

Generative LLMs and Textual Analysis in Accounting: (Chat)GPT as Research Assistant?

Ties de Kok
tdekok@uw.edu
University of Washington

October 2023

Working paper
Comments and feedback are welcome!

Abstract

Generative Large Language Models (GLLMs), such as the ChatGPT and GPT-4 models by OpenAI, are emerging as powerful tools for textual analysis tasks in accounting research. GLLMs can solve any textual analysis task solvable using non-generative methods, as well as tasks previously only solvable using human coding. This paper highlights the applications of GLLMs for accounting research and compares them to existing textual analysis methods. I also provide a framework for researchers to effectively utilize GLLMs in their work, addressing key considerations such as model selection, prompt engineering, and construct validity. In a case study, I demonstrate the capabilities of GLLMs by detecting non-answers in earnings conference calls, a traditionally challenging task to automate with high accuracy. I also demonstrate the flexibility of GLLMs by measuring various additional non-answer dimensions, such as the managers' justification and sentiment spin, and studying them during the COVID-19 pandemic. The non-answer dataset, and code examples, are made available through a companion website. Finally, I discuss the importance of addressing bias, replicability, and data privacy concerns when using GLLMs. Taken together, this paper equips researchers with the necessary knowledge and tools to harness the potential of GLLMs and editors and reviewers with the knowledge to better evaluate papers that use GLLM approaches.

LLM assistance statement: GPT-4 and ChatGPT were used to help with copy-editing and coding assistance.

Acknowledgements: I thank Beth Blankespoor, Ed deHaan, Andy Leone, Dawn Matsumoto, Brian Miller, and Xue Li for helpful comments. I thank Finnhub.io for providing the conference call dataset and Gow et al. (2021) for publicly sharing their code and data.

Companion website: https://github.com/TiesdeKok/chatgpt_paper

Generative LLMs and Textual Analysis in Accounting: (Chat)GPT as Research Assistant?

October 2023

Abstract

Generative Large Language Models (GLLMs), such as the ChatGPT and GPT-4 models by OpenAI, are emerging as powerful tools for textual analysis tasks in accounting research. GLLMs can solve any textual analysis task solvable using non-generative methods, as well as tasks previously only solvable using human coding. This paper highlights the applications of GLLMs for accounting research and compares them to existing textual analysis methods. I also provide a framework for researchers to effectively utilize GLLMs in their work, addressing key considerations such as model selection, prompt engineering, and construct validity. In a case study, I demonstrate the capabilities of GLLMs by detecting non-answers in earnings conference calls, a traditionally challenging task to automate with high accuracy. I also demonstrate the flexibility of GLLMs by measuring various additional non-answer dimensions, such as the managers' justification and sentiment spin, and studying them during the COVID-19 pandemic. The non-answer dataset, and code examples, are made available through a companion website. Finally, I discuss the importance of addressing bias, replicability, and data privacy concerns when using GLLMs. Taken together, this paper equips researchers with the necessary knowledge and tools to harness the potential of GLLMs and editors and reviewers with the knowledge to better evaluate papers that use GLLM approaches.

LLM assistance statement: GPT-4 and ChatGPT were used to help with copy-editing and coding assistance.

1. Introduction

Rapid advancements in natural language processing and machine learning 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 research assistants (RAs) while enjoying the scalability, speed, and cost benefits of machine learning methods. For example, a well-instructed GLLM can extract emission targets in ESG 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. Solving these problems using a GLLM can yield better results at a fraction of the researchers' time relative to using non-generative machine learning (ML) approaches or human coding. This paper demonstrates the application of GLLMs for accounting research and provides a framework and discussion that researchers, reviewers, and editors can use when working with or evaluating GLLMs.

GLLMs significantly improve on existing textual analysis approaches in several ways. First, they operate through natural language, which makes it easier to express a wide range of textual analysis tasks. Second, they can complete many tasks with no or little training data, which saves significant time and resources. Tasks that would previously take months might take only a couple of weeks to solve using a GLLM. For example, Hassan et al. (2019) identify discussions about political risk in earnings calls by training an algorithm on a large training dataset of political texts. Achieving this is a challenging and time-consuming task without GLLMs. In contrast, a state-of-the-art GLLM like GPT-4 can solve this problem out-of-the-box by simply showing it a call

excerpt and asking whether it discusses political risk or not.¹ Less powerful GLLM, such as GPT-3 or Llama2, can yield similar performance with appropriate instructions or training. Furthermore, GLLMs exhibit basic reasoning abilities and world understanding and can solve problems that would otherwise be infeasible to automate. For example, Gow et al. (2021) resort to a manual rule-based approach to detect non-answers in earnings conference calls. They find that existing machine learning methods are not able to deal with the complexity and nuance of non-answers. GLLM approaches can solve these types of complex tasks, as I demonstrate in the case study of this paper. Taken together, GLLMs show the potential to lower the barrier of entry to analyzing textual data as well as enable researchers to study new research questions previously not feasible.

While GLLMs are powerful, they come with their own challenges and limitations. GLLMs require a new way of working to ensure valid outputs and safeguard construct validity. In many ways GLLMs require a way of thinking that falls between interacting with a human and a non-generative machine learning model. Their rapid rise to popularity is accompanied by a lack of guidance on how to use GLLMs effectively and properly for academic research. For example, the recent overview of textual analysis methods in accounting by Bochkay et al. (2022) does not cover GLLMs. This paper provides a discussion, case study, and practical guidance to help researchers adopt and evaluate GLLMs in their research projects.

I start by comparing the GLLM approach to existing textual analysis techniques and to using human research assistants. The summary is that state-of-the-art GLLMs can solve any textual analysis task solvable using non-generative methods. However, non-generative techniques can be substantially cheaper and faster for less complex problems that are feasible to solve using simpler techniques. Where GLLMs start to become viable is for problems that would otherwise require

¹ See Appendix B for examples of GPT-4 completing these tasks.

complex ML machinery or human coding. GLLMs are substantially cheaper and faster relative to human RAs and can complete equivalent work for many textual analysis tasks. However, well incentivized RAs will be easier and more flexible to work with for small sample tasks.

I provide a five-step framework that researchers can use to solve a textual analysis problem using a GLLM. This framework focuses on the implementation details and challenges specific to using GLLMs and complements the general textual analysis guidance by Anand et al. (2020) and Bochkay et al. (2022). The framework covers understanding the task requirements, selecting the right GLLM model and instruction approach, developing the instruction prompt, evaluating the construct validity, and finally executing the task. I also provide code examples and practical resources through the companion website.

Next, I provide a case study that demonstrates the application of GLLMs to identify non-answers in earnings conference calls. Detecting non-answer is traditionally challenging because it involves a large degree of nuance and context dependence, which makes it a good case for testing the capabilities of GLLMs. The benchmark results indicate that there are various ways to use GLLMs to significantly improve on the accuracy of the Gow et al. (2021) approach. By combining GLLMs with existing methods, I am able to achieve an accuracy of 96% and a non-answer F1 score of 87% and deploy it at a reasonable cost. The case study further explores additional dimensions of non-answers, such as the managers' justification, the type of question, and whether the manager tries to give a positive spin to their non-answer. This analysis, using a fine-tuned ChatGPT model, underscores the flexibility and depth that GLLMs can bring and provides valuable insights into the communication dynamics of earnings calls. For example, I show that non-answers to forward-looking questions became more prevalent during the COVID-19 pandemic, which managers often accompanied by a qualitative discussion with a positive spin.

Finally, I provide a discussion on several key considerations when using GLLMs for academic research. These considerations include potential biases, attributability concerns, and limitations regarding recent events in the GLLM training data. Additionally, I provide guidance to address replicability challenges and data privacy and copyright concerns. Lastly, I discuss current GLLM developments, such as multi-modal support and GLLMs for other languages, and provide a discussion on the use of GLLMs for writing and programming tasks.

This paper makes four contributions. First, it extends Bochkay et al. (2022) by highlighting the use of Generative Large Language Models as a powerful new research method for accounting projects involving textual data. Second, it raises attention to the challenges and questions surrounding GLLMs, such as construct validity, biases and knowledge limitations, replicability, data privacy, and attributability concerns and provides guidance on how to address them. Third, it offers practical resources and guidance to help researchers get started with GLLMs for their own projects, including a conceptual framework, code examples, prompt engineering tips, and more through the companion website. Finally, the case study of non-answers in earnings conference calls adds to the disclosure literature on communication dynamics by studying the prevalence of non-answers and its various dimensions. I provide the non-answer dataset and its dimensions through the companion website for future research to use. Taken together, this paper equips researchers with the necessary knowledge and tools to harness the potential of GLLMs and editors and reviewers with the knowledge to better evaluate papers that use the GLLM approach.

2. Generative Large Language Models

2.1. What are GLLMs?

Generative Large Language Models (GLLMs) represent a class of machine learning models that consume and generate human-like text. Generative Pre-trained Transformer (GPT) models are

the most common type of GLLM, but other generative architectures exist. GLLMs have three key characteristics. Firstly, they are built on large-scale neural networks with billions of parameters, allowing them to capture intricate patterns and relationships in the data. This large scale enables GLLMs to develop a basic, yet limited, reasoning ability and world understanding. Secondly, GLLMs are generative in that they take natural language text as input (the *prompt*) and generate natural language text as output (the *completion*). Finally, foundational GLLMs are trained for general-purpose tasks, enabling them to adapt to a wide range of applications.

GPT and BERT (Bidirectional Encoder Representations from Transformers) models are both part of the transformer architecture family, which originates from the development of the attention mechanism (Bahdanau, Cho, and Bengio 2016; Vaswani et al. 2017). BERT models, introduced by Devlin et al. (2019), provide a way to represent text using its surrounding context instead of treating every word as independent. This makes BERT a powerful building block as part of a broader machine learning pipeline, such as for classification or other sequence-to-sequence tasks. In contrast, GPT models, developed by OpenAI, emphasize generative capabilities, and provide an end-to-end approach where both the input and output are natural language (Radford et al. 2018; 2019; T. Brown et al. 2020; OpenAI 2023). BERT models and GPT models share similarities, such as their transformer architecture and their transfer learning abilities (e.g., Siano and Wysocki 2021). The fundamental difference is how one adapts them to solve their textual analysis problem. A BERT model is adapted through the training procedure and ML machinery that one builds around it. In contrast, a GPT model is adapted through the development of a natural language prompt and potentially fine-tuning it for those types of prompts. The GPT approach makes it possible to get usable results more quickly for many textual analysis tasks. In the next section I discuss the pros and cons of GLLMs and highlight the types of tasks for which they are a good fit.

2.2. Comparison to current methods

2.2.1. Non-generative textual analysis and ML techniques

GLLMs have several features that make them stand out from current textual analysis approaches. Firstly, GLLMs take text as input and yield natural text as output. Natural text is a powerful way to embed and express complex tasks; it is how humans transfer information. Any task and information expressible through natural language can be part of the instruction to a GLLM and its output. This is significant because it makes it easier to adapt a GLLM to solve research tasks. Adapting a GLLM to a specific task only requires formulating the instructions using human language without the need to develop a custom machine learning setup. To illustrate, Hassan et al. (2019) identify conversation about political risk in conference calls by training a pattern-based sequence-classification method on a dataset of (non-)political texts. They construct a training dataset of political and non-political texts based on newspaper articles and textbooks, which is time-consuming. A powerful GLLM often doesn't require a training dataset. Instead, we can directly show the model a part of the earnings call and ask the question "*does this section discuss political risks, yes or no?*". Furthermore, the rigid nature of non-generative ML approaches can make it challenging, or impossible, to express more complex or specific tasks. Expressing a task through natural language helps in that regard as it is more flexible and easier to adjust. For example, Brown et al. (2018) try to identify how firms adjust their risk factor disclosures when a peer firm receives an SEC comment letter. Identifying specific types of changes to text is a challenging task so they limit their primary analysis to only looking at changes in the number of words. The GLLM approach would make it possible to identify and classify specific types of year-over-year changes. Moreover, one could even include details about the SEC comment letter to identify only changes that relate to the letter. One could achieve this by showing the risk factors

and a summary of the comment letter (which could also be generated by the GLLM), followed by the question “*Did the risk factor change in response to the comment letter, yes or no?*”.

Secondly, the scale of the original training procedure and the large size of GLLMs enable them to exhibit basic forms of reasoning and have a basic understanding of the world.² This makes it easier to instruct a GLLM to perform a task relative to a non-generative approach. Specifically, non-generative approaches require a large training sample or researcher defined rules (such as word lists) to perform a specific task. For GLLMs it is often enough to describe the task without the need for examples (called “zero-shot”) or with only a small number of examples as part of the prompt (called “few-shot”). For example, Li et al. (2021) use a complex word embedding approach and a manual check to extrapolate the culture seed-words from Guiso et al. (2015) to a broader word-list of the five culture values. This is valid, but a GLLM approach would make this easier. One could create an instruction showing the excerpt, the seed-words representing the culture values, and ask the model to identify all sentences that relate to one or more of the values. Similarly, Call et al. (2022) use a sophisticated machine learning algorithm to analyze conference call audio to identify cases of laughter in response to humor. They limit their focus to only successful cases of humor, as unsuccessful attempts to be funny do not elicit laughter. With current state-of-the-art GLLMs, such as GPT-4, one can instead show an earnings call excerpt by the CEO and ask the question “*did the CEO try to be funny, yes or no?*”. Which could identify both successful and unsuccessful attempts at humor.

