., 2025); see §A for background on these definitions and related work.

Claim 2. LLMs can tempt analysts into skipping conceptualization. In the pre-LLM era, text classification projects required human annotations. Although collecting human annotations is costly, they serve a useful forcing mechanism: human annotators require detailed coding instructions, forcing analysts to carefully define concepts in codebooks. LLMs, in contrast, can "fail silently": they can generate plausible labels, even in the absence of well-defined classes. In §3, we provide real-world examples of computational social scientists short-cutting the conceptualization step when using LLMs.

Claim 3. Neither post-hoc bias correction methods nor a more accurate LLM can overcome conceptualization-induced bias. The *science* aspect of computational social science implies that predicted labels are used in *downstream statistical inference*, e.g., a correlation (regression) analysis. In recent years, methodologists have proposed unbiased estimators that combine (noisy) LLM-predicted labels with gold-standard human labels, such as PPI (Angelopoulos et al., 2023), DSL (Egami et al., 2023), and CDI (Gligoric et al., 2025). Yet, expert annotators—who by Assumption 2 do not make scoring errors—can still produce incorrect labels if provided with "incomplete" codebooks. These errors will result in biased downstream estimates even after applying post-hoc correction methods such as PPI or DSL. Likewise, conceptualization-induced bias cannot be corrected solely by more accurate LLM classifiers: increasing LLM accuracy (with an incomplete codebook as input) cannot address bias. We provide simulation experiments to support this claim in §5.

Our three claims reinforce the thesis of this paper: in LLM-era text classification, conceptualization *remains* a first-order concern, as it was in pre-LLM era CSS text analysis.

In the next section, we introduce a running example of classifying PROTESTS and a set of (fictionalized) vignettes expanding Claims 1 and 2. We expand our conceptualization suggestions in §3 and LLM-scoring suggestions in §4. Finally, in §6, we recommend a path forward for LLM-era CSS measurement.

2 Running Example: Protests

To ground our arguments in a real-world application, we use a running example of PROTEST classification, a well-studied task in social science research; see Figure 2 and §B for additional examples. Consider a political scientist studying the effects of exposure to protests (independent variable) on anti-incumbent voteshare in a semi-autocratic country (dependent variable) across geographical regions in a single time period. They hypothesize that large, public, peaceful protests lower the perceived costs for others to express opposition. To test this theory, they collect a news corpus and require binary protest labels for each document in the corpus.

In Figure 1, we use this PROTEST example to illustrate the three main steps for classification in CSS: conceptualization, scoring documents, and downstream statistical inference. *Conceptualization* transforms a *background concept* (e.g., "a protest") to a *systematized concept* written in a codebook.[4] In the current LLM-era, *scoring* consists of an LLM which takes as input the codebook and documents to make predictions. Predictions are then used in *downstream estimates* of, e.g., the mean (prevalence) or correlation (regression) with other variables.

A *conceptualization error* arises from an incomplete codebook: the analyst's written definition does not fully specify the target concept, so even

[2] See §3.5 for discussion of "complete" versus "incomplete" codebooks. Fariss et al. (2020) use the term "translation validity" for this concept.

[3] §4 discusses potential tradeoffs between these.

[4] §A provides greater details on this terminology.

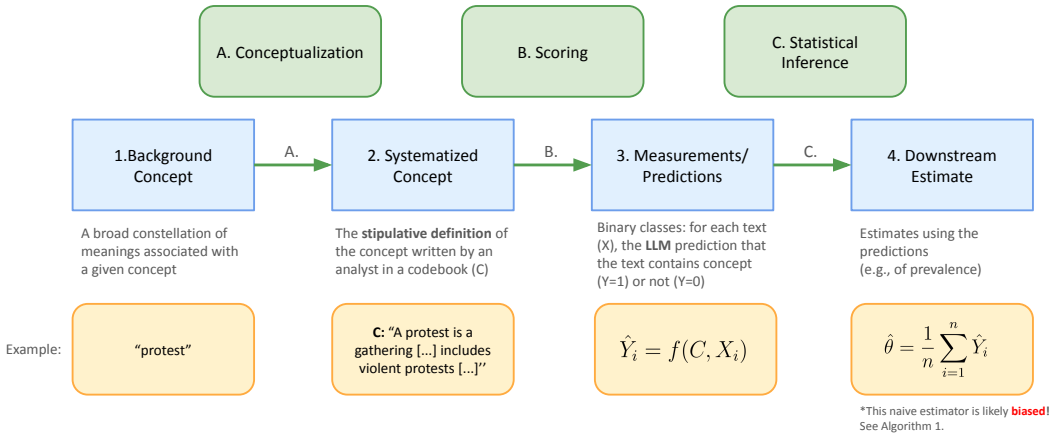

Figure 1: **Text analysis with codebooks and LLMs**. *Conceptualization* transforms a *background concept* (e.g., "a protest") to a *systematized concept* written in a codebook. In the current LLM-era, *scoring* consists of an LLM which takes as input the codebook and documents to make predictions. Predictions are then used in *downstream estimates* of, e.g., the mean (prevalence) or correlation (regression) with other variables.

an expert annotator under Assumption 2 could produce incorrect labels for some documents. For example, if the codebook defines PROTEST without addressing whether violent events are included, then a riot may be labeled as a protest despite the analyst's intent to exclude it. A *scoring error* arises during annotation: the codebook fully specifies the concept, but the annotator (human or LLM) misapplies it to a particular document, producing a different answer than that of an expert (oracle) annotator.

In the next sections, we present stylized extreme archetypes in vignette form—an LLM pessimist and optimist—to illustrate the tradeoffs at each extreme. We then introduce our recommended "pragmatist" approach. To simplify the math and make our points even clearer, we use an even simpler downstream inference goal: estimating the proportion of PROTEST documents in a corpus of size N: $\theta = \mathbb{E}[Y]$ where $Y = 1$ indicates a news article contains a protest (according to a fully specified codebook C) and $Y = 0$ indicates an article does not. Each analyst seeks to obtain **unbiased** estimates of this parameter with **low variance** and **low cost** (in terms of time, compute and annotation expenses).

2.1 Vignette: The LLM pessimist

Imagine an "LLM pessimist" who follows a traditional, pre-LLM process, eschewing any LLM-generated labels. They spend approximately a year obtaining a grant, then spend six months iteratively developing the codebook C,[5] and another six months obtaining the final gold-standard human annotations (Y_j) on $j = 1, 2, \ldots, n$ news articles, where sampled documents $n \ll N$.[6] Then their point estimates and 95% confidence intervals (CIs) under a normal approximation are

$$\hat{\theta}_{\text{pess}} = \frac{1}{n}\sum_{j=1}^{n} Y_j \tag{1}$$

$$\textbf{95\% CIs: } \hat{\theta}_{\text{pess}} \pm 1.96\sqrt{\frac{\hat{\sigma}_y^2}{n}} \tag{2}$$

where $\hat{\sigma}_y^2$ is the empirical variance of the gold-standard annotations, Y.

The advantage of the LLM pessimist approach is that their prevalence estimate, $\hat{\theta}_{\text{pess}}$, will be **unbiased** (under Assumption 2). However, this approach has the disadvantage that it is **extremely costly** in terms of analyst time and annotation expenses.

2.2 Vignette: The LLM optimist

Most CSS work has presented an optimistic view of LLMs replacing annotators; notably, Baumann et al. (2025) systematically review 103 papers that use or benchmark LLMs in CSS and find "88% of reviewed papers recommend using LLMs for data annotation tasks." Our fictional "LLM optimist"

[5]We think these are realistic estimates as Gerner et al. (2002) note, "CAMEO itself required about six months to develop, with between three and six people involved in the process at various times."

[6]We assume N is sufficiently large that it is cost-prohibitive to manually annotate the entire corpus.

