

# ChatGPT for Textual Analysis? How to Use Generative LLMs in Accounting Research

Ties de Kok<sup>a</sup>

<sup>a</sup>University of Washington, Seattle, Washington 98195

Contact: [tdekok@uw.edu](mailto:tdekok@uw.edu),  <https://orcid.org/0000-0001-9577-4057> (TdK)

Received: October 6, 2023

Revised: March 9, 2024; June 15, 2024

Accepted: June 24, 2024

Published Online in Articles in Advance:  
January 13, 2025

<https://doi.org/10.1287/mnsc.2023.03253>

Copyright: © 2025 INFORMS

**Abstract.** Generative large language models (GLLMs), such as ChatGPT and GPT-4 by OpenAI, are emerging as powerful tools for textual analysis tasks in accounting research. GLLMs can solve any textual analysis task solvable using nongenerative methods as well as tasks previously only solvable using human coding. Whereas GLLMs are new and powerful, they also come with limitations and present new challenges that require care and due diligence. This paper highlights the applications of GLLMs for accounting research and compares them with existing methods. It also provides a framework on how to effectively use GLLMs by addressing key considerations, such as model selection, prompt engineering, and ensuring construct validity. In a case study, I demonstrate the capabilities of GLLMs by detecting nonanswers in earnings conference calls, a traditionally challenging task to automate. The new GPT method achieves an accuracy of 96% and reduces the non-answer error rate by 70% relative to the existing Gow et al. (2021) method. Finally, I discuss the importance of addressing bias, replicability, and data sharing concerns when using GLLMs. Taken together, this paper provides researchers, reviewers, and editors with the knowledge and tools to effectively use and evaluate GLLMs for academic research.

**History:** Accepted by Eric So, accounting.

**Funding:** Supported by the Foster School of Business – University of Washington.

**Supplemental Material:** The online appendix and data files are available at <https://doi.org/10.1287/mnsc.2023.03253>.

**Keywords:** textual analysis • artificial intelligence • machine learning • GPT • large language models

## 1. Introduction

Rapid advancements in natural language processing (NLP) and machine learning (ML) have given rise to generative large language models (GLLMs), such as the ChatGPT and GPT-4 models by OpenAI. This new family of models holds significant potential for accounting research studying textual data. GLLMs can solve textual analysis tasks with the power, versatility, and ease of use of human coding, enjoying the scalability, speed, and cost benefits of machine learning methods. For example, a well-instructed GLLM can extract emission targets in environmental, social, and governance reports, find CEOs trying to be funny in conference calls, compare risk factor consistency across firms, cluster topics in employee reviews, extract corporate culture references in job postings, and much more. Whereas GLLMs are new and powerful, they also come with limitations and present new challenges that require care and due diligence. This paper provides researchers, reviewers, and editors with the knowledge and tools on how to effectively use and evaluate GLLMs in academic research.

GLLMs are a class of models that consume and generate text, like a powerful autocomplete. Generative pre-trained transformer (GPT) models are the most

common type of GLLM. Because GLLMs work through natural language, they are easier to use for a wide range of textual analysis tasks. The general-purpose capabilities of GLLMs also enable them to complete many tasks with no or little training data, which can save significant time and resources. For example, Hassan et al. (2019) identify political risk discussions in earnings calls by training an algorithm on a large training data set of political texts. Achieving this is a challenging and time-consuming task without GLLMs. In contrast, a state-of-the-art GLLM such as GPT-4 can solve this problem out of the box (i.e., zero shot) by simply showing it a call excerpt and asking whether it discusses political risk or not as shown in Appendix B. Smaller models, such as GPT-3 or Llama2, can yield similar performance with appropriate instructions or training. Furthermore, the scale of GLLMs enables a basic degree of reasoning ability and world understanding that helps them solve problems that would otherwise be infeasible to automate. For example, Gow et al. (2021) find that existing machine learning methods struggle with detecting nonanswers in earnings conference calls because of the nuance and heterogeneity of that task. GLLM approaches can solve such complex tasks as I demonstrate in a case study. Taken

together, GLLMs can make it easier for researchers to analyze textual data and study new research questions.

Whereas GLLMs are powerful, they come with limitations and present new challenges. The natural language aspect of GLLMs can make it feel like working with a capable human. However, even the best GPT models make mistakes and are not guaranteed to give meaningful responses, especially when using them out of the box. Care and due diligence by researchers can help mitigate this risk, but it requires carefully evaluating construct validity as with other ML methods. Furthermore, the large size of GLLMs can make them slow, expensive, and hard to control. Overcoming those limitations to unlock the potential of GLLMs requires working and thinking in new ways. However, the rapid rise in availability of GLLMs is accompanied by a lack of guidance on how to use these models in academic research. To illustrate, the recent overview of textual analysis methods in accounting by Bochkay et al. (2023) does not cover GLLMs. My paper provides an introduction, practical framework, case study, and discussion to help researchers effectively adopt and evaluate GLLMs for their research projects.

I start by introducing the GLLM approach to cover its basic workings and how it relates to existing large language models (LLMs), such as bidirectional encoder representations from transformers (BERT). I then compare the pros and cons of GLLMs relative to existing textual analysis techniques and human coding. The conclusion is that GLLMs can solve any textual analysis task solvable using nongenerative methods. However, nongenerative techniques can be substantially cheaper and faster for less complex problems that do not require the power of GLLMs. When GLLMs start to become viable is for problems that would otherwise require complex ML machinery or human coding. GLLMs are substantially cheaper and faster relative to human coding and can complete equivalent work for many textual analysis tasks. However, human coders are usually easier to instruct and preferable for complex tasks with smaller samples.

Next, I provide a four-step framework to help researchers use GLLMs for their research projects. The framework complements the guidance by Anand et al. (2020) and Bochkay et al. (2023) and focuses on the implementation details and challenges specific to GLLMs. The framework first covers defining the research task and identifying the types of information necessary to perform the task. GLLMs that lack the necessary information or context are prone to “hallucinate” by guessing or making up information. To help with that, the framework next provides guidance on selecting the appropriate GLLM approach and prompt design, including a flowchart in Figure 1. The last step provides guidance on how to ensure construct validity by designing an evaluation approach to assess model performance and

identify problems. Finally, the Online Appendix and companion website provide additional practical tips and code examples.<sup>1</sup>

To demonstrate the capabilities of GLLMs, I do a case study of identifying nonanswers in earnings conference calls. Detecting nonanswers is traditionally challenging because it involves a large degree of nuance and context dependence. The results indicate that GLLMs can solve this task in multiple different ways, all improving on the accuracy of the Gow et al. (2021) approach. By combining GLLMs with existing methods, I achieve an accuracy of 96% and a nonanswer F1 score of 87% at a reasonable cost.<sup>2</sup> This represents a 70% reduction in the nonanswer error rate from the Gow et al. (2021) method. To further demonstrate the flexibility and capabilities of GLLMs, I also fine-tune a ChatGPT 3.5 model to extract additional nonanswer dimensions, such as its justification, question type, and sentiment, using a training data set created with GPT-4.

Finally, I discuss several key considerations when using GLLMs in academic research. The training data of GLLMs raises concerns about biases, source material, and a lack of knowledge about recent events. Furthermore, working with GLLMs raises several new replicability challenges with which researchers need to be careful as models are not interchangeable, and third-party providers can remove or alter models at any time. Researchers should always store their raw prompts and completions under the assumption that they might lose access to the source model. Researchers should also be careful with sending their research data to third-party services as such data sharing might be prohibited or undesirable. Lastly, GLLMs are increasingly popular as writing and programming assistants. These tools can be useful productivity boosters, however, researchers should never blindly rely on the outputs of a GLLM when producing research and always scrutinize and check the outputs before using them.

This paper makes four contributions. First, it extends Bochkay et al. (2023) by showing how to use GLLMs as a powerful new textual analysis method. Second, it raises attention to and provides guidance on the challenges and limitations regarding GLLMs, such as ensuring construct validity, dealing with biases and knowledge limitations, and mitigating replicability and data-sharing concerns. Third, it offers practical resources and guidance to help researchers get started with GLLMs for their own projects, including a framework, technical guidance, code examples, and more through the Online Appendix and companion website. Finally, the case study of nonanswers in earnings conference calls adds to the disclosure literature on communication dynamics by studying the prevalence of nonanswers and its various dimensions. I provide the nonanswer data set and its dimensions through the companion website for future research to use. Taken together, this paper

equips researchers, reviewers, and editors with the knowledge and tools on how to effectively use and evaluate GLLMs in academic research.

## 2. Generative Large Language Models

### 2.1. What Are GLLMs and How Do They Work?

GLLMs are a class of machine learning models that consume and generate natural language. The most well-known example is ChatGPT by OpenAI; you ask ChatGPT a question, and it provides you with an answer. GPT models are the most common type of GLLM, but other generative architectures exist. The generative aspect is the key innovation of GLLMs that separates them from previous approaches. Specifically, GLLMs perform the task of predicting (i.e., generating) the next token based on the tokens that precede it, like autocomplete predictions when typing. A token represents a word, part of a word, or a character. For example, ChatGPT tokenizes the text “baseball is fun!” into the tokens “base”, “ball”, “is”, “fun”, “!”. Some words are a single token, and some words are multiple tokens.<sup>3</sup> The original input we provide to the GLLM is the prompt, and the resulting generation is the completion. Both the prompt and the completion are a collection of tokens, and the model predicts each token in the completion one at a time. For example, if we prompt ChatGPT with “what is the Tesla truck called?” and it provides us with the completion “Cybertruck.” the model has made three predictions, one for each output token (in bold):

1. What- is- the- Tesla- truck- called-? → “**Cyber**”
2. What- is- the- Tesla- truck- called-?- Cyber → “**truck**”
3. What- is- the- Tesla- truck- called-?- Cyber-truck → “.”

GLLMs can try to autocomplete any type of text and, by extension, try to solve any task expressible into words. The general applicability of GLLMs as well as the abundance of possible training data (i.e., any text) make it possible to train foundational models at parameter scales not previously feasible.<sup>4</sup> These large parameter scales and the diversity of its training data enable GLLMs to achieve a degree of language and world understanding that is significantly higher than previous models and textual analysis techniques.

Focusing on GPT models, many different model implementations exist, such as ChatGPT, GPT-3, GPT-4, Gemini (Bard), Claude, (Code)Llama2, CodeLlama, and Phi-2 to name a few. All these models work by predicting the next token based on the preceding tokens. The ways the models differ is primarily in their parameter size, tuning, and availability. The model size determines the out-of-the-box capabilities at the expense of cost and speed. The model tuning determines how the model responds and the types of tasks at which the model is

particularly good at. For example, the ChatGPT model is tuned to respond to your instructions like a human assistant, GPT-3 is tuned to continue your text, and CodeLlama is tuned to generate code. The availability determines whether the model is available to run locally or whether it is only accessible through a third-party service. Section 3.2 discusses how to choose the right model, and Online Appendix 1 provides a more detailed overview of model differences, popular models, and my recommendations to get started.

