Applying Large Language Models in Accounting: A Comparative Analysis of Different Methodologies and Off-the-Shelf Examples

Huaxia Li
Department of Accounting and Information Systems
Rutgers, The State University of New Jersey
huaxia.li@rutgers.edu

Miklos A. Vasarhelyi
Department of Accounting and Information Systems
Rutgers, The State University of New Jersey
miklosv@business.rutgers.edu

April 2024

Abstract

The emergence of Large Language Model (LLM) presents significant opportunities in accounting, including optimizing current processes, extracting new information, and updating accounting measurements. However, factors such as skill gaps, perceived complexity of integration, and cost constraints have limited its implementation in accounting. This study provides an overview of mainstream LLM utilization methods, including user interfaces and application programming interfaces, and introduces a novel approach via Robotic Process Automation (RPA) integration. The advantages and limitations of each method are discussed, accompanied by a current analysis of the time, labor, and monetary costs involved in employing LLMs for an accounting task. To facilitate practical applications, three off-the-shelf examples are provided. This study contributes to the literature and practice by summarizing and comparing LLM implementation methods, responding to the challenges raised by researchers and stakeholders, and bridging the gap between technology innovation and its practical application in accounting.

JEL Classifications: M41, O14, O33, D61

Keywords: Large language model (LLM); ChatGPT; Robotic Process Automation (RPA); User interface (UI); Application programming interface (API); Pros and cons; Cost analysis

I. INTRODUCTION

The emergence of Large Language Models (LLMs) has brought tremendous potential to reform current practices across various industries (Alberts et al. 2023; Pal, Bhattacharya, Lee, and Chakraborty 2023; Sallam 2023; Vasarhelyi, Moffitt, Stewart, and Sunderland 2023). However, researchers and practitioners have raised several challenges regarding the application of LLMs in the accounting domain, including skill gaps among accountants, the lack of off-the-shelf solutions, the perceived complexity of integration to local systems, and the cost constraints related to initial implementation and subsequent maintenance (Dext 2024; Essex 2020; O'Leary 2023). Currently, there is limited guidance on how to implement LLMs in accounting research and practice, where stakeholders generally lack a strong technical background. To address these challenges, this study reviews prevalent methods for deploying LLMs in the accounting domain, proposes two innovative processes that integrate robotic process automation (RPA) (Huang and Vasarhelyi 2019; Moffitt, Rozario, and Vasarhelyi 2018; Zhang, Issa, Rozario, and Soegaard 2023), and evaluates the benefits, limitations, and costs associated with each method. Additionally, the study offers three practical, off-the-shelf examples that accounting researchers and practitioners can utilize. This study aims to address the research question of how LLMs, through different implementation methods including user interfaces (UI), application programming interfaces (API), and RPA, can be effectively applied in accounting research and practices.[1]

Traditionally, deploying an advanced deep learning-based language model would require strong programming knowledge, with users needing to set up complex infrastructure, manage software and hardware, and handle intricate configurations and parameters. This technical barrier

[1] An API is a set of programming-level protocols that enable different software programs to communicate with each other. A UI is the part of a computer program or application that users can see and interact with. It encompasses various graphical elements like screens, buttons, menus, and other visual components, providing users with the means to interact with the program.

limits the accessibility of these powerful models to a relatively small group of experts in the field. Recent advances in LLMs have brought this technology to a broader spectrum of users through a user-friendly interface and a readily accessible API. With the launch of ChatGPT at the end of 2022, along with its UI and API, a plethora of research and projects leveraging LLMs have been conducted to accomplish various accounting tasks and propose new measurements (Eulerich and Wood 2023; Gu, Schreyer, Moffitt, and Vasarhelyi 2023; Hu, Liang, and Yang 2023; Li, Gao, Wu, and Vasarhelyi 2023; Vasarhelyi et al. 2023). Furthermore, the development of prompt engineering has played a pivotal role in facilitating the harnessing of LLMs, enabling users to tailor model behavior without the need for expensive training or fine-tuning (Gu et al. 2023; Li et al. 2023; Vasarhelyi et al. 2023).[2]

To summarize various methods for applying LLMs in the accounting domain, this study first reviews the accounting literature to examine common LLM application methods. Findings suggest that roughly half of the research uses a chat-based UI to implement LLMs, which usually focuses on case studies and design science research that does not require a large number of queries from LLMs. The other half of the research utilizes an API, focusing on empirical research that leverages LLMs as a novel tool to accomplish accounting tasks and measurements.

Although the UI and API methods have already lowered the application barrier for researchers and practitioners, there are still cases that these two methods cannot handle, particularly in the increasingly complex accounting domain where data can be transmitted between different working pipelines and applications. This problem can be mitigated by RPA, a technology that has been employed in the accounting domain for several years. RPA can interface with LLMs, allowing the combined approach to parse plain text with an LLM's contextual comprehension and

[2] Prompt engineering refers to the process of carefully crafting input queries or instructions to guide the output of LLM like ChatGPT without the need for extensive model retraining or adjustment.

transfer data across systems. This study proposes that LLMs can be applied through RPA in ways that offer significant benefits that UI or API methods alone cannot provide.