Algorithm 1 Prevalence (mean) estimator with LLMs; adapted from Angelopoulos et al. (2023)

Input: Unlabeled texts $\{X_i\}_{i=1}^N$, codebook C, texts with (gold-standard) human labels $\{(X_j, Y_j)\}_{j=1}^n$, LLM-predictions $\hat{Y} = f_{\text{LLM}}(\cdot)$, error level $\alpha = 0.05$

1: $\hat{\theta}^f \leftarrow \frac{1}{N}\sum_{i=1}^N f(C, X_i)$ ▷ LLM-only prevalence estimate

2: $\hat{\Delta} \leftarrow \frac{1}{n}\sum_{j=1}^n \big(f(C, X_j) - Y_j\big)$ ▷ empirical "rectifier" using human labels

3: $\hat{\theta}^{\text{PP}} \leftarrow \hat{\theta}^f - \hat{\Delta}$ ▷ "prediction-powered" estimate

4: $\hat{\sigma}_f^2 \leftarrow \frac{1}{N}\sum_{i=1}^N \Big(f(C, X_i) - \hat{\theta}^f\Big)^2$ ▷ empirical variance of LLM estimate

5: $\hat{\sigma}_{f-Y}^2 \leftarrow \frac{1}{n}\sum_{j=1}^n \Big(f(C, X_j) - Y_j - \hat{\Delta}\Big)^2$ ▷ empirical variance of empirical rectifier

6: $w \leftarrow 1.96\sqrt{\frac{\hat{\sigma}_f^2}{N} + \frac{\hat{\sigma}_{f-Y}^2}{n}}$ ▷ normal approximation to CIs

Output: Prevalence point estimate, $\hat{\theta}^{\text{PP}}$, and 95% confidence interval, $\mathcal{C}^{\text{PP}} = \big(\hat{\theta}^{\text{PP}} \pm w\big)$

takes these high-level findings at face value, and defaults to a simple approach. They input all N news articles into an API-based LLM, $f(\cdot)$, with the prompt *"Classify as protest: yes/no"*. Thus, their codebook definition ($\tilde{C}$) consists of only the surface-form of the class label, "protest", in contrast to the pessimist's fully specified codebook C. They obtain a point estimate and 95% CIs via

$$\hat{\theta}_{\text{optimist}} = \frac{1}{N}\sum_{i=1}^N f(\tilde{C}, X_i) \quad (3)$$

$$\textbf{95\% CIs: } \hat{\theta}_{\text{optimist}} \pm 1.96\sqrt{\frac{\hat{\sigma}_f^2}{N}} \quad (4)$$

where $\hat{\sigma}_f^2$ is the empirical variance of the LLM predictions.

The advantage of the optimist approach is that it is **extremely cheap**: they spent minimal time conceptualizing the concept PROTEST, the LLM predictions from an API were far cheaper than hiring human annotators, and the analyst did not have to install software and use expensive hardware to train supervised models. However, even despite obtaining labels for all N documents, they obtain **biased** estimates ($E[\hat{\theta}_{\text{optimist}}] - \theta \neq 0$) for two reasons. First, errors in the LLM's labels (scoring error) lead to biased estimates (Angelopoulos et al., 2023; Egami et al., 2023). Second, their limited codebook likely results in a conceptualization error: $\mathbb{E}[Y|X, \tilde{C}] \neq \mathbb{E}[Y|X, C]$. For example, based on the downstream inference goal we described in §2, they should exclude *violent* events from their definition of protests. We return to and expand these conceptualization and scoring errors in §3 and §5.

2.3 Vignette: The LLM pragmatist

Now, let us turn to a hybrid approach of an LLM pragmatist. The pragmatist analyst spends a few weeks conceptualizing PROTEST and writing a bespoke definition of PROTEST in a codebook. They spend $10 inputting all N news articles into an API-based LLM with their full codebook definition.[7] Finally, they spend one day manually annotating n randomly sampled examples. Combining their n gold-standard annotations with the N LLM predictions, they obtain a prevalence point estimate and confidence intervals via Angelopoulos et al. (2023)'s "prediction-powered inference" (PPI) algorithm, which we have adapted for codebooks and LLMs in Algorithm 1.

This pragmatist approach has the advantage that the estimates are **unbiased**; Angelopoulos et al. prove their estimator is unbiased (under Assumption 2) and has valid coverage for any error level. Also, the pragmatist approach is much **cheaper** than the labor-intensive approach of the pessimist. Because of these advantages, we recommend CSS analysts adopt this pragmatic approach in future studies. However, even a pragmatist can improve estimates by focusing on the following.

Avoiding conceptualization errors that come from incomplete codebooks and result in statistical bias. In §3, we provide real-world examples of possible definitions of PROTEST (that a pragmatist might use) and what constitutes conceptualization errors. In §5, we show that even post-hoc bias correction methods cannot correct for conceptualization-induced bias.

Reducing scoring errors from an LLM (increas-

[7] Alternatively, they may be able to use a supervised classifier instead of an LLM. See §4.2.

ing LLM accuracy) which reduces variance. As we see in Line 6 of Algorithm 1, decreasing LLM scoring (prediction) errors results in a decrease in $\hat{\sigma}^2_{f-Y}$ which results in estimates with lower variance (narrower CIs). While improving LLM accuracy is a large focus of modern NLP research, we elaborate on particular concerns for CSS analysts in §4.

3 Pragmatists Aim to Avoid Codebook Conceptualization Errors

Conceptualization is the process of transforming a *background concept* with a broad constellation of meanings ("protest") into a *systematized concept*—which we denote as PROTEST; see Step A in Figure 1. In this section, we describe how concepts have traditionally been systematized in CSS, explain how LLMs can tempt analysts to skip the conceptualization step (Claim 2), provide guidance on when conceptualization is sufficient, and discuss tradeoffs with LLM-assisted codebook development.

3.1 Definition types in codebooks can vary.

In Adcock and Collier's measurement framework, analysts transform background concepts into systematized concepts through *definitions*, which we categorize into three levels of specificity:

Type I. Surface form of label. The simplest definition consists solely of the surface form of the background concept, e.g., "Is this a *protest*?" (the conceptualization used by the LLM optimist vignette). This assumes that the concept's meaning is self-evident or can be learned from background corpora.

Type II. Dictionary entry. Alternatively, one could use a generic definition from a dictionary, e.g., "A protest is a public (often organized) manifestation of dissent."[8] A dictionary definition is intended to apply to many downstream use-cases.

Type III. Stipulative definition. Often, social scientists need a more specific definition than a dictionary entry. In this case, a domain expert could craft a *stipulative definition*, a *new* definition of a lexical unit for a specific context (Lycan and Lycan, 1994; Hitchcock, 2021). For example, Raleigh et al. (2010) defines PROTEST as *"an in-person public demonstration of three or more participants in which the participants do not engage in violence, though violence may be used against them."* This definition includes both inclusion criteria—a PROTEST must be in public and in-person (not online)—and exclusion criteria—violence by protesters, and protests with only one or two participants are excluded.

[8] WordNet (Miller, 1995) https://en-word.net/lemma/protest

3.2 The same background concept can map to multiple systematized concepts

For most substantive applications, stipulative definitions are required because the same background concept (e.g., "protest") can map onto multiple systematized concepts. Figure 2 illustrates this variability with four different stipulative definitions of PROTEST from real-world codebooks: the Automated Content Extraction (ACE) program (Doddington et al., 2004); the Armed Conflict Location and Event Data (ACLED), a widely used hand-coded dataset on violence and protest (Raleigh et al., 2010); the CAMEO event schema used by several machine-coded event datasets (Gerner et al., 2002); and the Crowd Counting Consortium (CCC)'s codebook for US-specific protests (Crowd Counting Consortium, 2024).

