Using Large Language Model Annotations for the Social Sciences: A General Framework of Using Predicted Variables in Statistical Analyses*

Naoki Egami† Musashi Hinck‡ Brandon M. Stewart§ Hanying Wei¶

This Version: May 7, 2024

Abstract

Social scientists use automated annotation methods, such as supervised machine learning and, more recently, large language models (LLMs), that can predict labels and generate text-based variables. While such predicted text-based variables are often analyzed as if they were observed without errors, we first show that ignoring prediction errors in the automated annotation step leads to substantial bias and invalid confidence intervals in downstream analyses, even if the accuracy of the automated annotations is high, e.g., above 90%. We propose a framework of *design-based supervised learning* (DSL) that can provide valid statistical estimates, even when predicted variables contain non-random prediction errors. DSL employs a doubly robust procedure to combine predicted labels and a smaller number of expert annotations. DSL allows scholars to apply advances in LLMs and natural language processing to social science research while maintaining statistical validity. We illustrate its general applicability using two applications where the outcome and independent variables are text-based.

Keywords: Large Language Models, Machine Learning, Prediction Errors, Doubly Robust Estimation

---

*The proposed methodology is implemented via our software R package, `dsl` (`http://dsl.software`). This paper extends and generalizes the methods we proposed in Egami *et al.* (2023). We appreciate the excellent research assistance by Songpo Yang, TaeJun Seo, and Benedikt Ströbl. We would like to thank Arthur Spirling, Katie Keith, Max Goplerud, Christian Fong, John Marshall, and Tom Robinson for their thoughtful comments on an early draft. We also appreciate comments from participants at the Political Methodology Summer meeting, the TADA conference, and the 2023 Neurips meeting.

†Corresponding Author. Assistant Professor, Department of Political Science, Columbia University. Email: naoki.egami@columbia.edu. URL: `https://naokiegami.com`

‡Postdoctoral Research Associate, Data-Driven Social Science Initiative, Princeton University. Email: mj2976@princeton.edu. URL: `https://muhark.github.io/about`

§Corresponding Author. Associate Professor, Department of Sociology and the Office of Population Research, Princeton University. Email: bms4@princeton.edu. URL: `https://brandonstewart.org`

¶Ph.D. student, Department of Political Science, Columbia University. Email: hw2893@columbia.edu. URL: `https://polisci.columbia.edu/content/hanying-wei`

1 Introduction

Over the last decade, social scientists have developed and applied a variety of text analysis and natural language processing methods to study a large collection of documents (Grimmer and Stewart, 2013; Gentzkow *et al.*, 2019). In text-as-data applications, one of the most common tasks is text annotation (or text classification) to generate text-based variables for subsequent statistical analyses. For example, Pan and Chen (2018) first annotate whether each online post accuses local Chinese officials of corruption so that they can later study whether and how much such online complaints are censored. Fowler *et al.* (2021) first annotate the tone of political ads and then analyze how politicians strategically change the tone of political advertising online and offline.

In an ideal world without any budget and time constraints, researchers, as domain experts, might want to carefully annotate all the documents they use in their main statistical analyses. However, this is often impossible for a large number of documents that social scientists analyze these days. To facilitate large-scale annotations, social scientists have used a variety of supervised machine learning (ML) methods to automate this text annotation step by training machines to mimic expert-coding (Hastie *et al.*, 2009; Barberá *et al.*, 2021). More recently, a growing number of papers propose using large language models (LLMs), such as ChatGPT, to automate text annotations by predicting text labels (e.g., Bommasani *et al.*, 2021; Ornstein *et al.*, 2022; Gilardi *et al.*, 2023; Linegar *et al.*, 2023; Ollion *et al.*, 2023; Pangakis *et al.*, 2023; Ziems *et al.*, 2023). Given that researchers can adapt LLMs to perform a wide range of text annotation tasks by simply changing prompts, automated LLM annotations present exciting opportunities for the social sciences.

While text annotation is essential, it is only the first step. Social scientists are often primarily interested in using text labels predicted by automated methods as key variables in subsequent statistical analyses (Hopkins and King, 2010; Egami *et al.*, 2022; Grimmer *et al.*, 2022). In the vast majority of current applications, researchers treat predicted text-based variables as if they were observed without any error: they ignore *prediction errors* in the first step of automated text annotation (Benoit *et al.*, 2009; Wang *et al.*, 2020; Fong and Tyler, 2021; Knox *et al.*, 2022). The natural intuition behind this common practice is that when the prediction accuracy is high enough, the underlying automated text annotation model, whether it is an LLM or a supervised ML model, "learned" how to label texts, and prediction errors are small enough that analysts can ignore them.

However, we clarify that ignoring such prediction errors in the first step of text annotation, even if the errors are small, leads to substantial bias, invalid confidence intervals, and wrong p-

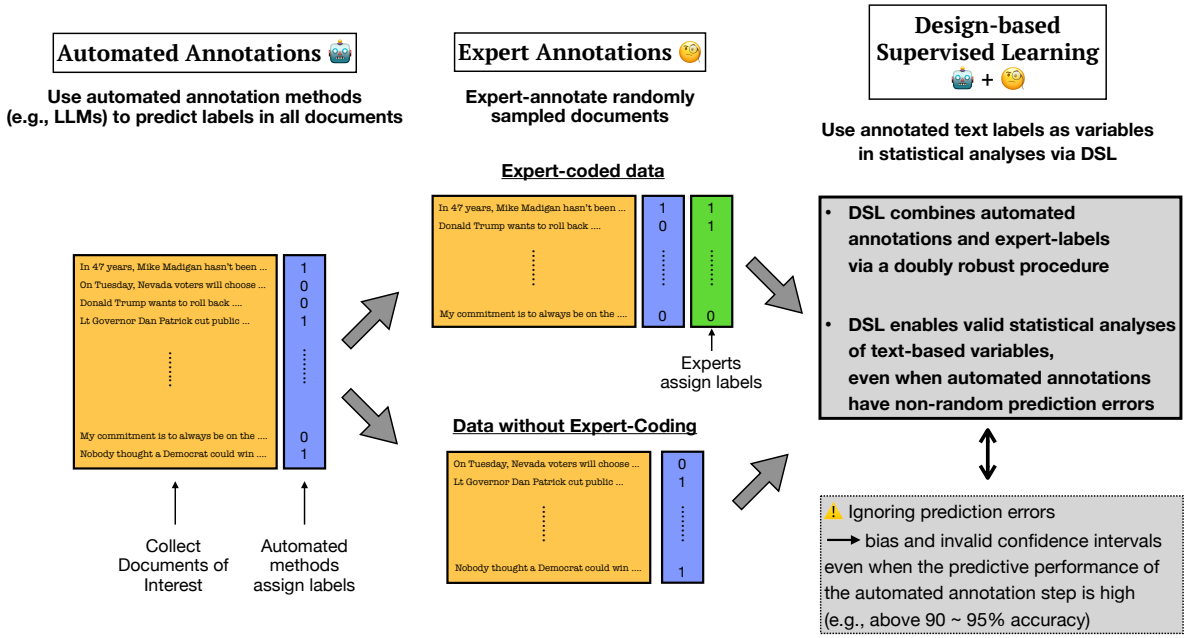


Figure 1: **Overview of the Design-based Supervised Learning (DSL).**

values in downstream statistical analyses of text-based variables. Biases from prediction errors exist even when the prediction accuracy in the text classification step is extremely high, e.g., above 90% or even at 95%. This is because prediction errors are not random—prediction errors are correlated with observed and unobserved variables we include in downstream analyses. In practice, this means that substantive and statistical conclusions can easily flip if researchers choose slightly different automated text annotation methods, as we empirically see in Section 5.

In this paper, we develop a general framework for using predicted variables in downstream statistical analyses without suffering from bias due to prediction errors. Unlike the existing approaches, the proposed approach, which we call *design-based supervised learning* (DSL), allows researchers to obtain statistically valid estimates and standard errors, even when automated text annotation methods have arbitrary non-random prediction errors. In practice, this means DSL enables researchers to use any recent advances in LLMs and natural language processing methods for automated text annotation, without worrying that downstream statistical analyses suffer from biases and non-random prediction errors.

Methodologically speaking, DSL combines large-scale (potentially biased) automated annotations and a smaller number of expert annotations (see Figure 1 for an overview of the method). In particular, DSL does it with a tailored bias-correction step building on the literature of doubly robust estimation (Robins *et al.*, 1994; Chernozhukov *et al.*, 2018). While DSL provides statistically valid estimates regardless of the prediction accuracy of the underly-

ing automated text annotation method, DSL can reduce standard errors when the underlying automated text annotation method becomes more accurate. Therefore, as LLMs improve over time, DSL becomes more efficient, too.

Most importantly, DSL only requires one assumption that researchers control the process through which documents are sampled for expert annotations. One of the most common scenarios is that researchers randomly sample documents for expert-coding. Researchers can also change the sampling probability for each document based on observed variables. This assumption is straightforward to guarantee by research design in many social science applications, which gives the name, *design-based* supervised learning. We do not make any assumptions about prediction errors in the underlying automated text annotation method.

Overall, DSL merges the complementary strengths of two annotation approaches: expert annotations are higher quality but expensive, while automated text annotations are scalable but have unknown prediction errors. Using only one of them in the main statistical analyses is suboptimal: researchers who only use expert-coded documents will miss so many documents they cannot expert-annotate, while researchers who only use automated text annotation methods will suffer from unknown large biases. DSL allows users to obtain statistically valid estimates, while gaining efficiency from automated text annotation methods.

Our proposed approach is a general-purpose method that works in a wide range of text-as-data applications. DSL can incorporate any automated text annotation methods, including the current and future LLMs, as well as the classical supervised ML method. DSL can be applied to a variety of common downstream analyses scholars conduct with text-based variables: linear, logistic, multinomial-logistic, Poisson, and linear fixed-effects regression, as well as the estimation of category proportions and causal inference with texts.[1] In Section 6, we provide practical guides to help researchers navigate several practical choices, such as how to determine the required number of expert annotations and what to do if expert annotations are not reliable enough. We offer an easy-to-use R package `dsl`, which can implement all the methods described in this paper with simple functions.

To concretely illustrate our proposed method, we use two empirical applications, Fowler *et al.* (2021) and Pan and Chen (2018), throughout the paper. Like many text-as-data studies, both applications first annotate documents and then use annotated text labels as the key variables in the main downstream statistical analyses. Using these applications, we will show how LLMs can

[1]In general, the proposed DSL framework can be applied to any statistical method that can be written as a convex optimization problem or a moment estimator. See our proof in Appendix B.

be incorporated into a wide range of text-as-data studies, while maintaining statistical validity.

Before we proceed, we want to clarify that when we use the term "expert-coding," we do not assume expert-coding is perfect or makes no error. Rather, we only assume that expert-coding is a procedure that defines the benchmark against which the quality of the automated text annotation is evaluated, as done in the established supervised machine learning literature for decades (Hastie *et al.*, 2009; Grimmer and Stewart, 2013). Importantly, while domain "human" experts provide the benchmark in most social science applications so far, our proposed method can use any high-quality, expensive annotations as the benchmark, and DSL does not require that human experts provide the benchmark. For example, if users want to correct annotations by lower-quality smaller LLMs with more expensive annotations by larger LLMs as the benchmark, the same proposed methodology can be applied. Because our method can use any user-specified coding procedure as the benchmark, researchers can also naturally incorporate widely used strategies for handling errors and uncertainties in expert annotations (e.g., Benoit *et al.*, 2009; Hopkins and King, 2010; Mikhaylov *et al.*, 2012). We provide additional discussion in Section 6.