This study compares the advantages and limitations of applying an LLM in accounting through a UI, API, UI combined with RPA (UI-RPA), and API combined with RPA (API-RPA). The results are summarized in Table 1. Generally, the UI method is the simplest to use but is not suitable for batch processing and customization. In contrast, the API approach allows for batch processing but can be challenging to integrate with certain existing processes, such as legacy enterprise resource planning (ERP) systems, and requires some programming expertise. The UI-RPA approach enhances productivity and consistency in results, yet demands considerable effort during initial setup. Lastly, the API-RPA approach offers optimal integration and efficiency, but it also necessitates significant setup and ongoing maintenance efforts.

Next, the time and monetary costs of each method are analyzed based on a concrete accounting task. The study finds that currently, the API-RPA is the most efficient method for large-scale accounting tasks.[3] On the other hand, the API and API-RPA approaches are the most expensive methods to apply under the current price rate of GPT-4 API.[4] Considering both cost reduction and increased efficiency, the user can select the UI-RPA approach, but it still depends on the nature of the task and the price rate. Table 2 offers users a useful basis for comparing and selecting the methods based on the current time and monetary cost.[5]

This study provides ready-to-use examples of Python code in Appendix 1 and UiPath bots in the supplementary documents to facilitate the application of LLMs by accounting researchers

[3] The UI-RPA approach is also efficient if the tasks are on a small scale, not restricted to the current prompt limitations set by LLMs.
[4] As time progresses, the limitations on the UI are expected to diminish, and the cost of the APIs is likely to decrease, with processes becoming more cost-effective, and tools evolving to be more comprehensive.
[5] The initial development costs of the API and RPA-related methods are not included in the time calculation. To help users save on these initial costs, this study provides examples for initiating the API code and RPA bots in Section VI.

and practitioners. The examples are not meant to satisfy all accounting tasks, but they can serve as a starting point for projects by providing key codes and bots designed for each method.

This study contributes to both literature and practice. By summarizing the mainstream usages of LLMs in the accounting domain, it provides a detailed discussion of each application method, along with its advantages, limitations, and associated time and monetary costs. This study responds to the challenges raised by accounting researchers and practitioners regarding the skill gaps among accountants, the lack of off-the-shelf solutions, the perceived complexity of integration with local systems, and the cost constraints related to initial implementation and subsequent maintenance (Dext 2024; Essex 2020; O'Leary 2023). The insights discussed in this study serve as an essential foundation for accounting researchers and practitioners in choosing among various methods to integrate LLMs into their current workflows.

This is the first study in the accounting literature to propose the application of LLMs through RPA integration and supply off-the-shelf codes, robots, and illustrations on critical accounting tasks that best fit each application method. The several accounting-oriented examples hold value for both researchers and practitioners, serving as a starting point for applying LLMs in this domain while recognizing the limitations of each method. Researchers and accountants can choose the optimal application approach based on the pros, cons, costs and accounting tasks illustrated in this study.

The remainder of the paper is organized as follows. Section II reviews the existing literature applying LLMs in accounting. Section III describes each method for using LLMs, and Section IV discusses the advantages and limitations of each approach. The cost analysis is outlined in Section V. Section VI provides off-the-shelf examples of three approaches. Subsequently, Section VII discusses and concludes the paper.

II. REVIEW OF LLM IMPLEMENTATIONS IN ACCOUNTING

LLMs have been deployed in the accounting literature for a variety of tasks, including auditing and risk analysis (Emett, Eulerich, Lipinski, Prien, and Wood 2023; Eulerich and Wood 2023; Föhr, Schreyer, Juppe, and Marten 2023; Gu et al. 2023), information extraction (Huang, Wang, and Yang 2023; Li et al. 2023), and sentiment analysis (de Kok 2023; Hansen and Kazinnik 2023; Lopez-Lira and Tang 2023). LLMs have also been used to develop new measurement methods for traditional accounting questions (Bernard, Blankespoor, de Kok, and Toynbee 2023; Jha, Qian, Weber, and Yang 2023). Common ways to leverage LLMs within current research include UI interaction and API. Eulerich and Wood (2023) use the ChatGPT UI to query an auditing relation task and retrieve the result from the output window. Following internal audit practices, they provide a series of audit-related example questions and input these into the ChatGPT UI for LLM understanding and response. Emett et al. (2023) interact with ChatGPT through the interface to examine whether LLMs can enhance the internal audit process. They design the prompts according to each internal audit step and type them into the ChatGPT interface for evaluation. To investigate whether LLMs can serve as tools for ensuring sustainability reports comply with the EU taxonomy, Föhr et al. (2023) leverage the ChatGPT interface as an auditor's assistant to examine the alignment of economic activities disclosed in sustainability reports that purport to use the EU taxonomy. Exploring LLM's ability to assist human auditors in various tasks, Gu et al. (2023) use three example auditing tasks—financial ratio analysis, post-implementation review, and journal entry testing—and utilize OpenAI's Playground to run the prompt protocol.[6]

[6] OpenAI's Playground is another type of UI that provides a web-based platform for users to experiment with and explore the capabilities of OpenAI's various LLM models.

Hu et al. (2023) also use the Playground to classify the sentiments of the MD&A sentences from the annual financial reports.