Finally, GLLMs can handle more text at once (i.e., the “context window”) relative to existing models such as BERT. The currently available 16k variant of ChatGPT has a context window of 16,385 tokens (about 26 pages of text). The Claude2 model by Anthropic has a context window of

² To illustrate, the GPT-4 model by OpenAI is likely trained on trillions of words and cost over \$100 million to train. See, <https://www.wired.com/story/openai-ceo-sam-altman-the-age-of-giant-ai-models-is-already-over/>

100k tokens, which is hundreds of pages of text. For reference, BERT models typically have a context window of only 512 tokens. These larger context windows make it easier to work with longer documents, which are common in business and disclosure contexts. For example, Gow et al. (2021) develop a regular expression approach to detect non-answers in conference calls. Extending their analysis to classify other types of answers using ML techniques using BERT would be challenging because the combination of a question and answer often wouldn't fit in 512 tokens. The ability of GLLMs to handle larger inputs makes it possible to show it a question, the answer(s), as well as a task like *"did the manager refuse to answer the question because of proprietary concerns, yes or no?"*. Section 4 provides a full case study demonstrating the use of GLLMs to identify non-answers in earnings calls.

Despite these benefits, GLLMs also have drawbacks when compared to non-generative textual analysis approaches. The primary disadvantage is that GLLMs are large and require more processing resources, making them more expensive and slower to use. Non-generative NLP techniques will be substantially cheaper and faster for simpler tasks that don't require the power of GLLMs. Furthermore, evaluating the construct validity of GLLMs can be more challenging due to the flexibility of the natural language aspect and the black-box nature of large models. Smaller non-generative models, or rule-based approaches, are generally more explainable and easier to evaluate. Finally, non-generative models can be easier to run and train without the need for using third-party APIs or access to expensive hardware. Taken together, GLLMs can solve any problem that traditional NLP methods can solve but that doesn't always make them the best or most cost-effective option. A general rule-of-thumb is to keep things as simple as possible and to only resort to more powerful techniques when necessary. GLLMs are a viable consideration for problems that would otherwise require a complex machine learning setup or human coding.

Finally, GLLMs and existing techniques are not mutually exclusive and combining them can yield the best of both worlds. One such combination is to use GLLMs to construct or extrapolate a training dataset for a non-generative (e.g., BERT) machine learning model that is cheaper and faster to scale. For example, Li (2010) creates a training dataset by manually classifying 30,000 forward-looking statements (FLS) along the tone and content dimensions. Using a GLLM to perform these initial training classifications would save significant researcher time. Another combination is to narrow down the sample using a non-generative method and then use a GLLM to perform the harder tasks. For example, Bozanic et al. (2018) provide a keyword list to identify forward-looking statement sentences. One could use this list to identify probable FLS sentence candidates and then feed them through a GLLM to improve the classification by removing bad matches and possibly extracting additional dimensions of FLS such as sentiment or time-horizon.

2.2.2. Human research assistants

Using GLLMs also has several advantages and disadvantages relative to human research assistants or performing the task manually yourself. The primary benefit of GLLMs is the potential for cost savings, as well as an increase in speed and scalability. GLLMs can also yield outputs that are more consistent as they are not subject to inter-rater variability. For example, Chuk et al. (2013) hand-collect management forecasts in earnings releases and compare it with a data aggregator. GLLMs can offer an approach to perform tasks like this automatically, which would save significant time and resources. Finally, GLLMs can exhibit super-human capabilities in terms of recalling and memorizing large quantities of information. For example, a research task that involves looking up facts in the FASB codification would be slow and challenging for most human research assistants. In contrast, a GLLM with access to that information (either through fine-tuning, or otherwise) could complete such a task in seconds.

Despite these advantages, there are limitations to using a GLLM relative to human work. The primary limitation is that human RAs are more flexible, better at reasoning, and have incentives to try hard and produce good work. A human can be held accountable for their work while a machine cannot. A well incentivized human RA will provide high quality outputs with only minimal instructions. In contrast, a GLLM approach might require prompt engineering or fine-tuning before it starts generating high quality outputs. This makes human RAs a better alternative for complex tasks with only small samples. For example, Brown et al. (2020) manually inspect a small random sample to evaluate their approach of identifying material filing revisions. This is likely automatable using a GLLM. However, the fixed costs of doing so are only worth it for larger samples. Domain knowledge can be another limitation. GLLMs might need to learn domain knowledge through explicit examples, which might be hard to obtain or formulate. For example, Hail et al. (2018) use human coding to identify accounting scandals that are morally wrong versus legally wrong. It might be challenging to communicate to a GLLM what it means for something to be morally wrong.

2.3. Current research using GLLMs

While a relatively new technology, GLLMs are already seeing use in a variety of accounting and finance working papers. Bernard et al. (2023) fine-tune a GPT-3 Babbage model on a sample of 100,000 10-K footnotes to measure the underlying business complexity. Kim et al. (2023) use a zero-shot ChatGPT approach to generate automatic summaries of financial disclosures to measure the level of disclosure bloat. Blankespoor et al. (2023) use ChatGPT and GPT-4 to identify and classify company Twitter messages that engage with different types of social conversations, including those relates to ESG and sustainability. Jha et al. (2023) use ChatGPT and conference calls to extract managerial expectations of corporate policies which they combine

into a firm-level investment score. 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 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. Lopez-Lira and Tang (2023) use ChatGPT and GPT-4 to define a sentiment score 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) compare a zero/few-shot GPT-3 model with the FinBERT model by Hu et al. (2023) for sentiment analysis. Their results indicate that GPT-3 as well as FinBERT outperform existing methodologies, but that the FinBERT model outperforms the zero/few-shot GPT-3 model. Hu et al. (2023) do not explore other GLLMs or fine-tuning.

The rapid emergence of these working papers and their impact highlights the relevance of the GLLM methodology as well as its applicability to a wide-range of NLP problems and documents.³

3. Using GLLMs to solve research tasks

The generative natural language aspects of a GLLM pose distinct challenges that are not covered by existing NLP overview papers such as Anand et al. (2020) and Bochkay et al. (2022). Working with GLLMs requires a way of thinking that falls between interacting with a human (due to the natural language aspect) and a non-generative machine learning model (due to the machine aspect). Specifically, while it is possible to instruct a GLLM like one would instruct a human, that

³ The Lopez-Lira and Tang (2023) paper on using ChatGPT for return prediction has, as of September 2023, 50,000 SSRN downloads and received coverage in 135 media outlets. Similarly, Matt Levine, from Bloomberg's Money Stuff, covered the results by Bai et al. (2023) as headline material.

will often yield unsatisfactory results. To achieve high quality, consistent, and efficient results researchers need to engineer GLLM-specific instructions, which is a skill. In this section, I provide a five-step framework that helps researchers implement GLLMs for research.

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 know 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 is lacking the necessary information to make a prediction or classification.⁴ Any information, such as specific institutional knowledge, that is likely not known to the GLLM will need to be provided, either through examples, instructions, or by fine-tuning.

One of the best ways to develop a deeper understanding of the task you are looking to automate is to perform the task manually yourself on a set of examples and ask 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 footnote text relating to FASB ASU 2014-09. You might be able to complete that task easily because you know what FASB ASU 2014-09 relates to, but the GLLM might have no idea, or worse, a flawed 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

Different GLLM models have different strengths and limitations, and the right model will

⁴ GLLMs sometimes “hallucinate”, which is the generation of incorrect or made-up information. This behavior is, in most instances, the result of the model’s inability to refuse to provide generations on topics or tasks for which it does not have sufficient context or background information.

depend on the task and the GLLM approach. There are three approaches that one can take to instruct and control the output of a GLLM: *zero-shot*, *few-shot*, and *fine-tuning*. The zero-shot approach works by providing the GLLM only with your instructions and the data on which you would like to perform the task, without giving any examples. The few-shot approach works like the zero-shot approach, except 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 by updating the model weights, i.e., the coefficients or parameter values of the neural net, through a training procedure called “fine-tuning”.

When selecting your approach, you need to keep the following aspects in mind: your “token budget”, the complexity of your task, and your accuracy vs. effort trade-off. The token budget refers to the length of the instructions to complete a single task as well as the total number of tokens required to complete all your generations. The time and costs required to complete all your generations with a GLLM model scale linear with the number of tokens in your prompt and completion. For example, including examples as part of your prompt will result in more tokens which can limit you to models with larger context windows and increase the time and money required. Furthermore, the complexity of your task is important when selecting the appropriate approach. Simple tasks work well with the zero-shot or few-shot approach, while more complex or domain-specific tasks may require fine-tuning. For example, classifying whether a text discusses employees might constitute a simple task while classifying whether a paragraph should reference FASB ASU 2014-09 might constitute a complex task. Lastly, fine-tuning can yield better results but requires significantly more effort relative to zero/few-shot approaches, so it is best to pick the simplest approach that yields satisfactory results.

Once settled on the approach you can evaluate which GLLM and provider would best suit your

needs. The GLLM in this context refers to the exact model, such as *gpt-3.5-turbo-16k*, and the provider refers to whoever is making the model available, such as OpenAI. Identifying the right GLLM and provider is generally the product of trial-and-error and picking the option which is cheapest and easiest. The companion website includes a list of common providers.

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 will generate based on your prompt. 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 will change for each observation in your data. For example, a simple prompt would be “*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.

When designing a prompt, the two primary objectives are (1) to communicate to the GLLM what you expect the completion to be and look like (*instructions + examples*) and (2) to provide the GLLM with the necessary information to generate the desirable completion (*context + focal data*). A model that misunderstands the instructions or that does not have access to the necessary information will generate completions of low quality.⁵ Any information that can be represented as text can be part of a prompt or completion. This includes not only narrative data but also, amongst others, tabular data, JSON, XML, HTML, and programming code. It is hard to anticipate how GLLMs will respond to an instruction, so it requires experimentation to get it right. The benefit of designing a GLLM instruction is that iteration is quicker than with human RAs. You can try an

⁵ Zero-shot and few-shot prompts will consist of three to four building blocks: instructions, relevant context, examples, and the focal data. For fine-tuned models the instructions, relevant context, and examples are already embedded into the model weights, so the prompt generally only includes the focal data. In other words, a fine-tuned model already “knows” what the task is because we trained it for it, so we don’t have to repeat the instructions.

infinite number of times, GLLMs don't get tired, distracted, or hungry after the N^{th} try. Appendix D includes a series of tips and tricks for designing successful and efficient 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 which can be hard, or impossible, to parse with your code if you don't design your completion with parsing in mind. For example, let's say we 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 dataset. Instead, we could include an instruction to return a JSON list of the format *"[(cashtag, sentiment), ...]"*. This would yield *"[('\$AAPL', 'bullish'), ('\$MMM', 'bearish')]"* which we can easily parse into a dataset.⁶

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. 2022). Evaluating generative completions comes with two primary challenges. The first challenge is that the zero-shot and few-shot approaches don't require a training sample to evaluate against. 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? To safeguard and ensure the construct validity when using the GLLM approach, I recommend the following best practices.

- (1) Always create an evaluation set of outcomes generated by human raters, such as yourself, co-authors, research assistants, domain experts, or third-party sources. This sample needs to be

⁶ Fine-tuning would happen after Step 3 and is further discussed in Appendix E.

large enough, generally between a hundred to a couple thousand. The evaluation sample needs to be large enough to represent the diversity of the dataset and task.

- (2) When possible, engineer your instructions such that you can evaluate the GLLM approach like a standard supervised machine learning approach. For example, returning an explicit outcome (e.g., 1 / 0) will be easier to evaluate relative to a narrative outcome.⁷ 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 (see e.g., section 4.3 in Bochkay et al. (2022)). Or when using a GLLM to make qualitative assessments, follow the standard best practices for evaluating qualitative interviews.
- (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 fall apart with closer inspection. Due to human confirmation bias, it is easy to over-estimate the quality of a GLLM completion.

3.5.Run the predictions and parse the results – Step 5

Once a satisfactory construct validity is reached, the approach can be scaled up to generate completions for all the data observations. This step is referred to as the “inference” step. There are two ways to generate completions for your full sample: interacting with an external third-party API or running the model locally on a powerful (Linux) machine with one or more accelerators

⁷ For example, when classifying sentiment using a GLLM one can design an instruction that yields a sentiment score in terms of a narrative description (“*The tweet is mostly positive, but some parts are negative*”), but that would be much more challenging to evaluate compared to a prompt that asks for discrete outcomes (e.g., is positive: 0/1).

(generally GPUs or TPUs). Using a third-party API, such as the OpenAI API, will be the easiest and fastest for most use cases. However, there are also GLLMs available that one can download and run locally. There are several advantages to running a model locally, for example in terms of replicability and data privacy, which are further discussed in Sections 5.2 and 5.3.

4. Case study – using GLLMs to identify non-answers in conference calls

One of the promises of GLLMs is their ability to automate tasks previously only feasible to solve using human coding. However, little empirical evidence exists to benchmark and demonstrate the GLLM approach. To help with that, I deploy GLLMs for a task challenging to accurately solve using traditional methods: identifying non-answers in earnings conference calls. Identifying non-answers in earnings conference calls is a hard NLP problem because questions and answers in earnings conference calls are diverse in their content and style, often involve a large degree of nuance and context, and frequently suffer lower data quality due to transcript errors.