### 2.2. How Do GLLMs Relate to Other LLMs Such as BERT?

The two most well-known types of LLMs are GPT and BERT models. Both are part of the transformer architecture family, which originates from the development of the attention mechanism (Bahdanau et al. 2016, Vaswani et al. 2017). Practically, the transformer architecture is a breakthrough because it allows for better parallelization as well as better modeling of long-range dependencies in text, making it possible to train larger models that understand text better. BERT models, introduced by Devlin et al. (2019), use a transformer architecture to convert text into a better input for machine learning models. In contrast, GPT models are generative and solve the task of predicting the next token like an autocomplete (Radford et al. 2018, 2019; Brown et al. 2020; OpenAI 2023).

Both GPT and BERT models are powerful textual analysis methods that utilize the benefits of the transformer architecture and its transfer learning abilities (e.g., Siano and Wysocki 2021). The fundamental difference is how one adapts them to solve a textual analysis problem. A BERT model is a powerful building block as part of a broader machine learning pipeline. For example, Huang et al. (2023) use a BERT model to represent financial documents, which they then feed into a standard machine learning classifier to classify the sentiment. In contrast, GLLMs are stand-alone in that we feed them a natural language prompt, the model generates a completion, and we retrieve the result by parsing that completion. For example, Lopez-Lira and Tang (2023) classify news sentiment by showing ChatGPT news headlines and asking, “Is this headline good or bad for the stock price of <company>? Answer YES if good news, NO if bad news, or UNKNOWN if uncertain.” They then parse the responses by ChatGPT to formulate their sentiment score.

I discuss the specific pros and cons of GLLMs versus existing methods in the next section. But, at a higher level, textual analysis tasks are often solvable using both a GPT as well as a BERT approach. The benefit of GPT models is that they are easier to use as they generally require less (or no) training data, and they are more flexible and easier to adapt to a wide range of tasks. The benefit of BERT approaches on the other hand is that once you manage to



train them, they are many magnitudes faster and cheaper to use and scale. The primary issue with GPT models is that they are expensive to scale, and the primary issue with BERT approaches is that they require costly training data sets. As a result, more complex textual analysis problems at small to medium scales align well with GPT approaches, whereas simpler and larger scale problems align well with BERT approaches. Finally, it is worth pointing out that the GPT method and BERT approaches are not mutually exclusive, and combining them can sometimes yield the best of both worlds. For example, a powerful GLLM such as GPT-4 can be a cost-effective way to create a high-quality training data set that can be used to train a BERT pipeline to make it cheaper and faster to scale the classifications to a large volume of documents.<sup>5</sup>

### 2.3. Pros and Cons Relative to Current Methods

#### 2.3.1. Comparison with Existing Textual Analysis Methods.

GLLMs have several benefits over existing textual analysis methods. The first is that large GPT models (such as ChatGPT and GPT-4) can often solve basic and objective textual analysis tasks without needing additional training. For example, Hassan et al. (2019) identify political risk conversation in conference calls by constructing a training data set and training a classifier on it. Creating large quantities of training data manually is costly and time-consuming. Instead, a model such as GPT-4 can solve this out of the box by showing parts of the earnings call and asking whether the section discusses political risks, yes or no. The second benefit is that natural language prompts are more flexible in expressing tasks relative to more rigid statistical approaches. For example, Brown et al. (2018) try to identify how firms adjust their risk factor disclosures when a peer firm receives a Securities and Exchange Commission (SEC) comment letter. They limit their measure to changes in the number of words as it is challenging to identify specific types of changes. The GLLM approach makes it possible to identify and classify specific types of year-over-year changes. Moreover, one could include details about the SEC comment letter to identify only changes that relate to the letter. This is done by formulating a prompt that shows the risk factors, a summary of the comment letter (which could also be GLLM-generated), followed by a question in the spirit of “did the risk factor change in response to the comment letter, yes or no?”.

The third benefit is that the basic reasoning and world understanding of large GPT models can make it possible to solve tasks that would be hard or impossible using existing NLP techniques. For example, Call et al. (2024) use a sophisticated machine learning algorithm to identify cases of laughter in response to humor during conference calls. They limit their focus to only successful cases of humor as unsuccessful attempts to be

funny do not elicit laughter. Current state-of-the-art GPT models can also identify unsuccessful cases of humor, for example, by showing an earnings call excerpt and asking the question “did the CEO try to be funny, yes or no?”. The final benefit is that GLLMs can handle longer inputs and documents relative to existing models. For example, BERT models can typically handle inputs of up to 512 tokens. In contrast, the most recent ChatGPT model can handle up to 16,385 tokens (about 26 pages of text), and the GPT-4-Turbo model can handle up to 128,000 tokens (a few hundred pages of text).

The GLLM approach also has important drawbacks and limitations. The primary disadvantage is that GLLMs are large and require more processing resources, making them more expensive and slower to use. Nongenerative textual analysis techniques are substantially cheaper and faster for simpler tasks that do not require the power of GLLMs. A good rule of thumb is to keep things as simple as possible and only resort to more powerful techniques when necessary. Furthermore, GLLMs such as ChatGPT and GPT-4 do make mistakes and are not guaranteed to give meaningful responses, especially when using them out of the box without any additional fine-tuning. The responses by GPT models can look good at first glance but not hold up to deeper inspection, so it is best to be skeptical of the performance until proven otherwise through evaluation. Achieving a high construct validity when using a GPT model requires researcher care and due diligence. I provide additional guidance on how to appropriately use and evaluate GLLMs in Section 3. Finally, large GPT models require expensive hardware and are most accessible through third-party providers. Sending research data to external parties is a risk and can be prohibited because of copyright and data restrictions. Using GLLMs locally can be hard or impossible, which can make them less suitable for tasks that require strict data controls.

**2.3.2. Comparison with Human Coding.** The power and out-of-the-box capabilities of GLLMs make them a possible alternative to human coding. For example, Chuk et al. (2013) hand-collect management forecasts in earnings releases to compare them with a data aggregator. A powerful GLLM can plausibly complete the same task. Using a GLLM would save significant time and resources as they are cheaper, faster, and yield more consistent results relative to human coding. GLLMs can also exhibit superhuman capabilities in terms of recalling and memorizing large quantities of information. However, even the most powerful GLLMs have important limitations, and there is no guarantee they yield responses equivalent to human coding. The primary benefits of research assistants (RAs) are their flexibility, stronger reasoning ability, and incentives to try hard and produce good work. Humans can be held accountable for their work, whereas machines cannot. A well-

incentivized RA usually provides high-quality outputs with only minimal instructions, whereas a GLLM might require prompt engineering or fine-tuning before it starts generating equivalent outputs. This makes human coding a better choice for complex tasks with smaller samples. Domain knowledge can be another limitation. GLLMs might need to learn domain knowledge through explicit examples, which can be hard to obtain or formulate. For example, Hail et al. (2018) use human coding to identify accounting scandals that are morally versus legally wrong. It would be challenging to communicate to a GLLM what it means for something to be morally wrong.

## 2.4. Current Research Using GLLMs

Even as a relatively new technology, GLLMs are already being used in working papers for a wide range of textual analysis tasks. Bernard et al. (2023) fine-tune a Llama 3 model on a large sample of 10-K footnotes to measure the underlying business complexity. Kim et al. (2024) use ChatGPT to generate automatic summaries of financial disclosures to measure the level of disclosure bloat. Rouen et al. (2023) use ChatGPT to extract seed words for every material topic in the Sustainability Accounting Standards Board standard by sector and industry. Jha et al. (2024) use ChatGPT and conference calls to measure managers' anticipated changes in capital expenditures. Bai et al. (2023) use a variety of GLLMs, such as ChatGPT, Google Bard, and an open-source model, to develop a measure of information content by comparing answers in the question-and-answer (Q&A) section of conference calls with those automatically generated by the GLLMs. Finally, Hansen and Kazinnik (2023) apply GPT models to study technical language by the Federal Reserve and show that GPT-4 provides explanations that are on par with human reasoning. Several papers are also applying GLLMs to model the sentiment of financial disclosures. For example, Lopez-Lira and Tang (2023) use ChatGPT and GPT-4 to define sentiment scores for news articles covering company news. Their results suggest that the GLLM sentiment scores outperform traditional sentiment methods, including BERT, in terms of return predictability and trading strategies. Alternatively, Hu et al. (2023) find that FinBERT by Huang et al. (2023) outperforms a zero-shot GPT-3 model for sentiment analysis.

## 3. A Framework for Using GLLMs to Solve Research Tasks

The generative natural language aspects of GLLMs pose distinct challenges that are not covered by existing NLP overview papers such as Anand et al. (2020) and Bochkay et al. (2023). Working with GLLMs requires a way of thinking that falls between interacting with a human

(because of the natural language aspect) and employing a nongenerative machine learning model (because of the machine aspect). Specifically, whereas it is possible to instruct a GLLM like instructing a human, that often yields unsatisfactory results. Achieving high-quality, consistent, and efficient results using the GLLM approach requires researcher care, effort, and due diligence. In this section, I provide a four-step framework that helps researchers implement GLLMs for their research tasks.

### 3.1. Define and Understand Your Problem: Step 1

The first step is to define and understand the problem you are looking to automate. Doing so requires the evaluation of what information is necessary to complete the task. This may include specific domain knowledge, contextual information, or other relevant details that are necessary to generate the desired output. Any model, irrespective of how powerful it is, will not perform well if it lacks the necessary information to make a prediction or classification. When GLLMs lack sufficient context or background information they are prone to hallucinate, which is the generation of incorrect or made-up information, especially when using them out of the box. Any information, such as specific institutional knowledge, that is likely not known to the GLLM needs to be provided either through examples or instructions or by fine-tuning.

To help with that, I recommend developing a deeper understanding of the task you are looking to automate by performing the task manually yourself on a set of examples and asking yourself why you made the decisions you made. What information did you use, and is it plausible that the GLLM you plan to use will "know" that information or will you need to provide it? For example, assume you want to identify whether a footnote text relates to Financial Accounting Standards Board Accounting Standards Update (FASB ASU) 2014-09. You might be able to complete that task easily because you know to what FASB ASU 2014-09 relates, but the GLLM might have no idea or, worse, a wrong understanding of it. Including an explicit description or definition of FASB ASU 2014-09 in the instruction will help with that.

### 3.2. Decide on the Approach and Model: Step 2

Each GLLM has different strengths and limitations, and the right model depends on the task and the approach. There are three approaches to instruct a GLLM: zero shot, few shot, and fine-tuning. The zero-shot approach works by providing the GLLM only your instructions and the data for the task without giving any examples. The few-shot approach works like the zero-shot approach except that you provide a few examples, usually one to five, as part of every prompt to complement your instructions. Finally, the fine-tuning approach works by training an existing GLLM on a larger set of

examples to update all or a part of the model weights (coefficients).