These codebooks differ as to whether a PROTEST includes "violence *by* protesters", the number of participants required, whether online protests are included, and several other aspects—see Appendix §B for the full definitions. Thus, at inference time, the same text—e.g., the sentence "a group of angry youth smashed the windows of businesses"—would result in different gold-standard class labels depending on the stipulative definition, i.e. $Y_i|X_i, C \neq Y_i|X_i, C'$. We emphasize that this does not imply that one stipulative definition is "more accurate" than the others. Instead, they reflect different conceptualizations that are consequential for their specific downstream applications.

3.3 Zero-shot LLMs can tempt analysts to skip conceptualization.

The advent of zero-shot classification with LLMs has fundamentally altered the conceptualization process by removing the forcing function that expert annotators provided. In the pre-LLM era, expert annotators effectively required Type III stipulative definitions. Faced with ambiguous cases in early pilot rounds of annotation, annotators might ask clarifying questions such as: "Do labor strikes count?" "How many people must be present?" "Does violence by protesters exclude an event from the PROTEST class?" This forced analysts to de-

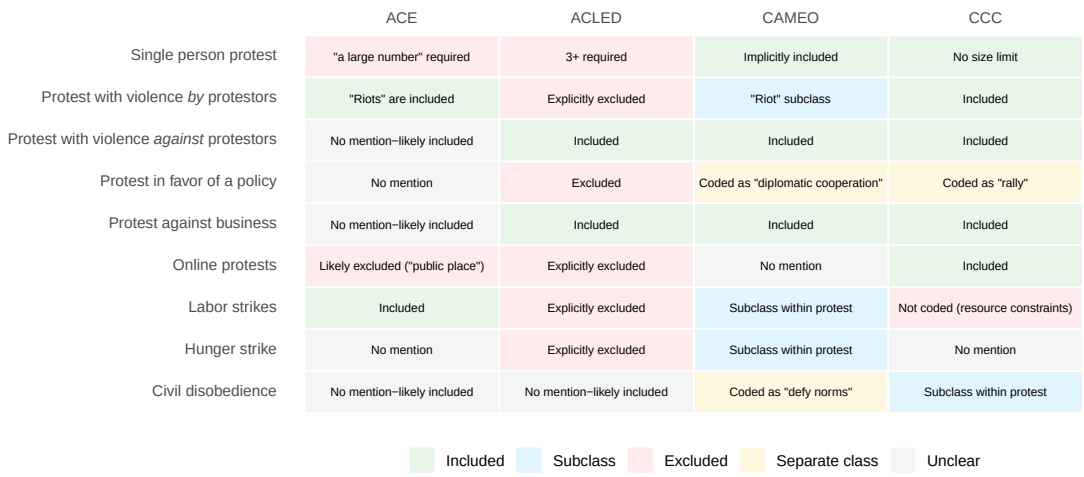

Figure 2: **Different stipulative definitions of PROTEST from real-world codebooks.** We manually categorize aspects of protest definitions from the codebooks of ACE (Doddington et al., 2004), ACLED (Raleigh et al., 2010), CAMEO (Gerner et al., 2002), and the Crowd Counting Consortium (2024) (CCC). The length of the definitions also varies: from around 40 (white-space) tokens in ACLED, to around 100 for CCC and ACE, to over 700 for CAMEO. See full definitions in §B.

velop precise stipulative definitions and explicitly address edge cases.

In contrast, LLMs can generate plausible labels even when provided only Type I or II definitions in their prompts. Instruction-tuned LLMs gained widespread appeal due to their high generalization performance given simple natural language instructions (Type I definitions) such as "Is the sentiment of this movie review positive or negative?" (Wei et al., 2022a).[9] This ability of LLMs to draw on concepts learned during training is useful, but creates the risk of "silent failure": the LLM can generate seemingly plausible labels without raising questions or signaling ambiguity an expert annotator would.

This temptation to skip conceptualization is evident in CSS papers in recent years in which researchers used zero-shot LLMs with Type I definitions without addressing conceptualization concerns. Brandt et al. (2024) provided LLMs with a document and list of event labels without additional definitions. Ziems et al. (2024)'s experiments across 25 CSS benchmarks largely excluded class definitions from LLM prompts.[10]

These current prompting practices—instructing the LLM to focus on generating particular labels—potentially limit the ability of the LLM to raise underspecification or ambiguity. In the next section we discuss a complementary practice: how LLMs could possibly be used in initial pilot rounds of codebook conceptualization.

3.4 Using LLMs for (semi-)automation of codebook conceptualization has tradeoffs.

Recent work has proposed incorporating LLMs into the process of codebook conceptualization (Dai et al., 2023; Gao et al., 2023; Xiong et al., 2025; Zhong et al., 2025). While these approaches have the advantages of decreasing analyst time used for conceptualization and potentially surfacing "edge cases", empirically evaluating their performance requires collecting expert labels across several versions of a codebook. Whether LLM-assisted codebooks overinflate analysts' confidence in the "completeness" of their codebooks is also an open question.

3.5 What constitutes a "complete" codebook?

Although no codebook can address all possible edge cases, we propose two potential complementary standards for when a codebook is sufficiently complete. We note that neither standard is fully complete and flag this as an area for future work.

Community consensus on relevant aspects. One completeness standard is that a codebook should address all aspects that a community of

[9] Focus on generalization with Type I definitions has been a long trend in NLP including "dataless classification" (Chang et al., 2008) and applying general "commonsense" or "world knowledge" (Zellers et al., 2018; Sap et al., 2019; Bisk et al., 2020, *inter alia*).

[10] See, for example, mappings.py from their replication materials.

domain experts agree are relevant for defining class membership and possibly affect downstream inference. For PROTEST, experts would likely judge a codebook as "complete" if it addresses all aspects shown in Figure 2—participant violence, minimum size, location, violence *against* protesters, etc.[11] This builds on an argument in the causal inference literature that it "is impossible to provide an absolutely precise definition of a version of treatment" (Hernán, 2016) (675), and instead, "[d]eclaring a version of treatment sufficiently well-defined is a matter of agreement among experts based on the available substantive knowledge" (676).

Expert agreement as an operational test. Drawing on our two assumptions (the existence of a gold standard label, and of an expert annotator who perfectly applies codebook definitions), a second practical standard is that two expert annotators, working independently with only the codebook (and no additional communication), should achieve near-perfect inter-annotator agreement, i.e. *construct reliability* (Jacobs and Wallach, 2021). However, the stringency of these two assumptions limits the practicality of this test.

These standards have an important implication: conceptualization cannot remain implicit. An analyst may have a clear mental model of PROTEST, but it must be written down in a codebook[12] to ensure *test-retest reliability* (Jacobs and Wallach, 2021), and then this complete codebook should be provided to both the expert annotators and LLM(s).

3.6 Conceptualization is *not* prompt engineering.

Finally, we assert that *conceptualization* and *prompt engineering* are not equivalent. A wide set of prompt engineering techniques exists to improve LLMs' performance by modifying their input, for example, by using few-shot learning, structured chain-of-thought reasoning, or specific formatting in the prompt. Because prompt engineering can improve LLM accuracy, analysts may believe that prompt optimization can overcome failures of conceptualization. However, as §5 shows, even a perfect annotator applying an incomplete codebook will yield biased estimates.

[11] As another applied example, consider the definition of civil war: how many annual battle deaths are required? Are anti-colonial wars included?

[12] In the appendix we further formalize this claim; the "reliability error" row in Table A1.

4 Pragmatists Aim to Reduce LLMs' Errors in Applying Codebooks

Post-hoc correction methods guarantee unbiased estimates—under Assumption 2—but analysts also want precise, low-variance estimates. Increasing LLM accuracy (decreasing scoring errors) yields downstream estimates with less variance (e.g., narrower CIs), but comes with cost tradeoffs. An enormous amount of work is focused on improving LLMs' ability to apply instructions or attend to definitions in their context (Zhou et al., 2023; Halterman and Keith, 2025; Nguyen et al., 2025). Here, we highlight a few points that we believe computational social scientists can actualize in their own work—intervening on the codebook and prompt, the model, and the parsing of the output—and also highlight the cost tradeoffs.