Two prior papers have studied conference call non-answers. 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) develop a rule-based technique using regular expressions to identify non-answers automatically for a large sample of earnings calls. Their approach is functional but leaves a lot of room for improvement in terms of classification quality. They try but are unable to improve performance through traditional machine learning. They highlight that identifying non-answers using machine learning is “*not as straightforward as it seems*” and “*we leave this challenge for future research*”. As such, the challenging task of identifying non-answers in earnings calls provides a case study to evaluate and benchmark the ability of GLLMs to perform tasks beyond what is currently feasible or possible.

4.1. Non-answer classification

4.1.1. Data and method

I retrieve the earnings call transcripts through the Finnhub.io API and limit the sample to CIKs that exist in the Gow et al. (2021) sample. To maximize Finnhub.io coverage and to enable an analysis of COVID-19, I focus on a 10-year sample period starting in 2013 and ending in 2022. This yields a sample of 63,959 earnings calls after requiring that the call includes a presentation section and includes at least three question and answer (Q&A) pairs. Processing each transcript and extracting the valid Q&A pairs results in a sample of 1,152,505 Q&A pairs.⁸

My objective is to develop a GLLM approach to solve the non-answer task, which I will refer to as the *GPT method*. To develop the GPT method, I draw several random samples of Q&A pairs and manually classify them to identify non-answers. After settling on the approach and prompts, I draw and manually classify a final sample of 500 Q&A pairs to serve as the evaluation sample.⁹ Through my development, two observations become clear. First, defining the exact boundary between an answer and a non-answer is challenging and arguably hard to define objectively. Second, it is generally hard to evaluate whether a response is a non-answer without the context of the underlying question. As a result, identifying non-answers requires a tolerance for ambiguity, a degree of reasoning ability, and the ability to evaluate the answer in the context of the question. Traditional machine learning approaches struggle with these exact aspects, which explains their

⁸ 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 where 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 aren’t relevant and including them inflates the classification statistics.

⁹ 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-sample” / “out-of-sample” logic helps with this. However, in practice this is challenging to do because humans can’t unsee evaluation signals like machines can. It is not realistic to create a new evaluation sample after every development iteration. Instead, I recommend loosely following the in/out sample methodology but supplementing it with plenty of output inspection. The thing that ultimately matters is that the outputs of the GLLM make sense, if they don’t, then the current setup isn’t working well irrespective of what the statistics are showing.

low performance on the task of identifying non-answers.

Using the available code, I replicate the Gow et al. (2021) measure and refer to it as the *baseline*. The results in Table 1 Column 2 show the performance results for the baseline.¹⁰ 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 lower recall of only 43%, with a higher Type II error of 57%. Combined, this yields a non-answer F1 score of only 49%. These results indicate that the baseline measure often misses non-answers as well as incorrectly label answers as non-answers.¹¹ Gow et al. (2021) acknowledge these limitations in their paper and it provides an opportunity for improvement using the GPT method.

I explore various approaches using GPT-3, ChatGPT, and GPT-4 before settling on the prompts shown in Appendix C and the remaining performance results in Table 1. Starting with the zero-shot approaches, the ChatGPT method (column 3) substantially improves over the baseline with an accuracy of 91%, a non-answer F1 score of 72%, and a substantially lower Type II error at 26%. The GPT-4 method (column 4) outperforms the previous approaches with an accuracy of 93%, a non-answer 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 examples or task-specific training.¹² The primary constraint is costs as the cheaper ChatGPT doesn't perform as well as GPT-4, but GPT-4 is expensive to deploy at a larger scale.

In columns 5 and 6 of Table 1, I explore an alternative approach of combining traditional methods with GPT models to achieve higher-than-GPT-4 performance at a more reasonable cost.

¹⁰ There are multiple reasons why the replication statistics vary, somewhat, from the original statistics in Gow et al. (2021). Most notably: I use a different transcript provider, period, way of processing the text, and I filter out the invalid Q&A pairs. Also, resolving edge case scenarios depends on the exact non-answer definition, which can vary slightly.

¹¹ As illustrated by this example, high accuracy can be deceiving when evaluating highly imbalanced classifications.

¹² The prompt engineering aspect is very important. For example, a minor modification to my prompt, e.g., removing the distributional statement (i.e., "... *these sentences are rare, in 65% ...*"), already significantly drops performance.

At a high level, I develop a funnel approach where I use cheaper methods to filter down the sample and resolve only the remaining harder classifications using more powerful methods.¹³ In the context of non-answers, I first use a simple keyword list to identify responses that have any possibility of being a non-answer. 42% of the Q&A pairs don't match any keywords and don't need to be further considered. Next, I apply a basic 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 non-answer.¹⁴ This reduces the possible non-answer Q&A pairs to only $\sim 1/3^{\text{rd}}$ of the original sample. The remaining sample contains the hard classification problem of separating out the true non-answers from the answers that look like non-answers, which I will solve using GLLMs.

To classify the remaining observations, I develop a new zero-shot ChatGPT prompt to identify the non-answer responses. To guide ChatGPT, I frame the prompt as an evaluation task and include the sentences possibly reflecting a non-answer. The full prompt is available in Appendix C.3. As shown in Column 5 of Table 1, this yields a performance that matches GPT-4. The resulting accuracy is 94% with an F1-score for non-answers of 82%, while being $1/35^{\text{th}}$ the price of GPT-4.

For most research applications this would suffice. But to fully demonstrate the potential of GLLMs, I add a final step by fine-tuning a ChatGPT model to further improve performance while simultaneously classifying multiple other non-answer dimensions. Doing so marginally increases the costs but it brings the performance up to an accuracy of 96%, with a non-answer precision and recall of 87%.¹⁵ Taken together, the results in Table 1 yield two primary take-aways. First, the performance statistics validate that GLLMs are capable and can solve hard NLP problems when

¹³ More detail on the approach, including the full keyword list, is available in Online Appendix 1.

¹⁴ I primarily include this step to demonstrate the versatility of fine-tuning a simpler GPT model. However, skipping this step would yield similar eventual performance at only a moderate increase in total costs.

¹⁵ The ChatGPT fine-tuning endpoint only became available after I had already completed the original approach in Column 5 of Table 1. Replacing the zero-shot ChatGPT step with the fine-tuning step would streamline the approach.

set up correctly. Second, the approaches in Columns 5 and 6 highlight that combining GLLMs with existing methods is a powerful approach to yield state-of-the-art performance at a reasonable cost. The approach in Table 1 column 6 acts as the *GPT method* for the remainder of my analyses.

4.1.2. *Full sample comparison*

Table 2 shows the results of applying the GPT method 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, shown in Table 2 Panel A (B), the GPT method identifies 13.9% (14.2%) of Q&A pairs as containing a non-answer. The Gow et al. method identifies 12.3% (12.7%) of the pairs as containing a non-answer. These results are close in aggregate, but the Venn diagram in Figure 1 shows that there is considerable disagreement between the two methods. Only 50% of the non-answers by the GPT method are also identified by the Gow et al. method. Furthermore, only 57% of the Gow et al. non-answers are considered non-answers by the GPT method. This is consistent with the performance statistics in Table 1. The magnitude of these results highlights that the accuracy improvement by the GPT method is substantial and yields a meaningfully cleaner measure of conference call non-answers.¹⁶

4.2. *Classification of non-answer types*

Next, I extend the approach and analysis to demonstrate the easy-of-use and flexibility of GLLMs as well as improve our understanding of conference call non-answers. Specifically, as part of the GPT method's last step I simultaneously classify the non-answers across several dimensions. I summarize each dimension below and highlight the classification options in *italics*.

- **The manager's justification for not providing an answer.** *Can't give* indicates that the manager justifies they can't answer because of reasons such as proprietary concerns, company

¹⁶ Online Appendix 2 includes examples of improvements of the GPT method over the Gow et al. method.

policy, or legal reasons. *Don't know* indicates that the manager justifies that it is too early, they do not have the answer, or they do not know or have the information.

- **The type of non-answer.** *Complete refusal* reflects a complete non-answer without the manager providing any information. *Qualitative* indicates that the manager does not directly answer but instead provides a qualitative answer. *Range or percentage* indicates that the manager does not directly answer but instead provides a quantitative range or percentage.
- **The question type preceding the non-answer.** *Breakdown* indicates that the participant requests the manager to provide more detail on something. *Forward looking* indicates that the participant requests the manager to provide guidance or a forward-looking statement. *Related party* indicates that the participant asks about things like acquisitions, partnerships, major customers, or suppliers. *R&D or regulation* indicates the participant asks about R&D projects, clinical trials, or regulatory/certification matters. And *Other* captures the remaining questions.
- **The sentiment spin of the non-answer** – *Optimistic* indicates the manager accompanies their non-answer with an optimistic statement (e.g., “I can’t give you the numbers, but things are looking good!”). And *Neutral or pessimistic* indicates the lack of an optimistic statement.

4.2.1. Methodology

To classify the non-answer dimensions, I demonstrate both a zero-shot GPT-4 approach as well as a fine-tuned ChatGPT approach. The full prompts are available in Appendix C.4 and C.5. After developing the approaches and prompts, I draw a random sample of 400 Q&A pairs that likely contain a non-answer (i.e., those remaining after the Column 5 approach in Table 1). Table 3 shows the performance results of evaluating the GPT-4 and fine-tuned ChatGPT approaches. The GPT-4 approach, shown in Columns 1 to 3, yields a high level of performance across all classification categories, with F1 scores ranging from 85% to 98%. However, the zero-shot GPT-4 approach has two downsides: it is not cost-efficient for long prompts, and it remains hard to communicate what exactly constitutes a non-answer, especially for edge cases. Fine-tuning can

help with both downsides. Fine-tuning a smaller model can yield similar performance to a large model at a fraction of the costs. Furthermore, fine-tuning makes it possible to communicate intricate concepts through examples rather than instructions, which can improve performance. The downside of fine-tuning is that it requires a high-quality training dataset, which is hard to obtain. An interesting hybrid approach is to fine-tune a smaller model on a training dataset that is put together based on the outputs of a larger model. This combination can yield the best of both worlds.

The high performance of GPT-4 on my non-answer classification task makes it suitable to demonstrate the hybrid approach. I start by drawing another random sample of ~2,100 Q&A pairs and feed them through the zero-shot GPT-4 approach. The performance statistics in Table 3 show that GPT-4 achieves near human performance for the four sub-dimensions, so I directly use those for the training data. The non-answer classification by GPT-4 can use improvement, so I manually go through all the training observations and improve the classification of non-answers with my own. I then use the improved training dataset and the OpenAI API to fine-tune a ChatGPT model using a reduced form of the GPT-4 prompt, which is shown in Appendix C.5.

The evaluation results for the fine-tuned ChatGPT model are shown in columns 4 to 6 of Table 3. In terms of costs, the fine-tuning approach is about 11x cheaper than the zero-shot GPT-4 approach. The results show that the fine-tuned ChatGPT model is better able to evaluate the validity of non-answers, with an overall F1 score of 93% versus the 85% of GPT-4. The performance on the other four classification categories is generally lower, but still high varying from an F1 score of 81% to 94%. One interesting observation is the performance comparison of GPT-4 versus fine-tuning for the less frequent categories, such as *Other*. Similar to traditional machine learning, fine-tuning requires enough examples for every classification outcome, which is hard for rare or diverse outcomes. As a result, the fine-tuned model struggles with the *Other*

category (F1 score of 64%). Zero-shot GLLM approaches are less susceptible to this problem, making them particularly suitable for tasks with traditionally challenging outcome distributions.

4.2.2. Full sample descriptive statistics

I apply the fine-tuned ChatGPT model to the plausible non-answer Q&A pairs, which yields the final *GPT method* as well as the full sample statistics shown in Table 4. The results in Table 4 show that managers justify their answer based on a lack of knowledge or information in about 37.5% of the cases. It is rare for the manager to provide a non-answer without any additional information, which only happens in 17.3% (*Complete refusal*) of the non-answers. Most of the time, in 70.9% (*Qualitative*) of the cases, the manager accompanies their non-answer by a qualitative discussion. In terms of question types, many non-answers relate to the participant asking for forward-looking information (*Forward looking* - 41.7%) or additional detail on something (*Breakdown* - 20.9%). About 12.8% (*Related party*) of the non-answers relate to questions about acquisitions, partnerships, or major customers. Finally, it is common for managers to give a positive spin to their non-answer (*Optimistic* - 55.1%).

4.3. Additional non-answer results

4.3.1. Non-answer determinants

I start by replicating Table 2 of Gow et al. (2021) to study how question characteristics relate to the propensity of a non-answer. I regress each question characteristic separately on the non-answer outcome with earnings call fixed effects, following Gow et al. (2021). To save space, the independent regressions results are shown below each other in Table 5 Panel A, columns 1 and 2. While the magnitudes differ, most of the results are similar. One notable difference is the result on question sentiment. Specifically, Gow et al. (2021) find a positive correlation between negative question sentiment and the likelihood of a non-answer. In contrast, the more accurate GPT measure

finds a negative correlation with both positive as well as negative question sentiment. These new results show that managers are *less* likely to provide a non-answer when an analyst asks a question with strong negative sentiment, although to a lesser degree than one with strong positive sentiment.