Several studies use the API as a tool to interact with LLMs for their tasks. Using the legacy version of GPT-3 technology, Hu et al. (2023) perform prompt engineering using the code-davinci-002, which can be accessed through the GPT-3 API.[7] Li et al. (2023) design a framework that leverages the GPT-4 API to batch-query contexts and performs prompt engineering to extract key financial data from unstructured sources, such as contracts, CSR reports, and annual reports. In their framework, users can supply the model name, API, and parameters to accommodate different source files and task types. Bai, Boyson, Cao, Liu, and Wan (2023) use the GPT-4 API and ask it to act as the executive team, answering questions raised by analysts during earnings conference calls. To limit the role and scope of the output, they assign different prompts to the "system," "assistant," and "user" within the API to better simulate the executive answering process. Eulerich et al. (2023) evaluate how well LLMs, specifically GPT-3.5 and GPT-4, can answer questions from accounting certification exams. They feed the exam questions to these models and analyze their ability to respond accurately in the context of professional accounting tests. The study observes that these models typically give better responses when given multiple sample questions and answers. To enhance accuracy, they include ten such examples in their prompts and adjust the API's "Temperature" setting to zero.[8] This adjustment reduces the models' creative responses, making them adhere more closely to the provided instructions and examples. Other researchers attempting to measure the business complexity of a company have incorporated the systemic

[7] code-davinci-002" is a specific version of the Davinci model series developed by OpenAI, designed for generating programming code and related text in a coherent and contextually relevant manner.

[8] The "Temperature" parameter of LLMs controls the degree of randomness in the model's responses, with a lower "Temperature" resulting in more predictable and conservative outputs, while a higher "Temperature" encourages more varied and creative responses.

design of their prompts into the GPT-4's API, including tags used by the companies and asking the model to classify them into the most appropriate standardized categories (Bernard et al. 2023). de Kok (2023) utilizes the API to fine-tune GPT-4 and classify non-answer types in earnings conference calls. Hansen and Kazinnik (2023) investigate whether GPT technology can interpret "fedspeak," which typically contains technical language. They use the text-davinci-003 version of GPT-3 and its API for this purpose.[9] The API's method of controlling the "Temperature" parameter aids in eliciting more consistent results. Lopez-Lira and Tang (2023) investigate whether LLMs could be leveraged to predict stock prices. To test the various models' abilities, they design their prompts with news headlines and used the API to batch prompt the GPT models.

III. LLM IMPLEMENTATION METHODS

The literature review indicates that UI and API are the primary methods for incorporating LLMs in accounting tasks. This study contributes to the accounting literature by integrating RPA, which, when combined with UI and API, offers increased efficiency. This section provides an overview of both UI and API methods in the field and further explores how RPA can be employed to improve interactions with LLMs.

UI

The most straightforward way to apply an LLM in the accounting domain is through the UI. A UI consists of the visual and interactive elements of a computer program or system that enable users to interact with and control its functionality. Typically, a UI includes graphical elements, menus, buttons, and other design components. For example, the ChatGPT UI, as shown

[9] text-davinci-003 was part of the GPT-3 model series by OpenAI and excels in generating coherent and contextually relevant text.

in Figure 1, provides users with a text box to type questions in, and clickable buttons to switch between models, or retrieve historical conversations. The typical process to use an LLM through its UI is for users to type prompts (questions) in the 'Message' box, click 'Send,' and receive the output (answers) from it.

API

An API is another common approach to using an LLM. Specifically, an API is a set of rules and protocols that enable different software applications to communicate with each other, facilitating the seamless exchange of data and functionality. It is an advanced method for utilizing LLMs, typically requiring a programming environment to implement. Essentially, users can leverage an API to establish connections between their local applications/systems and the LLM service, enabling data interaction between them. In the context of applying LLMs, users can programmatically send prompts to the LLM via the API and receive generated responses. An example API approach is shown in Appendix 1 and is further explained in Section VI.

UI-RPA

RPA (Huang and Vasarhelyi 2019; Moffitt et al. 2018; Zhang et al. 2023) offers great potential for integration with LLMs. RPA utilizes software robots to automate repetitive, rule-based tasks within accounting processes, thereby enhancing efficiency and minimizing human intervention. Research typically finds that RPA lacks adaptability, as it does not incorporate artificial intelligence into its analytical processes (Hong, Ly, and Lin 2023; Huang and Vasarhelyi 2019). However, RPA can be combined with LLMs to update most of the current working processes for researchers and practitioners. The UI method allows non-technical users to interact universally with LLMs. However, this method is only suitable for singular tasks that do not require repetition. UI-RPA expands the scope of the UI method by using RPA to automate the repetitive aspects of the tasks, enabling batch implementation of the UI method. Once all the prompts are developed, users can employ RPA to automate each query with the LLM, thus achieving a more efficient application of LLM through UI.

API-RPA

In addition to the UI approach, RPA can also be combined with LLM APIs to deepen integration across various applications. Real-world accounting tasks usually involve scenarios where users need to batch-process responses generated by LLM and then transmit those responses to another application, such as a company's ERP system. In such cases, the API method alone may be insufficient if the internal systems lack the necessary APIs to connect with the LLM platform.[10] The API-RPA method can address many such applications by bridging the local system to the LLM services through efficient processes. For instance, consider a scenario where an accounting firm needs to analyze and categorize numerous client emails for tax purposes. The firm uses an LLM to extract key information from the emails, such as dates, amounts, and tax-related terms. However, their in-house legacy ERP system lacks direct API integration with the LLM. By employing RPA, the firm can automate the transfer of the LLM-processed data into the ERP system. The RPA software mimics human interaction, logging into the ERP, and entering the data where necessary.