In the next section, we begin by discussing text annotations and clarify the promises and risks of using LLMs for automated text annotation. We then examine the problem of directly using predicted variables in downstream analyses (Section 3). We introduce our proposed method (Section 4) and illustrate its use using two empirical applications (Section 5). We provide practical guides (Section 6) before we conclude. Throughout the paper, we focus on text-as-data applications and prediction errors in automated annotation methods, but our proposed method is more general and can be used to handle any form of prediction error. In Section 7, we discuss the potential use of DSL for other areas of social science studies that rely on machine learning predictions, such as analyses of image, audio, and video.

Related Literature

This paper draws upon the foundational literature on double/debiased machine learning and doubly-robust estimation for missing data and causal inference (Robins *et al.*, 1994; Robins and Rotnitzky, 1995; Rotnitzky and Vansteelandt, 2014; Chernozhukov *et al.*, 2018). Like these papers, we exploit the influence function to derive debiased estimators. Our paper contributes to the growing literature on the use of predicted variables in statistical analyses. A number of papers develop methods for specific scenarios by making assumptions about the underlying data-generating process and prediction errors (e.g., Wang *et al.*, 2020; Fong and Tyler, 2021; Zhang, 2021; Knox *et al.*, 2022). In contrast to these papers, we only assume that researchers control the sampling process for expert annotations, and we do not make any assumption about the nature of prediction errors, which is particularly difficult to justify in applications of LLMs.

Our paper is most closely related to recent methods that build on the doubly robust estimation to deal with predicted variables (e.g., Angelopoulos *et al.*, 2023; Egami *et al.*, 2023; Mozer and Miratrix, 2023). These papers cover cases of text-based outcome variables but not text-based independent variables. By deriving a more general result, we cover cases where any subset of the outcome and independent variables are text-based and accommodate a much wider range of downstream analyses. This methodological generalization is fundamental because about 45% of applications use text-based variables as independent variables. In addition, we make practical contributions by providing new statistical software and clarifying detailed guides using two empirical applications. Katsumata and Yamauchi (2023) also develop a framework for using predicted variables while building on a different framework of control variates (Chen and Chen, 2000). We provide more technical discussions in Appendix A.

2 Automated Text Annotation

One of the most fundamental steps in many text-as-data research projects is to annotate documents. Over the last decade, scholars have used automated annotation methods to facilitate this time-consuming step by training machines to mimic expert-coding. In this section, we discuss how researchers can use the recent advances in LLMs for a wide range of text annotation tasks. We then clarify the potential risks of using LLM annotations, which motivates our main methodological contributions in Section 3 and Section 4.

2.1 Large Language Models as Text Classifier

Large language models are hard to define sharply, but they have undeniably become a central part of the natural language processing literature (Jurafsky and Martin, 2024). What distinguishes large language models from previous approaches is that they are trained on large collections of unlabeled data, gaining a proficiency in language that can be extended to downstream tasks with relatively little in-domain labeled data (Brown *et al.*, 2020). While the previous iteration of models captured this information in feature representations called static word embeddings (Mikolov *et al.*, 2013; Rodriguez and Spirling, 2022), the current generation of models has learned a generative model of language. This allows social scientists not only to generate new texts but also to use language itself (such as codebook directions) as the interface to get the desired functionality from the model.

An increasing number of social scientists use LLMs as automated text classifiers: researchers simply describe the annotation task in natural language instructions, and the LLM generates text labels by predicting the most appropriate text to follow such a request. For example, scholars have used LLMs to annotate sentiments, ideology, topics, hate speech, attitudes toward

immigrants, and support for a war, among others (Bommasani *et al.*, 2021; Ornstein *et al.*, 2022; Gilardi *et al.*, 2023; Ollion *et al.*, 2023; Pangakis *et al.*, 2023; Ziems *et al.*, 2023). See a wide range of examples we summarize in Appendix F.

2.1.1 How to Use LLMs as Text Classifiers

To illustrate this exciting potential, we use Fowler *et al.* (2021) as an example. The text annotation task here is to code the tones of ads into three categories ("Attack", "Contrast", and "Promote"). In the codebook developed in the well-known Wesleyan Media Project and used in Fowler *et al.* (2021), the tone of ads is defined as an answer to the following question. "In your judgment, is the primary purpose of the ad text to promote a specific candidate, attack a candidate, or contrast the candidates?" Instead of providing the codebook to trained expert coders, we can supply the same codebook to LLMs (see Figure 2-(a)) by first describing the codebook, then supplying Text to be classified, and finally prompt the LLM to Answer. In this example, when we use GPT 4, it understands the codebook and annotates a given document correctly as "Attack" (a response from the LLM is in a gray box).

More generally, text classification by LLMs can be performed in two steps. First, researchers need to choose which LLM to use. For example, famous commercial models like GPT 4 and GPT 3.5 are currently popular and famous, but researchers can also use open-source LLMs like Llama 2. We note that our discussion is general and applicable to any LLMs, including those developed in the future. In the second step, researchers decide on a *prompt*, i.e., a codebook and an instruction about a given annotation task. Many papers show that recent LLMs can perform a wide range of text annotation tasks by simply changing this prompt (e.g., Bommasani *et al.*, 2021; Ornstein *et al.*, 2022; Gilardi *et al.*, 2023; Ziems *et al.*, 2023). See Appendix F for examples of different types of prompts used in a wide range of social science annotation tasks. Researchers can also provide some examples (several pairs of texts and labels), also known as few-shot learning or in-context learning, to improve the prediction accuracy.[2] In Figure 2-(b), while we keep the codebook, a text to be classified, and an answer box (parts in boldface), we added three pairs of texts and answers as examples in the middle (parts in a non-bold face). The biggest benefit of using LLM annotations is that researchers can finish this automated text annotation step within a day for most applications, which is even faster than the classical supervised ML approach.

[2]How to choose examples for few-shot learning is an active area of research in natural language processing. In practice, we recommend including illustrative examples that researchers would include when creating a codebook of interest and training human coders.

In your judgment, is the primary purpose of the ad text to promote a specific candidate, attack a candidate, or contrast the candidates? Answer either "contrast", "promote", or "attack".

Text: """In 47 years, Mike Madigan hasn't been able to fix Illinois - and with JB Pritzker as his rubber stamp, there will only be higher taxes and more corruption. """

Answer:

attack

(a) Zero-shot learning (no exemplar)

In your judgment, is the primary purpose of the ad text to promote a specific candidate, attack a candidate, or contrast the candidates? Answer either "contrast", "promote", or "attack".

Text: """ On Tuesday, Nevada voters will choose between a local problem solver and a con man who funneled money from a children's charity into a failed political campaign. It's no wonder voters have rejected Danny Tarkanian 5 times."""

Answer: contrast

Text: """Donald Trump wants to roll back women's choice through the Supreme Court. I'll protect women's health care, and fully fund Planned Parenthood, in Florida. """

Answer: promote

Text:"""Lt Governor Dan Patrick cut public education funding by over five billion dollars, cut more than ten thousand teaching positions, and cut support for pre-K. With fewer teachers and larger class sizes, Dan Patrick won't let teachers teach and students learn. We need new leadership in Texas - vote Mike Collier for Lt Governor. """

Answer: attack

Text: """In 47 years, Mike Madigan hasn't been able to fix Illinois - and with JB Pritzker as his rubber stamp, there will only be higher taxes and more corruption. """

Answer:

attack

(b) Few-shot learning (exemplars in non-bold face)

Figure 2: **How to Use LLMs as Text Classifiers.**
Note: In (a) zero-shot learning, the basic prompt consists of a codebook (the first two lines), a text to be classified (the next two lines), and an answer box (the last line). A response from an LLM is represented in a gray box. In (b) few-shot learning, while keeping the basic components (parts in a bold face), users can add examples (parts in a non-bold face) in the middle.

2.1.2 Empirical Illustration of LLM Annotation

While the idea of using LLMs for text classification sounds promising, does it work in practice? In this section, we use two empirical applications of ours and the literature review to empirically illustrate the performance in a wide range of settings, which clarifies the promise and challenges.

Our first application is based on Fowler *et al.* (2021). In particular, we use all the 13,040

ads that are expert-coded by the original authors and examine the prediction accuracy of LLMs classifying the tone of ads. The second application is based on Pan and Chen (2018). We use their expert-coded $1,412$ citizen complaints to evaluate how well LLMs can classify whether each complaint accuses of wrongdoing by prefecture-level officials in China.[3]

Panels (a) and (b) in Figure 3 report F1 scores[4] for six versions of LLMs: GPT 4, GPT 3.5, and Llama 2 with zero-shot and few-shot learning. We provide the exact implementation details in Appendix G and I. Several points are worth noting. First, most of the LLMs can achieve F1 scores about $75 \sim 90\%$. This is promising and surprising given that these LLMs were *not* trained for these text annotation tasks, and LLMs were only given the codebook and a couple of examples (in the case of few-shot learning).

Second, the prediction performance varies across models and applications. In these two applications, F1 scores range from 48% to 95%. To further illustrate this wide variation in prediction performance, we also analyze a diverse set of empirical validation studies. In particular, based on a review paper by Ollion *et al.* (2023), we collected eight recent papers that examine the performance of LLM annotations in the social sciences, and we analyzed 113 text annotations tasks in total (see more details in Appendix F.2). We find that F-1 scores range from as low as 20% to more than 95%, and many tasks show about $70 \sim 80\%$ (see Panel (c) in Figure 3). This huge variation in prediction accuracy is a common feature of LLM annotations in the social sciences, and it is one of the key potential challenges of using LLMs, which we turn to next.

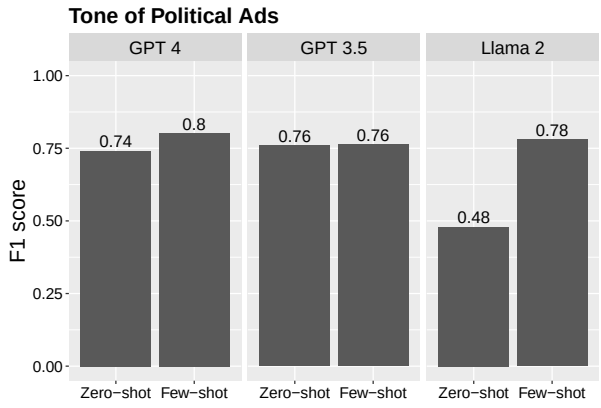
2.2 Potential Risks of LLM Annotation

As with any new technology, we have to carefully understand the potential risks as well as its promises. Even though LLMs have huge potential in many different text annotation tasks, they are, of course, not perfect and make *prediction errors*. While prediction errors can arise in any prediction method, including the classical supervised ML method, prediction errors in LLM classification are particularly difficult to understand for many reasons.