Next, I expand the analysis by correlating the same question features with the different types of non-answers. The results highlight that the question feature correlations are conditional on the type of non-answer. A few notable results. First, having a question with a strong positive sentiment does not impact the likelihood of managers giving a positive spin to their non-answer (-1.666 , *Column 4*). A question with a strong negative sentiment, on the other hand, does significantly decrease the likelihood of the manager giving a positive spin (-24.240^{***} , *Column 4*). Second, question complexity (*i.e.*, Q - *Fog index*) generally predicts a higher likelihood of a non-answer. However, this is driven by non-answers with a qualitative alternative (0.196^{***} , *Column 6*). Complete refusal and range or percentage non-answers instead are less likely for complex questions (-0.050^{***} , *Column 5* and -0.030^{**} , *Column 7*). Finally, questions that ask about specific amounts, locations, or dates are generally more likely to receive a non-answer. However, these categories are significantly less likely to receive a complete refusal and more likely to receive a qualitative non-answer (-0.191^{***} , -0.291^{***} , -0.146^{***} in *Column 5*).

Finally, I focus on the sub-sample of non-answers and independently correlate the question type with the non-answer outcome type. The results are shown in Panel B of Table 5. Several interesting patterns emerge. First, managers generally justify not answering a question regarding detailed breakdowns or related parties by declining to answer for policy or proprietary reasons (21.225^{***} & 21.220^{***} , *Column 1*). On the contrary, they are more likely to justify their non-answer by stating they don't know for questions about forward-looking information or R&D and regulation (-17.545^{***} & -9.988^{***} , *Column 1*). Second, managers primarily give a positive spin

to their non-answer when analysts ask about forward looking statements, but less so for the other categories (10.048^{***} , *Column 2*). Lastly, managers are more likely to completely refuse answering questions about related parties or R&D and regulations (9.815^{***} & 8.606^{***} , *Column 3*). Qualitative non-answers are common for forward-looking questions whereas range or percentage non-answers are more common for questions asking for a breakdown (5.130^{***} in *Column 4* and 9.119^{***} in *Column 5*).

Taken together, these results highlight that non-answers differ on interesting dimensions. Depending on the research question, it can make sense to focus on certain types of non-answers over others. For example, if one tries to study managerial ability it might make sense to focus on non-answers where the manager doesn't know. Similarly, if one tries to study a manager's willingness to share information, it can make sense to focus on just complete refusals. The full dataset split by the non-answer types is available on the companion website for researchers to use.

4.3.2. *Non-answer time-trends and COVID-19 pandemic*

I perform a time-series analysis to compare the GPT and Gow et al. (2021) method and to study how the COVID-19 pandemic impacted the non-answer behavior of managers. The results are shown in Figure 2. Regarding the methods comparison, Figure 2a shows that the GPT method is substantially more stable over time relative to the Gow et al. method. This is in part because the Gow et al. rules were developed based on an older sample period. But more generally, the topics of earnings calls change over time and the GPT method is more flexible to deal with these changes due to the broader and stronger language understanding of GLLMs. Looking at the COVID-19 period, both measures indicate a substantial increase in the non-answer likelihood following the start of the pandemic. However, the lower temporal stability of the Gow et al. (2021) measure, incorrectly, suggests that the COVID-19 spike is part of a broader time-trend. The GPT method

paints a clearer picture, showing a clear increase in non-answers that deviate from the time-trend.

Diving into the specific types of non-answers, Figure 2b shows that the COVID-19 pandemic resulted in a short-lived increase in neutral or pessimistic non-answers and a longer-lived increase in optimistic non-answers. Looking at the non-answer justifications, Figure 2c shows an increase in the number of non-answers justified by uncertainty or lack of information during the pandemic. Regarding the types of non-answers, Figure 2d shows a slight downward trend in the likelihood of managers completely refusing to give a non-answer. Furthermore, combining Figures 2d and 2e shows that the increase in non-answers during the COVID-19 pandemic is primarily driven by managers giving qualitative non-answers to forward-looking questions.

5. Discussion

5.1. GLLM training data: biases, attributability, and recent events

Researchers using GLLMs should be aware of the potential biases, attributability 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, preliminary analysis suggests that job postings generated by ChatGPT are less inclusive and more biased relative to humans.¹⁷ These biases can 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.

Attributability concerns arise because GLLMs are trained on a large diverse set of sources making it difficult to trace the origin of generated text. This may lead to issues regarding intellectual property, the need to provide attribution, and the accuracy of the information produced. To help with that it is better to control the source of important information by explicitly passing it

¹⁷ <https://www.businesswire.com/news/home/20230308005543/en/>, last accessed April 2023.

to your model through your prompt or fine-tuning. Directly providing the information makes attribution easier and helps safeguard the accuracy of that information. For example, generations by a GLLM might suggest that it understands the FASB taxonomy. However, it is unclear what source material was used to develop that understanding and it might be incorrect or out-of-date. Passing it the explicit portion of the FASB taxonomy that you would like 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 will not be 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 replicable research 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, which makes it hard, if not impossible, to fully understand the decision criteria applied by the model. Second, the expensive computational resources required to run most GLLMs limit the accessibility of these models. The easiest way to access a GLLM is through a third-party API, which is convenient but comes with replicability concerns. Models accessed through a third-party API can change or become unavailable over time. These concerns are analogous to working with human coders, who might not be available in the future either. There are several things a researcher can do to mitigate replicability concerns when working with GLLMs.

The first method is to not rely on third-party APIs and use GLLMs for which you can download the model weights and run it locally. Appendix E provides an example and discussion

of locally fine-tuning a Llama2 model, for example. 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 possible, or require a prohibitive degree of technical know-how and expertise.¹⁸ Furthermore, running the same model with the same prompt on different sets of hardware can still create minor differences in the completions, which makes exact replication challenging.¹⁹ For many projects the benefits of using a third-party API will outweigh the challenges that come with running a GLLM locally.

Another alternative method that works across all methods is to 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, human coders, or any other type of machine learning model. These best practices include:

- (1) Take detailed notes and be transparent in your empirical choices, such as your data, your task, your prompt design, your approach, and your model(s). Sharing your code, when possible, is a straightforward way to reduce ambiguity and improve replicability.
- (2) Always store and backup the raw prompts and completions as they come out of the model. Having access to the raw prompts and completions enables replication of all the analyses that follow and is the second-best thing next to having the source of the generations.
- (3) Try to make your GLLM generations as deterministic as possible. GLLMs inherently exhibit a degree of randomness to enable “creative” responses. The “temperature” hyper-parameter manages this randomness. Setting the temperature to zero will make the GLLM pick the highest probability tokens, which makes the generations as close to deterministic as possible.²⁰

5.3. Data privacy and copyright

When using GLLMs researchers need to be mindful of the privacy of their data, specifically

¹⁸ Currently, most state-of-the-art GLLMs can only run on powerful NVIDIA GPUs, some of which are unavailable in countries such as China due to export or import restrictions.

¹⁹ This limitation is not specific to GLLMs and applies to most models based on large neural networks.

²⁰ Note that due to technical reasons small variations can still exist, even when setting the temperature to zero.

when using third-party APIs. Sharing research data as part of your prompts or fine-tuning can result in data privacy concerns. Firstly, you are sharing your research data with whoever is running the API. Secondly, the API provider might use your prompts and generations to improve and re-train their models. This retraining can result in GLLMs unintentionally disclosing sensitive information in their future outputs to others.²¹ The same logic applies to data that is under strict copyright restrictions. To mitigate this risk, researchers should implement data anonymization techniques when preparing their training and prompt datasets. Furthermore, it is best to avoid using GLLMs through 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

GLLM developments are happening at a quick pace in a few notable areas. Current GLLMs can only accept natural language data as input. However, multi-modal support is on the horizon and models like ChatGPT and GPT-4 will support image inputs in addition to text inputs. Multi-modal support will result in an even greater range of tasks and problems that are automatable using GLLMs, such as those involving data visualizations, challenging PDF files, corporate presentations, video material, and audio transcripts.

Speed optimization and lower processing requirements are another area of progress. These optimizations will make GLLMs more usable as well as more affordable and scalable. To illustrate, the ChatGPT API (*GPT-3.5-turbo*) is already ten times cheaper relative to its predecessor GPT-3. Furthermore, there are multiple 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.,

²¹ For example, you can try to ask ChatGPT what it knows about your name, and it will generally yield a response detailing (some) public information about yourself. That information about you was part of the original training data collected from the internet. Any prompts you provide might be added to the training data and surface in the same way.

Touvron, Lavril, et al. 2023; Touvron, Martin, et al. 2023; Wang and Komatsuzaki 2021; Black et al. 2022). Better open-source models will make it easier to run GLLMs locally, which will help reduce concerns around replicability and data privacy, on top of further reducing costs.

Another area of work is to connect GLLMs to information sources beyond their original training data (e.g., “retrieval-augmented-generations”). These implementations, such as the ChatGPT Plugins by OpenAI, are showing potential to improve the knowledge and information lookup capabilities of GLLMs.²² For example, one could create a plugin that can search the FASB taxonomy and hook it up to a GLLM to perform a domain-specific textual analysis task on 10-K footnotes. Connecting GLLMs to external sources has many applications but also raises questions about reliability and replicability, which future researchers will need to be mindful of.

Finally, most current GLLM models focus on the English language. These English GLLMs do have basic capabilities in other languages but the state-of-the-art performance is generally for English. 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 GLLM models will make it easier to conduct textual analysis research on non-English documents.

5.5. GLLMs for writing & programming

While this paper focuses on using GLLMs for textual analysis tasks, it is worth discussing their increasing applications in other parts of the research process, such as AI copy editors and programming assistants. Many of the above discussions apply to these applications as well although with a few specific challenges. For example, when using GLLMs to adjust, check, or improve academic writing, it is critical to ensure proper attribution. Generated content requires fact-checking as GLLMs can produce plausible-sounding but false information or references.

²² <https://openai.com/blog/chatgpt-plugins> , last accessed September 2023.

Researchers should verify the accuracy of the writing generated and exercise caution in relying on the model-generated content without thorough examination.

In the context of programming, GLLMs can serve as useful productivity tools that facilitate code development and debugging. However, the research and coding problems encountered in academia are often rare and unusual, differing from 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 the final results. Code review is essential when working with code written or modified by a GLLM. This is especially true when GLLM coding assistants write code that the researcher wouldn't be able to write otherwise. To help with that, I echo the call by Bochkay et al. (2022) for researchers to (still) invest time in developing textual analysis and coding skills and for institutions to dedicate resources to provide the necessary training and guidance.

6. Conclusion

Generative Large Language Models (GLLMs) offer significant potential for accounting research involving textual data. GLLMs provide a powerful, versatile, and cost-effective alternative to traditional machine learning methods and human research assistants. This paper highlights the advantages of GLLMs, addresses challenges such as construct validity, biases, knowledge limitations, replicability, data privacy, and attributability concerns, and offers practical resources and guidance for researchers. By providing a discussion, case study, and practical guidance this paper equips researchers with the necessary knowledge and tools to harness the potential of GLLMs and assists editors and reviewers in evaluating papers that employ GLLM approaches.

References

- Anand, Vic, Khrystyna Bochkay, Roman Chychyla, and Andrew Leone. 2020. "Using Python for Text Analysis in Accounting Research." *Foundations and Trends® in Accounting* 14 (3–4): 128–359.
- Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. 2016. "Neural Machine Translation by Jointly Learning to Align and Translate." arXiv.
- Bai, John (Jianqiu), Nicole M. Boyson, Yi Cao, Miao Liu, and Chi Wan. 2023. "Executives vs. Chatbots: Unmasking Insights through Human-AI Differences in Earnings Conference Q&A." SSRN Scholarly Paper. Rochester, NY.
- Bernard, Darren, Elizabeth Blankespoor, Ties de Kok, and Sara Toynbee. 2023. "Confused Readers: A Modular Measure of Business Complexity." SSRN Scholarly Paper. Rochester, NY.
- Black, Sid, Stella Biderman, Eric Hallahan, Quentin Anthony, Leo Gao, Laurence Golding, Horace He, et al. 2022. "GPT-NeoX-20B: An Open-Source Autoregressive Language Model." arXiv.
- Blankespoor, Elizabeth, Ties de Kok, and Xue Li. 2023. "Corporate Engagement in Public Discourse.", Working Paper.
- Bochkay, Khrystyna, Stephen V. Brown, Andrew J. Leone, and Jennifer Wu Tucker. 2022. "Textual Analysis in Accounting: What's Next?" *Contemporary Accounting Research*.
- Bozanic, Zahn, Darren T. Roulstone, and Andrew Van Buskirk. 2018. "Management Earnings Forecasts and Other Forward-Looking Statements." *Journal of Accounting and Economics* 65 (1): 1–20.
- Brown, Nerissa C., Richard M. Crowley, and W. Brooke Elliott. 2020. "What Are You Saying? Using Topic to Detect Financial Misreporting." *Journal of Accounting Research* 58 (1): 237–91.
- Brown, Stephen V., Xiaoli (Shaolee) Tian, and Jennifer Wu Tucker. 2018. "The Spillover Effect of SEC Comment Letters on Qualitative Corporate Disclosure: Evidence from the Risk Factor Disclosure." *Contemporary Accounting Research* 35 (2): 622–56.
- Brown, Tom, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, et al. 2020. "Language Models Are Few-Shot Learners." In *Advances in Neural Information Processing Systems*, 33:1877–1901. Curran Associates, Inc.
- Call, Andrew C., Rachel W. Flam, Joshua A. Lee, and Nathan Y. Sharp. 2022. "Managers' Use of Humor on Public Earnings Conference Calls." SSRN Scholarly Paper. Rochester, NY.
- Chuk, Elizabeth, Dawn Matsumoto, and Gregory S. Miller. 2013. "Assessing Methods of Identifying Management Forecasts: CIG vs. Researcher Collected." *Journal of Accounting and Economics* 55 (1): 23–42.
- Gow, Ian D., David F. Larcker, and Anastasia A. Zakolyukina. 2021. "Non-Answers During Conference Calls." *Journal of Accounting Research* 59 (4): 1349–84.
- Guiso, Luigi, Paola Sapienza, and Luigi Zingales. 2015. "The Value of Corporate Culture." *Journal of Financial Economics*, NBER Conference on the Causes and Consequences of Corporate Culture, 117 (1): 60–76.
- Hail, Luzi, Ahmed Tahoun, and Clare Wang. 2018. "Corporate Scandals and Regulation." *Journal of Accounting Research* 56 (2): 617–71.
- Hansen, Anne Lundgaard, and Sophia Kazinnik. 2023. "Can ChatGPT Decipher Fedspeak?" SSRN Scholarly Paper. Rochester, NY.
- Hassan, Tarek A, Stephan Hollander, Laurence van Lent, and Ahmed Tahoun. 2019. "Firm-Level Political Risk: Measurement and Effects*." *The Quarterly Journal of Economics* 134 (4): 2135–2202.
- Hollander, Stephan, Maarten Pronk, and Erik Roelofsen. 2010. "Does Silence Speak? An Empirical Analysis of Disclosure Choices During Conference Calls." *Journal of Accounting Research* 48 (3): 531–63.
- Hu, Nan, Peng Liang, and Xu Yang. 2023. "Whetting All Your Appetites for Financial Tasks with One Meal from GPT? A Comparison of GPT, FinBERT, and Dictionaries in Evaluating Sentiment Analysis." SSRN Scholarly Paper. Rochester, NY.