The primary trade-off between the zero-/few-shot and fine-tuning approaches is convenience versus control. Using GLLMs out of the box in a zero-/few-shot way is relatively easy as it avoids having to create a training data set and fine-tuning the model with it. This is especially true when using a model already set up through an external application programming interface (API). However, even the most powerful GLLMs, such as GPT-4, do make mistakes, and there is no guarantee they give meaningful responses. The primary way to control a zero-shot GLLM is through the prompt you provide, which can make it hard and sometimes impossible to get consistent and satisfactory results. As with a more traditional machine learning pipeline, fine-tuning a GLLM works by instructing the model through examples in a training data set instead of through prompt instruction. A high-quality training data set yields more control over the behavior of the model, but creating training data are costly, and fine-tuning a GLLM is significantly more complex.<sup>6</sup> I provide details and guidance on fine-tuning a GLLM in Online Appendix 5 and the companion website.

You also need to consider the number of tokens required to complete all your predictions. The time and costs of using a GLLM increase linearly with the tokens in your prompt and completion. Bigger models are generally slower and more expensive and, thus, lead to lower token budgets for a given amount of money and time. This has implications for your model choice and your decision between zero shot, few shot, and fine-tuning as well as your prompt design. For example, a few-shot prompt can work better relative to a zero-shot prompt, but including examples as part of your prompt significantly increases the number of tokens required to process an observation.

There are many providers and models from which to choose. Online Appendix 1 provides an overview of the popular GPT models currently available and the differences between models. The flowchart in Figure 1 can help you decide what GLLM approach to consider for a given research project. I generally recommend using the convenience of third-party party APIs (versus a local model) when sharing your data is possible. I also recommend first evaluating whether a zero-shot approach would work and only switching to model fine-tuning if a zero-shot approach does not yield satisfactory construct validity. Once you have selected the approach, try different model sizes using a smaller sample and pick the smallest model that yields satisfactory performance, which is not necessarily the model with the highest performance. For example, if a ChatGPT (or Llama 13b) model yields reasonable error rates for your task, then it is unnecessary to spend a lot more time and money

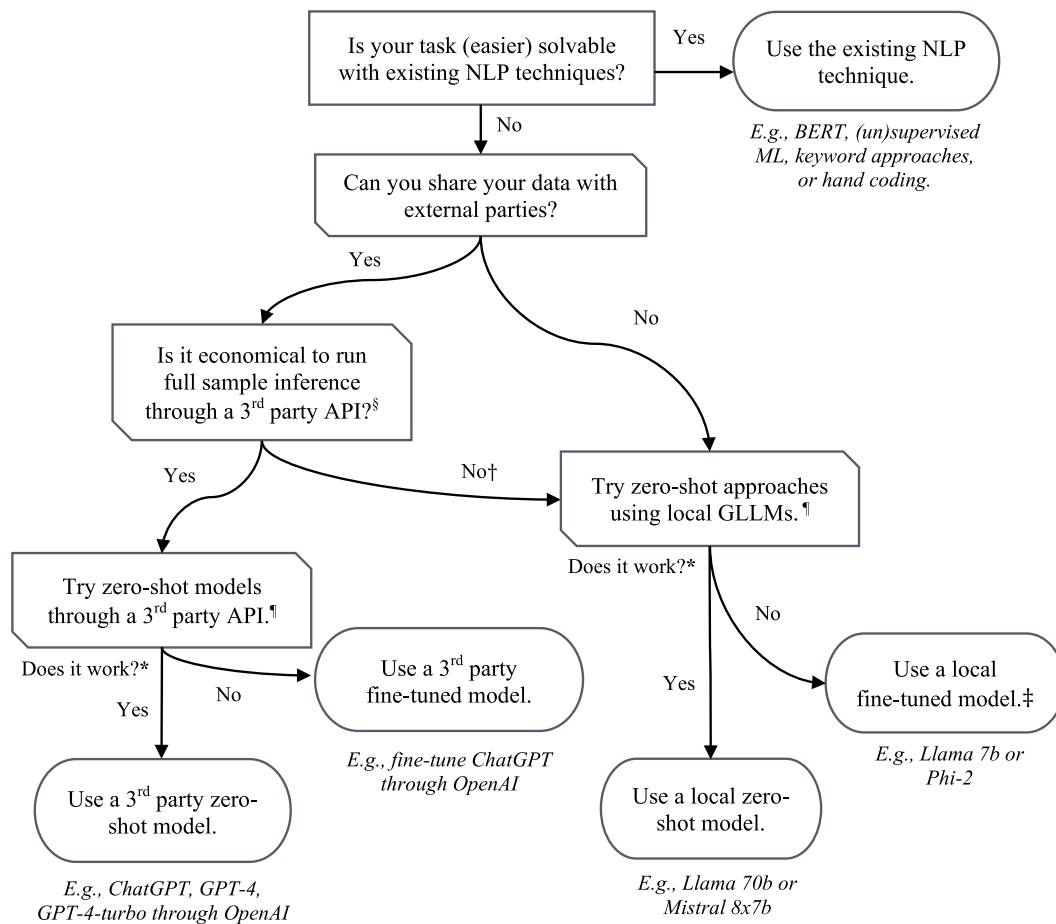
using a GPT-4 (or Llama 70b) model to only marginally increase performance.

### 3.3. Develop Your Prompt (i.e., Prompt Engineering): Step 3

A prompt is the input you provide to the model, and a completion is the output the model generates. Both are expressed in natural language and take the form of a string of text. The input prompt is the primary way to control the output generated by the GLLM as the prompt contains your instructions, any contextual information, and the focal data. The focal data represents the information that changes for each observation in your data. For example, a simple prompt is “Does the following tweet relate to politics, yes or no? Tweet: {the focal tweet}.” Engineering a good prompt is the key to getting reliable and high-quality outputs, especially when using a zero-/few-shot approach. Prompt engineering is less influential when using fine-tuning because the model learns the task primarily through the training data examples in that approach.

When designing a prompt, the two primary objectives are to communicate to the GLLM what you expect the completion to be and to provide the GLLM with the necessary information to generate the desired completion. A model that misunderstands the instructions or that does not have access to the necessary information will generate completions of low quality. Technically, any information representable as text can be part of a prompt or completion. This includes not only narrative data but also tabular data, JavaScript Object Notation (JSON), Extensible Markup Language, Hypertext Markup Language, programming code, and others. However, zero-shot models generally perform best with regular text; alternative formats might require fine-tuning. It is hard to anticipate how GLLMs will respond to instructions, so it requires experimentation against an evaluation sample to determine the right prompt that achieves high construct validity. Appendices B–D provide prompt examples as well as practical prompt engineering tips to help develop effective prompts.

When designing the completion portion of your instruction, the primary objective is to make the output machine readable and easy to parse. Completions are natural text that can be hard or impossible to parse with your code if you do not design your completion with parsing in mind. For example, we might want to extract cashtag mentions out of Twitter messages and classify them as bullish or bearish. If we ask a GLLM to perform that task without being mindful of the output, it might yield something like “The first cashtag is \$AAPL and it is bullish and the second cashtag is bearish (MMM).” This would be tedious and difficult to parse into a data set. Instead, we could include an instruction to return a JSON list of the format “[{cashtag, sentiment}, ...].”

**Figure 1.** GLLM Approach Flowchart

**Notes.** \*Visual inspection of a few generations is not sufficient to know whether a zero-shot approach works or not. Section 3.4 discusses how to appropriately evaluate construct validity. Also, zero-shot approaches are generally more expensive as they require more tokens, so they might not work because of cost constraints.

<sup>†</sup>The scale economies of third-party APIs can make them relatively economical compared with running an equivalent model yourself. However, academic institutions commonly have existing computing infrastructure that one could utilize with a local model at low to no costs. Furthermore, local models make it possible to explore smaller GPT models, which can be significantly cheaper to use and fine-tune.

<sup>§</sup>The costs of using a third-party API scale linearly with the number of tokens you input and output. I recommend observing or calculating the number of tokens for a small sample of prompts + completions and extrapolating those costs to the size of your entire sample to estimate how much it would cost in total.

<sup>¶</sup>There are many models from which to choose. I recommend trying different models and model sizes using a small test sample and picking the smallest model that yields satisfactory performance, which is not necessarily the model with the highest performance because of time and cost considerations. See Section 3.2 and Online Appendix 1 for more guidance.

<sup>‡</sup>Fine-tuning a local GLLM is the most technically advanced option and I recommend only pursuing it if fine-tuning or training an existing ML pipeline, such as using BERT, is not working or feasible for your task.

This would yield “[('\$AAPL', 'bullish'), ('\$MMM', 'bearish')],” which we can easily parse into a data set.

### 3.4. Evaluate the Construct Validity: Step 4

Evaluating the construct validity of any textual analysis approach is an important responsibility of the researcher (Bochkay et al. 2023). Evaluating generative completions comes with two primary challenges. The first challenge is that evaluating zero- and few-shot approaches still requires an evaluation data set even when they do not require a training sample. The second challenge is that GLLMs can yield completions that are

not compatible with standard performance metrics. For example, it is easy to develop a prompt that asks many questions at once. However, how do we evaluate the quality of the completion if it consists of multiple individual elements? I recommend the following best practices to evaluate the construct validity of a GLLM:

1. Always create an evaluation set of outcomes generated by human raters, such as yourself, coauthors, research assistants, domain experts, or third-party sources. This sample needs to be large enough, generally between a hundred to a couple thousand. The evaluation sample needs to be large enough to represent



the diversity of the data set and task. All approaches require an evaluation sample, including zero- and few-shot approaches.

2. When possible, engineer your instructions such that you can evaluate the GLLM approach like a standard supervised machine learning approach. For example, classifying sentiment as “positive = 0/1” is easier to evaluate relative to a narrative description, such as “sentiment = the text is mostly positive, but some parts are negative.” When returning multiple outcomes at once, I recommend evaluating and reporting each outcome independently.

3. When it is not possible to evaluate your completion as a classification task, I recommend following the evaluation best practices for the closest non-GLLM approach that one would use for that specific task. For example, when using a GLLM for a clustering task, one can follow the best practices for unsupervised machine learning (e.g., section 4 in Bochkay et al. 2023).

4. Be skeptical of the completion quality. GLLMs are trained to mimic human responses as closely as possible. However, GLLM responses are not human responses. Outputs generated by GLLMs tend to look great at first inspection but can be less accurate upon closer inspection. Because of human confirmation bias, it is easy to overestimate the quality of a GLLM completion.

Once a satisfactory construct validity is reached, the approach can be scaled up to generate completions for all the data observations. This step is commonly referred to as model inference.

## 4. Case Study: Using GLLMs to Identify Nonanswers in Earnings Conference Calls

One of the promises of GLLMs is to help automate tasks that would otherwise be challenging. To demonstrate and evaluate their capabilities, I deploy GLLMs for the challenging task of identifying nonanswers in earnings calls. Identifying nonanswers is a hard NLP problem because questions and answers in earnings calls are diverse in their content and style, often involve a large degree of nuance and context, and frequently suffer from lower data quality because of transcript errors.