4.1 Optimizing prompts

The easiest point of intervention for researchers seeking to improve LLM performance is to modify the inputs to the LLM, specifically the prompt format and the codebook. LLMs' accuracy is sensitive to even small changes in prompt format (Sclar et al., 2024; Schulhoff et al., 2024), including in CSS (Atreja et al., 2025).

Longer codebooks increase computational costs. The length and detail of codebooks imposes a tradeoff between conceptualization errors and LLMs' errors in applying the codebook. Adding longer clarifications, positive or negative examples, or "FAQs" on edge cases may more precisely define their classes, but LLMs are known to struggle with longer contexts, even prompts that fit within their maximum context length (Hsieh et al., 2024; Levy et al., 2024; Liu et al., 2024; Hong et al., 2025). It is thus possible that a very detailed codebook may *decrease* accuracy. Moreover, there is some evidence that (stipulative) definitions in the prompt may not always override the LLM's pretrained sense of a word or concept (Nguyen et al., 2025).

Automated prompt optimization requires additional annotations. Analysts may turn to automatic prompt engineering techniques such as AutoPrompt (Shin et al., 2020), DSPy (Khattab et al., 2023), or using LLMs themselves to optimize prompts (Zhou et al., 2022). However, these techniques still require (1) an initial prompt and (2) additional gold-standard labeled examples against which the prompts are optimized.

4.2 Optimizing models

Second, researchers can reduce LLM's labeling error via the choice or modification of the LLM itself. In general, larger models are more accurate (Kaplan et al., 2020) but require more compute (either locally or behind APIs). In addition to this obvious choice of *which* LLM to use, applied CSS researchers may want to consider two other options that have been shown to increase performance, of course, with tradeoffs.

Fine-tuning requires open-weight models and more annotations. Halterman and Keith (2025) show that parameter-efficient supervised fine-tuning (SFT) can improve LLM performance on three CSS datasets. However, SFT precludes the use of closed-weight LLMs and requires a large investment of time and computational resources. It also imposes additional data requirements: a gold standard example can be used either for SFT or for post-hoc bias correction, but not both.

Using multiple models increases computational costs. An exciting area of current research is how to best combine multiple LLMs, or combine LLMs with older-generation (smaller or encoder-only) models. Pangakis and Wolken (2024) and Halterman (2025) propose using *distillation* methods: using a larger LLM (the teacher model) to generate "silver standard" labels and then fine-tuning smaller open-weight models or supervised classifiers (the student model) on these generated labels.

4.3 Optimizing model outputs

Analysts may also be able to improve performance by focusing on the outputs of an LLM (at inference time), again with costs.

Scaling test-time compute increases computational costs. Iterative inference methods, e.g., *chain-of-thought* reasoning (Wei et al., 2022b) and attempts to "scale test-time compute" (Snell et al., 2024) have been shown to improve LLM accuracy, but at the cost of many additional output tokens.

Calibrated probabilistic classifications require more annotations. Prevalence estimation and other downstream inference estimators can often achieve better estimates by using "soft" classification probabilities, $\hat{P}(Y = 1|X)$ (Keith and O'Connor, 2018). Obtaining well-calibrated probabilities from a generative LLM, either using token probabilities or "verbalized confidence scores," is still an open challenge (Tian et al., 2023; Xiong et al., 2024).[13] Future work could examine how to optimally allocate a portion of one's gold-standard annotation budget to supervised methods for calibrating confidence scores, i.e., Platt scaling (Platt et al., 1999).

For a given CSS project, the benefits of increasing LLM accuracy and decreasing variance of downstream estimates may be worth the costs. However, we emphasize that none of these LLM techniques alone are able to correct for conceptualization-induced bias, which we elaborate on in the next section.

5 Simulation Experiments

This section provides simulation evidence for Claim 3: that conceptualization errors cannot be fixed solely by post-hoc bias correction methods. We focus only on the "pragmatist" approach that combines LLM-generated labels and gold-standard (human) labels.[14] To simplify experiments and avoid noise from natural language text, we do not generate synthetic text or apply an LLM, and instead we generate (simulated) structured data to represent different codebooks and annotations on documents.

We simulate a dependent variable Z_i that depends on whether a (true, non-violent) protest occurs ($D_i = 1$), and whether the event is violent ($V_i = 1$):

$$Z_i = \beta_0 + \tau D_i + \beta_1 V_i + \beta_2 X_i + \epsilon_i \quad (5)$$

where $\tau = 1$ is the true positive effect of peaceful protest, $\beta_1 = -5$ is the negative effect of violent events, X_i is a covariate, and $\epsilon_i \overset{\text{iid}}{\sim} \mathcal{N}(0, \sigma^2)$

Under a complete codebook, expert annotations are $Y_i^{\text{complete}} = D_i$, where D_i is the true protest indicator that by construction excludes violent events (V_i). Under an incomplete codebook that does not address violence, expert annotations become $Y_i^{\text{incomplete}} = D_i \vee V_i$: violent events are labeled as protests because the incomplete codebook gives annotators no basis to exclude them.

Our simulated LLM annotations follow the same codebook-dependent labels with additional random noise (see §D). We generate $n = 1K$ gold-standard human annotations deterministically given the complete and incomplete codebooks respectively, and generate $N = 10K$ noisy "LLM" annotations, $\hat{Y}_i$.

[13] Even extracting hard classification labels from a generative model can be difficult (Schulhoff et al., 2024).

[14] See §D for additional simulations under the "pessimist" and "optimist" paradigms.

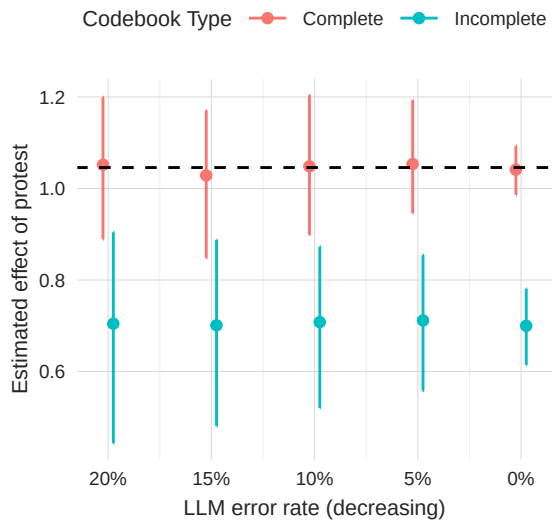

Figure 3: **DSL-corrected estimates from simulated data** (§5); we display the mean estimate (dot) and 95% empirical intervals across 250 simulations (bars). $N = 10,000$ per simulation, and the true effect is the dashed line. **Takeaway:** Decreasing annotator scoring error reduces variance, but use of an incomplete codebook always results in biased estimates.

For estimation, we provide the gold-standard human annotations, LLM annotations, and dependent variable to the DSL correction method (Egami et al., 2023).[15] We repeat the data generating process (DGP) under 250 different random seeds and use the empirical central 95th interval as the confidence interval. See §D for the full DGP and additional simulation details.

The results in Figure 3 support Claim 3: the DSL correction yields unbiased estimates when the codebook is complete, with lower LLM errors producing estimates with lower variance. In contrast, when the codebook is incomplete, both the LLM and expert labels are systematically biased because the *violence* aspect is not included in the codebook but affects the dependent variable in the regression. In other words, if the expert annotators are given a codebook that does not reflect the researcher's concept of protest, then their annotations cannot recover the researcher's desired estimand. This reinforces our emphasis on "complete" codebook conceptualizations.

[15]We note that Algorithm 1 presents PPI for prevalence estimation, while here we use DSL (Egami et al., 2023) for adjusting regression coefficients; both share the same core logic of correcting LLM predictions using a small set of gold-standard labels.