- Jha, Manish, Jialin Qian, Michael Weber, and Baozhong Yang. 2023. "ChatGPT and Corporate Policies." SSRN Scholarly Paper. Rochester, NY.
- Kim, Alex G., Maximilian Muhn, and Valeri V. Nikolaev. 2023. "Bloated Disclosures: Can ChatGPT Help Investors Process Information?" SSRN Scholarly Paper. Rochester, NY.
- Kwon, Woosuk, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. "Efficient Memory Management for Large Language Model Serving with PagedAttention." arXiv.
- Li, Feng. 2010. "The Information Content of Forward-Looking Statements in Corporate Filings—A Naïve Bayesian Machine Learning Approach." *Journal of Accounting Research* 48 (5): 1049–1102.
- Li, Kai, Feng Mai, Rui Shen, and Xinyan Yan. 2021. "Measuring Corporate Culture Using Machine Learning." *The Review of Financial Studies* 34 (7): 3265–3315.
- Lopez-Lira, Alejandro, and Yuehua Tang. 2023. "Can ChatGPT Forecast Stock Price Movements? Return Predictability and Large Language Models." SSRN Scholarly Paper. Rochester, NY.
- OpenAI. 2023. "GPT-4 Technical Report." arXiv.
- Radford, Alec, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. "Improving Language Understanding by Generative Pre-Training."
- Radford, Alec, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. "Language Models Are Unsupervised Multitask Learners."
- Siano, Federico, and Peter Wysocki. 2021. "Transfer Learning and Textual Analysis of Accounting Disclosures: Applying Big Data Methods to Small(Er) Datasets." *Accounting Horizons* 35 (3): 217–44.
- Sun, Yu, Shuohuan Wang, Shikun Feng, Siyu Ding, Chao Pang, Junyuan Shang, Jiaxiang Liu, et al. 2021. "ERNIE 3.0: Large-Scale Knowledge Enhanced Pre-Training for Language Understanding and Generation." arXiv.
- Touvron, Hugo, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, et al. 2023. "LLaMA: Open and Efficient Foundation Language Models." arXiv.
- Touvron, Hugo, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, et al. 2023. "Llama 2: Open Foundation and Fine-Tuned Chat Models." arXiv.
- Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. "Attention Is All You Need." In *Advances in Neural Information Processing Systems*. Vol. 30. Curran Associates, Inc.
- Wang, Ben, and Aran Komatsuzaki. 2021. "GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model." Software.

Appendix A: Glossary

Terminology	Definition
<i>API</i>	A communication protocol to interact with an internet service.
<i>Completion</i>	The text output generated by the GLLM.
<i>Few-shot</i>	Providing a prompt with a few examples as part of the instructions.
<i>Fine-tuning</i>	Retraining (part of) the model weights for a specific task.
<i>GLLM</i>	Generative Large Language Model.
<i>GPT</i>	Generative Pre-trained Transformer.
<i>Inference</i>	Using a GLLM to make a generation.
<i>LLM</i>	Large Language Model.
<i>Model weights</i>	The parameter values (“coefficients”) of the neural network.
<i>Prompt</i>	The text input provided to the GLLM.
<i>Prompt engineering</i>	The task of developing a prompt for your task.
<i>Token</i>	A token is the unit of observation for a GLLM. A single word generally consists of multiple tokens. One token is on average four characters in the English language.
<i>Zero-shot</i>	Providing a prompt without any examples, just the instruction.

Appendix B: Simple examples

This appendix provides a few examples of how one would use GLLMs and prompt-engineering to automate various types of textual analysis tasks. To save space, the prompts are shown as their zero-shot version, but they would be easily adapted to the few-shot or fine-tuning approaches.²³

Identifying statements about political risk:

Consider a research project that requires identifying statements in earnings calls related to political risk. A GLLM can solve this task by prompting it with:

```
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 =
```

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

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

²³ I've 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.

Identifying CEO Humor:

Consider a research project that requires classifying CEO statements in an earnings conference call as trying to be funny (yes) or not (no). A GLLM can solve this task by prompting it with:

```
Classify the CEO statements below as trying to be funny (yes) or not (no):
```

```
1. Thank you, everyone, for joining us today. I must say, our quarterly results are so good, even my dog started investing in our stock!
```

```
2. Thank you, everyone, for joining us today. I hope that you have a wonderful day!
```

```
Return the result in the following JSON format:
```

```
[
    {"index" : i, "funny" : Yes/No}
]
```

```
JSON =
```

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

```
[
    {"index" : 1, "funny" : "Yes"},
    {"index" : 2, "funny" : "No"}
]
```

Extracting EBITDA-like numbers for the MD&A section:

Consider a research project that requires extracting EBITDA-like numbers from a company's Management Discussion and Analysis (MD&A) section. A GLLM can solve this task by prompting it with:

```
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 =

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

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

Clustering task:

Consider a research project that requires clustering company press release headlines into unsupervised topic categories. A GLLM can solve this task by prompting it with:

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 to 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 =

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

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


Data cleaning task:

Consider a research project that needs to work with PDF documents that are challenging to convert to text. The extract text is often split and cut-off, making it hard to extract sentence. A GLLM can fix this problem by prompting it with:

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. The 2025 Sustainability
Goals we are now pursuing reflect
a heightened commitment to
going beyond compliance and
thinking holistically about how our
operations and products affect
the world and every life in it. For
additional information on the development
of our 2025 Sustainability
Goals, see global challenges.

Return the result in the following JSON format:

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

JSON =

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