Two prior papers study earnings call nonanswers. Hollander et al. (2010) use manual hand-coding to study a small sample of earnings calls. This coding process took multiple months to complete. More recently, Gow et al. (2021, p. 4) develop a rule-based technique using regular expressions to identify nonanswers automatically for a large sample of earnings calls. That approach is functional, but their classifications leave significant room for improvement. They try but are unable to improve performance through traditional machine learning. They highlight that identifying nonanswers using machine learning is “not as straightforward as it seems,” and “we

leave this challenge for future research.” As such, identifying nonanswers in earnings calls is a good case study to evaluate the ability of GLLMs to solve challenging textual analysis problems.

### 4.1. Nonanswer Classification

I retrieve earnings call transcripts through the Finnhub.io API and limit the sample to Central Index Keys that exist in the Gow et al. (2021) sample. To maximize Finnhub.io coverage and enable an analysis of the COVID-19 period, I focus on a 10-year sample period from 2013 to 2022. This yields a sample of 63,959 earnings calls after requiring that the call includes a presentation section and includes at least three Q&A pairs. Processing each transcript and extracting the valid Q&A pairs results in a sample of 1,152,505 Q&A pairs.<sup>7</sup>

To develop my GLLM approach (i.e., the GPT method), I first examine several random samples of Q&A pairs to identify nonanswers and help select and refine the best approach and prompt structure. After selecting my final approach and prompts, I draw and manually classify a final sample of 500 Q&A pairs to serve as the evaluation sample.<sup>8</sup> The manual classifications make two observations clear. First, defining the exact boundary between an answer and a nonanswer is challenging and arguably somewhat subjective. Second, it is generally hard to evaluate whether a response is a nonanswer without the context of the underlying question. As a result, identifying nonanswers requires a tolerance for ambiguity, a degree of reasoning ability, and the ability to evaluate the answer in the context of the question. Traditional ML approaches struggle with these aspects, which explains their low performance for the task of identifying nonanswers.

I replicate the baseline Gow et al. (2021) measure using their available code. Table 1, column (2) shows the performance results for this baseline.<sup>9</sup> The baseline measure shows an accuracy of ~86% and a precision of ~58%, which is in line with Gow et al. (2021). My replication of the baseline measure yields a higher type II error of 57% resulting in a lower nonanswer F1 score of 49%. These results indicate that the baseline measure often misses nonanswers as well as incorrectly label answers as nonanswers, which Gow et al. (2021) acknowledge in their paper.<sup>10</sup>

I explore various approaches using ChatGPT and GPT-4 before settling on the prompts shown in Appendix D and Online Appendix 2. Starting with the zero-shot approaches, the ChatGPT method (Table 1, column (3)) substantially improves over the baseline with an accuracy of 91%, a nonanswer F1 score of 72%, and a substantially lower type II error of 26%. The GPT-4 method (column (4)) outperforms the previous approaches with an accuracy of 93%, a nonanswer F1 score of 79%, and a type II error of only 8%. It is worth highlighting that ChatGPT and GPT-4 provide this level of performance only through prompt engineering without receiving any



**Table 1.** Nonanswers: Classification Report

	(1) Manual	(2) Gow et al. (2021)	(3) ChatGPT zero shot	(4) GPT-4 zero shot	(5) Keyword + GPT-3 filter + ChatGPT	(6) (5) + ChatGPT FT filter
Answer	423	443	418	398	408	423
Nonanswer	77	57	82	102	92	77
Accuracy		0.86	0.91	0.93	0.94	0.96
Type I error		0.06	0.06	0.07	0.05	0.02
Type II error		0.57	0.26	0.08	0.10	0.13
Nonanswers						
Precision		0.58	0.70	0.70	0.75	0.87
Recall		0.43	0.74	0.92	0.90	0.87
F1 score		0.49	0.72	0.79	0.82	0.87
Total						
Precision		0.85	0.91	0.94	0.94	0.96
Recall		0.86	0.91	0.93	0.94	0.96
F1 score		0.86	0.91	0.93	0.94	0.96
<i>N</i>	500	500	500	500	500	500
Mean tokens per Q&A pair			435	550	300	335
Costs per 1,000		\$0	\$0.66	\$16.98	\$0.48	\$1.06

Notes. This table reports the performance statistics of the various nonanswer classification approaches. ChatGPT refers to the gpt-3.5-turbo model. A full description of the methods is available in Section 4.1 and Online Appendix 3. See Endnote 2 for a conceptual introduction to the precision, recall, and F1 scores. The scores are based on the number of true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN). Accuracy =  $(TP + TN)/(TP + TN + FP + FN)$ , type I error =  $FP/(FP + TN)$ , type II error =  $FN/(FN + TP)$ , precision =  $TP/(TP + FP)$ , recall =  $TP/(TP + FN)$ , and F1 score =  $TP/(TP + 0.5 \times (FP + FN))$ .

examples or task-specific training. But the prompt engineering is important; even minor changes, such as removing the distributional statement (i.e., "... these sentences are rare, in 65% ..."), already significantly drop performance. Taken together, the primary downside of these approaches is that scaling zero-shot GPT-4 to the full sample is expensive, and the ChatGPT results could be improved.

In columns (5) and (6) of Table 1, I explore an alternative approach of combining traditional methods with GPT models to outperform zero-shot GPT-4 at a more reasonable cost. At a high level, I develop a funnel approach with which I use cheaper methods to filter down the sample and resolve only the remaining harder classifications using GLLMs. The full approach is available in Online Appendix 3. In summary, I first use a keyword list to identify responses that have any possibility of being a nonanswer. Forty-two percent of the Q&A pairs do not match any keywords and, thus, do not require further processing. Next, I train a machine learning model (a fine-tuned GPT-3 Babbage model in this case) to further filter out sentences that are highly likely not reflecting a nonanswer. This reduces the possible nonanswer Q&A pairs to only approximately a third of the original sample. The remaining sample contains the most difficult problem of separating out the true nonanswers from those appearing to be nonanswers. To solve this task, I frame a zero-shot ChatGPT prompt (see Appendix D.2) as an evaluation task and highlight the sentences possibly reflecting a nonanswer. Column (5) of Table 1 shows that this approach matches the

performance of GPT-4 with an accuracy of 94% and a nonanswer F1-score of 82%, being only 1/35th the price of GPT-4.

For most research tasks, this would suffice. But to fully demonstrate the potential of GLLMs, I add a final step by fine-tuning a ChatGPT (GPT-3.5) model to further improve performance, simultaneously classifying multiple other nonanswer dimensions. Doing so marginally increases the costs, but it brings the performance up to an accuracy of 96% with a nonanswer precision and recall of 87%.<sup>11</sup> This reflects a 70% reduction in nonanswer error rates over the Gow et al. (2021) method.<sup>12</sup> Taken together, the results in Table 1 show that GLLMs can solve hard NLP problems when set up correctly. The approaches in columns (5) and (6) also highlight that combining GLLMs with existing methods can help achieve state-of-the-art performance at a reasonable cost.<sup>13</sup>

Table 2 shows the results of applying the GPT method (i.e., Table 1, column (6)) and the Gow et al. (2021) method on the complete sample of 1,152,505 earnings call Q&A pairs. At the Q&A pair (filing) level, the GPT method identifies 13.9% (14.2%) of Q&A pairs as containing a nonanswer. The Gow et al. (2021) method identifies 12.3% (12.7%) of the pairs as containing a nonanswer. These results are close in aggregate, but the Venn diagram in Figure 2 shows that there is considerable disagreement between the two methods. Only 50% of the nonanswers by the GPT method are also identified by the Gow et al. (2021) method. Furthermore, only 57% of the Gow et al. (2021) nonanswers are considered nonanswers by the GPT method. Consistent

**Table 2.** Nonanswers: Large Sample Analysis

	(1) Observations	(2) Mean	(3) Standard deviation	(4) P5	(5) P25	(6) P50	(7) P75	(8) P95
Question-answer pair level								
GPT method - % nonanswer	1,152,505	0.139	0.35	0	0	0	0	1
Gow method - % nonanswer	1,152,505	0.123	0.33	0	0	0	0	1
Earnings call level								
Number of Q&A pairs	63,959	18.019	8.93	6	12	17	23	34
GPT method - # of nonanswers	63,959	2.502	2.20	0.00	1.00	2.00	4.00	7.00
GPT method - % of nonanswers	63,959	0.142	0.12	0.00	0.06	0.13	0.20	0.36
Gow method - # of nonanswers	63,959	2.210	2.10	0.00	1.00	2.00	3.00	6.00
Gow method - % of nonanswers	63,959	0.127	0.12	0.00	0.04	0.11	0.19	0.33

*Notes.* This table reports the nonanswer statistics for the full sample. The GPT method corresponds to the method from Table 1, column (6), and the Gow method corresponds to the Gow et al. (2021) method.

with Table 1 and shown in Online Appendix 4, the predictions by the GPT method are considerably better with fewer errors.

#### 4.2. Classification of Nonanswer Dimensions

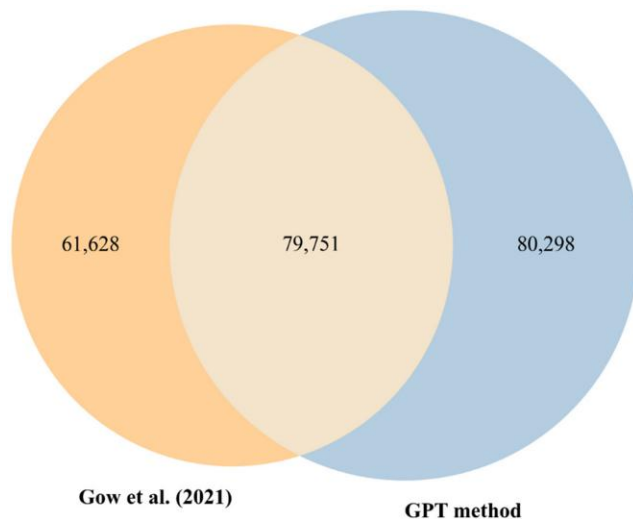
The flexibility of GLLMs makes it possible to construct prompts that yield multiple classifications at once with little additional costs. To demonstrate this, I fine-tune a ChatGPT model in the last step of the GPT method to also categorize nonanswers across multiple additional dimensions: the justification for not providing an answer, the type of nonanswer, the type of question preceding

the nonanswer, and whether the manager tries to give a positive sentiment spin to the nonanswer. Appendix A lists the possible outcomes for each dimension, and Online Appendices 2 and 3 provide the full prompt as well as a full description of the fine-tuning approach. In summary, I use zero-shot GPT-4 to create an initial training data set, I manually go through these responses to correct mistakes, and then I fine-tune the ChatGPT model on the resulting data set.<sup>14</sup>

Table 3 shows the out-of-sample performance of both GPT-4 and the fine-tuned ChatGPT model, split out by each of the dimensions. The GPT-4 approach (columns (1)–(3)) yields a high level of performance across all classification categories with F1 scores ranging from 85% to 98% but at a steep cost. The fine-tuned ChatGPT approach (columns (4)–(6)) yields a better nonanswer classification performance of 93% but slightly lower F1 scores for the four additional dimensions with F1 scores ranging from 81% to 94%. Importantly, the fine-tuning approach is many magnitudes cheaper than the GPT-4 approach, yielding comparable performance. Table 3 also highlights that fine-tuned models tend to underperform for classifications with diverse outcomes such as “other” because of insufficient examples. Conversely, zero-shot GLLMs can excel in these scenarios by leveraging their ability to generalize from seen to unseen outcomes.