6 Conclusion and A Path Forward

In text classification CSS projects, LLMs are almost always used in a broader pipeline that involves the definition of domain-specific concepts and downstream statistical inference. In this work, we show LLMs have already tempted analysts to speed through the conceptualization process and use the surface-form of labels (e.g., "protest") instead of full stipulative definitions. In the presence of a conceptualization error (e.g., not distinguishing a sub-aspect of a protest such as violence *by* protesters), post-hoc bias correction methods will still yield biased results due to bias in the human annotations, posing a challenge for substantive conclusions in the social sciences. These points reinforce our thesis: **conceptualization remains a first-order concern, even in the LLM-era**.

Going forward, we make the following recommendations to CSS analysts who aim to obtain low-cost, unbiased, low-variance estimates. First, we recommend using the "pragmatist" approach which combines gold-standard annotations and LLM-generated predictions with post-hoc bias correction methods. We note, however, that it is an open problem as to which correction method performs best with finite samples for a given task; see de Pieuchon et al. (2025). Second, LLMs cannot fully replace the consensus from a community of peer experts on whether a codebook is "complete", so we recommend that human domain expertise be heavily incorporated into early pilot rounds of creating the codebook. Finally, analysts will likely be able to decrease variance in their estimates by moving beyond vanilla *zero-shot* LLM approaches, e.g., by increasing LLM test-time compute, fine-tuning an LLM on a supervised dataset, or using an ensemble of LLMs and supervised classifiers. Ultimately, we believe that thinking *holistically* about the entire pipeline: *conceptualization*, *scoring*, and *downstream statistical inference* can help solidify LLM-based classification as a rigorous methodology in the social sciences.

Acknowledgments

We thank Maria Antoniak for initial questions that led us to writing this paper. We thank Adel Daoud and Alexander Hoyle for detailed and helpful comments on an earlier draft. We also thank anonymous ARR reviewers for helpful suggestions. KK received support from a YI Grant from Allen Institute for Artificial Intelligence (AI2) and National

Science Foundation under Grant No. 2451403. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of AI2 or the National Science Foundation.

Limitations

Because our paper is primarily a conceptual one, we see only a few limitations to our work. First, as we describe in the introduction, we only consider classification tasks that are not subjective (Assumption 1). Post-hoc bias correction methods (e.g., DSL or PPI) also make this assumption; however, there remain CSS text classification tasks that do not fall under this paradigm. Additionally, we only describe *associative* downstream statistical inference tasks, but likely some of the same conceptualization-induced bias concerns could apply to *causal* tasks as well; see Feder et al. (2022) for a review on causal inference with text.

Second, we only discuss *text* classification in the paper, but our argument could also be extended to other data modalities, such as images, satellite data, audio, or video, where gold standard labels are defined using codebooks.

Third, for expository purposes, we only focus on a single running example from political science: protest classification. However, there are many other complex classification tasks from the social sciences that could similarly be examined using our framework, such as the task of classifying the degree of human rights violations from annual human rights reports, or the task of identifying populist rhetoric in text. These examples would both involve careful work to precisely define human rights violations and populism, respectively, and then obtain labels from text.

Finally, we note that the landscape of LLMs and NLP research is changing very quickly, so new LLM methods that help with conceptualization may emerge (see §3.4).

References

Robert Adcock and David Collier. 2001. Measurement validity: A shared standard for qualitative and quantitative research. *American Political Science Association*.

Anastasios N Angelopoulos, Stephen Bates, Clara Fannjiang, Michael I Jordan, and Tijana Zrnic. 2023. Prediction-powered inference. *Science*, 382(6671):669–674.

Shubham Atreja, Joshua Ashkinaze, Lingyao Li, Julia Mendelsohn, and Libby Hemphill. 2025. What's in a prompt?: A large-scale experiment to assess the impact of prompt design on the compliance and accuracy of LLM-generated text annotations. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 19, pages 122–145.

David Bamman, Kent K Chang, Li Lucy, and Naitian Zhou. 2024. On classification with large language models in cultural analytics. In *CHR 2024: Computational Humanities Research Conference*.

Pablo Barberá, Amber E Boydstun, Suzanna Linn, Ryan McMahon, and Jonathan Nagler. 2021. Automated text classification of news articles: A practical guide. *Political Analysis*, 29(1):19–42.

Joachim Baumann, Paul Röttger, Aleksandra Urman, Albert Wendsjö, Flor Miriam Plaza-del Arco, Johannes B Gruber, and Dirk Hovy. 2025. Large language model hacking: Quantifying the hidden risks of using LLMs for text annotation. *arXiv preprint arXiv:2509.08825*.

Emily M Bender and Alex Lascarides. 2022. *Linguistic fundamentals for natural language processing II: 100 essentials from semantics and pragmatics*. Springer Nature.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. 2020. PIQA: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*.

Patrick T Brandt, Sultan Alsarra, Vito J D'Orazio, Dagmar Heintze, Latifur Khan, Shreyas Meher, Javier Osorio, and Marcus Sianan. 2024. ConfliBERT: A language model for political conflict. *arXiv preprint arXiv:2412.15060*.

Ming-Wei Chang, Lev-Arie Ratinov, Dan Roth, and Vivek Srikumar. 2008. Importance of semantic representation: Dataless classification. In *AAAI*, volume 2, pages 830–835.

Crowd Counting Consortium. 2024. *crowdcounting.org*.

Shih-Chieh Dai, Aiping Xiong, and Lun-Wei Ku. 2023. LLM-in-the-loop: Leveraging large language model for thematic analysis. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 9993–10001.

Nicolas Audinet de Pieuchon, Adel Daoud, Connor Thomas Jerzak, Moa Johansson, and Richard Johansson. 2025. Benchmarking debiasing methods for LLM-based parameter estimates. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 19768–19783.

George R Doddington, Alexis Mitchell, Mark A Przybocki, Lance A Ramshaw, Stephanie Strassel, and Ralph M Weischedel. 2004. The automatic content extraction (ACE) program-tasks, data, and evaluation. In *LREC*, volume 2, page 1.

Vito D'Orazio, Steven T Landis, Glenn Palmer, and Philip Schrodt. 2014. Separating the wheat from the chaff: Applications of automated document classification using support vector machines. *Political analysis*, 22(2):224–242.

Naoki Egami, Musashi Hinck, Brandon Stewart, and Hanying Wei. 2023. Using imperfect surrogates for downstream inference: Design-based supervised learning for social science applications of large language models. *NeurIPS*.

Jackson Ehrenworth and Katherine Keith. 2023. Literary intertextual semantic change detection: Application and motivation for evaluating models on small corpora. In *Proceedings of the 4th Workshop on Computational Approaches to Historical Language Change*, pages 1–14.

Matan Eyal, Shoval Sadde, Hillel Taub-Tabib, and Yoav Goldberg. 2022. Large scale substitution-based word sense induction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4738–4752.

Christopher J Fariss, Michael R Kenwick, and Kevin Reuning. 2020. Measurement models. *The SAGE handbook of research methods in political science and international relations*, pages 353–370.

Amir Feder, Katherine A Keith, Emaad Manzoor, Reid Pryzant, Dhanya Sridhar, Zach Wood-Doughty, Jacob Eisenstein, Justin Grimmer, Roi Reichart, Margaret E Roberts, et al. 2022. Causal inference in natural language processing: Estimation, prediction, interpretation and beyond. *Transactions of the Association for Computational Linguistics*, 10:1138–1158.

Jie Gao, Yuchen Guo, Toby Jia-Jun Li, and Simon Tangi Perrault. 2023. CollabCoder: A GPT-powered workflow for collaborative qualitative analysis. In *Companion Publication of the 2023 Conference on Computer Supported Cooperative Work and Social Computing*, CSCW '23 Companion, page 354–357, New York, NY, USA. Association for Computing Machinery.