[10] The lack of API access is common in legacy systems or proprietary software tailored for specific business needs. These systems often predate modern API standards or are designed without external connectivity in mind, focusing instead on specific internal functionalities (MLTECHsoft 2023).

IV. PROS AND CONS

The inherent pros and cons of LLMs, such as their capacity for understanding and summarizing human language alongside issues like hallucination and inconsistency, are well-discussed in recent literature (Gu et al. 2023; Li, de Freitas, Lee, and Vasarhelyi 2024). This section specifically delves into the advantages and limitations of various LLM implementation methods, particularly when applied to accounting tasks or integrated into existing workflows (see Table 1).[11]

UI

The UI method is most accessible for accounting researchers and practitioners seeking to implement LLMs, as it simply requires an internet-connected computer. Currently, leading LLM products like ChatGPT, Gemini, Bing Chat, and Claude all offer UI access. Therefore, it requires the least skills and costs. The key benefits for researchers and practitioners in accounting when using LLMs through UIs include a no-code environment, ease of demonstration, and an interactive experience. Stakeholders in this domain may not have the technical expertise for a coding-based approach. The UI method eliminates the need for underlying programming, thus facilitating easier access to LLMs through intuitive graphical interfaces and dialog windows. For example, basic Question-and-Answer (Q&A) tasks with LLMs can be performed by entering a question into a dialog box and receiving a response in the reply window. Another significant implementation of the UI approach is ChatGPT Plugins, launched in March 2023. With user-friendly UI elements, users can effortlessly select from a dropdown list of plugins, choosing one or more that align with their specific tasks. This approach empowers accounting researchers and practitioners to leverage LLMs and their diverse built-in features, irrespective of their technical background.

[11] Hallucination refers to the phenomenon where the model generates information that is not grounded in the input data, often resulting in the creation of plausible but factually incorrect content.

The ease of understanding and the ability to demonstrate processes are additional benefits of the UI approach. It allows users to visually trace the process of sending textual prompts, images, or documents and receiving responses, similar to a real-world conversation. This visualization aids accountants in assessing the responses from the LLMs and quickly identifying which step or prompt needs adjustment if the desired outcome is not achieved. Furthermore, when presenting the use of LLMs in projects or processes to clients, accounting professionals can rely on UI for intuitive demonstrations. Compared to the API approach, this is invaluable for showcasing their work and enhancing the audience's comprehension.

The UI's interactive capabilities are especially advantageous for tasks where users need to ask follow-up questions based on prior responses from LLMs. A UI supports displaying replies in various formats, including markdown, which is mostly less intuitive through the API method. For example, as shown in Figure 2, it can organize responses into bullet points, tables, or highlight key parts of the answer in bold. This range of presentation styles greatly aids in user comprehension, making the UI method particularly suitable for accounting tasks requiring interactive functions.

However, several limitations exist when applying LLMs through UIs. These include low scalability, limited customization, and constraints related to input and output token limitations. Implementing LLMs via UI requires users to input prompts into a dialog box, wait for the generation of responses, and then manually copy or download these responses, potentially integrating them into existing workflows. This process is viable for singular tasks or small groups of tasks that do not necessitate significant repetitive processes, such as the verification of the figures in the annual report. However, for a large number of repetitive tasks, the UI approach becomes less appropriate. Since accounting tasks often involve repetition, this should be a consideration for researchers and practitioners when applying LLMs.

Additionally, most UIs offer only limited options for adjusting model parameters.[12] Changing parameter settings of an LLM can profoundly impact its outcomes. For example, the "Temperature" parameter is often used to control the creativity of LLM responses. For creative tasks, such as financial report generation, a high "Temperature" setting (e.g., 0.6) may be preferred to foster more creative responses. Conversely, for tasks requiring precise information extraction or verification from the provided context, such as compliance checking, a lower "Temperature" setting is advisable to minimize variations. Therefore, the limited parameter control in UI approaches should be carefully considered by users who want to tune LLMs to accomplish specific accounting tasks.

Another significant drawback currently encountered when utilizing UIs for LLMs is the token limitation on inputs and outputs.[13] As of October 2023, the input token limit for GPT-4's UI stands at 4,096 tokens.[14] This restriction significantly hampers the processing of long prompt inputs and response outputs, a situation that accounting tasks usually involve, such as in tasks involving extensive financial report audits, tax compliance cross-verification with rules, or in-depth report generation based on various financial analyses. Although ongoing developments in LLM technologies and computing power suggest that token limitations might become less constraining in future UI applications, this progress could be moderated by factors like abuse prevention and economic considerations in accounting tasks, leading to a more nuanced evolution of these limitations.

[12] The authors note that Bing Chat offers basic options to toggle between more 'Creative' and 'Precise' responses in its UI. However, there remain significant limitations in terms of granular customization when using LLMs through UIs. OpenAI's Playground is one of the few UI methods that provide parameter-setting options.

[13] 'Token' typically refers to a unit of text, such as a word or part of a word, that the model processes. Token limitation means that there is a maximum number of tokens the model can handle in a single input or output sequence.

[14] Comparatively, as of the time of this research, the GPT-4 Turbo API method can handle inputs with a context window of 128K.