Figure 3 portrays the 160,049 nonanswers by each dimension for the full sample. Only ~17% of the nonanswers reflect a complete refusal to answer the question; instead, managers often provide an alternative qualitative answer (~71%). Most commonly, nonanswers are a response to requests for a breakdown (~21%) or forward-looking information (~42%). In 55% of the nonanswers, the managers try to paint a positive picture. Table 4 shows the correlation between question types and the nonanswer dimensions. Managers generally justify not answering a question about breakdowns or related parties by declining to answer for policy or proprietary reasons (21.225\*\*\* and 21.220\*\*\*).<sup>15</sup> On the

**Figure 2.** (Color online) Nonanswers: Venn Diagram



*Notes.* The Venn diagram shows the overlap in nonanswer classifications between the GPT and Gow et al. (2021) methods. The classification report in Table 1 highlights that the GPT method significantly outperforms the Gow et al. (2021) method. To further illustrate the improvement, I show 15 random examples of nonanswers from each of the three areas of the Venn diagram in Online Appendix 4. As the random samples show, both approaches make mistakes. However, the rate of mistakes is much lower for the GPT method. The right outer slice primarily consists of nonanswers missed by the Gow et al. (2021) method, and the left outer slice primarily consists of answers falsely classified as nonanswers by the Gow et al. (2021) method.

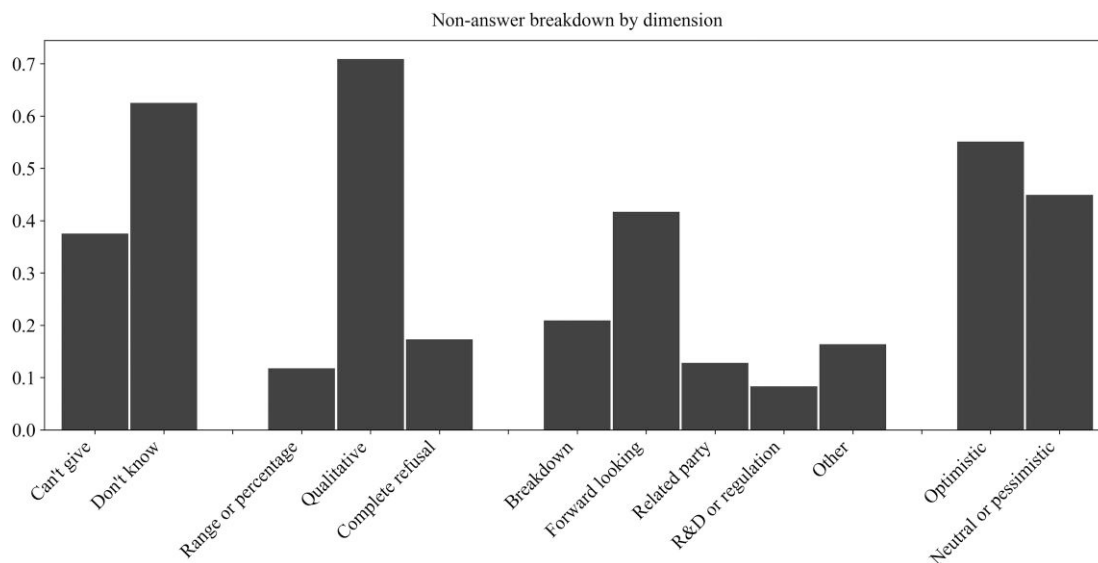
**Table 3.** Nonanswer Dimensions: Model Performance

	(1)	(2) GPT-4 - zero shot		(3)	(4) Fine-tuned ChatGPT		(6)
	Precision	Recall	F1 score	Precision	Recall	F1 score	
Answer	1.00	0.41	0.58	0.93	0.73	0.82	
Nonanswer	0.86	1.00	0.93	0.93	0.98	0.96	
	<i>0.89</i>	<i>0.88</i>	<i>0.85</i>	<i>0.93</i>	<i>0.93</i>	<i>0.93</i>	
Cannot give	0.98	0.96	0.97	0.93	0.91	0.92	
Do not know	0.98	0.99	0.98	0.95	0.96	0.95	
	<i>0.98</i>	<i>0.98</i>	<i>0.98</i>	<i>0.94</i>	<i>0.94</i>	<i>0.94</i>	
Range or percentage	0.88	0.86	0.87	0.73	0.73	0.73	
Qualitative	0.95	0.92	0.93	0.90	0.92	0.91	
Complete refusal	0.84	0.93	0.88	0.90	0.85	0.87	
	<i>0.92</i>	<i>0.92</i>	<i>0.92</i>	<i>0.88</i>	<i>0.88</i>	<i>0.88</i>	
Breakdown	0.96	0.86	0.90	0.83	0.79	0.81	
Forward-looking	0.96	0.94	0.95	0.89	0.87	0.88	
Related party	0.89	0.93	0.91	0.70	0.84	0.76	
R&D or regulation	0.85	0.92	0.88	0.86	0.90	0.88	
Other	0.78	0.87	0.82	0.65	0.63	0.64	
	<i>0.91</i>	<i>0.91</i>	<i>0.91</i>	<i>0.82</i>	<i>0.81</i>	<i>0.81</i>	
Optimistic	0.92	0.93	0.92	0.87	0.88	0.87	
Neutral or pessimistic	0.92	0.91	0.91	0.86	0.85	0.86	
	<i>0.92</i>	<i>0.92</i>	<i>0.92</i>	<i>0.87</i>	<i>0.87</i>	<i>0.87</i>	
N		400			400		
Mean tokens per Q&A pair		1,267			324		
Costs per 1,000, \$		45.53			4.18		

Notes. This table reports the performance statistics for the last step of the GPT method. The statistics are based on an evaluation sample of 400 random Q&A pairs that remain after the first three steps of the GPT method. Online Appendices 2 and 3 include the GPT-4 prompt and a full description of the ChatGPT fine-tuning procedure. The weighted average scores are shown in italic.

contrary, they are more likely to state they do not know the answer to forward-looking questions or those about R&D and regulation ( $-17.545^{***}$  and  $-9.988^{***}$ ). Managers mostly give a positive spin to their nonanswers

when analysts ask about forward-looking statements ( $10.048^{***}$ ). Lastly, managers are most likely to directly refuse to answer questions about related parties or research and development (R&D) and regulations

**Figure 3.** Nonanswer Breakdown by Dimension

Notes. The chart shows the nonanswers split out by each of the nonanswer dimensions. Within each dimension group, the bars add up to 100% to account for all the nonanswers.

**Table 4.** Nonanswer Dimensions: Single Variable Regressions

	(1) Nonanswer: cannot give	(2) Nonanswer: positive	(3) Nonanswer: full reject	(4) Nonanswer: qualitative	(5) Nonanswer: range or percentage
<i>Q - Breakdown</i>	21.225*** (0.750)	−5.175*** (0.561)	−3.368*** (0.309)	−5.751*** (0.447)	9.119*** (0.249)
<i>Q - Forward-looking</i>	−17.545*** (0.878)	10.048*** (0.563)	−5.836*** (0.399)	5.130*** (0.574)	0.706** (0.305)
<i>Q - Related party</i>	21.220*** (1.135)	−2.111*** (0.426)	9.815*** (0.380)	−3.580*** (0.426)	−6.235*** (0.320)
<i>Q - R&amp;D or regulation</i>	−9.988*** (1.088)	−8.191*** (0.973)	8.606*** (0.732)	−1.400 (0.812)	−7.206*** (0.371)
<i>Q - Other</i>	−6.222*** (0.910)	−5.331*** (0.471)	1.682*** (0.321)	1.530*** (0.392)	−3.211*** (0.165)
Multivariate	No	No	No	No	No
N	160,049	160,049	160,049	160,049	160,049

*Notes.* This table reports the regression results of regressing the question characteristic independently of the nonanswer outcomes at the Q&A pair level. The results are shown in columnar format for expositional reasons, they are not multivariate. Standard errors are clustered by firm and year and reported in parentheses.

\*, \*\*, and \*\*\* represent significance levels of 0.10, 0.05, and 0.01, respectively.

(9.815\*\*\* and 8.606\*\*\*). Online Appendix 6 provides an additional analysis in which I regress question features, such as sentiment and readability, on the likelihood of nonanswers and the various dimensions.

Lastly, I perform a time series analysis to study nonanswers over time and evaluate the impact of the COVID-19 pandemic. The results in Figure 4(a) show a relatively stable trend with an increase in nonanswers during COVID-19. The breakdown in Figure 4(b) reveals that this increase in nonanswers during COVID-19 is driven by an increase in managers providing qualitative nonanswers to forward-looking questions and justifying them by not knowing the answer.

## 5. Discussion

### 5.1. GLLM Training Data: Biases, Source Material, and Recent Events

Researchers using GLLMs should be aware of the potential biases, source concerns, and limitations regarding recent events in the GLLM training data. GLLMs are trained on extensive textual corpora, and they can reproduce biases present in that data. For example, Kaplan et al. (2024) find that ChatGPT tends to reproduce gender-based language biases that are commonly present in human-written reference letters although these differences do vary across prompts and language categories. These biases could also surface, for example, when using ChatGPT to classify job postings for a research project. Researchers need to evaluate the outputs for potential biases and, when necessary, mitigate them through prompt engineering or fine-tuning procedures.