First, as we saw in Section 2.1.2, the amount and direction of prediction errors in LLM clas-

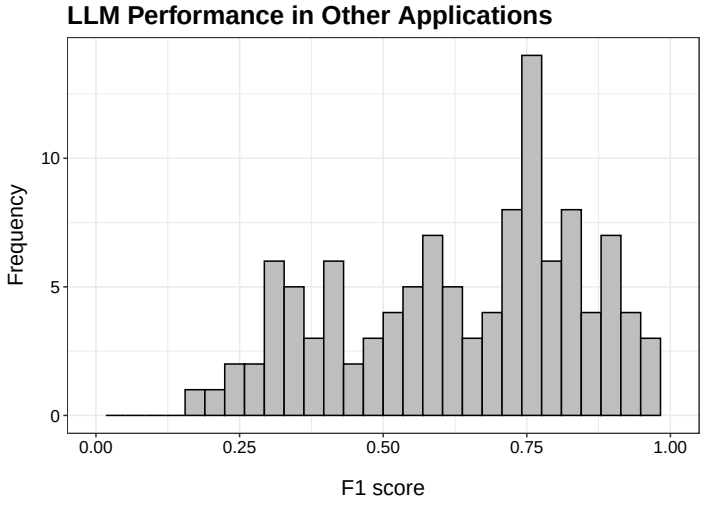
[3]See Section 5.2 for the details on Pan and Chen (2018).

[4]F1 score is a harmonic mean of the recall and precision, i.e., $\text{F1} = 2/(\text{recall}^{-1} + \text{precision}^{-1})$, and it is the most standard measure of prediction performance when categories are imbalanced. Classification accuracy is another popular measure, but it can be artificially high when categories are imbalanced. For readers more familiar with accuracy, we also report the figures based on accuracy in Appendix G and I, finding qualitatively similar results.



(a) Fowler *et al.* (2021)

(b) Pan and Chen (2018)



(c) 113 annotation tasks in 8 papers

Figure 3: **Prediction Performance of LLMs as Text Classifiers.**

sification can substantially vary depending on tasks, prompts, LLM models, and other unknown parameters in models. Most importantly, these variations in prediction errors are unknown and unpredictable to users. Recent LLMs are large-scale black box models that exhibit incredible language capacities in many different tasks for which LLMs were *not* trained (Bommasani *et al.*, 2021). We are constantly surprised by how well LLMs can perform many different tasks, but this also means that we do not fully understand when and why they might fail.

Second, social scientists often study complex, nuanced concepts expressed in documents, and thus, text annotation tasks in the social sciences are inherently difficult. Indeed, for many applications, even domain experts need several rounds of pilot coding and extensive discussions

to create, polish, and finalize a codebook to define how to label texts. Given the inherent difficulty of the task, prediction errors are inevitable.

Third, many recent LLMs lack the basic scientific requirement of transparency and replicability. In particular, many recent successful LLMs, e.g., GPTs, are proprietary methods, and as a result, users and research communities, in general, do not know the training data or exact training procedures that LLMs use (Spirling, 2023). Without access to the training data and training procedures, it is nearly impossible for users to understand the prediction errors that LLMs make.

Finally, a large number of papers have shown that LLMs also inherit unknown social, political, and racial biases contained in the unknown large-scale training data (see, e.g., Bender *et al.*, 2021). Prediction errors in LLMs can come not only from technical reasons but also from deeper reasons related to ethics and fairness, which further complicates the understanding of prediction errors.

In sum, it is extremely difficult or nearly impossible to fully understand how prediction errors occur in LLM classification. In Section 3, we clarify such prediction errors in LLM classification, and more generally in any automated text classification approach, can significantly bias downstream text analyses, if the errors are ignored when using predicted text labels.

3 Predicted Text Labels as Variables in Downstream Analyses

While document-level text classification is essential, text annotation is rarely the end goal of social science research. It is only the first step. Social scientists are often interested in using predicted text labels as variables in subsequent statistical analyses. Even though researchers often analyze predicted text-based variables as if they were observed without any error, this section clarifies that ignoring such *prediction errors*[5] in the first step of text annotation, even if the errors are small, leads to substantial bias, invalid confidence intervals, and wrong p-values in downstream statistical analyses of text-based variables. Biases from prediction errors exist even when the prediction accuracy in the text classification step is extremely high, e.g., above 90% or even at 95%. This is because prediction errors are not random—prediction errors are correlated with observed and unobserved variables we include in downstream analyses.

[5]In this paper, we define prediction errors to be the discrepancy between the text labels predicted by an automated text annotation method and the text labels that would have been assigned by expert-coding. As emphasized in Section 1, expert-coding is defined as a procedure that defines the benchmark against which the quality of the automated text annotation is evaluated, and DSL does not require that human experts provide the benchmark.

3.1 Setup and Quantity of Interest

Before describing the problem of prediction errors, we begin by defining statistical analyses we conduct after text annotation. Here, we focus on the most common regression analyses, and we discuss other common analyses, such as causal inference with texts, in Section 6.

Suppose researchers are interested in analyzing N documents. For each document i, we define Y_i as the outcome of interest and $\mathbf{X}_i$ as independent variables. Using general notation, we can define the quantity of interest as coefficients β of a generalized linear model.

$$\mathbb{E}(Y_i \mid \mathbf{X}_i) = f(\mathbf{X}_i^\top \beta) \tag{1}$$

where $f(\cdot)$ is an inverse of a canonical link function for the generalized linear model. This general setup incorporates a wide range of common statistical analyses, such as linear, logistic, multinomial logistic, Poisson, and linear fixed-effects regression, as well as the estimation of category proportions over time or across groups.[6] Importantly, we only view coefficients β as a low-dimensional summary, and thus, this paper does not assume the underlying data-generating process follows a specified parametric model (Lundberg *et al.*, 2021).

Researchers might also be interested in estimating the first differences or other quantities that are functions of coefficients rather than coefficients themselves (King *et al.*, 2000). Our proposed methods can be applied not only to coefficients but also to any function of coefficients. We provide such examples in Section 5.2.

3.2 Current Practice: Directly Using Predicted Labels as Variables

In text analyses, a subset of the outcome Y and independent variables $\mathbf{X}$ are based on some forms of text labels, and creating such text-based variables requires text annotation. When using automated text annotation methods to predict text labels, regardless of the exact choice, statistical analyses take the following steps in general. First, researchers check the accuracy of prediction against expert-coded data, e.g., using cross-validation. If the accuracy is "low,"[7] researchers retrain the model until it gets better (e.g., using different LLMs or ML models

[6]Our literature review of the ten political science journals finds that our setup covers common statistical models used in more than 91% of applications using text annotations: Linear regression (49% of applications), Logistic regression (21%), Category proportions over time or across groups (Subgroup means) (19%), and Poisson regression (2%).

[7]Scholars use different criteria for deciding how much is "low" and "high enough," but many scholars use $80 \sim 90\%$ as rough thresholds. In our literature review, the final prediction models researchers chose have the accuracy of 89.8% and the F1 score of 84.6%, on average.

and adding more informative predictors). Then, once the accuracy becomes "high enough," they now use predicted text labels directly in downstream analyses as if those variables were directly observed and not predicted. The idea is that when the prediction accuracy is high, the prediction model sufficiently mimics expert-coding, and thus, prediction errors are small enough that they do not affect downstream analyses significantly.

More concretely, most researchers use one of the following two ways to use predicted variables in downstream text analyses. The first approach, which we call LLM-Only Estimation, is to use LLMs to predict text labels for every document (see Section 2 for different ways to improve LLM-prediction).

LLM-Only Estimation

Step 1: Predict text labels using LLMs for each document.

Step 2: Sample a subset of documents for expert-coding.

Step 3: Check the prediction accuracy using the expert-coded data. Repeat Step 1 until the prediction accuracy is high.

Step 4: Use LLM-predicted variables in downstream text analyses.

The second and more classical approach is to use the supervised machine learning model (e.g., random forest, lasso, and so on) to predict text labels. The main steps are essentially the same as those in the LLM-Only Estimation, and the only difference is the way in which researchers produce predictions (via LLMs or the supervised ML method estimated with the expert-coded data).

Classical Supervised Learning Estimation

Step 1: Sample a subset of documents for expert-coding.

Step 2: Train a supervised machine learning model with the expert-coded data.

Step 3: Check the prediction accuracy using the expert-coded data via cross-validation. Repeat Step 2 until the prediction accuracy is high.

Step 4: Use ML-predicted variables in downstream text analyses.

3.3 The Methodological Challenges of the Current Practice

Ignoring prediction errors in the text annotation step, even if the errors are small, leads to bias, invalid confidence intervals, and wrong p-values in the subsequent statistical analyses of text-based variables. This is because prediction errors are *not completely random*—prediction errors

are correlated with observed and unobserved variables we include in downstream analyses.[8] Even small prediction errors can bias downstream analyses in any direction by any amount. Because exactly the same problem applies to the LLM-only estimation and the classical supervised learning estimation, we do not distinguish them, and we discuss prediction errors in general.

To concretely illustrate the problem, we focus on a simple case where the outcome variable Y requires text annotation, and researchers want to regress Y on independent variables $\mathbf{X}$ to estimate coefficients β defined as,

$$\mathbb{E}(Y_i \mid \mathbf{X}_i) \;=\; \mathbf{X}_i^\top \beta. \tag{2}$$

Researchers can easily obtain the ordinary squares estimates of β when the outcome variable of interest Y is observed for every document. However, when Y requires text annotation and Y itself is not observed for each document, researchers instead regress the predicted outcome variable $\widehat{Y}$ on independent variables $\mathbf{X}$. This linear regression with the predicted outcome variable will lead to unbiased coefficient estimation when prediction error, $e_i = \widehat{Y}_i - Y_i$, is zero on average across all different combinations of $\mathbf{X}$.

$$\mathbb{E}(e_i \mid \mathbf{X}_i) \;=\; 0. \tag{3}$$

Even though this expression might seem similar to the standard exogeneity assumption, it turns out that this condition implies much stronger assumptions. Formally, researchers can ignore prediction errors only when prediction errors are completely random, i.e., prediction errors are not affected by the independent variable, the outcome variable, or any unobserved confounder. Unfortunately, this condition is untenable in almost all social science applications. While we focused on one setting where Y is text-based, similar stringent conditions are required when other types of variables (e.g., independent variables) are text-based. We offer additional discussions and the general bias formula in Appendix B.

The literature has explored several approaches to address this problem. First, researchers might consider incorporating bootstrap to capture the uncertainty of the text-prediction step, with the hope of addressing prediction errors. Unfortunately, the central problem of prediction errors is the bias correlated with observed and unobserved variables in downstream analyses and how fast the bias goes to zero asymptotically. Thus, simply adding bootstrap to the current practice cannot eliminate this problem. Second, researchers might make a stringent modeling

[8]We note that this problem of ignoring prediction errors is common not only in text-as-data applications but also in many other social science applications (see similar discussions, e.g., Fong and Tyler, 2021; Zhang, 2021; Knox *et al.*, 2022; Katsumata and Yamauchi, 2023).