Timnit Gebru, Jamie Morgenstern, Briana Vecchione, Jennifer Wortman Vaughan, Hanna Wallach, Hal Daumé III, and Kate Crawford. 2021. Datasheets for datasets. *Communications of the ACM*, 64(12):86–92.

Deborah J. Gerner, Philip A Schrodt, Omür Yilmaz, and Rajaa Abu-Jabr. 2002. Conflict and mediation event observations (CAMEO): A new event data framework for the analysis of foreign policy interactions. *International Studies Association, New Orleans*.

Fabrizio Gilardi, Meysam Alizadeh, and Maël Kubli. 2023. ChatGPT outperforms crowd workers for text-annotation tasks. *PNAS*.

Kristina Gligoric, Tijana Zrnic, Cinoo Lee, Emmanuel Candes, and Dan Jurafsky. 2025. Can unconfident LLM annotations be used for confident conclusions? In *NAACL-HLT*.

Justin Grimmer, Margaret E Roberts, and Brandon M Stewart. 2022. *Text as data: A new framework for machine learning and the social sciences*. Princeton University Press.

Andrew Halterman. 2025. Synthetically generated text for supervised text analysis. *Political Analysis*.

Andrew Halterman and Katherine A Keith. 2025. Codebook LLMs: Evaluating LLMs as measurement tools for political science concepts. *Political Analysis*.

Miguel A Hernán. 2016. Does water kill? a call for less casual causal inferences. *Annals of epidemiology*, 26(10):674–680.

David Hitchcock. 2021. *Definition: A practical guide to constructing and evaluating definitions of terms*. Windsor studies in Argumentation.

Kelly Hong, Anton Troynikov, and Jeff Huber. 2025. Context rot: How increasing input tokens impacts LLM performance. Technical report, Chroma.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, and Boris Ginsburg. 2024. Ruler: What's the real context size of your long-context language models? In *First Conference on Language Modeling*.

Abigail Z Jacobs and Hanna Wallach. 2021. Measurement and fairness. In *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*, pages 375–385.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Katherine Keith and Brendan O'Connor. 2018. Uncertainty-aware generative models for inferring document class prevalence. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4575–4585.

Omar Khattab, Arnav Singhvi, Paridhi Maheshwari, Zhiyuan Zhang, Keshav Santhanam, Sri Vardhamanan, Saiful Haq, Ashutosh Sharma, Thomas T. Joshi, Hanna Moazam, Heather Miller, Matei Zaharia, and Christopher Potts. 2023. DSPy: Compiling declarative language model calls into self-improving pipelines. *ICLR*.

Mosh Levy, Alon Jacoby, and Yoav Goldberg. 2024. Same task, more tokens: the impact of input length on the reasoning performance of large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173.

Li Lucy, Jesse Dodge, David Bamman, and Katherine Keith. 2023. Words as gatekeepers: Measuring discipline-specific terms and meanings in scholarly publications. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 6929–6947.

William G Lycan and William G Lycan. 1994. Stipulative definition and logical truth. *Modality and Meaning*, pages 263–282.

George A Miller. 1995. WordNet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Tu Nguyen, Kevin Du, Alexander Miserlis Hoyle, and Ryan Cotterell. 2025. How persuasive is your context? In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 32085–32111.

Nicholas Pangakis and Samuel Wolken. 2024. Knowledge distillation in automated annotation: Supervised text classification with LLM-generated training labels. In *The Sixth Workshop on Natural Language Processing and Computational Social Science*, page 113.

John Platt et al. 1999. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3):61–74.

Inioluwa Deborah Raji, Emily Denton, Emily M Bender, Alex Hanna, and Amandalynne Paullada. 2021. AI and the everything in the whole wide world benchmark. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Clionadh Raleigh, Rew Linke, Håvard Hegre, and Joakim Karlsen. 2010. Introducing ACLED: An armed conflict location and event dataset. *Journal of Peace Research*, 47(5):651–660.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. 2019. Social IQA: Commonsense reasoning about social interactions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4463–4473.

Dominik Schlechtweg, Barbara McGillivray, Simon Hengchen, Haim Dubossarsky, and Nina Tahmasebi. 2020. SemEval-2020 task 1: Unsupervised lexical semantic change detection. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 1–23.

Sander Schulhoff, Michael Ilie, Nishant Balepur, Konstantine Kahadze, Amanda Liu, Chenglei Si, Yinheng Li, Aayush Gupta, HyoJung Han, Sevien Schulhoff, et al. 2024. The prompt report: A systematic survey of prompting techniques. *arXiv preprint arXiv:2406.06608*.

Melanie Sclar, Yejin Choi, Yulia Tsvetkov, and Alane Suhr. 2024. Quantifying language models' sensitivity to spurious features in prompt design or: How I learned to start worrying about prompt formatting. *ICLR*.

Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting knowledge from language models with automatically generated prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2024. Scaling LLM test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*.

Rosamond Thalken, Edward Stiglitz, David Mimno, and Matthew Wilkens. 2023. Modeling legal reasoning: LM annotation at the edge of human agreement. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9252–9265.

Katherine Tian, Eric Mitchell, Allan Zhou, Archit Sharma, Rafael Rafailov, Huaxiu Yao, Chelsea Finn, and Christopher D Manning. 2023. Just ask for calibration: Strategies for eliciting calibrated confidence scores from language models fine-tuned with human feedback. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

Petter Törnberg. 2024. Large language models outperform expert coders and supervised classifiers at annotating political social media messages. *Social Science Computer Review*, page 08944393241286471.

Pat Verga, Sebastian Hofstatter, Sophia Althammer, Yixuan Su, Aleksandra Piktus, Arkady Arkhangorodsky, Minjie Xu, Naomi White, and Patrick Lewis. 2024. Replacing judges with juries: Evaluating LLM generations with a panel of diverse models. *arXiv preprint arXiv:2404.18796*.

Hanna Wallach, Meera Desai, A Feder Cooper, Angelina Wang, Chad Atalla, Solon Barocas, Su Lin Blodgett, Alexandra Chouldechova, Emily Corvi, P Alex Dow, et al. 2025. Position: Evaluating generative AI systems is a social science measurement challenge. In *International Conference on Machine Learning*, pages 82232–82251. PMLR.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2022a. Finetuned language models are zero-shot learners. In *ICLR*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022b. Chain-of-thought prompting elicits reasoning in large language models. *NeurIPS*.

John Wilkerson and Andreu Casas. 2017. Large-scale computerized text analysis in political science: Opportunities and challenges. *Annual Review of Political Science*, 20:529–544.

Chenfei Xiong, Jingwei Ni, Yu Fan, Vilém Zouhar, Donya Rooein, Lorena Calvo-Bartolomé, Alexander Miserlis Hoyle, Zhijing Jin, Mrinmaya Sachan, Markus Leippold, et al. 2025. Co-detect: Collaborative discovery of edge cases in text classification. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 354–364.

Miao Xiong, Zhiyuan Hu, Xinyang Lu, Yifei Li, Jie Fu, Junxian He, and Bryan Hooi. 2024. Can LLMs express their uncertainty? an empirical evaluation of confidence elicitation in LLMs. In *ICLR*.

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. SWAG: A large-scale adversarial dataset for grounded commonsense inference. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 93–104.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2023. Judging LLM-as-a-judge with MT-bench and Chatbot Arena. *Advances in neural information processing systems*, 36:46595–46623.

Mian Zhong, Pristina Wang, and Anjalie Field. 2025. Hicode: Hierarchical inductive coding with llms. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 31048–31066.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023. Instruction-following evaluation for large language models. *arXiv:2311.07911*.

Naitian Zhou, David Bamman, and Isaac L Bleaman. 2025. Culture is not trivia: Sociocultural theory for cultural nlp. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 25869–25886.

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2022. Large language models are human-level prompt engineers. In *The eleventh international conference on learning representations*.

Caleb Ziems, William Held, Omar Shaikh, Jiaao Chen, Zhehao Zhang, and Diyi Yang. 2024. Can large language models transform computational social science? *Computational Linguistics*.