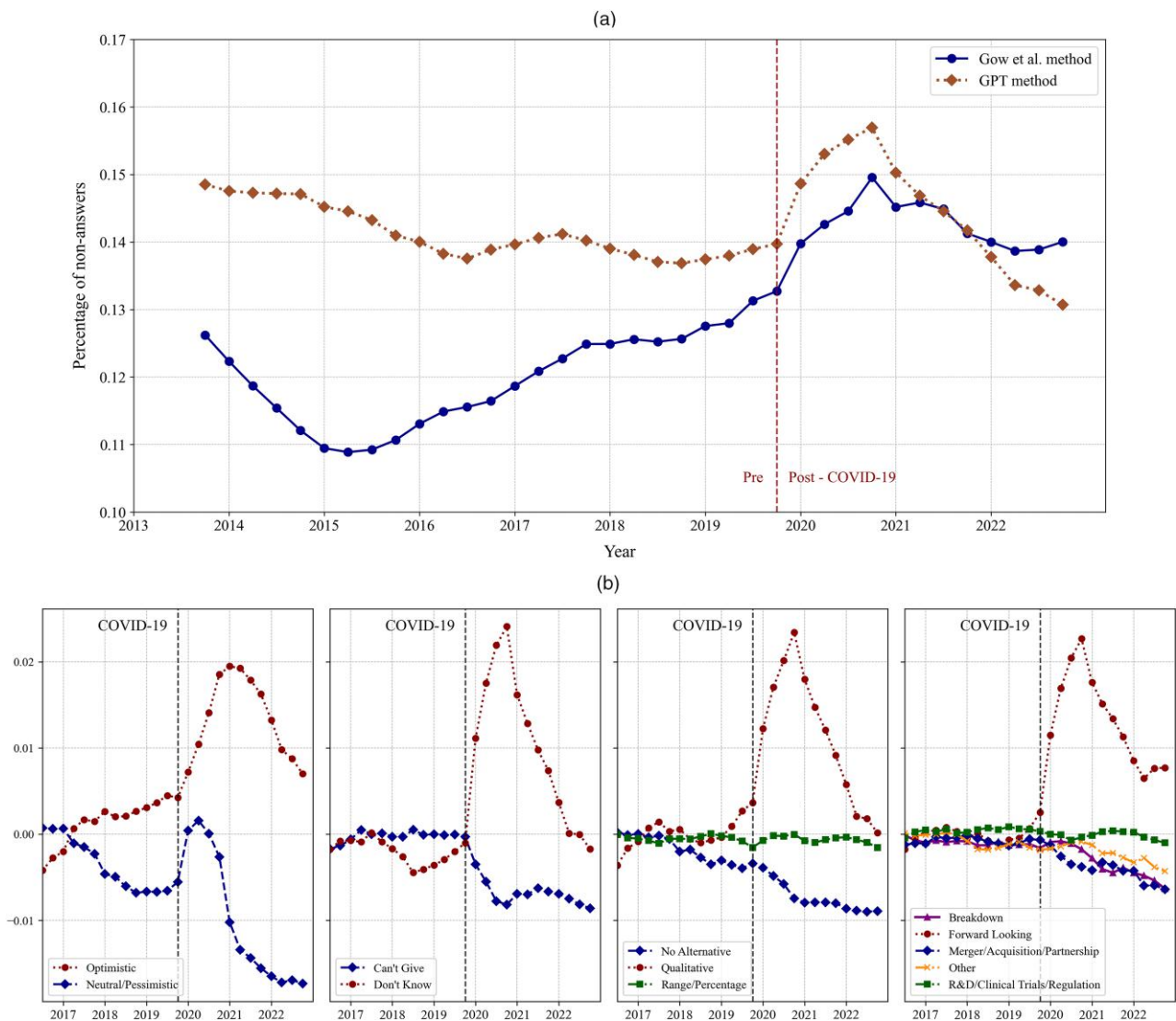
Source concerns arise because GLLMs are trained on a large, diverse set of sources, making it difficult to know exactly what information on which the model is relying to make its generations. The lack of a clear

source can make attribution challenging as well as make the model behavior harder to predict. Explicitly providing the important information to your model through prompt or fine-tuning can reduce those concerns. For example, generations by a GLLM might suggest that it understands the FASB taxonomy. However, the source material used to develop that understanding is unclear and might be incorrect or out of date. Passing it the explicit portion of the FASB taxonomy that you want it to use as part of your prompt resolves that concern and gives greater control. Lastly, GLLMs are trained on historical data up to a certain point. Any events or changes outside of the training sample are not reflected in the output of the model. Researchers must account for these limitations when interpreting the results and, if necessary, incorporate up-to-date information through their prompts or by fine-tuning the model using more recent data.

### 5.2. Replicability

Ensuring replicability is a challenging but important task, especially when working with complex data and procedures. The GLLM approach faces two specific replicability challenges. First, GLLMs are powered by large neural networks with billions of parameters that work in ways we do not fully understand. Outputs for a given prompt can vary significantly between different models, versions of the same model, and sometimes even from generation to generation. For example, Chen et al. (2023) document significant differences in the outputs from GPT-4 when comparing the variant from March 2023 with that of June 2023. Second, accessing GLLMs is most convenient through third-party providers. However, these providers can change the models or make them unavailable at any time, which can make it difficult to ensure full replicability.



**Figure 4.** (Color online) Nonanswers: Time Trends and COVID-19 Impact

**Notes.** The two time series plots show the trend of nonanswers over time. Panel (a) shows the percentage of nonanswer Q&A pairs for each quarter, including both the Gow et al. (2021) and GPT methods. Each quarterly percentage is normalized by the four-quarter moving average. Panel (b) shows the same results split out by each of the nonanswer dimensions. To make the panels comparable, I center the quarterly percentage by the average of the corresponding dimension in the period prior to COVID-19, which are the years 2014–2020. (a) Nonanswer percentage over time (four-quarter moving average). (b) Change in four-quarter moving average relative to the pre-COVID-19 period.

The first option to improve replicability is not to rely on third-party APIs and use only GLLMs that you can store and run locally. Online Appendix 5 and the companion website provide a discussion and code examples of how to locally use and fine-tune GLLMs. Running the model yourself yields more control and replicability. However, it does not resolve all replicability concerns. For example, access to the required specialized hardware can be expensive, not available, or require a prohibitive degree of technical know-how and expertise for others to replicate. Furthermore, running neural networks on different sets of hardware can still create minor differences in the outputs, making exact replication not a guarantee. For many projects the benefits of

using a third-party API will outweigh the challenges that come with running a GLLM locally.

A more broadly applicable recommendation is to always assume that one can lose access to the exact source of the generations. The exact source could be a local GLLM, a GLLM behind an API, or even human coders. With that assumption in mind, it becomes critical to always store and backup the raw prompts and completions exactly as they enter in and exit out of the source. Having access to the raw prompts and completions enables replicating all later analyses and is second best to a deterministic source. Making your generations as deterministic as possible also helps. Setting the “temperature” hyperparameter to zero makes the GLLM always pick

the highest probability tokens, which makes the generations as close to deterministic as possible. Finally, being transparent in your empirical choices, such as your data, your task, your prompt design, your approach, and your model(s) is always a good practice. Sharing your code, when possible, is also a straightforward way to reduce ambiguity and improve replicability.

### 5.3. Data Privacy and Copyright

Researchers using GLLMs need to be careful about sharing their research data with third-party providers of APIs or other computing resources. Sharing research data with external parties as part of your prompts or fine-tuning can result in data privacy, copyright, or other contractual violations. Besides the legal ramifications, API providers might use your prompts and generations to improve and retrain their models. This retraining can result in GLLMs unintentionally disclosing sensitive information in their future outputs to others. To mitigate this risk, researchers should avoid using third-party APIs when data sharing is not allowed or desirable, especially when the API providers state in their terms of service that prompts can serve as additional training data.

### 5.4. Current GLLM Developments

GLLMs are developing quickly in a few notable areas. Speed optimization and lower processing requirements are a primary area of progress. These optimizations will continue to make GLLMs more usable as well as more affordable and scalable. Furthermore, there are many ongoing developments to improve open-source models to make them more competitive with closed-source models, such as the Llama2 models and others (e.g., Wang and Komatsuzaki 2021; Black et al. 2022; Touvron et al. 2023a, b). Better open-source models will make it easier to run GLLMs locally, which helps reduce concerns around replicability and data privacy on top of further reducing costs.

The primary focus of this paper is on using GLLMs to process text. However, recently, multimodal GLLMs have become increasingly capable and available. Multimodal models can accept other types of inputs in addition to text, such as audio and images. For example, GPT-4-Vision by OpenAI and Gemini Pro Vision by Google can handle both text and images as input. Multimodal models enable an even greater range of tasks and problems that are automatable using GLLMs, such as analyzing data visualizations, challenging PDF files, corporate presentations, video interviews, and audio transcripts. In addition to multimodal models, there are also multiple platforms that specialize in making it easier to incorporate specific types of alternative data into text prompts, such as the PDF.ai API to incorporate PDF data more easily as part of your prompt.

Another area of work is to connect GLLMs to information sources beyond their original training data, such as retrieval augmented generations. These implementations demonstrate the potential to improve the knowledge and information lookup capabilities of GLLMs. For example, one could enable a GLLM to search the FASB taxonomy and perform a domain-specific textual analysis task. Connecting GLLMs to new sources has many applications but also raises questions about reliability and replicability, of which future researchers need to be mindful.

Finally, most current GLLMs focus on the English language. These models commonly have basic capabilities in other languages, but the state-of-the-art performance is generally limited to English texts. Multiple developments are in progress to provide GLLMs with other language focuses, such as the Ernie model for Chinese NLP tasks (Sun et al. 2021). These non-English GLLMs will make it easier to conduct textual analysis research on non-English documents.

### 5.5. Using GLLMs for Writing and Programming

GLLMs are also increasingly popular as writing and programming assistants, which raises specific concerns and challenges. In the context of writing, the primary concern with GLLMs is attributability and the factuality of the writing. For example, Bhattacharyya et al. (2023) find in the context of medical research that only 7% of the academic references generated by ChatGPT are authentic and accurate. GLLMs can help proofread or copyedit academic writing, but do not blindly rely on the writing of a GLLM. Instead, carefully check the generations for inaccuracies, misattributions, or other mistakes.

In the context of programming, GLLMs can serve as useful productivity tools to write and debug code. However, the research and coding problems encountered in academia are often rare and unusual unlike commonly used coding problems found on the internet. Current GLLMs excel at solving coding problems that are prevalent in the training data but can struggle with more complex, rare, or domain-specific coding tasks. Minor mistakes in the code base for a research project can have major impacts on the validity of results and inferences. Code written or modified by a GLLM should always be validated for its accuracy. Researchers should not use GLLM coding assistants to write code they do not understand and cannot appropriately evaluate and scrutinize. In that regard, I echo the call by Bochkay et al. (2023) for researchers to still invest time in developing textual analysis and coding skills and for institutions to dedicate resources to provide training and guidance. The return on investment of these skills is even higher now that GLLMs exist as they enable you to leverage your skills more effectively.

## 6. Conclusion

GLLMs offer significant potential for academic research analyzing textual data. GLLMs provide a powerful and versatile alternative to traditional machine learning methods and human coding. This paper provides an introduction and framework on how to effectively utilize GLLMs for a research project, also highlighting their limitations and challenges in terms of controllability, construct validity, replicability, and data sharing. In a case study, I demonstrate the capabilities of GLLMs by detecting nonanswers in earnings conference calls, a traditionally challenging task to automate. The new GPT method achieves an accuracy of 96%, which reflects a 70% reduction in nonanswer error rates over the Gow et al. (2021) method. The Online Appendix and companion website provide additional practical resources, such as a detailed comparison of models and code examples to get started with GLLMs. Taken together, this paper equips researchers, reviewers, and editors with the knowledge and tools to effectively use and evaluate GLLMs in academic research.