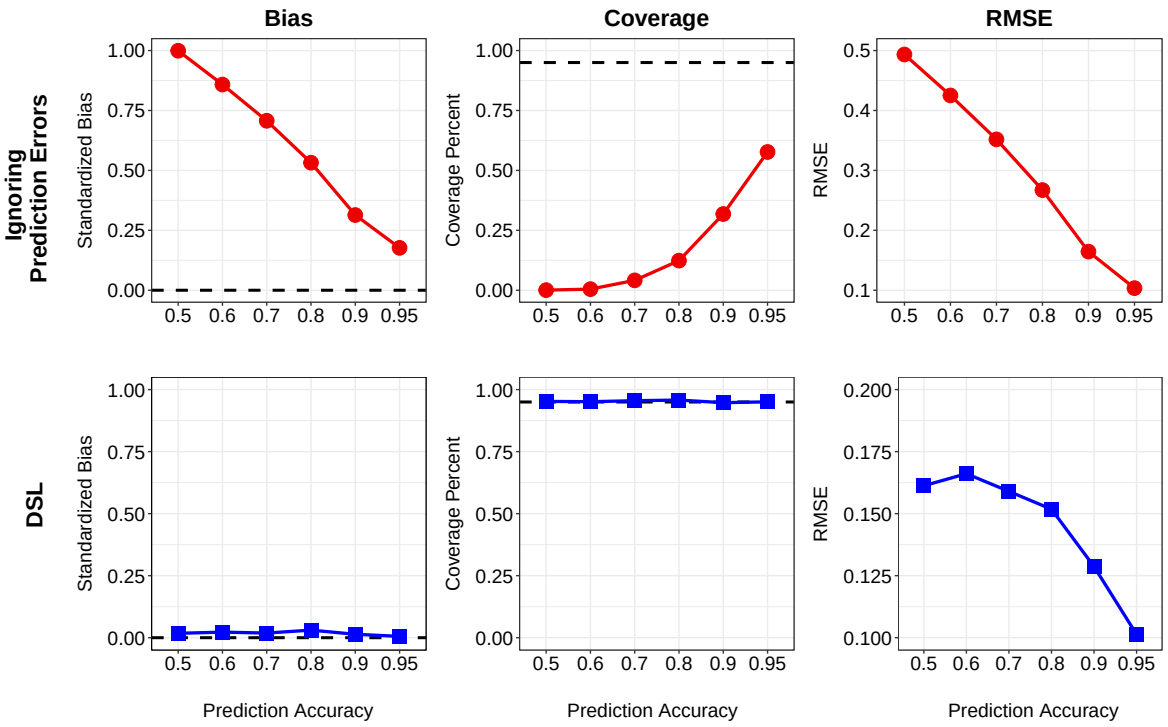


Figure 4: **Ignoring Prediction Errors Lead to Bias and Invalid Confidence Intervals.** *Note*: The first row shows the results of estimators ignoring prediction errors, and the second row previews the results of DSL introduced in the next section. The first, second, and third columns represent bias, coverage rates of 95% confidence intervals, and root mean squared errors (RMSE), respectively. X-axis shows varying prediction accuracy of the underlying automated text annotation method.

assumption to explicitly model $\mathbb{E}(e_i \mid \mathbf{X}_i)$ (i.e., how prediction errors vary with $\mathbf{X}$) (e.g., Wang *et al.*, 2020). This type of method works only when the model for prediction errors is correct and the outcome variable is text-based, while they cannot produce valid confidence intervals or p-values even under the correct model of prediction errors. Our proposed approach will not make any modeling assumptions about how prediction errors occur.

3.4 Simulation Study

We now use a simulation study to illustrate the problem of ignoring prediction errors. While our method is general, we consider logistic regression with the text-based outcome as a common example. We vary the prediction accuracy of the underlying automated annotation methods—from as low as 50% to as high as 95%—and evaluate how ignoring prediction errors affects downstream regression analyses. We detail the data generation process in Appendix D.

The first column in Figure 4 shows the average bias across coefficients standardized by the true coefficients. When prediction errors are ignored (the first row), bias decreases as the accuracy of the underlying prediction method goes up. However, bias can be as large as 30%

and 18% of the true coefficients even when the underlying prediction accuracy is 90% and 95%. The second column in Figure 4 shows the coverage rate of the 95% confidence intervals (the probability of reported confidence intervals covering the true coefficients), and the coverage rates should be at least 95% if a given estimation method is statistically valid. Unfortunately, when prediction errors are ignored, the coverage rate of 95% confidence intervals is as low as 32% and 58% even when the underlying prediction accuracy is 90% and 95%. Bias and coverage rates are, of course, much worse when the accuracy of the prediction method is about $80 \sim 90\%$ or lower, as we see in most applications. These results demonstrate that researchers cannot ignore prediction errors even when the underlying prediction method has excellent prediction accuracy.

The second row of Figure 4 previews the results of the proposed DSL. As we show in the next section, DSL is theoretically guaranteed to have asymptotically unbiased estimates and valid confidence intervals, regardless of the accuracy of the underlying prediction method (see the first and second columns in the second row). When the underlying prediction method becomes more accurate, DSL also gets more accurate and has smaller standard errors, which is shown by the reduction in root mean squared error (RMSE) (see the third column in Figure 4). When the accuracy of the underlying prediction method goes up from 50% to 95%, RMSE reduces from 0.16 to 0.10, which is equivalent to 37.5% reduction in standard errors.

4 Design-based Supervised Learning

We propose a general method, which we call *design-based supervised learning* (DSL), to use predicted variables in downstream analyses without introducing bias from prediction errors. This general framework allows researchers to use LLM annotations or text labels predicted by ML methods in downstream text analyses while maintaining statistical validity.

4.1 Overview

We first provide an overview of the proposed DSL method. While the proposed framework can accommodate various versions of implementations, we first focus on the basic version and then discuss other extensions later.

Design-based Supervised Learning Estimator (DSL)

Step 1: Predict text labels using LLMs for each document.

Step 2: Sample a subset of documents for expert-coding.

Step 3: Train an ML model to improve LLM-prediction with the expert-coded data.

Step 4: Combine expert-coded labels and predicted variables in the DSL regression.

Importantly, most steps (Steps 1–3) are similar to existing approaches and thus are already familiar to applied researchers. In the first step, like the LLM-only estimation, we predict text labels using LLMs. In the second step, like the existing approaches, we sample a subset of documents for expert-based coding. In the third step, researchers can use expert-coded documents as the training data and train a supervised machine learning model where we predict the expert-coded labels with predictors that include LLM annotations generated in Step 1 and any other variables that are predictive (e.g., term-document matrices).[9] This step is similar to Step 2 in the classical supervised learning estimation, and the only difference is that users can also incorporate LLM annotations as predictors for the expert-coded labels. The fourth step is an essential defining feature of DSL. We combine expert-based coding and predicted variables in a tailored fashion, which we describe in detail in the next sections.

4.2 Assumption: Design-based Sampling

Before we describe the details of the DSL regression estimator, we clarify the central assumption behind DSL. In particular, we require that researchers know the process through which documents are sampled for expert-coding. Formally, we make the following assumption by defining π_i to be the probability of sampling document i for expert-coding.

Assumption 1 (Design-based Sampling for Expert-Coding)
The probability of sampling documents for expert-coding π_i is known to researchers, and π_i is larger than zero for every document.

Assumption 1 holds when the researchers can choose which documents to be coded by experts. For example, if the researchers have 10000 documents and sample 100 of them to expert-annotate at random, $\pi_i = \frac{100}{10000} = .01$ for all documents. Here, the sampling probability for each document is decided by the researchers and is greater than zero. We also allow more complex stratified or block sampling schemes (i.e., change the sampling probability of documents based on document-level observed covariates) and can cover any case where the sampling probability π_i depends on the LLM annotation, document-level covariates, independent variables, or the outcome variable, as long as π_i is known. This generality is important because researchers might want to over-sample documents that are difficult to annotate. For example, in Fowler *et al.* (2021), if researchers a priori expect that longer political ads are more difficult to annotate, they can change the sampling probability based on the length of the ads. In Section 5.1.3, we discuss how to determine the required number of expert annotations in each application.

[9]Researchers can also skip this third step, and doing so is equivalent to using the identity function for predicting expert-coded labels with LLM labels.

Importantly, Assumption 1 does rule out some applications, and two are worth noting: (1) Researchers use external coding (rather than their own expert-coding) to measure text-based variables of interest, and it is unknown why only a subset of documents were coded. For example, Hager and Hilbig (2020) analyze speech documents published by the German government. For 47% of all documents, the topic of the speech is assigned by the German government, but the rest of the documents are published without an explicit topic assignment. In this case, researchers do not decide which document to be sampled for the expert-labeling, and thus, the assumption is violated. (2) Another scenario occurs when researchers need to analyze documents in real-time as soon as they obtain text data, e.g., making polling predictions based on social media posts on election day. In such cases, it might be inevitable to use expert-coded documents from the past, but in this example, social media posts on election day have the probability of being sampled for expert-coding is zero, and thus, the assumption is violated. However, if researchers have time to sample a subset of social media posts on election day for expert-coding, Assumption 1 holds because now every document they analyze has the probability of being labeled greater than zero. Therefore, when researchers need to collect documents over time, researchers can guarantee Assumption 1 by making sure to sample documents for expert-coding from each time period they analyze.

While we do not cover all text-as-data scenarios, our approach covers the vast majority of social science research applications where researchers need to annotate a corpus of documents that are available in total before analyzing data. Even more importantly, Assumption 1 can be guaranteed by research design alone. This is akin to how randomized experiments can guarantee the absence of unmeasured confounding by randomizing treatments. Our assumption is transparent and easy to justify. This is the reason why our method is named *design-based* supervised learning.

4.2.1 Assumptions We Do Not Make

Understanding assumptions we do not make is as important as understanding the assumptions we do make. In particular, we make *no* assumptions about prediction errors and allow for arbitrary prediction errors. Researchers do not need to assume how prediction errors arise in LLMs or ML prediction. We do not need to assume LLM annotations are unbiased and accurate, or the fitted supervised ML model is correctly specified, unbiased, and accurate. In practice, this means that researchers can use predictions from LLMs or supervised ML without worrying about their prediction errors or inherent biases.

This is in sharp contrast to existing alternatives. Both the LLM-only estimation and the classical supervised learning approach have to assume prediction errors are completely random.

This assumption is often severely violated in practice, and most importantly, researchers cannot guarantee this assumption by research design.

4.3 DSL Regression

We now examine how the proposed DSL estimator can incorporate predicted variables without introducing bias under Assumption 1. To provide intuition, we start with a simple case and generalize it step by step.

4.3.1 Building Intuition with Estimation of Category Proportion

Suppose researchers are interested in estimating the proportion of documents belonging to a particular category, e.g., the proportion of political ads attacking opponents. Define $Y_i \in \{0, 1\}$ to denote whether a given political ad attacks opponents. When using the LLM-only estimation or the classical supervised ML methods, users would first predict whether each ad attacks opponents $\widehat{Y}_i$ and then average it over ads to estimate the proportion of attacking ads.