```
[  
  {"i" : 1, "sentence" : "In 2018, we entered the third year of an  
ambitious 10-year goal cycle."},  
  {"i" : 2, "sentence" : "The 2025 Sustainability Goals we are now  
pursuing reflect a heightened commitment to going beyond compliance  
and thinking holistically about how our operations and products  
affect the world and every life in it."},  
  {"i" : 3, "sentence" : "For additional information on the  
development of our 2025 Sustainability Goals, see global  
challenges."}  
]
```

Appendix C: Earnings call non-answer prompts

C.1 - ChatGPT zero shot (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 =

C.2 - GPT-4 zero shot (Table 1 column 4):

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 unambiguously indicates:

1. A clear absence of specific details or information that prevents them from providing an answer.
2. A conscious decision not to divulge particular information at the moment.

Important criteria for valid sentences:

- Must state a lack of knowledge about a specific detail.
- Must clearly indicate a choice not to share a particular detail.

Exclude:

- Statements expressing general uncertainty or vagueness.
- Sentences about company plans or potential future actions.
- Disclaimers, deferrals to others, or any remarks that still convey a type of answer or opinion.

****In essence, the qualifying sentence should firmly suggest: "I don't have that specific detail" or "I choose not to share that specific detail now."****

Provide your response using the following JSON format: JSON = {{
 "sentences" : []
}}

Examples of sentences to exclude / ignore:

- "I'll let another person handle the specifics."
- "But time will tell."
- "We will decide that in the future."
- "The exact details are yet to be decided."

Be extremely selective. Only include sentences you are 100% sure about.

JSON =

C.3 - ChatGPT zero shot 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 =

C.4 - GPT-4 zero shot filter with categories (Table 3a columns 1 to 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.

****Task 2 - reasons:****

Why is the manager reluctant, unable, or unwilling to share the specific information?

Your choices:

- dont_know -> the manager does not know or have the numbers and/or it is too early to tell
- cant_give -> the manager states that they don't provide that information as per policy, norm, or principle
- proprietary -> the manager states that the information is proprietary or sensitive for competitors
- invalid_match -> the answer is incorrectly flagged as containing a refusal or hesitation to answer, select if no sentences.

****Task 3 - alternative:****

Does the manager still provide an alternative answer, if so what do they provide? Your choices:

- range_or_perc -> the manager instead provides a range estimate or another type of quantitative estimate such as a percentage
- comparison -> the manager instead makes a quantitative or qualitative comparison to some other point of reference
- qualitative -> the manager instead provides a qualitative statement without sharing exact details or numbers
- no_alternative -> the manager provides no further information or only explains their reasoning for not answering
- invalid_match -> the answer is incorrectly flagged as containing a refusal or hesitation to answer, select if no sentences.

****Task 4 - topic:****

What is the part of the question about that the manager is not, or only partially, answering? Your choices:

- forward_looking - the manager refuses to provide guidance or other forward-looking information
- breakdown - the manager refuses to provide a breakdown or more granularity or detail on something
- rd_clinical_trails - the manager refuses to provide more information on a research & development projects or clinical trails
- regulation_certification - the manager refuses to speak about a regulatory matter, a compliance issue, or an ongoing certification by a third-party
- customer_or_supplier - the manager refuses to provide details on major customers or suppliers
- major_investment - the manager refuses to speak about a merger and/or acquisition or an other type of major investment, collaboration, or partnership
- other

****Task 5 - sentiment:****

In the manager response, are they trying to paint a positive, negative, or neutral picture?

- optimistic -> the manager is trying to paint a (somewhat) positive picture with their alternate answer
- pessimistic -> the manager is trying to paint a (somewhat) cautionary or pessimistic picture with their alternative answer
- neutral -> the manager is painting is trying to paint a neutral picture

Provide your response using the following JSON format: JSON = {{
 "sentences" : [],
 "reasons-discussion" : a brief discussion of your choice for "reasons",
 "reasons" : dont_know / cant_give / proprietary / invalid_match,
 "alternative-discussion" : a brief discussion of your choice for "alternative",
 "alternative" : range_or_perc / comparison / qualitative / no_alternative / invalid_match,
 "topic-discussion" : a brief discussion of your choice for "topic",
 "topic" : forward_looking / breakdown / rd_clinical_trails / regulation_certification / customer_or_supplier / major_investment,
 "sentiment-discussion" : a brief discussion of your choice for "sentiment"
 "sentiment" : optimistic / pessimistic / neutral
 }}

JSON =

C.5 - ChatGPT fine-tuned filter with categories (Table 3a columns 4 to 6) – prompt + completion

Analyst question:

> {question}

Manager response:

> {answer}

####

{json} <|end|>

Appendix D: Prompt engineering tips and best practices

GLLMs generate text left to right, so their own generations become part of the prompt.

The left-to-right aspect of GLLMs is useful to keep in mind as it (1) might create unexpected behavior when you are trying to perform multiple tasks at once, and (2) can be utilized to increase the quality of the completions. A helpful technique to improve completion quality is to ask the model to explain itself before generating a final outcome. Such “chain of thought prompting” will generally create better and more consistent outputs as the model is using its own explanation as part of the classification decision, which improves quality.

Optimize the number of tokens used by your prompt and completion.

Tools like *tiktoken* and the OpenAI *tokenizer* are helpful tools to understand how many tokens your prompt and completion will use.²⁴ Optimizing the number of tokens can save substantial time and money. Reducing the number of tokens by 10% will generally yield in a 10% speed up and 10% cost saving. Besides prompt optimization, one can also save tokens by classifying multiple items with a single instruction, for example by having it return a list instead of a single item.

Be specific and to the point, less is often more.

When using a zero-shot, or few-shot, approach it can happen that the GLLM ignores certain parts of your instructions, which can make the completions hard to control. This behavior is usually the result of your instruction being too vague or ambiguous or because your instructions are too long. In my experience, the model will weigh certain parts of the instructions more or less heavily in a way that is hard to predict. To alleviate this, it can help to remove less important instructions and stick with instructions that are short and explicit. Another approach that one can take is to explicitly tell the GLLM what instructions are more important. For example, by ordering your instructions in order of importance and including a note to explicitly explain the order.

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” often cause the model to generate significantly different outputs relative to when they are excluded. The influence of a

²⁴ <https://github.com/openai/tiktoken> and <https://platform.openai.com/tokenizer>, last accessed April 4th 2023.

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.

If you can't get your prompt to work, consider using a smaller model with fine-tuning.

The largest GLLM models are best for zero-shot or few-shot approaches. However, sometimes a task is too complex or specific for a zero-shot prompt. In those cases, you can achieve better performance by taking a smaller GLLM model and fine-tuning it on a reasonable number of training examples. A fine-tuned smaller model will often outperform a larger model at a fraction of the costs.

Combining prompts (i.e., prompt-chaining).

For more complex tasks it might not be possible or effective to try and do everything through a single prompt. In this scenario a powerful technique is to chain a series of prompts together, called “prompt chaining”. For example, you might want to analyze the entire risk factor section of an annual filing, which likely won't fit inside a single prompt due to the length. To deal with that, you can split up the risk factors, feed each risk factor into the GLLM with a prompt to summarize it, and then combine all the summarized risk factors together into the final prompt. Similarly, to increase the quality of your completions you can feed the model completions back into the model with a prompt asking it to check or improve the previous completion and generate a new completion in the process. The *LangChain* Python library provides functionality to make it easier to chain prompts together.²⁵

Other useful prompt engineering resources:

- <https://github.com/openai/openai-cookbook>
- <https://github.com/dair-ai/Prompt-Engineering-Guide>
- <https://github.com/f/awesome-chatgpt-prompts>

²⁵ <https://github.com/hwchase17/langchain>

Appendix E: Using and fine-tuning a local GLLM

Using a third-party API, such as those by OpenAI or Anthropic, is generally the easiest and quickest way to interact with a GLLM. However, using a third-party API raises concerns about data privacy and replicability. For example, OpenAI announced in August 2023 that they would stop supporting (i.e., delete) all existing fine-tuned models to facilitate a transition to a new generation of fine-tunable models. Section 5.2 of the paper discusses strategies that researchers can deploy to minimize replicability risks when using third-party providers. An alternative way to mitigate data privacy and replicability concerns is to use a GLLM with downloadable weights. These GLLMs can be run locally or in the cloud and give the researcher possession of the weights, which can increase replicability and save costs.

But local GLLMs also come with downsides. First, the state-of-the-art performance by models such as ChatGPT and GPT-4 by OpenAI and Claud2 by Anthropic are only available through their APIs. Zero-shot performance of downloadable GLLMs is generally worse out-of-the-box relative to GLLM providers. Second, running GLLMs directly requires specific hardware, such as GPUs with sufficient vram for the model you want. Getting access to this hardware can be challenging for very large models. Third, the inference speed (i.e., tokens per minute), is generally slower with a local GLLM than with a third-party provider. There are techniques to help speed up GLLMs (e.g., vLLM) but integrating them requires researcher time and can lead to compatibility problems. Finally, running a GLLM directly does not create perfect replicability. Differences in the hardware and software can result in minor differences depending on the system, even for the same model.

With these caveats in mind, downloadable GLLMs are being developed at a rapid pace and are becoming better, faster, and easier to use. To illustrate, I provide a few anecdotal examples of replicating parts of the paper using the Llama2 models, a cloud instance with a GPU, and the vLLM Python library by Kwon et al. (2023). Code examples are available on the companion website. I start by provisioning a cloud instance through Vast.AI with an A6000 GPU. The A6000 GPU makes it possible to run the 7 billion (i.e., 7b) and 13b chat variations of the Llama 2 model. The Llama(2) models are made available for free by Meta and have permissible licenses for research. Both chat and non-chat versions are available. I explore two replications, a zero-shot replication of the results in Table 1 and a fine-tuning replication of the results in Table 3.

I feed the evaluation sample from Table 1 through the 7b and 13b Llama2 models, using the same prompt used for ChatGPT (Appendix C.1).²⁶ The outputs show that the Llama2 models struggle to follow the instructions. The Llama2 models frequently return text beyond the JSON response or provide JSON that is not parseable. Around 16% (8%) of the 500 evaluations are unparseable with the 7b (13b) model. Furthermore, a manual inspection of the parseable results indicates that the performance is not good out-of-the-box. For example, the 13b model marks 86% of the answers as containing a non-answer. However, there are a few important caveats to these results: detecting non-answers is an unusually hard task and more importantly I did not adapt the prompts to the Llama2 models to improve the results. For simpler tasks it is plausible that one could get these models to work quite well with sufficient prompt engineering. Furthermore, running zero-shot predictions using the Llama2 models is about 10 times cheaper than the ChatGPT OpenAI API.