## Acknowledgments

The author thanks Eric So (editor), Beth Blankespoor, Ed deHaan, Andy Leone, Dawn Matsumoto, Brian Miller, Tim Martens, Xue Li, an anonymous associate editor, and two anonymous reviewers for helpful comments. The author thanks FinnHub.io for providing the conference call data set and Gow et al. (2021) for publicly sharing their code and data. A prior version of this paper was titled “Generative LLMs and Textual Analysis in Accounting: (Chat)GPT as Research Assistant?”. GPT-4 and ChatGPT were used to help with copyediting and coding assistance. The companion website can be found at [https://github.com/TiesdeKok/chatgpt\\_paper](https://github.com/TiesdeKok/chatgpt_paper).

## Appendix A. Definitions

### A.1. Common GLLM Terminology

Terminology	Definition
API	A communication protocol to interact with an internet service
Completion	The text output generated by the GLLM
Few shot	Providing a prompt with a few examples as part of the instructions
Fine-tuning	Retraining (part of) the model weights for a specific task.
GLLM	Generative large language model
GPT	Generative pretrained transformer
Inference	Using a GLLM to make a generation
LLM	Large language model
Model weights	The parameter values (coefficients) of the neural network
Prompt	The text input provided to the GLLM
Prompt engineering	The task of developing a prompt for your task
Token	The unit of input for a GLLM—can be a word, part of a word, or a character
Zero shot	Providing a prompt without any examples, just the instruction

### A.2. Nonanswer Dimensions

Dimension	Outcome	Definition
Justification	Cannot give	The manager justifies the inability to answer because of reasons such as proprietary concerns, company policy, or legal reasons.
	Do not know	The manager justifies the inability to answer because it is too early, they do not know the answer, or do not have the information.
Nonanswer type	Complete refusal	The manager completely refuses to provide any answer or information.
	Qualitative	The manager does not directly answer, but instead provides a qualitative alternative answer.
	Range or percentage	The manager does not directly answer, but instead provides a quantitative range or percentage.
Question type	Breakdown	The participant requests more detail on something. In most cases, they request a number breakdown, such as by segment.
	Forward-looking	The participant requests guidance or a forward-looking statement.
	Related party	The participant asks about things such as acquisitions, partnerships, major customers, or suppliers.
	R&D or regulation	The participant asks about R&D projects, clinical trials, or regulatory/certification matters.
	Other	A question that falls outside the above four question types.
Sentiment spin	Optimistic	A nonanswer accompanied by an optimistic statement (e.g., “I can’t give you the numbers, but things are looking good!”)
	Neutral or pessimistic	The manager does not provide an optimistic statement.

## Appendix B. Simple Examples

This appendix provides simple examples of how one would use GLLMs and prompt engineering to solve various types of textual analysis tasks. To save space, the prompts are shown as their zero-shot version, but they are easily adapted to the few-shot or fine-tuning approaches.<sup>16</sup>

Identifying statements about political risk:

Identify whether the earnings call transcripts below discuss political risks or not.

1. gaming industry is currently supporting a ballot initiative to amend the constitution to authorize an increase in the –BET– limits

2. Now on to the third quarter results. Cano Health continued to achieve steady organic

growth in Q3. Total membership grew 4.6% from the second quarter to approximately 295,000 members, an increase of 40% year-over-year.

3. governments both President Obama's budget proposal and separate legislation –PENDING– in Congress would provide funding to continue the Medicaid stimulus for another

Return the result in the following JSON format:

```
[{"discusses_political_risk": Yes/No}]  
JSON =
```

This yields the following output when passed to GPT-4:

```
[{"discusses_political_risk": "Yes"},  
 {"discusses_political_risk": "No"},  
 {"discusses_political_risk": "Yes"}]
```

Extracting earnings before interest, taxes, depreciation, and amortization (EBITDA)-like numbers from the management discussion and analysis section:

Extract all EBITDA-related references and quantitative amounts from the paragraph below. Do not convert values and report them as reported.

> In the third quarter of 2021, our EBITDA increased to \$1.2 billion, primarily driven by higher sales volumes and improved operational efficiency. Our Debt-to-EBIDAR increased by 20%. This growth was partially offset by increased raw material costs and unfavorable foreign exchange impacts.

Return the result in the following JSON format:

```
[{"name": ..., "value": ...}]  
JSON =
```

This yields the following output when passed to GPT-4:

```
[{"name": "EBITDA", "value": "$1.2 billion"},  
 {"name": "Debt-to-EBIDAR increase", "value":  
  "20%"}]
```

Topic classification for press release headlines:

Cluster the following company press releases into common categories:

1. Company A announces a new partnership with Company B to develop innovative solutions for the healthcare industry.

2. Company B reports its financial results for the second quarter of 2021, highlighting a 15% increase in revenue compared with the same period last year.

3. Company C unveils its latest initiative to reduce their carbon emissions, a \$2 Billion investment in an all-electric fleet of company vehicles.

Return the result in the following JSON format:

```
[{"index": ..., "category": ...}]  
JSON =
```

This yields the following output when passed to GPT-4:

```
[{"index": 1, "category": "Partnerships"},  
 {"index": 2, "category": "Financial Results"},  
 {"index": 3, "category": "Sustainability  
 Initiatives"}]
```

Extract sentences from broken text:

Extract valid sentence from the following text that got split up:

In 2018, we entered the third year of an ambitious 10-year goal cycle. Our 2025 goals reflect a heightened commitment

Return the result in the following JSON format:

```
[{"i": ..., "sentence": ...}]  
JSON =
```

This yields the following output when passed to GPT-4:

```
[{"i": 0, "sentence": "In 2018, we entered the  
third year of an ambitious 10-year goal cycle."},  
 {"i": 1, "sentence": "Our 2025 goals reflect a  
heightened commitment."}]
```

## Appendix C. Prompt Engineering Tips

### GLLMs Generate Text Left to Right, so Their Own Generations Become Part of the Prompt

A helpful technique to improve completion quality is chain-of-thought prompting, which involves asking the model to first explain itself before making the classification. Because of the left-to-right nature of GLLMs, the explanation becomes part of the context that can help create better and more consistent classifications relative to only asking the model to do the final classification.

### Optimize the Number of Tokens Used by Your Prompt and Completion

Optimizing the number of tokens can save substantial time and money. Reducing the number of tokens by 10% generally yields a 10% speed up and 10% cost saving. Classifying multiple observations through a single prompt can also help save on the number of overall tokens.

### Be Specific and to the Point: Less Is Often More

GLLMs sometimes ignore part of the instructions, which can make them harder to control. Sticking with instructions that are short and explicit can help with this. It can also help to order your instructions in order of importance and include a note in your prompt to explain the order.

### Chaining Prompts Together

For more complex tasks, it might not be possible to do everything through a single prompt. Chaining prompts together can help in this scenario. For example, feeding the first model completions back into the model with another prompt asking it to check or improve the previous completions for specific types of errors can help improve the final classification quality.

### Small Differences in the Prompt Can Make Big Differences in the Completion

Small changes in the prompt can make a big difference when instructing a GLLM. For example, including uncertain words such as "might," "possibly," or "could" can



have a big impact. The influence of a single word or sentence is hard to predict, but trial and error and experimentation are effective ways to tweak the prompt to improve the completion quality and improve consistency.

### It Can Help to Provide Distributional Statements as Part of Your Prompt

A challenge with zero-/few-shot approaches is achieving the desired outcome distribution. Every generation is independent, and the model does not keep track of the distribution of the outcomes. It can help to mention in your prompt that an outcome is rare or that it is encouraged to favor one outcome over another. I demonstrate this technique in my prompt in Online Appendix 2.1, for example.

### It Can Help to Track the Prediction Confidence of the Model

GLLMs can vary in their confidence from prediction to prediction. Instructing the model to also return its confidence level can provide insight into with what type of documents the model is struggling. Model-provided confidence scores can be a helpful indicator, however, they are token generations as any other, and they can be wrong. A model stating it has high confidence does not guarantee a correct prediction, and it remains important to evaluate construct validity using the guidance in Section 3.4.

### Consider Switching Models If You Cannot Get Your Prompt to Work

As I highlight in Online Appendix 1, models can differ in nuanced ways. As a result, it is worth trying different models to see which ones yield the best out-of-the-box understanding of your task. You might find that it is hard to communicate your task to one model but significantly easier to communicate it to another even if the models are of a similar size and tuning.

## Appendix D. Examples of Earnings Call Nonanswer Prompts

This appendix provides two examples of the types of prompts I use for the nonanswer results. The full prompts for all the specifications in Tables 1 and 3 are shown in Online Appendix 2.

E.1: ChatGPT zero-shot prompt (Table 1, column (3)):

In the provided text, the analyst posed the question:

```
> {question}
```

The manager responded with:

```
> {answer}
```

```
**Task 1 - sentences:**
```

From the manager's response, extract only those sentences where the manager explicitly indicates either:

- They currently lack specific details or information to provide.
- They are deliberately choosing not to share specific information at the moment.

Please exclude sentences that discuss general uncertainties, company plans, or any actions the company might take in the future.

Also exclude disclaimers that are immediately followed by an answer.

Important: these sentences are rare, in 65% of the cases this will be an empty list. It is ok to return an empty list.

Provide your response using the following JSON format: JSON = {{

```
  "sentences": []
}}
```

JSON =

E.2: ChatGPT zero-shot prompt after keyword/GPT-3 filter (Table 1, column (5)):

Analyst question:

```
{question}
```

Manager response:

```
{answer}
```

A research assistant has marked the above response as including a statement that reflects unwillingness or inability to answer (part) of the analysts' question, because of the following comment(s):

```
> {comments}
```

Based on the question and full response above, provide a detailed assessment whether the manager's response includes a statement, explanation, or justification indicating an inability or unwillingness to answer the question. If you classify the response as reflecting inability or unwillingness to answer, justify your classification with specific phrases or sentences from the manager's response. If there's no such indication, explain why not.

Your response should be in the following JSON format:

```
JSON = {{
  "assessment": a detailed assessment unique
to this evaluation following the instructions
above,
  "your_classification": 1 or 0
}}
```

JSON =

## Endnotes

<sup>1</sup> See [https://github.com/TiesdeKok/chatgpt\\_paper](https://github.com/TiesdeKok/chatgpt_paper).

<sup>2</sup> The F1 score combines the precision and recall scores. A high precision means the approach effectively avoids false positives, meaning most of the identified nonanswers are true nonanswers. A high recall means the approach effectively avoids false negatives, meaning it captures most of the true nonanswers. Consequently, a high F1 score means the approach is effective at minimizing both false positives and false negatives and performs well overall.

<sup>3</sup> Tokens share some resemblance to syllables, but they are not the same. For example, ChatGPT can tokenize the word "microwave" to the tokens "mic" and "rowave." Tokenizers vary by model but are generally case sensitive and include whitespaces. The OpenAI tokenizer is available online: <https://platform.openai.com/tokenizer>.

<sup>4</sup> For example, the GPT-4 model by OpenAI is likely trained on trillions of words and costs more than \$100 million to train. See <https://www.wired.com/story/openai-ceo-sam-altman-the-age-of-giant-ai-models-is-already-over/>.