In contrast, DSL uses the following design-adjusted outcome.

$$\widetilde{Y}_i = \underbrace{\widehat{Y}_i}_{\substack{\text{Predicted} \\ \text{Outcome}}} - \underbrace{\frac{R_i}{\pi_i}(\widehat{Y}_i - Y_i)}_{\text{Bias-Correction Term}}, \tag{4}$$

where Y_i is the outcome of interest coded by experts, R_i is a binary variable taking 1 if document i is expert-coded and 0 otherwise, and π_i (defined in Section 4.2) is the probability of sampling document i for expert-coding.[10] This estimator has deep theoretical connections to doubly robust estimation in the causal inference literature (Robins *et al.*, 1994; Chernozhukov *et al.*, 2018), and the bias-correction term is similar to the one in the augmented inverse probability weighting estimator.

In the most simple case of random sampling with equal probabilities ($\pi = n/N$ where n is the number of expert-coded documents and N is the total number of documents), the DSL

[10]The design-adjusted outcome is equal to $\widehat{Y}_i$ when $R_i = 0$ and is equal to $\widehat{Y}_i - (\widehat{Y}_i - Y_i)/\pi_i$ when $R_i = 1$. It might be counter-intuitive to change the outcome for documents $R_i = 1$ when the outcome of interest Y_i is observed. Importantly, the goal is not to correct the prediction error at each document level, but at the level of quantities of interest (average in this case, as we show in equation (5) below). In fact, we can only estimate the prediction error from documents with $R_i = 1$, which is used to correct outcomes for documents with $R_i = 0$ on average. More generally, it is usually impossible but also unnecessary to bias-correct outcomes at each document level (Hopkins and King, 2010).

estimator becomes simple.

$$\frac{1}{N}\sum_{i=1}^{N}\widetilde{Y}_i = \underbrace{\frac{1}{N}\sum_{i=1}^{N}\widehat{Y}_i}_{\substack{\text{Mean of}\\ \text{Predicted Outcomes}}} - \left(\underbrace{\frac{1}{n}\sum_{i:R_i=1}\widehat{Y}_i}_{\substack{\text{Mean of}\\ \text{Predicted Outcomes}\\ \text{in Labeled Data}}} - \underbrace{\frac{1}{n}\sum_{i:R_i=1}Y_i}_{\substack{\text{Mean of}\\ \text{Observed Outcomes}\\ \text{in Labeled Data}}} \right) \tag{5}$$

The main idea is to use the expert-coded data to estimate bias from prediction errors (the difference between the second and third terms on the right-hand side), which we subtract from the conventional estimator that relies only on predicted labels (the first term on the right-hand side). For example, suppose users have $N = 10,000$ ads and randomly sampled $n = 100$ ads for expert annotation. The first term on the right-hand side estimates the proportion of attacking ads by averaging the predicted labels in all $N = 10,000$ documents (suppose it is 20%). Because this first term suffers from bias due to prediction errors, the second and third terms on the right-hand side estimate the bias to be subtracted. In particular, the second term estimates the proportion of attacking ads by averaging the predicted labels in $n = 100$ expert-coded documents (suppose it is 18%), and the third term estimates the proportion of attacking ads by averaging the expert-coded labels in $n = 100$ expert-coded documents (suppose it is 10%). Because the expert-coded data are randomly sampled, we can estimate the bias by taking the difference between the second and third terms, $18 - 10 = 8\%$, which we subtract from the first term (i.e., the original naive estimator that only uses predicted labels). In this simple example, the DSL estimate is $20 - (18 - 10) = 12\%$.

4.3.2 DSL Linear Regression

While the previous section focuses on the estimation of category proportions under simple random sampling, the DSL framework can be applied to linear regression as well (under any user-specified sampling strategy). Specifically, the DSL linear regression simply needs to regress the design-adjusted outcome $\widetilde{Y}_i$ (equation (4)) on independent variables $\mathbf{X}_i$. Formally, the DSL regression estimator can be written as

$$\widehat{\beta}_{DSL} = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\widetilde{\mathbf{Y}} \tag{6}$$

where $\widetilde{\mathbf{Y}} = (\widetilde{Y}_1, \ldots, \widetilde{Y}_N)$ and ith row of matrix $\mathbf{X}$ is $\mathbf{X}_i$. Under Assumption 1, the DSL estimator is consistent and asymptotically normal when we use cross-fitting (Chernozhukov *et al.*, 2018) to generate predictions.[11] The corresponding confidence intervals can be constructed with the

[11]This step corresponds to Step 3 in the DSL workflow (see Section 4.1). More technical details are in Appendix B.

usual standard error formula. Valid statistical inference is possible because the design-adjusted outcomes correct the prediction error on average across all different combinations of $\mathbf{X}$.

$$\mathbb{E}(\widetilde{Y}_i - Y_i \mid \mathbf{X}_i) \;=\; 0. \tag{7}$$

We provide proof of these theoretical properties in Appendix B.

Importantly, the DSL estimator corrects bias only under Assumption 1 without making any assumption about prediction errors in $\widehat{Y}_i$. In practice, this means that researchers can use any LLMs and supervised ML methods to construct the predicted outcomes, even if LLMs and ML methods contain arbitrary prediction errors. While the DSL regression allows for any prediction error, it becomes more accurate (i.e., standard errors are smaller, and confidence intervals are narrower) when the prediction errors are smaller. Therefore, researchers can exploit the recent advances in LLMs and supervised ML methods without sacrificing valid statistical inference, while reducing standard errors as the prediction step becomes more accurate. We illustrated these desirable properties in simulation studies (Section 3.4) and will show more results in empirical applications (Section 5).

4.3.3 Generalization of DSL

Finally, we emphasize that the same general idea applies to a large class of generalized linear models we introduced in Section 3.1 (e.g., logistic, multinomial-logistic, Poisson, and linear fixed-effects regression) and to general cases where any subset of the outcome variable and independent variables are text-based. The only but crucial difference is that we have to bias-correct not the outcome variable itself (as we did in equation (4)) but the underlying moment function. In general, define $m(Y_i, X_i; \beta)$ to be the subgradient of the convex optimization problem defining a generalized linear model. Then, the moment function for the DSL estimator can be written as

$$m(\widehat{Y}_i, \widehat{X}_i; \beta) - \frac{R_i}{\pi_i}\Big(m(\widehat{Y}_i, \widehat{X}_i; \beta) - m(Y_i, X_i; \beta)\Big) \tag{8}$$

where $\widehat{Y}_i$ and $\widehat{X}_i$ are predictions for the outcome Y_i and independent variables X_i. We provide technical details in Appendix B.

5 Empirical Applications

We now use empirical applications to illustrate how to apply DSL in a wide range of settings. The first application based on Fowler *et al.* (2021) considers settings where the outcome variable is text-based, while the second based on Pan and Chen (2018) examines cases where the independent variables are text-based.

5.1 Text as Outcome: Fowler *et al.* (2021)

Fowler *et al.* (2021) examine how and whether the tone of political ads varies across Facebook and television. To test this question, after annotating the tone of ads, the original authors run a linear fixed effects model that regresses the tone of ads on the main independent variable indicating whether a given ad is from Facebook or television, while including candidate-fixed-effects.[12]

In this section, we conduct empirical validation using the expert-coded political ads from Fowler *et al.* (2021). In particular, we use $13,040$ expert-coded political ads as the target population of documents ($N = 13,040$). But we pretend that we can only sample $n = 1000$ documents for expert-coding (less than 8% of the original number of expert-coding) and use automated text annotations methods to predict the tone of ads for the remaining $12,040$ documents. We then assess how well DSL and other methods, which are based on 1000 expert-coded documents with predicted $12,040$ documents, can recover the benchmark estimates, which use the entire $13,040$ expert-coded documents. By doing so, we can illustrate the use of DSL step by step, while testing how DSL and other methods perform when the underlying automated text annotation methods have non-random prediction errors.

5.1.1 Setup

DSL requires four simple steps. First, we generate LLM annotations for the entire population of documents. As we discussed in Section 2, we here consider six versions: GPT 4, GPT 3.5, and Llama 2 with zero-shot and few-shot learning. In the second step, we randomly sample 1000 documents for expert-coding (we will discuss how to determine the number of expert-coding in Section 5.1.3).[13] In the third step, using the expert-coded data, we further improve LLM predictions by cross-fitting the generalized random forest (Athey *et al.*, 2019) to predict the expert-coded labels with LLM annotations produced in the first step. Finally, we combine expert-coded labels and predicted labels in the DSL linear fixed-effects regression, where we regress the design-adjusted outcome on the same independent variables used in the original paper, that is, the Facebook dummy variables and candidate-fixed-effects. The main quantity of interest in the original paper is the coefficient of the Facebook dummy variable. Our companion

[12]We follow the original paper's definition, and the tone of ads is computed by the weighted average of the tone of ads in a given pair of a candidate and a platform where weights are proportional to the expenditure of a given ad.

[13]In this empirical validation, we rely on expert-coding from the original authors, so we simply reveal expert-coding for sampled documents.

R package `dsl` can implement the third and fourth steps with one function, while taking LLM annotations (Step 1) and expert-coding (Step 2) as inputs from users.

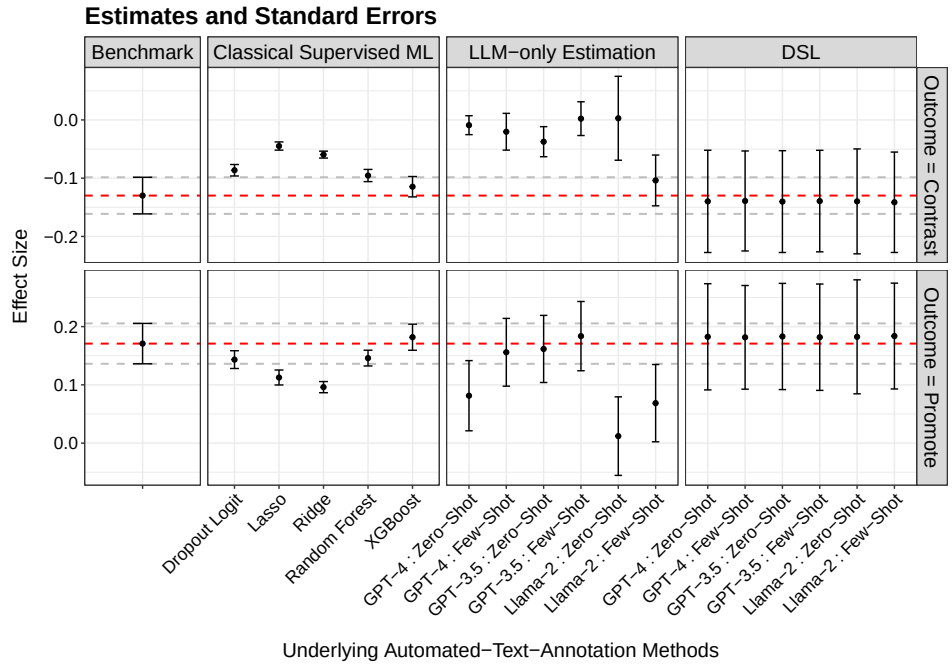
We compare DSL against the classical supervised learning approach and the LLM-only estimation. For the classical supervised learning approach, we examine five widely used supervised ML methods: drop-out regularized logistic regression (used in the original paper), lasso, ridge, random forest, and XGBoost. We use a set of predictors used in the original paper (more than 7000 variables) that are constructed by processing the ad's text, images, video, and audio. For the LLM-only estimation, we consider the same six versions of LLM annotations. We expect that these existing approaches can provide unbiased estimates with valid confidence intervals, only when prediction errors are completely random, while DSL provides valid statistical guarantees even with arbitrary prediction errors.