Large proprietary models shine for zero-shot tasks, whereas smaller models shine when fine-tuned. So next, I replicate the ChatGPT fine-tuning step in Table 3 by fine-tuning a Llama2 7b chat model using a similar Vast.ai setup but swapping out the A6000 for an RTX 4090.²⁷ Fine-tuning took about an hour on the ~2,100 training observations. I then apply the fine-tuned model on the evaluation set and the performance results are significantly better and more usable. The model strictly follows the instructions and produces valid JSON, which wasn't possible with the zero-shot tests. Untabulated results show that the fine-tuned Llama2 7b model yields an accuracy (F1 score) of 81% (77%) for the non-answers, 90% (90%) for the non-answer justification, 76% (77%) for the non-answer type, 71% (69%) for the question type, and 83% (83%) for the non-answer sentiment. These statistics are getting close to those of the fine-tuned ChatGPT model, at 1/100th the cost. It only costs around \$0.04 to generate 1,000 predictions using the Llama2 7b model, versus \$4.18 for the same 1,000 predictions using the fine-tuned ChatGPT model through OpenAI.

These anecdotal results highlight that it is worth exploring local GLLM models when your research task hits cost constraints using third-party providers, requires strict data privacy, or has constraints surrounding replicability and maintaining possession of the model weights.

²⁶ You can apply to download the Llama2 model weights here: <https://ai.meta.com/llama/>

²⁷ The A6000 has 48gb vram relative to the 24gb vram of an RTX 4090. However, the RTX 4090 trains much faster.

Figure 1: Non-answers - Venn diagram

A Venn diagram showing the overlap in non-answers between the full GPT method and the Gow et al. (2021) method. Online Appendix 2 provides representative examples of answer excerpts that fall in the two outer slices.

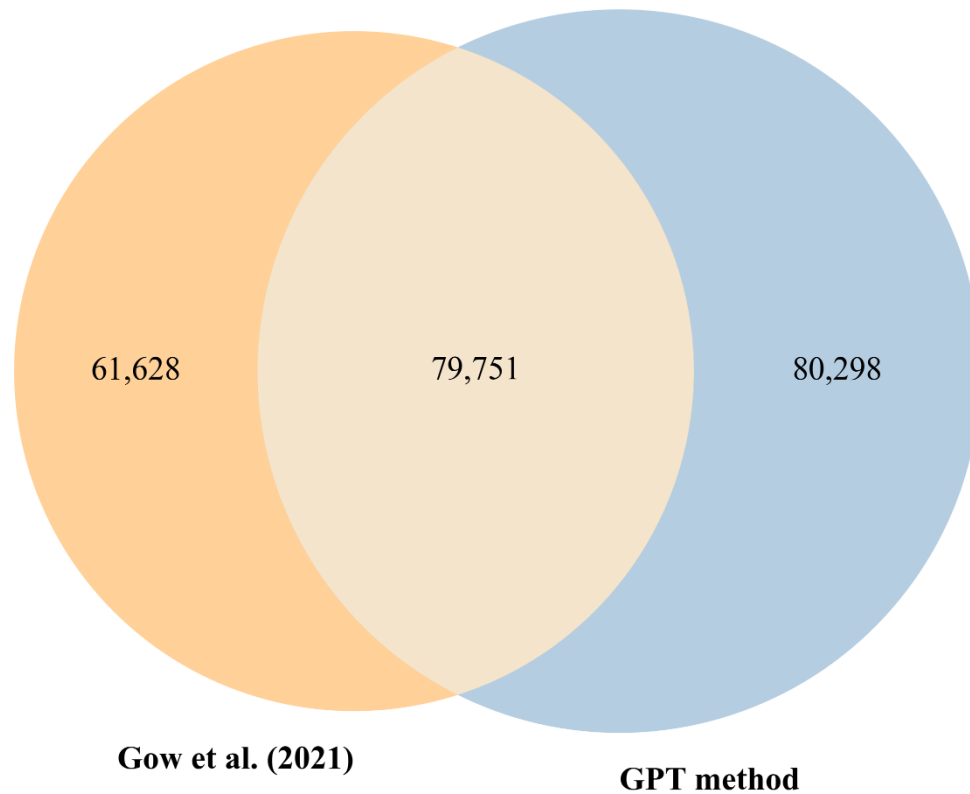


Figure 2: Non-answers – time-trends and COVID-19 impact

Figure 2a: Non-answer % over time (4-quarter moving average)

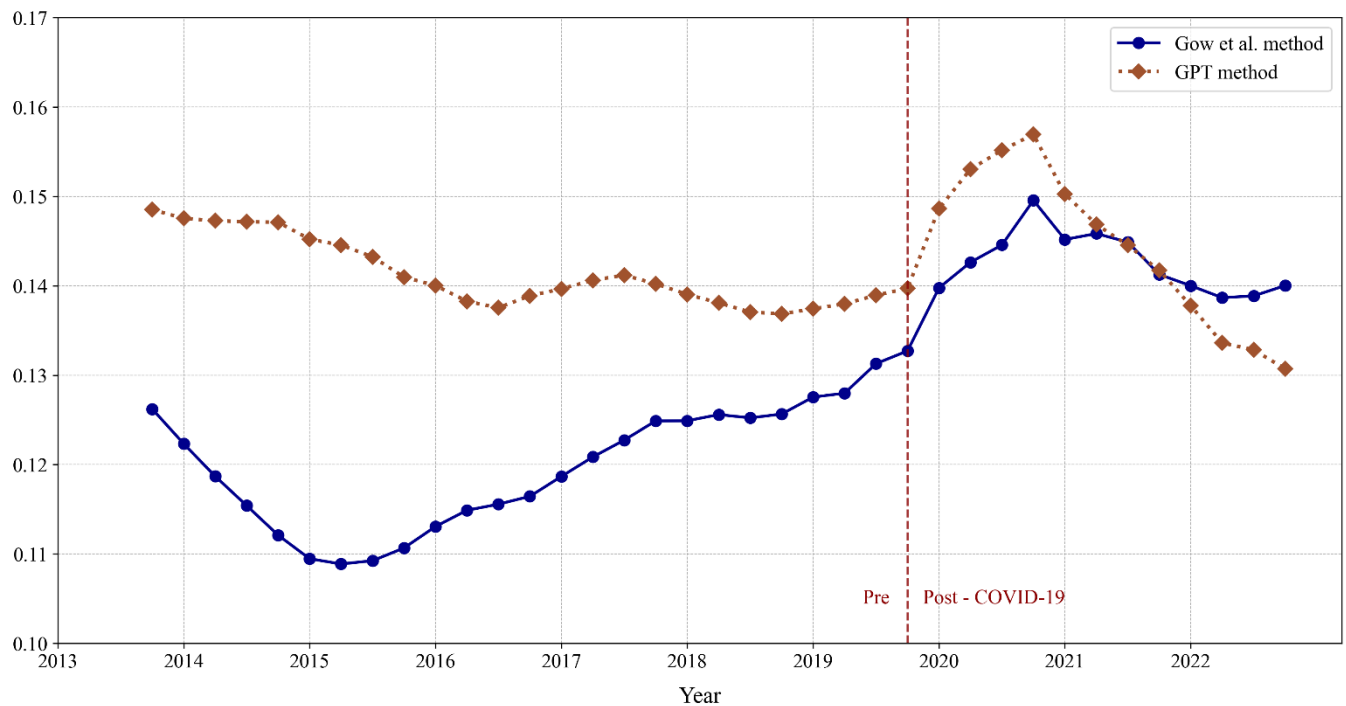


Figure 2b: Non-answer % over time split by non-answer sentiment (4-quarter moving average)

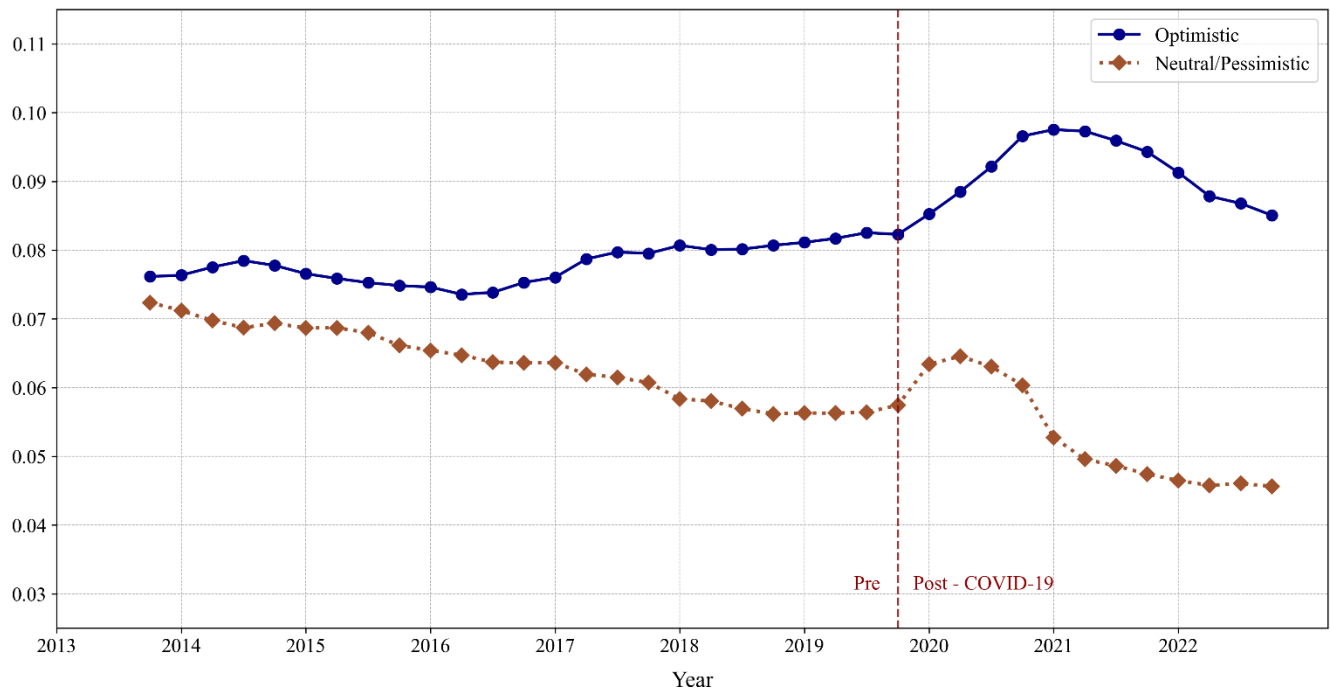


Figure 2c: Non-answer % over time split by non-answer justification (4-quarter moving average)

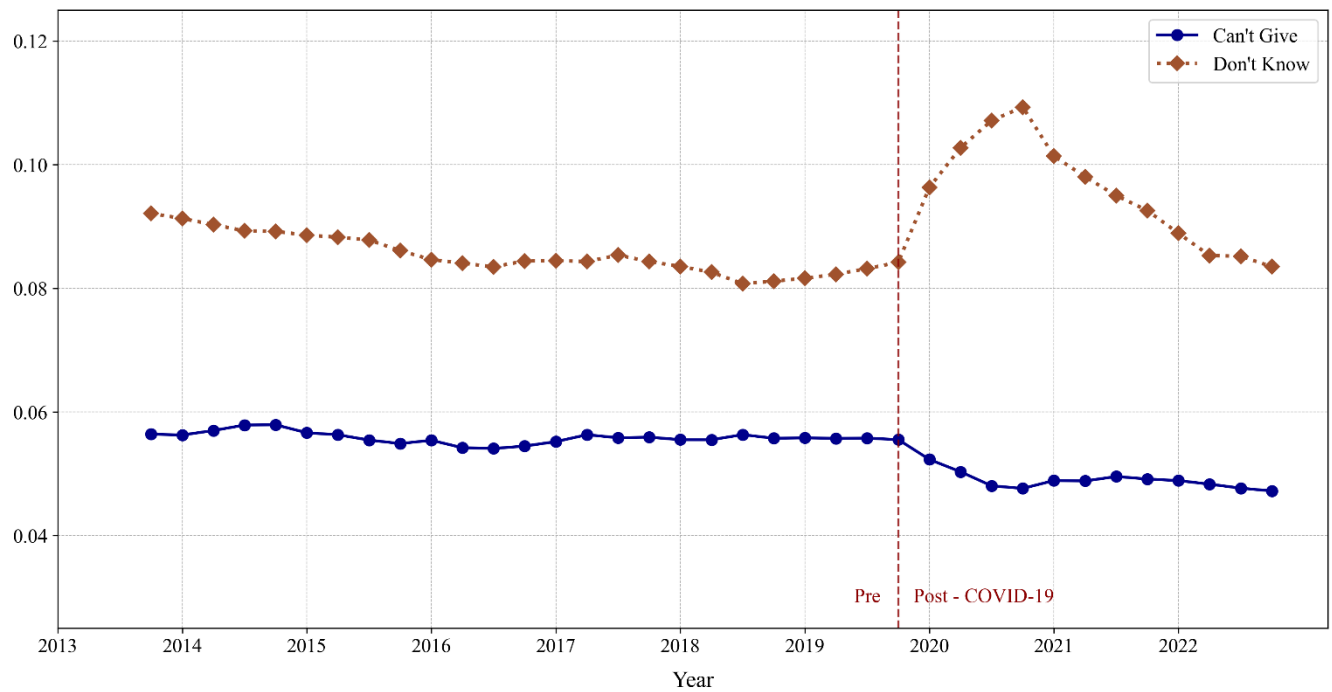


Figure 2d: Non-answer % over time split by non-answer alternative (4-quarter moving average)

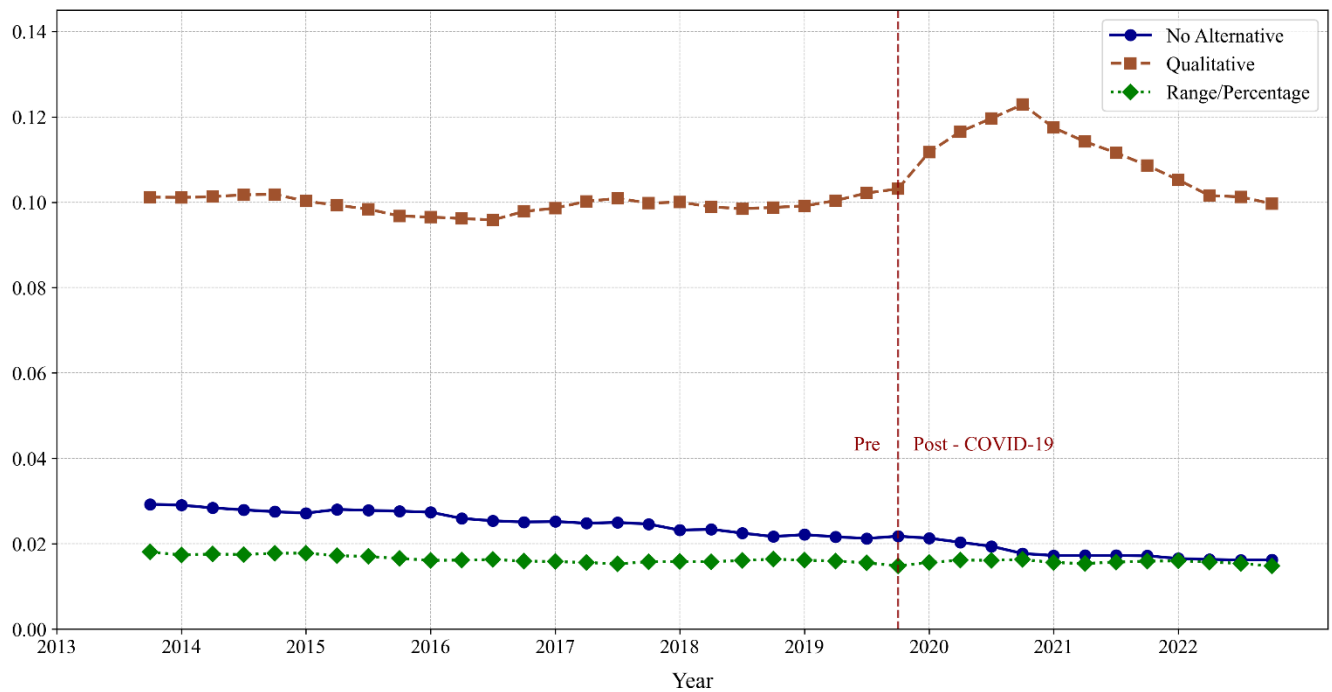


Figure 2e: Non-answer % over time split by question type (4-quarter moving average)

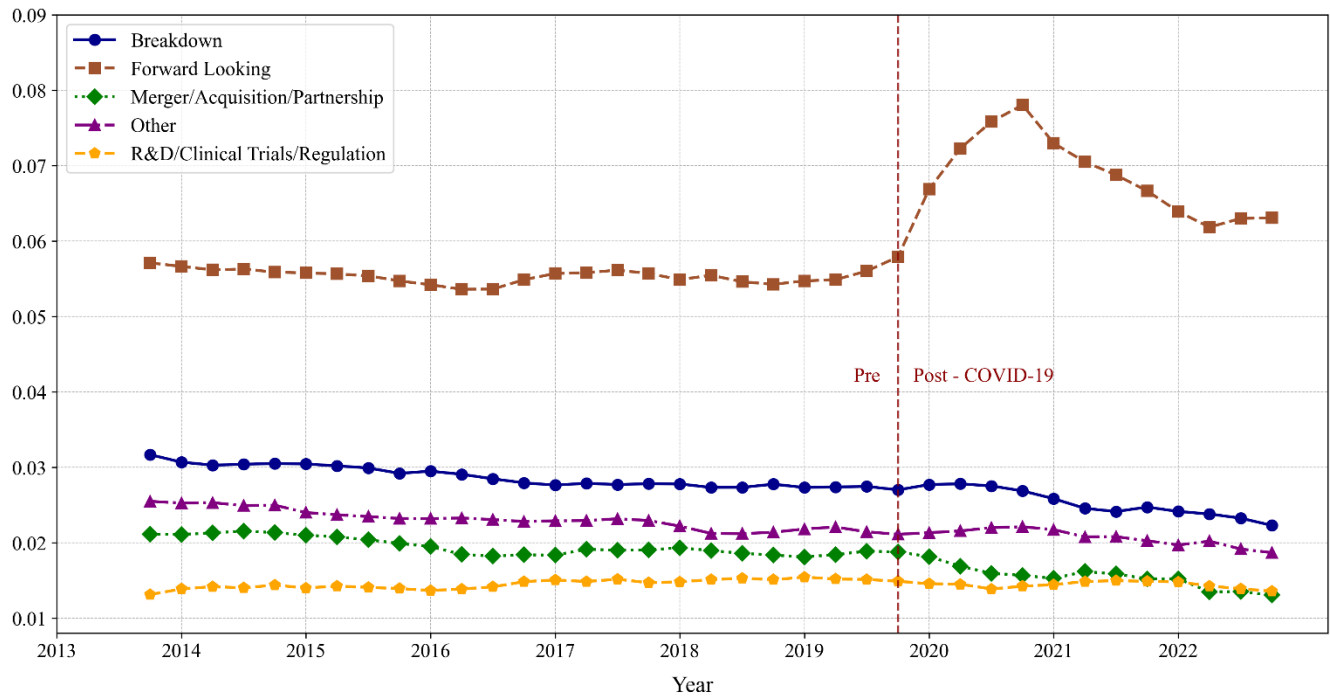


Table 1: Non-answers - classification report

	(1)	(2)	(3)	(4)	(5)	(6)
	Manual	Gow et al. (2021)	ChatGPT Zero-shot	GPT-4 Zero-shot	Keyword + GPT-3 Filter + ChatGPT	(5) + ChatGPT FT Filter
Answer	423	443	418	398	408	423
Non-answer	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
Non-answers:						
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
N	500	500	500	500	500	500
Mean tokens per Q&A pair			435 tokens	550 tokens	300 tokens	335 tokens
Costs per 1,000		\$0	\$0.66	\$16.98	\$0.48	\$1.06

This table reports the performance statistics for the various non-answer classification approaches. ChatGPT refers to the *gpt-3.5-turbo* model.

Table 2: Non-answers – large sample analysis*Panel A: Question-answer pair level*

	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)
	Obs.	Mean	Std. Dev	P5	P25	P50	P75	P95
<i>GPT method</i>								
Non-answer	1,152,505	0.139	0.35	0	0	0	0	1
<i>Gow et al. (2021)</i>								
Non-answer	1,152,505	0.123	0.33	0	0	0	0	1
<i>Question features</i>								
Positive tone	1,152,505	0.011	0.02	0.00	0.00	0.00	0.02	0.04
Negative tone	1,152,505	0.011	0.02	0.00	0.00	0.00	0.02	0.04
Uncertainty	1,152,505	0.014	0.02	0.00	0.00	0.01	0.02	0.05
Fog index	1,152,505	10.23	4.55	4.33	7.50	9.61	12.14	18.07
Contains numbers	1,152,505	0.394	0.49	0.00	0.00	0.00	1.00	1.00
Contains money or percent	1,152,505	0.160	0.37	0.00	0.00	0.00	0.00	1.00
Contains location	1,152,505	0.136	0.34	0.00	0.00	0.00	0.00	1.00
Contains date or time	1,152,505	0.544	0.50	0.00	0.00	1.00	1.00	1.00

This table reports the non-answer statistics for the full sample. The *GPT method* corresponds to the method from Table 1 Column 6. Panel A reports the statistics at the question-answer pair level and Panel B reports the statistics at the earnings call level.

Table 2: Non-answers – large sample analysis (cont.)*Panel B: Earnings call level*

	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)
	Obs	Mean	Std. Dev	P5	P25	P50	P75	P95
Number of Q&A pairs	63,959	18.019	8.93	6	12	17	23	34
<i>GPT method</i>								
Number of non-answers	63,959	2.502	2.20	0.00	1.00	2.00	4.00	7.00
Percentage of non-answers	63,959	0.142	0.12	0.00	0.06	0.13	0.20	0.36
<i>Gow et al. (2021) method</i>								
Number of non-answers	63,959	2.979	2.45	0.00	1.00	2.00	4.00	8.00
Percentage of non-answers	63,959	0.127	0.12	0.00	0.04	0.11	0.19	0.33

Table 3: Non-answer dimensions – model performance

	(1)	(2)	(3)	(4)	(5)	(6)
	GPT-4 - zero shot			Fine-tuned ChatGPT		
	Precision	Recall	F1 score	Precision	Recall	F1 score
Answer	1.00	0.41	0.58	0.93	0.73	0.82
Non-answer	0.86	1.00	0.93	0.93	0.98	0.96
	0.89	0.88	0.85	0.93	0.93	0.93
Can't give	0.98	0.96	0.97	0.93	0.91	0.92
Don't know	0.98	0.99	0.98	0.95	0.96	0.95
	0.98	0.98	0.98	0.94	0.94	0.94
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
	0.92	0.92	0.92	0.88	0.88	0.88
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
	0.91	0.91	0.91	0.82	0.81	0.81
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
	0.92	0.92	0.92	0.87	0.87	0.87
N		400			400	
Mean tokens per Q&A pair		1,267 tokens			324 tokens	
Costs per 1,000		\$45.53			\$4.18	

This table reports the performance statistics for the non-answer sub-classifications. ChatGPT refers to the *gpt-3.5-turbo* model.

Table 4: Non-answer dimensions – full sample results

	(1)	(2)	(4)	(5)	(6)
	Question-answer pair level			Filing level	
	Count	Proportion	Mean	Std. Dev	Median
Answer	30,482	16.0%			
Non-answer	160,049	84.0%	2.50	2.20	2
Can't give	60,082	37.5%	0.94	1.24	1
Don't know	99,967	62.5%	1.56	1.63	1
Range or percentage	18,908	11.8%	0.30	0.59	0
Qualitative	113,446	70.9%	1.77	1.71	1
Complete refusal	27,695	17.3%	0.43	0.81	0
Breakdown	33,488	20.9%	0.52	0.85	0
Forward looking	66,704	41.7%	1.04	1.24	1
Related party	20,424	12.8%	0.32	0.66	0
R&D or regulation	13,239	8.3%	0.21	0.76	0
Other	26,194	16.4%	0.41	0.73	0
Optimistic	88,199	55.1%	1.38	1.44	1
Neutral or pessimistic	71,850	44.9%	1.12	1.44	1
N	190,531 (full) - 160,049 (non-answers only)		Number of filings: 63,959		

This table reports the full sample statistics for the non-answer dimension classifications at the question-answer pair level and the filing level.

Table 5: Non-answer determinants – single variable regressions*Panel A: Gow et al. replication*

	(1) Non-answer GPT approach	(2) Non-answer Gow et al.	(3) Non-answer Can't give	(4) Non-answer Positive	(5) Non-answer Complete refusal	(6) Non-answer Qualitative	(7) Non-answer Range or Perc.
Q - Positive sentiment	-48.598*** (3.019)	-34.180*** (4.062)	-14.813*** (0.641)	-1.666 (3.764)	-14.421*** (0.844)	-22.937*** (3.700)	-11.240*** (0.432)
Q - Negative sentiment	-14.763*** (3.020)	6.892** (2.788)	-3.505* (1.761)	-24.240*** (1.455)	-1.209 (1.147)	-13.947*** (1.680)	0.394 (0.628)
Q - Uncertainty	70.868*** (3.754)	50.331*** (3.376)	27.300*** (0.911)	42.506*** (3.075)	10.081*** (1.299)	50.329*** (3.391)	10.458*** (0.870)
Q - Fog index	0.116*** (0.013)	0.226*** (0.020)	0.010* (0.005)	0.170*** (0.007)	-0.050*** (0.006)	0.196*** (0.010)	-0.030*** (0.002)
Q - Contains numbers	3.262*** (0.269)	2.302*** (0.191)	0.770*** (0.098)	2.149*** (0.203)	0.008 (0.036)	2.302*** (0.212)	0.952*** (0.045)
Q - Contains money	1.550*** (0.290)	1.347*** (0.159)	0.487*** (0.102)	0.981*** (0.230)	-0.191*** (0.056)	0.724** (0.235)	1.017*** (0.052)
Q - Location	0.745*** (0.159)	1.074*** (0.096)	-0.034 (0.046)	0.867*** (0.091)	-0.291*** (0.060)	1.127*** (0.111)	-0.091** (0.038)
Q - Date or time	3.299*** (0.163)	2.725*** (0.174)	0.999*** (0.063)	2.455*** (0.148)	-0.146*** (0.025)	2.703*** (0.130)	0.742*** (0.038)
Call FE	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Multivariate	No	No	No	No	No	No	No
N	1,152,473	1,152,473	1,152,473	1,152,473	1,152,473	1,152,473	1,152,473

This table reports the regression results of regressing each question characteristic independently on the non-answer outcomes at the Q&A pair level. 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.

Table 5: Non-answer determinants – single variable regressions (cont.)*Panel B: GPT method extension*

	(1) Non-answer Can't give	(2) Non-answer Positive	(3) Non-answer Full reject	(4) Non-answer Qualitative	(5) Non-answer Range or Perc.
Q - Breakdown	21.225*** (0.750)	-5.175*** (0.561)	-3.368*** (0.309)	-5.751*** (0.447)	9.119*** (0.249)
Q - Forward looking	-17.545*** (0.878)	10.048*** (0.563)	-5.836*** (0.399)	5.130*** (0.574)	0.706** (0.305)
Q - Related party	21.220*** (1.135)	-2.111*** (0.426)	9.815*** (0.380)	-3.580*** (0.426)	-6.235*** (0.320)
Q - R&D or regulation	-9.988*** (1.088)	-8.191*** (0.973)	8.606*** (0.732)	-1.400 (0.812)	-7.206*** (0.371)
Q - Other	-6.222*** (0.910)	-5.331*** (0.471)	1.682*** (0.321)	1.530*** (0.392)	-3.211*** (0.165)
Call FE	No	No	No	No	No
Multivariate	No	No	No	No	No
N	160,049	160,049	160,049	160,049	160,049

Online Appendix

O.A. 1. – Full description of GPT method

This section provides additional detail on the four steps of the *GPT method*. This approach demonstrates the various ways to use GLLMs, which makes it more extensive than necessary. A two-step approach (e.g., keyword filter + fine-tuned GLLM) is sufficient for most use-cases.

Step 1: Keyword filter

I first narrow down the Q&A pairs to those that possibly include a non-answer. Identifying possible non-answer is an easier task than identifying exact non-answers. Specifically, any false positives (i.e., sentences matched as a possible non-answer, but aren't) will be taken care of in the later stages. So, the objective is to reduce the sample of possible non-answers while minimizing on false negatives. Minimizing false negatives is the primary concern as any Q&A pairs without a keyword match will be marked as an answer and not be analyzed by the more powerful methods later.

To develop the keyword list, I use the words in the regular expressions by Gow et al. (2021) as a starting point. I then manually extend this list with unigrams, bigrams, or trigrams that help reduce false negatives. The full keyword list is shown below. I apply the keyword search at the sentence level and mark the Q&A pairs with one or more keyword match for further processing.

Trigrams:

call_it_out, at_this_time, at_this_point, at_this_moment, break_it_out, don_t_have, don_t_know

Bigrams:

not_going, will_not, won_t, by_region, get_into, that_level, are_not, don_t, do_not, give_you, break_out, splice_out, tell_you, too_early, can_t, can_not, not_ready, right_now, no_idea, not_give, not_sure, wouldn_t, haven_t

Unigrams:

cannot, comment, commenting, comments, unable, guidance, guide, guiding, forward, hard, talk, range, disclose, report, privately, forecast, forecasts, forecasting, specific, specifics, detail, details, public, publicly, provide, breakout, statement, statements, update, announcement, announcements, answer, answers, quantify, share, sharing, information, discuss, mention, sorry, apologies, apologize, recall, remember, without, specifically, difficult, officially

Step 2: Basic machine learning filter

The keyword approach is simple and fast, but it is naïve in its implementation as it doesn't take any context into account. An alternative filtering approach is to use a basic supervised machine learning algorithm. To demonstrate, I next train a classifier using the OpenAI API to classify sentences that possibly contain a non-answer. Like in step 1, the objective is to reduce the sample while minimizing false negatives. To train the classifier, I randomly draw a sample of 500 Q&A pairs and classify the 2,100 answer sentences. The “in-sample” performance statistics show a non-answer recall score of 97%, which indicates a minimal number of false negatives. I apply this classifier to the KW matched sentences and retain Q&A pairs with one or more sentence matches.

The prompt and completion templates for the GPT-3 Babbage model are simple and shown below:

```
prompt_template = """
<|input|> {sentence} -> """

completion_template = """ {answer - 0/1} <|end|>
```

Steps 3 and 4: ChatGPT methods

The two filtering steps reduce the sample of Q&A pairs with possible non-answers to about 1/3rd of the full sample. These remaining pairs contain the answers that are hardest to classify as they all have some element of a non-answer to them. I demonstrate two ways, zero-shot and fine-tuning, of using a ChatGPT model to solve the remaining classifications. These final classifications are done at the Q&A pair level, not the sentence level, to provide the full question and answer context.

For the zero-shot ChatGPT approach (i.e., step 3) I design the prompt shown in Appendix C.3. The zero-shot prompt has a few notable design considerations. First, I am only applying this prompt on the sample of Q&A pairs that remain after the filtering step, which makes it a distinct task from the full sample zero-shot prompt in Appendix C.1. Second, I help ChatGPT by providing it with the matched sentences from steps 1 and 2 (i.e., the “comments” portion in the prompt). Third, I frame the task in terms of “evaluate the work of a research assistant”, to help nudge ChatGPT to make more detailed and nuanced judgments for trickier cases where the classification isn’t immediately obvious. And finally, I apply the chain-of-thought technique to improve classifications by asking ChatGPT to first make a detailed assessment before its final classification.

The zero-shot ChatGPT approach works well, although it sometimes over-estimates the probability of a non-answer. This is a challenge with zero-shot approaches as it is hard to convey complex concepts in a few lines of text, especially when the task is niche and context specific. Fine-tuning methods can help with that, as they convey the instructions through examples rather than prompt instructions. So, to further improve the classifications coming out of step 3, I feed all the classifications through a ChatGPT model that is fine-tuned on the zero-shot classifications by GPT-4. The full prompts are in Appendix C.4 and C.5. A few notable things to point out:

- I identify the common types of false positives by step 3 (e.g., statements about not pursuing company actions being flagged as non-answers) and tailor the GPT-4 prompt to specifically filter these out. Providing more task specific context is especially helpful with GPT-4.
- Instead of a 0 / 1 classification, I require the model to identify the exact non-answer phrases. This is more challenging but, in my testing, increasing the number of tokens to generate reduces outcome variability and can improve performance. Ex-post, I transform the matched sentences into a 0 / 1 variable, where no sentences receive a 0 and a 1 otherwise.
- In my GPT-4 prompt (Appendix C.4), I ask the model to make five classifications at once. Doing so makes performance slightly harder to evaluate, however, in my testing it can yield better overall performance. The types of questions themselves act as part of the instructions, making it easier to convey to the GLLM what you want it to do. For example, asking GPT-4 to also classify the non-answer justification seems to improve its ability to understand what a non-answer is in the first place.

O.A.2 – Examples of GPT method improvements

This appendix provides a few illustrative examples of the types of false positives and false negatives that the Gow et al. (2021) measure is susceptible to, but not the GPT method.

Examples of Gow et al. (2021) false positives:

Note: bolded parts are the portion matched by the Gow et al. (2021) regular expressions.

- So it's been positive across the board have **not had a single negative comment**.
- From a pure rent standpoint **we don't have the** productivity yet.
- **I don't know** whether private equity or somebody else is going to buy that business.
- So, we have the ability to communicate with them on a very regular basis over long periods of time which we don't get to do when you are **offline**.
- Well, we are seeing a number of customers gearing up to try to be participants in the PMI, **we don't know** who will win that ultimately, and I don't think they do either frankly.
- So, we have the contracts to and we've had the commitments that we're going to be delivering future homes, but generally the builder **won't release those orders** until he thinks he can get that jobsite started.
- So many times in this call, we would get the question of why aren't your margins -- in a tight market, why **aren't you getting enhancement in your margins**, et cetera, and we would always say, but that's not our model.
- But we are **not in the current guide** that we have.

Examples of Gow et al. (2021) false negatives:

- But we're not in a position yet to make any further detailed announcements regarding their work.
- No, I mean I don't think we plan on getting that granular.
- Anything more than that, I don't -- I want to stay away from specifics.
- So I think it's a bit of a mix in '23, and we'll give you more guidance on that towards the end of the year.
- Truly not possible to tease out individual contributions because the way the auction works, everything is dependent, so if we have fewer ads that's going to play out with different pricing and other sorts of things, so I can give you sort of our gut feel for it, but I can't quantify precisely.
- As for future guidance on new product cadence, that's not something that we can give right now,
- Well, I mean, Linda, again, that's a forward looking question. And we shy away from those as much as we possibly can, no; we shy away from a period. So unfortunately, you got to sort that out on your own.
- As we said, we don't want to dribble out things piecemeal.
- Well, first off, on gigabit, yeah, it is too early to talk about it.