<sup>5</sup> The cost savings of combining a GLLM with a nongenerative ML model can be significant. For example, it might cost \$0.004 to classify a paragraph using GPT-4-turbo. For one million paragraphs, that would cost around \$4,000. Instead, it would cost around \$200 to classify 50,000 paragraphs using GPT-4-turbo to form a training sample and a few hundred dollars in compute costs to train a BERT pipeline and run a million inferences through it. In contrast, hiring humans to classify 50,000 paragraphs would be magnitudes more expensive and significantly slower.

<sup>6</sup> Fine-tuning a GLLM shares similarities to fine-tuning a BERT pipeline. However, GLLMs have more parameters, making them more powerful but also significantly harder to fine-tune using local hardware. As such, it is only worth fine-tuning a GLLM over a BERT approach when the task is too complex for a BERT model, the task is only expressible through a natural language prompt, or the task requires a large context length.

<sup>7</sup> My number of Q&A pairs per call is lower relative to existing papers for a few reasons. First, if a question receives multiple answers, I collapse them into a single answer. Second, I only consider Q&A pairs for which the question is at least 30 characters long, the answer is at least 10 characters long, and the combination of question + answer is at least 75 characters long. This helps remove many Q&A pairs that only say things such as “Q: Thank you – A: Thanks!” or “Q: Bye – A: See you next call.” These pairs are not relevant, and including them inflates the classification statistics.

<sup>8</sup> As in Gow et al. (2021), there is a risk of overfitting the human learning (e.g., prompt development) to the evaluation set. The in-/out-of-sample logic helps with this. However, in practice, it is not realistic to create a new unseen evaluation sample after every prompt iteration. Instead, I recommend loosely following the in/out sample methodology together with much output inspection. The objective is to get meaningful outputs, not blindly maximizing the statistics.

<sup>9</sup> There are multiple potential reasons why the replication statistics vary slightly from the original statistics in Gow et al. (2021). For example, I use a different transcript provider, period, way of processing the text, and filtering out of invalid Q&A pairs. Also, resolving edge case scenarios depends on the exact nonanswer definition, which can vary slightly.

<sup>10</sup> As illustrated by this example, a high accuracy can be a less useful metric with highly imbalanced classifications.

<sup>11</sup> The ChatGPT fine-tuning endpoint only became available after I had already completed the original approach in column (5) of Table 1. If I would redo my GPT method as of 2024, I would utilize the lower API costs to simplify the pipeline by only including the keyword filter and feeding the rest directly into a fine-tuned ChatGPT model.

<sup>12</sup> I calculate the 70% reduction with this equation:  $((1 - \text{Accuracy}_{\text{GPT}}) - (1 - \text{Accuracy}_{\text{Gow}})) / (1 - \text{Accuracy}_{\text{Gow}})$ .

<sup>13</sup> The total cost of classifying the ~1.2 million Q&A pairs using my final GPT method was around \$1,300. Running all Q&A pairs through zero-shot ChatGPT or GPT-4 would cost around \$800 and \$20,500 based on the September 2023 prices. With the significantly lower February 2024 prices, my final approach would cost around \$600, zero-shot ChatGPT would cost around \$350, and zero-shot GPT-4-turbo would cost around \$7,200 in total.

<sup>14</sup> Manually going through the responses to correct mistakes is quick and only takes a few hours. In contrast, manually coding all observations from scratch for all dimensions would take a few days up to a few weeks for this task.

<sup>15</sup> \*, \*\*, and \*\*\* represent significance levels of 0.10, 0.05, and 0.01, respectively.

<sup>16</sup> I have chosen prompts that are easier to read for humans, but there are multiple optimizations that one could do to improve their token budget. For example, removing unnecessary newlines and spaces, and shortening the output.

## References

- Anand V, Bochkay K, Chychyla R, Leone A (2020) Using Python for text analysis in accounting research. *Foundations Trends Accounting* 14(3–4):128–359.
- Bahdanau D, Cho K, Bengio Y (2016) Neural machine translation by jointly learning to align and translate. Preprint, submitted September 1, <https://arxiv.org/abs/1409.0473>.
- Bai JJ, Boyson NM, Cao Y, Liu M, Wan C (2023) Executives vs. chatbots: Unmasking insights through human-AI differences in earnings conference Q&A. Preprint, submitted June 22, <https://dx.doi.org/10.2139/ssrn.4480056>.
- Bernard D, Blankespoor E, de Kok T, Toynbee S (2023) Using GPT models to measure the complexity of business transactions. Preprint, submitted June 25, <https://dx.doi.org/10.2139/ssrn.4480309>.
- Bhattacharyya M, Miller VM, Bhattacharyya D, Miller LE (2023) High rates of fabricated and inaccurate references in ChatGPT-generated medical content. *Cureus* 15(5):e39238.
- Black S, Biderman S, Hallahan E, Anthony Q, Gao L, Golding L, He H, et al. (2022) GPT-NeoX-20B: An open-source autoregressive language model. Preprint, submitted April 14, <https://arxiv.org/abs/2204.06745>.
- Bochkay K, Brown SV, Leone AJ, Tucker JW (2023) Textual analysis in accounting: What's next? *Contemporary Accounting Res.* 40(2):765–805.
- Brown SV, Tian XS, Tucker JW (2018) The spillover effect of SEC comment letters on qualitative corporate disclosure: Evidence from the risk factor disclosure. *Contemporary Accounting Res.* 35(2):622–656.
- Brown T, Mann B, Ryder N, Subbiah M, Kaplan JD, Dhariwal P, Neelakantan A, et al. (2020) Language models are few-shot learners. Larochelle H, Ranzato M, Hadsell R, Balcan MF, Lin H, eds. *Advances in Neural Information Processing Systems*, vol. 33 (Curran Associates Inc., Red Hook, NY), 1877–1901.
- Call AC, Flam RW, Lee JA, Sharp NY (2024) Managers' use of humor on public earnings conference calls. *Rev. Accounting Stud.* 29:2650–2687.
- Chen L, Zaharia M, Zou J (2023) How is ChatGPT's behavior changing over time? Preprint, submitted July 18, <https://arxiv.org/abs/2307.09009>.
- Chuk E, Matsumoto D, Miller GS (2013) Assessing methods of identifying management forecasts: CIG vs. researcher collected. *J. Accounting Econom.* 55(1):23–42.
- Devlin J, Chang MW, Lee K, Toutanova K (2019) BERT: Pre-training of deep bidirectional transformers for language understanding. Preprint, submitted October 11, <https://arxiv.org/abs/1810.04805>.
- Gow ID, Larcker DF, Zakolyukina AA (2021) Non-answers during conference calls. *J. Accounting Res.* 59(4):1349–1384.
- Hail L, Tahoun A, Wang C (2018) Corporate scandals and regulation. *J. Accounting Res.* 56(2):617–671.
- Hansen AL, Kazinnik S (2023) Can ChatGPT decipher Fedspeak? Preprint, submitted April 7, <https://dx.doi.org/10.2139/ssrn.4399406>.
- Hassan TA, Hollander S, van Lent L, Tahoun A (2019) Firm-level political risk: Measurement and Effects. *Quart. J. Econom.* 134(4):2135–2202.
- Hollander S, Pronk M, Roelofsens E (2010) Does silence speak? An empirical analysis of disclosure choices during conference calls. *J. Accounting Res.* 48(3):531–563.

- Hu N, Liang P, Yang X (2023) Whetting all your appetites for financial tasks with one meal from GPT? A comparison of GPT, FinBERT, and dictionaries in evaluating sentiment analysis. Preprint, submitted April 26, [https://papers.ssrn.com/sol3/papers.cfm?abstract\\_id=4426455](https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4426455).
- Huang AH, Wang H, Yang Y (2023) FinBERT: A large language model for extracting information from financial text. *Contemporary Accounting Res.* 40(2):806–841.
- Jha M, Qian J, Weber M, Yang B (2024) ChatGPT and corporate policies. Preprint, submitted July 26, <https://dx.doi.org/10.2139/ssrn.4521096>.
- Kaplan DM, Palitsky R, Arconada Alvarez SJ, Pozzo NS, Greenleaf MN, Atkinson CA, Lam WA (2024) What's in a name? Experimental evidence of gender bias in recommendation letters generated by ChatGPT. *J. Medical Internet Res.* 26: e51837.
- Kim AG, Muhn M, Nikolaev VV (2024) Bloated disclosures: Can ChatGPT help investors process information? Preprint, submitted April 21, <https://dx.doi.org/10.2139/ssrn.4425527>.
- Lopez-Lira A, Tang Y (2023) Can ChatGPT forecast stock price movements? Return predictability and large language models. Preprint, submitted April 10, <http://dx.doi.org/10.2139/ssrn.4412788>.
- OpenAI (2023) GPT-4 technical report. Preprint, submitted March 15, <https://arxiv.org/abs/2303.08774>.
- Radford A, Narasimhan K, Salimans T, Sutskever I (2018) Improving language understanding by generative pre-training. *OpenAI* (June 11), <https://openai.com/index/language-unsupervised/>.
- Radford A, Wu J, Child R, Luan D, Amodei D, Sutskever I (2019) Language models are unsupervised multitask learners. *OpenAI* (February 14), <https://openai.com/index/better-language-models/>.
- Rouen E, Sachdeva K, Yoon A (2023) Sustainability meets substance: Evaluating ESG reports in the context of 10-Ks and firm performance. Preprint, submitted March 15, <http://dx.doi.org/10.2139/ssrn.4227934>.
- Siano F, Wysocki P (2021) Transfer learning and textual analysis of accounting disclosures: Applying big data methods to small(er) datasets. *Accounting Horizons* 35(3):217–244.
- Sun Y, Wang S, Feng S, Ding S, Pang C, Shang J, Liu J, et al. (2021) ERNIE 3.0: Large-scale knowledge enhanced pre-training for language understanding and generation. Preprint, submitted July 5, <https://arxiv.org/abs/2107.02137>.
- Touvron H, Lavril T, Izacard G, Martinet X, Lachaux MA, Lacroix T, Rozière B, et al. (2023a) LLaMA: Open and efficient foundation language models. Preprint, submitted February 27, <https://arxiv.org/abs/2302.13971>.
- Touvron H, Martin L, Stone K, Albert P, Almahairi A, Babaei Y, Bashlykov N, et al. (2023b) Llama 2: Open foundation and fine-tuned chat models. Preprint, submitted July 18, <https://arxiv.org/abs/2307.09288>.
- Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Ł, Polosukhin I (2017) Attention is all you need. Guyon I, Von Luxburg U, Bengio S, Wallach H, Fergus R, Vishwanathan S, Garnett R, eds. *Advances in Neural Information Processing Systems*, vol. 30 (Curran Associates Inc., Red Hook, NY).
- Wang B, Komatsuzaki A (2021) GPT-J-6B: A 6 billion parameter autoregressive language model. <https://github.com/kingoflolz/mesh-transformer-jax>.

Copyright of Management Science is the property of INFORMS: Institute for Operations Research & the Management Sciences and its content may not be copied or emailed to multiple sites or posted to a listserv without the copyright holder's express written permission. However, users may print, download, or email articles for individual use.