5.1.2 Results

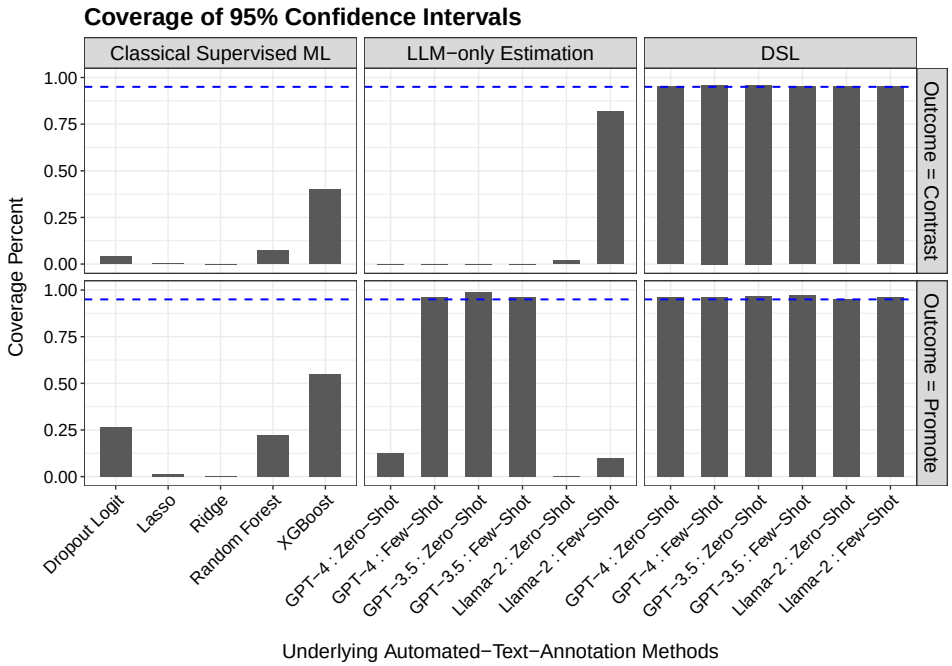
The results are reported in Figure 5. Due to the space constraints, we focus on two outcomes "Contrast" and "Promote" in the main text as these two outcomes have the highest and lowest LLM performances, while reporting the results on "Attack" in Appendix H. In Figure 5-(a), the leftmost column reports the benchmark estimates based on the entire sample of $13,040$ expert-coded documents. The remaining columns show point estimates and standard errors of different methods, and the X-axis shows the underlying automated text annotation method. Figure 5-(b) reports coverage rates of the 95% confidence intervals.[14] If a method can produce valid confidence intervals, coverage rates of its 95% confidence interval should be at least 95%.

We now discuss each method in order. First, we look at the LLM-only estimation. Figure 5-(a) shows that point estimates have large variations depending on the underlying LLM method used for automated text annotations. This is because the LLM-only estimation ignores differential prediction errors that each LLM method makes, and as a result, estimates of the quantity of interest are biased. Some methods have small biases for one outcome (e.g., Few-shot learning with Llama-2 when the outcome is "Contrast"), but no method has small biases across both outcomes. Crucially, in the real-world application where researchers cannot observe the "Benchmark" estimate (unless they expert-code every single document), it is impossible for users to decide which estimates to report and trust. Indeed, depending on which LLMs users choose, they could reach statistically and substantively different results. For example, when researchers use GPT-4 with Few-shot learning, they might conclude the effect on "Promote" is

[14] We compute this as the probability of confidence intervals covering the benchmark estimate over 500 repeated sampling of the population of documents and expert-coding.



(a)



(b)

Figure 5: **Comparisons of DSL and Existing Approaches using Fowler *et al.* (2021).** *Note*: In Panel (a), red dotted lines represent point estimates of the "Benchmark" estimates, and gray dotted lines represent their 95% confidence intervals. To show the average performance across random sampling of expert-coding, we report the average point estimates and standard errors across 500 repeated sampling. In Panel (b), blue dotted lines represent 95%.

substantively and statistically indistinguishable from zero, whereas they would find a statistically significant, negative effect when they use Llama-2 with Few-shot learning. Figure 5-(b) shows that confidence intervals based on the LLM-only estimation are, in general, invalid (i.e., cannot cover the benchmark estimates with 95%) due to large biases. In sum, this large variation in estimates is the fundamental problem of ignoring prediction errors: researchers can get statistically and substantively different estimates depending on the choice of LLMs, and there is no way to decide which estimate is the most credible. More generally, the LLM-only estimation has no statistical guarantees in the presence of non-random prediction errors, i.e., some methods had good point estimates for one outcome in this application, but that was a statistical coincidence.

Next, we look at the classical supervised ML method, which has the same problem of ignoring prediction errors. Just like the LLM-only estimation, point estimates have large variations depending on the underlying ML method used for automated text annotation. Importantly, this variation exists even though each supervised ML method has roughly the same prediction performance. Interestingly, estimates from XGBoost have small biases for both outcomes. However, getting a good point estimate is not sufficient in social science analyses, and it is crucial to report a valid uncertainty measure. As we discussed in Section 3, unfortunately, the classical supervised ML method underestimates standard errors, and as a result, it has invalid confidence intervals. Figure 5-(b) shows that, even for "XGBoost" that have good point estimates, the 95% confidence intervals only cover the true effect about 50%, which in practice means that reported standard errors are severely underestimated and reported p-values are wrong. As discussed in Section 3, these problems cannot be solved by simply adding bootstrap.[15]

Finally, we discuss how the proposed DSL overcomes the shortcomings of the existing methods. Several points are worth emphasizing. First, unlike the existing methods, point estimates of DSL are stable regardless of the underlying automated text annotation methods users choose, and they all have small biases. This property is fundamental in empirical research because researchers do not need to worry that statistical and substantive conclusions might change if they happen to use different LLMs. Therefore, researchers can justify the use of LLMs without assuming that predictions from LLMs are unbiased or accurate. Here, we focus on prediction based on LLMs, but the DSL regression can also be used to correct biases when the automated text annotation is done with the classical supervised ML method. Second, as we see in Fig-

[15]Researchers might wonder about extremely small confidence intervals for the supervised ML method. This is due to both regularization bias and the failure to incorporate prediction uncertainty, which both lead to smaller invalid standard errors.

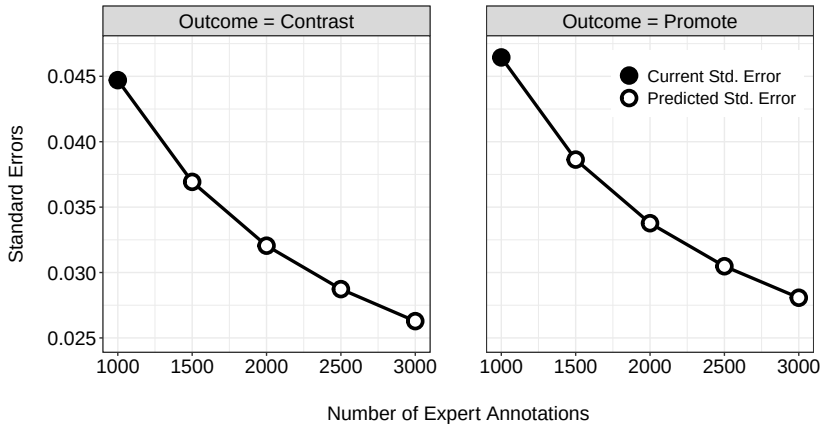


Figure 6: **Power Analysis to Determine the Required Number of Expert Annotations.** *Note*: Each panel reports the current standard errors (1000 expert annotated samples) and predicted standard errors for different numbers of expert annotations. The left and right panels consider DSL analyses when the outcome is "Contrast" and "Promote", respectively.

ure 5-(b), DSL gives valid standard errors and confidence intervals (i.e., reported confidence intervals have a coverage rate of 95%), unlike the existing methods that significantly underestimate the true uncertainty. Taken together, DSL provides stable, unbiased point estimates and valid confidence intervals regardless of which LLMs they use to automate annotations. This is because DSL explicitly takes into account prediction errors through the design-based sampling of expert-coding.

Researchers might wonder about the wider confidence intervals of DSL relative to other methods. First, DSL estimators rightly have larger standard errors because they properly take into account prediction errors. In contrast, by ignoring prediction errors, the confidence intervals of the existing methods are invalid and underestimate the true uncertainties. Indeed, falsely narrow confidence intervals around biased estimates are exactly what we should avoid: we do not want to be falsely confident about wrong estimates. Second, in practice, researchers can conduct a power analysis to decide the number of expert-coded documents necessary for reducing standard errors of DSL to a certain level, which we discuss next.

5.1.3 Power Analysis

The number of documents experts need to annotate depends on applications. To help researchers in each specific application, we develop a data-driven power analysis: after annotating a small number of documents, we can predict how many more documents researchers need to annotate

in order to achieve a user-specified size of standard error.[16] Figure 6 predicts how standard errors reduce as the number of expert annotations increases. For example, as in traditional power analysis, suppose researchers expected a coefficient of the Facebook dummy variable to be -0.08 when the outcome is "Contrast". To detect this effect size with sufficient statistical power, scholars ordinarily need standard errors smaller than 0.04. From this figure, researchers can predict that randomly sampling 500 additional documents for expert annotations will reduce the current standard errors from 0.045 to about 0.0375.

5.2 Text as Independent Variables: Pan and Chen (2018)

We now use Pan and Chen (2018) to consider settings where the independent variables are text-based. This application illustrates the general applicability of our proposed approach: unlike the previous application, documents of interest are written in Chinese, and the original authors use logistic regression as the downstream statistical model.