A Additional Related Work

Literature on Measurement. The terminology surrounding measurement is used inconsistently across the literatures we draw on. Adcock and Collier (2001) distinguish four stages of measurement—background concept, systematized concept, indicators, and scores—connected by conceptualization, operationalization, and scoring. Wallach et al. (2025) adopt a closely related four-level framework but rename the first task "systematization" and treat operationalization as encompassing the development of measurement instruments. Fariss et al. (2020) collapse the background/systematized distinction into a single "conceptualization" stage and use "operationalization" to refer broadly to the design of the data-generating procedure, including the statistical measurement model. Jacobs and Wallach (2021) use "operationalization" in the narrower sense of instantiating a construct as a latent variable in a measurement model and do not use "conceptualization" as a stage label.

Codebook-based measurement compounds this potential confusion because the codebook includes both the systematized definition of the concept and the operational decision rules, examples, and instructions for applying that definition. However, other components of the "operationalization" stage, including the choice of LLM, the prompting strategy, sampling parameters, output parsing, and the annotation interface, live outside the codebook. To minimize ambiguity, we use "conceptualization" in a broad sense to include any part of writing the codebook, recognizing that this elides the conceptual/operational distinction within the codebook itself but reflects how codebook development actually proceeds in practice. We use "scoring" or "applying the codebook" to refer to the LLM or annotator producing labels from documents using the codebook. To avoid confusion, we limit our use of the term "operationalization".

Lexical Semantics. Linguists differentiate between semantic meaning derived from linguistic form and semantic meaning derived from the content to which a speaker publicly commits (Bender and Lascarides, 2022, p. 11). A social scientist's *act* of writing a new stipulative definition of a label in a codebook can be seen as a new public commitment to the meaning of that label. This act could be viewed as *meaning transfer* or a *deliberate semantic shift* (Bender and Lascarides, 2022, p. 52) because there is typically a salient connection between a label's existing sense (the "background concept") and the "nonce" sense in the codebook.

These deliberate semantic shifts are related to *polysemy* (e.g., "kernel" meaning something different in computer science, math, and nutrition) and the NLP tasks of *semantic change detection* (Schlechtweg et al., 2020; Ehrenworth and Keith,

2023) and *word sense induction* (Eyal et al., 2022; Lucy et al., 2023). However, codebook definitions differ in that they represent instantaneous, intentional meaning shifts, rather than natural semantic changes within a larger community over time.

Data Benchmarks. Echoes of our emphasis on conceptualization and local stipulative definitions also appear in work that formalizes and critiques data benchmarks for machine learning and NLP. Raji et al. (2021) warn against "de-contextualized" data and claims of "general knowledge or general-purpose capabilities" in data benchmarks. Wallach et al. (2025) also observe that AI "researchers and practitioners appear to jump straight from background concepts to measurement instruments, with little to no explicit systematization in between." Zhou et al. (2025) call for the community to go beyond static cultural benchmarks and call for "localization" over "generalization." *Datasheets for Datasets* (Gebru et al., 2021) asks authors *"Is there a label or target associated with each instance? If so, please provide a description."* However, they do not prescribe whether the description should be Type II or Type III definition.

Other NLP work. While our work focuses on corpus-centered classification (computational social science), other work aims to evaluate LLM generative output using LLMs themselves, the so called "LLM-as-a-judge" (Zheng et al., 2023), or an ensemble/"jury" of LLM judges (Verga et al., 2024).

B Full definitions of PROTEST from codebooks

This appendix provides the definitions of PROTEST used in the four codebooks we examine in the main text and Figure 2.

B.1 ACE

ACE uses the class label "demonstrate" to refer to the concept of PROTEST in other datasets.

> A DEMONSTRATE Event occurs whenever a large number of people come together in a public area to protest or demand some sort of official action. DEMONSTRATE Events include, but are not limited to, protests, sit-ins, strikes, and riots.

B.2 Crowd Counting Consortium (CCC)

The CCC dataset classifies "protests" in a general sense. It includes a specific "PROTEST" class, and distinguishes between PROTEST, RALLY, MARCH, etc. events. It defines the PROTEST class as

> A crowd gathering in public to express disagreement with, or disapproval or anger or frustration toward, a specific individual or organization that is at or near the crowd's gathering point (e.g., a politician giving a speech, a corporate headquarters, a bank branch, a construction site, a city hall), or in negative reaction to a recent or current event (e.g., the killing of George Floyd, the reversal of Roe v. Wade).

B.3 ACLED

ACLED has the definition:

> A 'Protests' event is defined as an in-person public demonstration of three or more participants in which the participants do not engage in violence, though violence may be used against them. Events include individuals and groups who peacefully demonstrate against a political entity, government institution, policy, group, tradition, business, or other private institution.

In their codebook, they describe how the following are *not* recorded as 'Protests' events:

- "symbolic public acts such as displays of flags or public prayers (unless they are accompanied by a demonstration);"
- "legislative protests, such as parliamentary walkouts or members of parliaments staying silent;
- strikes (unless they are accompanied by a demonstration); and"
- "individual acts such as self-harm actions like individual immolations or hunger strikes."

B.4 CAMEO

The CAMEO codebook defines protests hierarchically. A protest is an event that fits any of the following subclasses:

- Engage in political dissent, not otherwise specified

- Demonstrate or rally
- Conduct hunger strike
- Conduct strike or boycott
- Obstruct passage, block [in protest]
- Protest violently, riot

Each of these subclasses in turn has further subclasses referring to the target or demand of the protest. Each also includes examples, which we omit from the word count provided in the paper. Figure A1 illustrates an entry from the codebook.

CAMEO	1451
Name	**Engage in violent protest for leadership change**
Description	Protest forcefully, in a potentially destructive manner, to demand leadership change.
Usage Notes	Target should be the actor who is expected to relinquish power. Riots that demand new elections should also be coded here.
Example	Egyptian demonstrators rioted following a peaceful demonstration calling for the immediate removal of President Hosni Mubarak from office.

Figure A1: Example entry for the CAMEO codebook, illustrating the hierarchical definition of PROTEST. The "protest" class includes "protest violently, riot" as a subclass, which has a further subclass of "engage in violent protest for leadership change".

C Breakdown of the pragmatic approach

In Table A1 we describe some additional ways one may approach downstream estimation goals. Our recommendation—the pragmatist approach with a complete codebook—is highlighted in green and the setting we explore in our simulation experiments (§5) with an incomplete codebook is highlighted in red.

We note there are two other settings which look similar to the pragmatist approach, but should be avoided. First there could be a **procedural error**, in which a complete codebook (C) is provided to the LLM but an incomplete codebook ($\tilde{C}$) is provided to the expert annotators. We note that if the complete codebook exists, it should also be provided to annotators. Second, there could be a **reliability** error in which an expert can provide gold-standard labels with the conceptualization "in their own head" and result in biased estimates. In §3.5, we discuss how this violates *construct reliability*.

D Additional Simulation Details

Here, we provide additional details on the simulation experiments described in §5.

D.1 True DGP

Motivated by the real-world PROTEST codebooks (Figure 2), we first simulate four aspects of a protest that will not individually affect the dependent variable. A political science domain expert (an author) selected the hyperparameters to represent a (fairly) realistic real-world scenario. For each instance, i, we sample

- $Z_i^1 \sim \text{Bernoulli}(0.2)$, meeting a basic protest definition
- $Z_i^2 \sim \text{Bernoulli}(0.96)$, for events that are not hunger strikes,
- $Z_i^3 \sim \text{Bernoulli}(0.9)$, for protests *against* someone
- $Z_i^4 \sim \text{Bernoulli}(0.88)$, for greater than 3 attendees.