Based on the pros and cons discussed, some critical accounting tasks that deserve the application of UI approach include client consultation, basic financial analysis and reporting, and basic compliance checking. The interactive and intuitive nature of UIs makes them ideal for conducting detailed client Q&A sessions, where clients may have various inquiries regarding their financial standing, tax obligations, or investment strategies. The UI's ability to display LLM responses in a visually engaging format facilitates a more dynamic consultation experience. Basic financial analysis and reporting can also leverage the benefits of UI-driven LLMs. Accountants often deal with the generation of financial statements, budget forecasts, and performance reports, which require not only the processing of numerical data but also the interpretation of financial trends and the creation of narrative reports. With the UI approach, the accountants can manually input raw financial data and receive narrative analyses. The UI allows accountants to easily modify prompts to refine outputs. Finally, compliance checking is another area where UI-based LLMs can make a significant impact. The change in financial regulations requires accountants to continuously update their knowledge and ensure that all financial transactions and reports comply with relevant laws and standards. UIs can simplify this task by providing a platform for accountants to quickly check the compliance of financial documents against current regulations. By inputting relevant sections of financial reports into the LLM via the UI, accountants can build customized LLM (such as GPTs) that give feedbacks on potential compliance issues, suggestions for rectification, and references to the relevant regulatory texts.

API

Implementing LLMs through APIs offers several advantages over other methods for accounting tasks. First, the API approach can integrate seamlessly into existing workflows without significantly altering their structure. APIs enable LLMs to function as modular tools that can be easily added, modified, or removed within the overall working pipeline. Consider a process designed to classify objects into different asset types based on their textual descriptions. Without LLMs, this might be performed through human reading, keyword matching, or other data analytics techniques. If accountants decide to incorporate LLMs, they can establish API connections between the process and the LLMs. This allows for the easy transmission of input (textual descriptions of assets) to the LLMs and the subsequent transfer of output (asset type classification) back to the system. Additionally, accountants can effortlessly disconnect the API, replace the LLM method with a traditional manual approach, or switch to a different LLM by changing APIs. Thus, applying LLMs through an API is a flexible method that integrates easily and does not necessitate re-engineering the existing workflow, which is suitable for optimizing most of the established accounting tasks.

Second, the API approach enables scalable processing, which is especially suitable for accounting tasks that usually involve repetitive processes. Unlike the UI method, which requires manual input and retrieval of prompts and responses, the API method can automate the LLMs' implementation process. Once the API connection is established and deployed within the existing environment, it can automatically perform each task. This makes the API method particularly suitable for handling bulk tasks with LLMs.

Finally, the API approach allows for parameter setting and customization, which is typically not available in the UI method. The UI method is designed for general-purpose tasks, which may not need parameter adjustment to satisfy specific needs. In contrast, the API method offers users customizable options that can enhance the relevance of responses to specific tasks. This allows the researchers and accountants to design tailored accounting processes. For example, the GPT-4 API allows users to set the model type, "Temperature," maximum token count, role context and task context. These parameters enable users to tune the appropriate model for different accounting tasks, adjust the creativity level of the response, and determine the desired output length. The GPT-4 API also enables setting roles such as "User," "System," and "Assistant" in prompts, providing structured instructions for complex prompts and facilitating prompt engineering.

However, the implementation of the API approach is not without its challenges. Initially, setting up the API connection and maintaining it, particularly during updates or workflow changes, requires skilled personnel. If the connection is not established correctly or the parameters are not fine-tuned for the task's specific nature, the API method's performance may be inferior to that of the UI approach. To aid in deploying LLMs through APIs, Section VI of this research presents a real-world case of integrating the GPT-4 API into an accounting project. This section offers example code and insights on how to effectively leverage the API approach.

Another limitation is the potential incompatibility of the existing workflow with API connections. The researcher and accountant’s existing process may be based on legacy technology that does not support API integrations or a secure system that prohibits external communication. For instance, a financial manager wishing to use LLMs to detect cybersecurity risks in server logs may face challenges if the server does not support automatic API access (Centric 2023; MLTECHsoft 2023). To analyze the logs, they would have to manually download them and then feed them into the LLMs. This incompatibility creates barriers to seamlessly integrating LLMs into an existing workflow. Additionally, API integration could raise privacy concerns, especially in sectors like accounting, where data often contains sensitive information. Linking the workflow to an external API might lead to data breaches if the connection is compromised. Lastly, using an API could make the processes more opaque to users. While LLMs are already perceived as "black boxes," an API approach can add another layer of obscurity. This can be problematic when errors occur, as it becomes difficult for users to determine whether the issue lies with the system or the LLM. This ambiguity could result in users spending a significant amount of time trying to trace the source of the problem.

With these pros and cons discussed, some accounting tasks that benefit most from the API approach include basic financial data extraction, transaction classification and verification, and basic fraud detection. Information extraction is critical for accurate record-keeping and financial analysis but is often labor-intensive and prone to human error when done manually. LLMs can analyze text to identify and extract data points such as transaction amounts, dates, and parties involved. The application of LLMs through API revolutionizes this process by extracting key financial details with high precision and efficiency. Transaction classification and verification involves categorizing financial transactions according to predefined accounting standards and verifying their accuracy and legitimacy. Applying LLMs through API can help interpret large amounts of transaction descriptions, classify them into the appropriate categories, and flag transactions that deviate from expected patterns for further review. It can also be deployed continuously within the system. Finally, fraud detection also benefits from the integration of LLMs through APIs. Through API integration, LLMs' capabilities in identifying patterns from text can be incorporated directly into the continuous monitoring systems, allowing for efficient fraud detection.