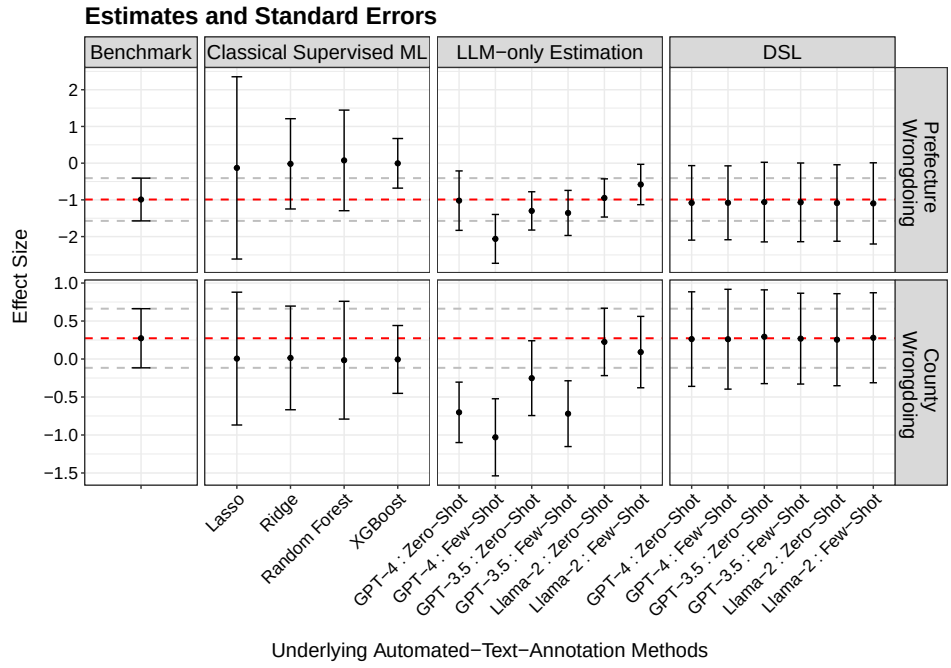
The key research question in this study asks whether Chinese officials systematically conceal complaints of corruption from upper-level authorities. To test this question, after annotating whether each citizen complaint accuses of prefecture-level or county-level wrongdoing (*Prefecture Wrongdoing* and *County Wrongdoing*), the original authors run a logistic regression that regresses the upward reporting (i.e., whether a given complaint is reported upward to provincial-level officials) on the aforementioned two independent variables (*Prefecture Wrongdoing* and *County Wrongdoing*) and other control variables.

In this section, we again check the performance of DSL and existing methods against the benchmark estimate based on the entire $1,412$ expert-coded complaints. We pretend that we can only sample $n = 500$ documents for expert-coding and use automated text annotation methods to predict *Prefecture Wrongdoing* and *County Wrongdoing* for the remaining documents. We then assess how well DSL and other methods can recover the benchmark estimates. While the implementation of each method is similar to the one we illustrate in the previous application, we provide all the details in Appendix J.

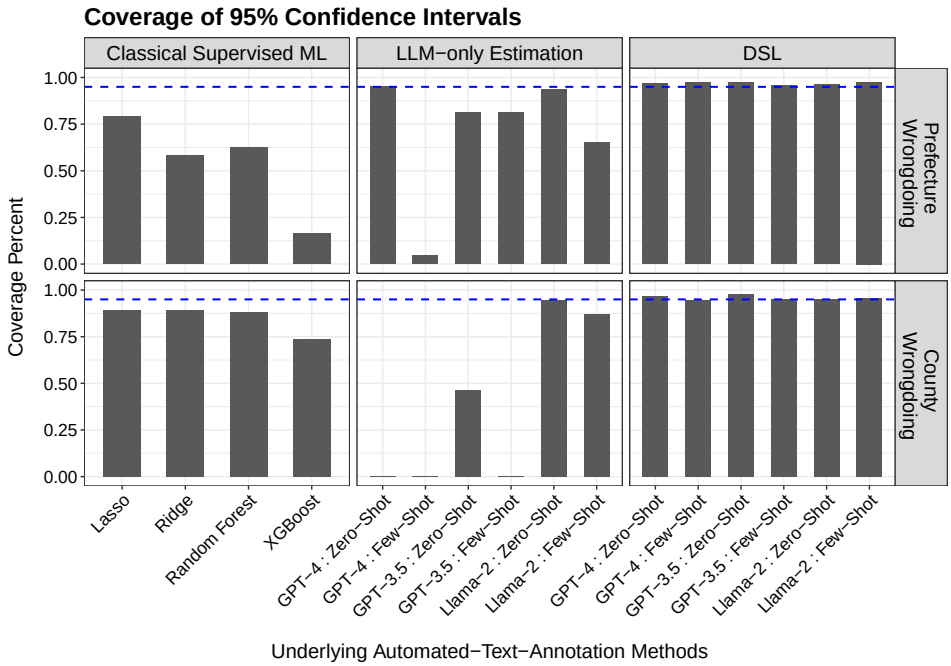
Figure 7 shows estimated coefficients of *Prefecture Wrongdoing* and *County Wrongdoing* as well as the coverage rates of their 95% confidence intervals.[17] As in the previous application, estimates from the LLM-only-estimation are biased, and importantly, substantive and statistical conclusions can flip depending on which LLMs users choose for automated text annotation. These variations exist even though the prediction accuracy of different LLMs is roughly similar

[16] Our `R` package `dsl` implements this power analysis with one function.

[17] Appendix J reports results based on the first differences, which reveals the same findings.



(a)



(b)

Figure 7: **Comparisons of DSL and Existing Approaches using Pan and Chen (2018).** *Note*: In Panel (a), red dotted lines represent point estimates of the "Benchmark" estimates, and gray dotted lines represent their 95% confidence intervals. To show the average performance across random sampling of expert-coding, we report the average point estimates and standard errors across 500 repeated sampling. In Panel (b), blue dotted lines represent 95%.

(see Appendix I). We emphasize that some methods (e.g., Llama 2) happened to have small biases and reasonable coverages in this application, but this is simply a statistical coincidence without any theoretical guarantee. In the real-world application where researchers cannot see the "Benchmark" estimate, it is impossible for users to decide which estimate is the most credible. As in the previous application, estimates from the classical supervised ML approach are heavily biased and have invalid confidence intervals.

In contrast to these existing approaches, DSL is theoretically guaranteed to be asymptotically unbiased and have valid confidence intervals, as we can clearly see in Figure 7. In practice, this means that researchers can get valid statistical estimates regardless of the choice of the underlying automated text annotation methods.

6 Practical Guide

In this section, we provide practical recommendations regarding the most frequently asked questions. Given the space constraints, we offer additional, comprehensive practical guides in Appendix L, including how to choose LLMs, how to use more complex sampling strategies (e.g., active learning), and what to do if the performance of LLM annotations is poor, among others.

6.1 Errors in Expert Annotations

In practice, expert annotation, which is defined in this paper as a procedure that acts as the benchmark against which the quality of the automated text annotation is evaluated, can contain errors. Completely random errors in expert annotations do not affect the validity of downstream analyses with DSL. However, in some applications, users might worry that errors in expert annotations can be systematic.

Importantly, concerns of such errors in expert annotations are far from new, and they equally apply to almost all existing text-as-data methods, including any supervised machine learning methods and any unsupervised learning methods that are validated by expert-reading of documents (Grimmer and Stewart, 2013). Thus, there already exist various strategies and recommendations for handling errors and uncertainties in expert annotations (e.g., Benoit *et al.*, 2009; Hopkins and King, 2010; Mikhaylov *et al.*, 2012), and researchers can straightforwardly apply them to DSL as well.

Building on this literature, we recommend having multiple expert coders and implementing the following strategies. First, we recommend prioritizing quality over quantity when it comes to expert annotations. It is better to have high-quality expert annotations on a small number of documents than to lower the quality to increase the number of expert-coded documents. In particular, users can reduce the risk of non-random errors by having stricter disambiguation rules

(e.g., all the coders re-annotate any text that has at least one disagreement). This expensive high-quality annotation is not possible for a large number of documents, but it is extremely valuable if it is done even for a small number of documents. Automated text annotation methods like LLMs can provide the quantity. DSL allows researchers to combine these two complementing annotation methods: high-quality, expensive expert annotations and lower-quality, large-scale automated annotations.

Second, researchers can also empirically evaluate the robustness of statistical estimates to errors in expert annotations. In particular, users can treat annotations from one expert coder as if they were the only expert annotations and check whether and how much DSL estimates change depending on which expert annotations are used. When the intercoder reliability between experts is low, and disagreements are systematic, DSL estimates will vary substantially across annotations from different expert coders. In this case, researchers have to re-assess a codebook and disambiguate any systematic disagreement they have in expert annotations. Users can compute the misclassification matrix (e.g., Mikhaylov *et al.*, 2012) to guide disambiguation steps. When the intercoder reliability between experts is high, and disagreements are non-systematic, DSL estimates will be similar across annotations from different expert coders. Finally, when users can justify modeling assumptions about errors, they can also apply simulation-based methods, such as SIMEX (see, e.g., Hopkins and King, 2010; Mikhaylov *et al.*, 2012).

6.2 What if LLM annotations are Very Good?

If LLM annotations have extremely high predictive performance, DSL is going to have small standard errors because bias-correction terms are small (the second part of equation (4) is close to zero), while maintaining statistical validity. However, as long as LLM annotations are not perfect, even if they have excellent performance, the LLM-only estimation is not statistically valid (as seen in our applications and simulations). So, DSL is preferred to the LLM-only estimation even when LLM annotations have high predictive performance.[18]

6.3 Reporting Standards

To transparently report the DSL results, we recommend reporting the following three aspects in the paper. (1) The specification of automated text annotation methods (e.g., the choice of LLM and prompts used to predict labels), (2) sampling method for expert annotations and the number of expert annotations (e.g., randomly sample 500 documents for expert annotations with equal probabilities), and (3) the specification of the DSL model, as scholars typically

[18]In a hypothetical scenario when LLM annotations have no error at all, DSL is going to be the same as the LLM-only estimation.

report statistical models (e.g., we run the DSL linear regression where the dependent variable is the tone of political ads and the independent variables are the Facebook dummy variable and candidate-fixed effects).

6.4 Limitations

Finally, automated text annotation, in particular, the use of LLM annotations, is a rapidly advancing technology. While we propose a generic method that can incorporate any automated text annotation methods with any prediction error, it might sometimes be possible to derive an application-specific automated text annotation method that can statistically guarantee completely random prediction errors or can model prediction errors. When researchers can justify additional assumptions about how prediction errors arise in the automated text annotation step, they can potentially get smaller standard errors than DSL. DSL is a general-purpose method that is most useful when researchers want to avoid stringent assumptions about prediction errors in automated text annotation methods.

7 Wide Applicability

While we so far focused on regression analyses in text-as-data applications, which are the most common downstream analyses, the DSL framework can be used for a broader range of statistical analyses in the social sciences. Our general framework is applicable to any application where researchers use predictive methods to scale up measurements.

- **Estimation of Category Proportions over Time or across Groups**: Many scholars are interested in estimating the proportion of all documents in each user-specified category (e.g., Hopkins and King, 2010; Keith and O'Connor, 2018; Card and Smith, 2018; Jerzak *et al.*, 2023). For example, we might study how the proportion of censored documents changes over time or how the proportion of social media posts containing hate speech differs across groups, such as Democrats and Republicans. These questions can be analyzed within the DSL framework. For instance, using whether a document is censored as the outcome and time indicators as the independent variable in the DSL linear regression, researchers can estimate how the proportion of censored documents changes over time.

- **Causal Inference with Texts**: An increasing number of scholars make causal inference with textual data (Fong and Grimmer, 2021; Egami *et al.*, 2022; Feder *et al.*, 2022; Mozer and Miratrix, 2023). DSL can be used in causal inference applications where the outcome, treatment, or confounders are text-based. In randomized experiments, researchers can use the DSL regression to perform the difference-in-means or covariate-adjusted linear

regression for estimating the average treatment effect.[19] In observational studies, under corresponding causal identification assumptions, researchers can apply the DSL two-stage-least squares for the instrumental variable design, the DSL local linear regression for the regression discontinuity design, and the DSL two-way fixed effects estimator for the difference-in-differences design.