Then, we sample whether the protest is violent which directly affects the dependent variable:

$$V_i \sim \text{Bernoulli}(0.05) \tag{6}$$

Using these aspects, we deterministically set the binary true protest label:

$$D_i = Z_i^1 \wedge Z_i^2 \wedge Z_i^3 \wedge Z_i^4 \wedge \neg V_i \tag{7}$$

We also sample a single *covariate* that will be input into the downstream estimate

$$X_i \sim \mathcal{N}(0, 1) \tag{8}$$

In a real-world example, this covariate might be something like voteshare in a prior election, the degree of urbanization in the geographic unit, or the union density of a region, each of which might be a confounder for the number of protests (independent variable) and later voteshare (dependent variable).

D.2 Simulating annotations

Given the instances from the DGP, we simulate the following annotations. Following the pragmatist framework, we sample $n = 1,000$ gold-standard (human) annotations and $N = 10,000$ LLM annotations. Like the rest of our paper, our assumptions are that (1) gold-standard human annotations are under an assumption that the expert human annotations will contain no labeling errors, i.e., will completely adhere to the codebook they are provided; and (2) the LLMs are not perfectly accurate and will have a certain amount of annotation error.

LLM label	Gold label	Inference	Description
–	$Y = \text{Expert}(X, C)$	Unbiased (high var.)	**Pessimist**. High variance from only human labels ($n << N$)
$\hat{\tilde{Y}} = \text{LLM}(X, C)$	–	Biased	**Optimist.** No correction for annotation/scoring errors
$\hat{Y} = \text{LLM}(X, C)$	$Y = \text{Expert}(X, C)$	Unbiased	**Pragmatist**. A complete codebook permits post-hoc correction.
$\hat{Y} = \text{LLM}(X, C)$	$\tilde{Y} = \text{Expert}(X, \tilde{C})$	Biased	**Procedural error**. Gives incomplete $\tilde{C}$ to expert
$\hat{\tilde{Y}} = \text{LLM}(X, \tilde{C})$	$Y = \text{Expert}(X, \cdot)$	Unbiased	**Reliability error**. Assumes an expert can label without C, see §3.5
$\hat{\tilde{Y}} = \text{LLM}(X, \tilde{C})$	$\tilde{Y} = \text{Expert}(X, \tilde{C})$	Biased	**Conceptualization error**. Incomplete codebook, see §5

Table A1: **Comparing approaches to downstream inference**, many of which combine LLM zero-shot predictions with human expert gold-standard labels. Here, C is a complete codebook and $\tilde{C}$ is an incomplete codebook. We highlight the **pragmatist** row as the recommended approach, and also highlight the conceptualization error row which corresponds to Claim 3.

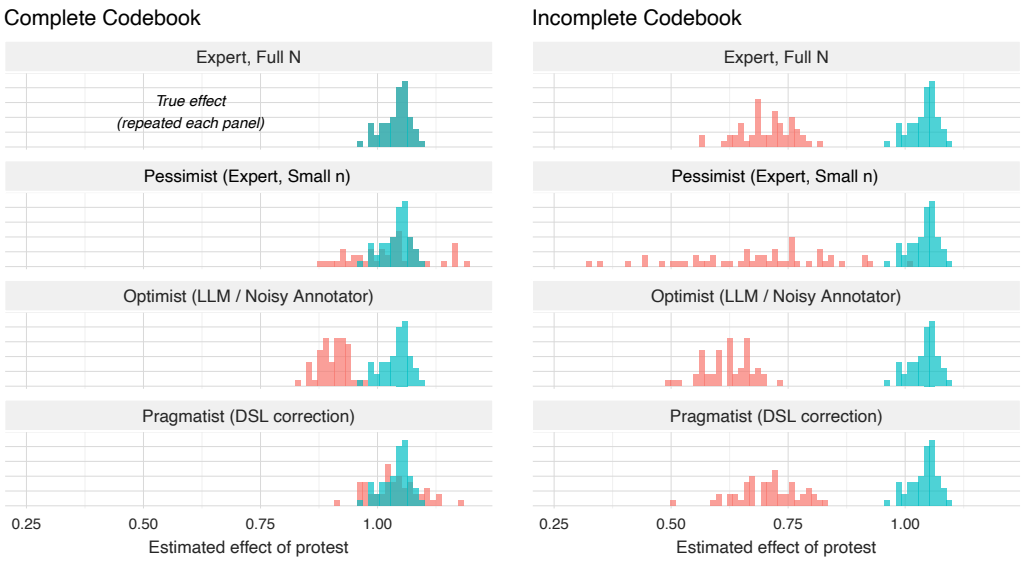

Figure A2: **Simulation results** comparing four estimation strategies across "complete" and "incomplete" codebook conditions across 50 simulations with $N = 10,000$ documents and a regression model where peaceful protests have a positive effect and violent protests have a negative effect. Results in blue (repeated in each panel) show estimates from expert annotation of all N documents. "Small N" and "DSL" conditions use 10% expert annotated documents, and "LLM" uses labels for all N documents with 10% random error. The "incomplete" codebook omits the instruction to exclude violent protests.

Complete codebook annotations. Under the assumptions above, we simulate the human labels under a complete codebook as a deterministic mapping from the true protest labels,

$$Y_i^{\text{complete}} = D_i. \tag{9}$$

Under the complete codebook, we sample LLM-generated labels as additive noise from the true protest labels with noise level δ,

$$P_i \sim \text{Uniform}(0, 1) \tag{10}$$

$$\hat{A}_i^{\text{complete}} = \begin{cases} D_i & \text{if} \quad P_i > \delta \\ |D_i - 1| & \text{if} \quad P_i \leq \delta \end{cases} \tag{11}$$

Incomplete codebook annotations. Under a *complete codebook*, the true protest label (D) and the violent event label (V) are mutually exclusive, i.e., via Eqn. 7 we can never have an instance for which both $D_i = 1$ and $V_i = 1$. Under an *incomplete* codebook, annotators (both human and LLM) only receive limited aspects of the protest definition, and they do *not* have access to the aspect that excludes violent events. Thus, human annotations using an incomplete codebook are

$$Y_i^{\text{incomplete}} = D_i \vee V_i. \tag{12}$$

Likewise, the LLM is given the incomplete codebook so also does not know to exclude violent events and also annotates instances with error level

δ,

$$P_i \sim \text{Uniform}(0, 1) \tag{13}$$

$$\hat{Y}_i^{\text{incomplete}} = \begin{cases} D_i \vee V_i & \text{if} \quad P_i > \delta \\ |(D_i \vee V_i) - 1| & \text{if} \quad P_i \leq \delta. \end{cases} \tag{14}$$

D.3 Estimation with DSL

The inference goal is to estimate $\hat{\tau}$. For each of the two codebook settings, as input into the DSL method (Egami et al., 2023), we provide the LLM labels $\{\hat{Y}_i\}_{i=1}^N$, the covariates $\{X\}_{i=1}^N$, and the dependent variable $\{Y\}_{i=1}^N$ for all N instances. We also input the gold-standard labels $\{Y_i\}_{i=1}^n$ where $n \ll N$ (as we assume in §2). Specifically, we fix $n = \frac{N}{10}$. We also provide DSL with the assumed linear additive structure of the regression coefficients we aim to estimate, $Y \sim \hat{Y} + X$.

D.4 Results

As we show in Figure 3, an incomplete codebook results in bias that cannot be corrected for, even as the LLM error approaches 0%.

In Figure A2, we use the same DGP but also provide estimates for the other approaches (besides pragmatists) described in §2. The regression setup here is similar to our straightforward example of mean estimation in §2: LLM "optimists" use the LLM-produced protest labels ($\hat{Y}$) directly in a regression without any post-hoc bias correction. The LLM "pessimists" use only the gold-standard labels (Y) in the regression. Similarly to the mean estimation in §2, the "optimist" obtains biased regression coefficients, both from LLM's annotation error and conceptualization error, while the "pessimist" obtains unbiased estimates, but higher variance estimates from their smaller n.