UI-RPA

The accounting literature typically only utilizes LLMs through UI and API. The combination of UI and RPA in applying LLMs offers several benefits that the UI method alone does not. This combination can achieve batch querying with LLMs while maintaining ease of integration with existing workflows (e.g., local payroll systems). The API method itself might encounter limitations, such as difficulties in integrating into the existing workflow due to legacy systems that do not support automatic data import or export. However, the RPA method can bypass this limitation. Specifically, RPA can mimic human interactions with existing systems by recognizing graphical elements and transmitting data from one application or system to another. Thus, even if the existing system does not support underlying programming-level interaction, RPA can still connect it with the LLMs' UI to enable automatic querying.

Another advantage is that the combination of RPA and UI can enhance the accuracy and consistency of query results while also maintaining a high level of verifiability. The pure UI method, which requires human input and output of accounting data from the LLMs, can lead to clerical errors. While the API solution can resolve issues like accuracy and consistency, the relatively low transparency of the process tends to make the results questionable to users if an error occurs, further increasing the cost of verifiability. The UI-RPA approach can enhance accuracy and consistency since machines excel at processing repetitive work. At the same time, the entire querying process can be visually recorded and timestamped for future reference. In the event of a systematic error, users can quickly trace back to when the query was performed and examine the error from there. In terms of accounting tasks, this feature is especially important since it enhances the accuracy of the results and adds another layer to data assurance.

Additionally, the UI-RPA method can also be combined with manual efforts. Situations exist where some complicated accounting tasks require human judgment. In these cases, the process could be re-engineered to be a UI with an Attended RPA workflow, allowing the automatic process to pause and wait for human instruction (Zhang, Thomas, and Vasarhelyi 2022). After receiving input from a human, the bot will continue to finish its remaining steps and proceed to the next task.

Finally, this combined method caters to a broader range of users with diverse interests and needs, particularly in fields like accounting, where tasks may involve various types of information, including documents, tables, and figures. The wide array of functions provided by RPA, such as table extraction, computer vision, and document extraction features in UiPath, complements the current functional limitations of LLMs. For example, LLMs struggle to directly read and summarize online tables spanning multiple pages. With RPA assistance, the process can be designed to initially use the Table Extraction function from the RPA to read the entire online table within the website, then feed this data into the LLM for high-level understanding and summarization. With the continuous development of automatic packages in RPA software, this combined approach could become increasingly powerful.

The most challenging part of applying LLMs through UI-RPA is the initial setup process for accounting researchers and practitioners, which requires users to have a clear understanding of the process, as well as the ability to program the bot to handle different elements between various applications and the LLMs' UI. Meanwhile, the maintenance process will also require skilled personnel who can update the bots based on changes in the UI and the working process. This study provides an example RPA bot in Section VI that could serve as a starting point for users who wish to modify and implement it in their existing processes.

Another challenge is that not every system integrates well with RPA, especially those requiring CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart), a program designed to differentiate between humans and machines. If the accounting tasks involve the integration with websites that employ CAPTCHA, such as online banking systems for financial reconciliations or using government portals for tax filing, it becomes difficult for RPA to pass since these tests typically require human intelligence to solve (Karatas 2023).

Finally, introducing new software might create additional privacy and cybersecurity issues, especially for accounting data. Although RPA is a mature technology used by many companies worldwide, it still introduces new risks if the process is compromised without authorization. Additional evaluation should be conducted before its use.

Based on the pros and cons, UI-RPA is suitable for accounting tasks such as expense management and auditing, asset management and depreciation scheduling, as well as budgeting and forecasting that require interaction between LLM and local systems. Expense management and auditing are vital yet cumbersome tasks. The application of UI-RPA can streamline this process by automating the extraction and categorization of expense data from manually supplied receipts, credit card statements, and other documents, and allowing it to navigate through various expense management software to input, categorize, and reconcile expenses. Asset management and the scheduling of depreciation are another areas ripe for enhancement. By employing UI-RPA, organizations can automate the tracking, categorization, and depreciation calculation of assets. It can seamlessly interact with databases and spreadsheets to update and maintain accurate records of asset lifecycles, ensuring compliance with accounting standards. Finally, budgeting and forecasting can be enhanced through automated data aggregation from various sources, including sales figures, operational costs, and market trends using RPA. By further integrating LLM capabilities, organizations can analyze this data to generate interactive and dynamic budget forecasts.

API-RPA

In addition to the advantages related to RPA mentioned above, integrating LLMs through an API-RPA offers the benefit of achieving the most in-depth integration with existing workflows. This maximizes the efficiency of implementing LLMs in the accounting domain. While the API method alone enables the instant application of LLMs without manually typing each prompt into the UI, it has the drawback that the generated data might be difficult to integrate into existing processes, such as internal ERP systems. The aforementioned UI-RPA approach could address this issue by leveraging RPA's ability to transfer data between systems. However, it suffers from the inherent efficiency limitations associated with the UI approach. The API-RPA method, however, emerges as the optimal solution for this type of accounting task. Particularly, RPA enables the process to robotically collect raw data from existing systems by recognizing graphical-level elements and inputting them into the LLM via the API to achieve efficient queries. After the LLM's processing, the bot can automatically retrieve the output and transmit it back to the internal systems, thus achieving the most in-depth integrations.