- **Statistical Analyses of Unstructured Data, e.g., Images, Audios, Videos**: Social scientists have begun to utilize a wider range of new data sources, such as image, audio, and video (e.g., Knox and Lucas, 2021; Torres and Cantú, 2022; Tarr *et al.*, 2023). Like the text-as-data literature, researchers often use medium-specific automated annotation methods (e.g., convolutional neural networks and recent foundation models) before analyzing such annotated data in the main downstream statistical analyses. However, as in the automated text annotation, they inevitably contain non-random prediction errors. DSL can be applied to handle such prediction errors in image, audio, and video annotation tasks as well.

8 Concluding Remarks

In this paper, we propose a general framework for using recent advances in automated text annotation methods (e.g., LLMs) and, more generally, generative artificial intelligence (AI) in the social sciences. The proposed framework guarantees statistical validity of downstream analyses, without suffering from bias due to unknown non-random prediction errors in AI models.

Due to the recent rapid advances in AI, we can happily expect that new AI models will be developed every month or even faster. This also means that we will continue to have a suite of models that have high predictive performance but lack scientific and theoretical understanding about their prediction errors and various biases (political, racial, gender, social, and so on). However, the existing approaches (i.e., ignoring prediction errors) exactly need to justify how prediction errors arise. Currently, researchers have to pretend that new AI models have completely random prediction errors, or they have to miss those recent advances. DSL overcomes this tradeoff by incorporating a small number of high-quality, expensive expert annotations. DSL allows users to incorporate any future and current AI models because we do not make any assumptions about prediction errors. With DSL, researchers can always apply state-of-the-art AI models to their social science studies, without worrying that prediction errors and biases in

[19]Previous studies (e.g., Fong and Grimmer, 2021; Egami *et al.*, 2022) have clarified the challenges of inferring a codebook and causal estimates from the same data, especially when using unsupervised learning approaches. In contrast, this paper relies on a supervised learning framework where a codebook is given by researchers rather than estimated by a model.

such AI models might invalidate their scientific and statistical conclusions. We hope that this paper provides a foundation for future work considering this exciting intersection of the social sciences, machine learning, and AI.

References

Angelopoulos, A. N., Bates, S., Fannjiang, C., Jordan, M. I., and Zrnic, T. (2023). Prediction-powered inference. *Science* **382**, 6671, 669–674.

Athey, S., Tibshirani, J., and Wager, S. (2019). Generalized Random Forests. *The Annals of Statistics* **47**, 2, 1148 – 1178.

Barberá, P., Boydstun, A. E., Linn, S., McMahon, R., and Nagler, J. (2021). Automated Text Classification of News Articles: A Practical Guide. *Political Analysis* **29**, 1, 19–42.

Bender, E. M., Gebru, T., McMillan-Major, A., and Shmitchell, S. (2021). On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*, 610–623.

Benoit, K., Laver, M., and Mikhaylov, S. (2009). Treating Words as Data with Rrror: Uncertainty in Text Statements of Policy Positions. *American Journal of Political Science* **53**, 2, 495–513.

Bommasani, R., Hudson, D. A., Adeli, E., Altman, R., Arora, S., von Arx, S., Bernstein, M. S., Bohg, J., Bosselut, A., Brunskill, E., *et al.* (2021). On the Opportunities and Risks of Foundation Models. *arXiv preprint arXiv:2108.07258* .

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., *et al.* (2020). Language Models are Few-Shot Learners. *Advances in neural information processing systems* **33**, 1877–1901.

Card, D. and Smith, N. A. (2018). The importance of calibration for estimating proportions from annotations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, 1636–1646.

Chen, Y.-H. and Chen, H. (2000). A Unified Approach to Regression Analysis under Double-Sampling Designs. *Journal of the Royal Statistical Society Series B: Statistical Methodology* **62**, 3, 449–460.

Chernozhukov, V., Chetverikov, D., Demirer, M., Duflo, E., Hansen, C., Newey, W., and Robins, J. (2018). Double/Debiased Machine Learning for Treatment and Structural Parameters. *Econometrics Journal* **21**, C1 – C68.

Egami, N., Fong, C. J., Grimmer, J., Roberts, M. E., and Stewart, B. M. (2022). How to make causal inferences using texts. *Science Advances* **8**, 42, eabg2652.

Egami, N., Hinck, M., Stewart, B., and Wei, H. (2023). Using Imperfect Surrogates for Downstream Inference: Design-based Supervised Learning for Social Science Applications of Large Language Models. *Advances in Neural Information Processing Systems* **36**.

Feder, A., Keith, K. A., Manzoor, E., Pryzant, R., Sridhar, D., Wood-Doughty, Z., Eisenstein, J., Grimmer, J., Reichart, R., Roberts, M. E., *et al.* (2022). Causal inference in natural language processing: Estimation, prediction, interpretation and beyond. *Transactions of the Association for Computational Linguistics* **10**, 1138–1158.

Fong, C. and Grimmer, J. (2021). Causal Inference with Latent Treatments. *American Journal of Political Science* .

Fong, C. and Tyler, M. (2021). Machine learning predictions as regression covariates. *Political Analysis* **29**, 4, 467–484.

Fowler, E. F., Franz, M. M., Martin, G. J., Peskowitz, Z., and Ridout, T. N. (2021). Political Advertising Online and Offline. *American Political Science Review* **115**, 1, 130–149.

Gentzkow, M., Kelly, B., and Taddy, M. (2019). Text as Data. *Journal of Economic Literature* **57**, 3, 535–574.

Gilardi, F., Alizadeh, M., and Kubli, M. (2023). ChatGPT Outperforms Crowd-Workers for Text-Annotation Tasks. *arXiv preprint arXiv:2303.15056* .

Grimmer, J., Roberts, M. E., and Stewart, B. M. (2022). *Text as data: A new framework for machine learning and the social sciences*. Princeton University Press.

Grimmer, J. and Stewart, B. M. (2013). Text as Data: The Promise and Pitfalls of Automatic Content Analysis Methods for Political Texts. *Political analysis* **21**, 3, 267–297.

Hager, A. and Hilbig, H. (2020). Does Public Opinion Affect Political Speech? *American Journal of Political Science* **64**, 4, 921–937.

Hastie, T., Tibshirani, R., Friedman, J. H., and Friedman, J. H. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, vol. 2. Springer.

Hopkins, D. J. and King, G. (2010). A Method of Automated Nonparametric Content Analysis for Social Science. *American Journal of Political Science* **54**, 1, 229–247.

Jerzak, C. T., King, G., and Strezhnev, A. (2023). An improved method of automated nonparametric content analysis for social science. *Political Analysis* **31**, 1, 42–58.

Jurafsky, D. and Martin, J. H. (2024). Speech and language processing. Tech. rep., Stanford University.

Katsumata, H. and Yamauchi, S. (2023). Statistical Analysis with Machine Learning Predicted Variables. Tech. rep., Working Paper.

Keith, K. and O'Connor, B. (2018). Uncertainty-aware generative models for inferring document class prevalence. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 4575–4585, Brussels, Belgium. Association for Computational Linguistics.

King, G., Tomz, M., and Wittenberg, J. (2000). Making the Most of Statistical Analyses: Improving Interpretation and Presentation. *American Journal of Political Science* 347–361.

Knox, D. and Lucas, C. (2021). A Dynamic Model of Speech for the Social Sciences. *American Political Science Review* **115**, 2, 649–666.

Knox, D., Lucas, C., and Cho, W. K. T. (2022). Testing Causal Theories with Learned Proxies. *Annual Review of Political Science* **25**, 419–441.

Linegar, M., Kocielnik, R., and Alvarez, R. M. (2023). Large Language Models and Political Science. *Frontiers in Political Science* **5**, 1257092.

Lundberg, I., Johnson, R., and Stewart, B. M. (2021). What is Your Estimand? Defining the Target Quantity Connects Statistical Evidence to Theory. *American Sociological Review* **86**, 3, 532–565.

Mikhaylov, S., Laver, M., and Benoit, K. R. (2012). Coder Reliability and Misclassification in the Human Coding of Party Manifestos. *Political Analysis* **20**, 1, 78–91.

Mikolov, T., Yih, W.-t., and Zweig, G. (2013). Linguistic Regularities in Continuous Space Word Representations. In *Proceedings of the 2013 conference of the north american chapter of the association for computational linguistics: Human language technologies*, 746–751.

Mozer, R. and Miratrix, L. (2023). Decreasing the Human Coding Burden in Randomized Trials with Text-based Outcomes via Model-Assisted Impact Analysis. *arXiv preprint arXiv:2309.13666* .

Ollion, E., Shen, R., Macanovic, A., and Chatelain, A. (2023). Chatgpt for Text Annotation? Mind the Hype! *SocArXiv. October* **4**.

Ornstein, J. T., Blasingame, E. N., and Truscott, J. S. (2022). How to Train Your Stochastic Parrot: Large Language Models for Political Texts. Tech. rep., Working Paper.

Pan, J. and Chen, K. (2018). Concealing Corruption: How Chinese Officials Distort Upward Reporting of Online Grievances. *American Political Science Review* **112**, 3, 602–620.

Pangakis, N., Wolken, S., and Fasching, N. (2023). Automated Annotation with Generative AI Requires Validation. *arXiv preprint arXiv:2306.00176* .

Robins, J. M. and Rotnitzky, A. (1995). Semiparametric efficiency in multivariate regression models with missing data. *Journal of the American Statistical Association* **90**, 429, 122–129.

Robins, J. M., Rotnitzky, A., and Zhao, L. P. (1994). Estimation of Regression Coefficients When Some Regressors Are Not Always Observed. *Journal of the American Statistical Association* **89**, 427, 846–866.

Rodriguez, P. L. and Spirling, A. (2022). Word Embeddings: What Works, What Doesn't, and How to Tell the Difference for Applied Research. *The Journal of Politics* **84**, 1, 101–115.

Rotnitzky, A. and Vansteelandt, S. (2014). Double-robust methods. In *Handbook of missing data methodology*, 185–212. CRC Press.

Spirling, A. (2023). Why Open-Source Generative AI Models Are An Ethical Way Forward For Science. *Nature* **616**, 7957, 413–413.

Tarr, A., Hwang, J., and Imai, K. (2023). Automated Coding of Political Campaign Advertisement Videos: An Empirical Validation Study. *Political Analysis* **31**, 4, 554–574.

Torres, M. and Cantú, F. (2022). Learning to See: Convolutional Neural Networks for the Analysis of Social Science Data. *Political Analysis* **30**, 1, 113–131.

Wang, S., McCormick, T. H., and Leek, J. T. (2020). Methods for Correcting Inference based on Outcomes Predicted by Machine Learning. *Proceedings of the National Academy of Sciences* **117**, 48, 30266–30275.

Zhang, H. (2021). How using machine learning classification as a variable in regression leads to attenuation bias and what to do about it. SocArXiv.

Ziems, C., Held, W., Shaikh, O., Chen, J., Zhang, Z., and Yang, D. (2023). Can Large Language Models Transform Computational Social Science? *arXiv preprint arXiv:2305.03514* .