All of the aforementioned limitations related to RPA and API, including the complexity of initial setup and maintenance, are also present in this API-RPA approach. To assist with these challenges, this research provides a practical example of combining API and RPA in Section VI. This example is aimed at helping the application of this combined method into existing processes.

In general, some accounting tasks that are most suitable for applying API-RPA include systematic financial data extraction and analysis, regulatory compliance and reporting, and trail analysis and fraud detection. Financial statements, invoices, and other accounting documents often contain unstructured data that can be challenging to process manually (Li et al. 2023). By leveraging RPA's ability to collect raw data from various sources and feed it into the LLM via an

API, the process can be automated and scaled to handle large volumes of data. The extracted data could be further transferred back to the internal system through RPA. For compliance checking, accounting professionals usually navigate a complex web of regulations and compliance requirements, which often involves interpreting dense legal documents and generating reports. The API-RPA approach allows for the efficient collection of data from various sources, passing it to the LLM for analysis, and automating the generation and publishing of compliance reports. Lastly, for trail analysis and fraud detection, LLMs can be trained to analyze audit trails, transaction logs, and other relevant data. The API-RPA approach enables the efficient collection and integration of data from various systems, passing it to the LLM for analysis, transferring the data back to the system, and automating the generation of audit reports or fraud alerts.

V. COST ANALYSIS

Timing and Labor Aspects

This section discusses the varying time and labor costs associated with each LLM implementation method. The pure UI method is the slowest and most labor-intensive, as it requires human effort to query the LLMs and retrieve the results. To be specific, the time spent on each task depends on its complexity, which might require multiple steps to interact with LLMs. For example, a task involving audit analytics, which usually includes a series of interactive questions and answers with an LLM, would be more time-consuming than a simpler task, such as checking a document for specific information. Additionally, the time and labor costs of the UI method significantly increase for batch tasks. In contrast, the UI-RPA method could be much faster and labor-efficient, with the most limiting factor being the UI's prompt limitations.[15] However, these concerns could be mitigated as LLM providers have progressively increased their capabilities since launch, a trend expected to continue as costs decrease and technology matures.

The API-related methods are more efficient and automated, performing queries through an underlying coding environment that eliminates manual processes. Therefore, the time cost of the API approach relates only to the limitations of the LLM service provider if considering only pure querying time without subsequent integration into the working process.[16] However, when considering the time required to integrate LLM responses into the existing workflow, the pure API method might not be the most efficient. This is because transferring generated data into the workflow still demands time and labor. In such cases, the API-RPA method proves more efficient. For example, in the context of using LLMs to retrieve data from an ERP system, querying the LLM via the API only automates the process of performing the query. This method does not assist in the retrieval of data from, or its input back into, the ERP system, thereby limiting overall efficiency. Since RPA can facilitate the transfer of data between different applications and processes, it has the potential to save a significant amount of time.

For a clearer understanding, this study compares the time cost of various methods in a specific accounting project involving LLM. The task involves using GPT-4 to extract key indicators from 500 unstructured financial statements in text format and then inputting these indicators into the company's internal system, which lacks an API connection. With the UI method, users need to manually enter each statement and its prompt into GPT-4, then copy and paste the

[15] For instance, the current query limit for GPT-4 UI is 50 prompts per three hours, amounting to only 400 prompts per 24 hours. Although GPT-3.5 UI claims no limit on queries, the authors' tests indicate a soft limitation of 700-1,000 prompts per 24 hours.

[16] To provide an illustration, at the time of this research, OpenAI's API allows 500 requests per minute and 10,000 requests per day for tier one users. This rate would be mostly sufficient for normal users in the accounting research and practice domains.

results into the internal system. The API method automates querying GPT-4 but still requires manual entry of the results into the internal system. By contrast, the UI-RPA method automates the entire process, subject only to GPT-4 UI's request speed and prompt limitations. Lastly, the API-RPA method also automates the entire process, but without the constraints of GPT-4 UI's query limitations.

Table 2 columns (1) – (3) display the estimated time and labor effort required to complete this task using each method. The total time for the UI approach to extract 500 statements is 208 minutes, assuming no breaks for the user.[17] However, due to GPT-4's current prompt limitation of 50 prompts every three hours, the total time to complete all 500 tasks increases to 1,800 minutes.[18] In contrast, the API method only takes 141 minutes, as it is unaffected by prompt limitations.[19] The UI-RPA significantly reduces time, taking only 67 minutes, due to automated data input into the internal system, assuming prompt limitations are disregarded.[20] However, owing to the same prompt restrictions on UI requests, the final time cost also amounts to 1,800 minutes. Finally, the API-RPA approach is the most efficient, requiring only 41 minutes and is not subject to prompt limitations.[21] Regarding labor effort, both the UI and API methods necessitate manual input of results into the internal system. Conversely, the two RPA-enabled approaches eliminate labor effort since the bot automates the entry of the LLM results into the system.

[17] Considering 10 seconds to search the statement for values and 15 seconds to fill the value into the system: (10s + 15s) * 500 / 60s = 208 minutes. The time for searching and filling the values is estimated by the author performing a small sample of the extraction tasks.

[18] Based on the GPT-4 UI’s limit of 50 prompts every three hours as of October 2023, completing 500 prompts would require (500 / 50) * 3h * 60m = 1,800 minutes.

[19] Considering 2 seconds to search the statement for values and 15 seconds to fill the value into the system: (2s + 15s) * 500 / 60s = 141 minutes.

[20] Considering 5 seconds to search the statement for values and 3 seconds to fill the value into the system: (5s + 3s) * 500 / 60s = 67 minutes.

[21] Considering 2 seconds to search the statement for values and 3 seconds to fill the value into the system: (2s + 3s) * 500 / 60s = 41 minutes.

Monetary Considerations

The monetary cost of utilizing LLMs varies, depending on whether the model is free or fee-based. For fee-based models, expenses may include subscription fees or charges for API tokens. Localized or open-source models are typically free, provided the necessary infrastructure is in place. There are free LLM providers like GPT-3.5, Claude 2, and Bing Chat, but they often have limitations regarding response speed and the number of prompts available. When a model is offered by a third-party service and is not free, costs are generally linked to membership fees and token pricing.[22] Table 2, column (4), employs OpenAI's fee structure as of October 2023 to compare the costs of different methods for the data extraction tasks from the 500 statements. Assuming each statement is, on average, 1,024 tokens long, including text, numbers, and punctuation, the total cost for processing all 500 statements via the API method is approximately $15.36.[23] By contrast, using the UI method for the same task would incur a cost of about $0.83, which is only 5.4% of the API method's cost.[24]

Balanced Choice

The time and monetary costs associated with different LLM implementation methods can vary significantly, influenced by current pricing rates and specific strategies used. It is crucial for users to balance both aspects when selecting a method. The examples provided earlier aim to elucidate the cost-benefit considerations involved. The UI method, for instance, typically incurs low monetary but high time costs due to the LLM's prompt limitation. To address this, users might

[22] For example, as of this research, OpenAI's GPT-4 UI version costs $20 per month. With a limit of 50 prompts every three hours, the cost per prompt works out to approximately $0.0017. In contrast, OpenAI's API approach charges about $0.03 per 1,000 input tokens and $0.06 per 1,000 output tokens.

[23] 1.024 * $0.03 * 500 = $15.36. The token length of the output, consisting of only a few digits, is too small to significantly affect the estimated price when applying the API approach.

[24] Based on the monthly cost of $20 (assuming 30 days per month), the price per prompt would be $20 / (50 * (30d * 24h / 3h)) = $0.0017. Thus, extracting 500 statements would result in a cost of $0.0017 * 500 = $0.83.

employ multiple UI methods simultaneously. For example, using four accounts for the UI approach would raise the monetary cost in the previously mentioned scenario to $3.32. However, this strategy would substantially decrease the time required to complete the task, reducing it to just 363 minutes.[25] While these calculations are based on current pricing, researchers and accountants should assess both time and monetary costs, adapting their approach to meet the specific requirements of their tasks and budget constraints. These trade-offs and considerations are likely to evolve with advancements in LLM technology and other factors that will inevitably become more cost-effective.

VI. REAL-WORLD EXAMPLES

In light of the initial setup challenges associated with the methods discussed in Section IV, this section provides three examples for utilizing OpenAI's GPT-4 through the API, UI-RPA, and API-RPA approaches. These examples are designed to use GPT-4 to extract key indicators from 500 unstructured financial statements in text format and then input these indicators into the company's internal system, which lacks an API connection, aligning with the tasks detailed in Section V. Extracting structured financial data from unstructured sources is a critical task for researchers and practitioners in the accounting domain, and is also the foundation for many accounting tasks, such as risk analysis, budget forecasting, and compliance checking. Structured data facilitates empirical research and informs investments and regulatory activities. With the continuous growth of digitalization and the vast amount of unstructured textual data being generated, the ability to extract key financial information from such sources becomes paramount.

[25] Implementing the same number of tasks across four accounts would quadruple the cost to $0.83 * 4 = $3.32. Splitting the 500 statements among these four accounts results in 125 statements per account. Therefore, processing them would require two three-hour cycles, and the remaining 25 statements would take an additional 6h * 60m + (5s + 3s) * 25 / 60s = 363 minutes.

While this example focuses on financial statements, the methods can be applied to various types of unstructured text, such as email communications, event logs, online posts, and news releases. The corresponding code snippets and RPA bots necessary for implementing these approaches are provided. While it is important to note that these example codes and RPA bots may not align with all accounting tasks, they serve as valuable starting points for accounting researchers and practitioners, eliminating the need to start from scratch.

The code for connecting to the GPT-4 API is provided in Appendix 1. It consists of three main components: importing the necessary libraries, defining functions, and performing the extraction tasks. The provided function and the parameters allow users to customize key variables to suit various tasks, such as the "model" and "Temperature." Furthermore, the function can be executed in batches with the defined loop function. Users can replace the API Key with their own credentials and run the code either in the local Python environment or on Google Colab.

An illustration of the UI-RPA process is provided in Figure 3, as well as the bot files in the supplementary document. This example utilizes UiPath to design an automated bot that can connect the LLM UI query process with the local system. To implement this RPA solution, we use the Google Chrome web browser. Before running the RPA process, users need to log into their ChatGPT account in the Chrome web browser and select the specific model they wish to utilize. After this setup, users can close the browser and proceed to execute the RPA workflow.

Figure 3 offers a concise overview of the example workflow of the UI-RPA. It utilizes a local Excel file to store all statements and prompts.[26] Each statement and its prompt is inputted into the ChatGPT UI using RPA. Upon receiving the responses from GhatGPT, the RPA captures them and saves them back into the Excel file. The process is also designed to handle exceptions,

[26] This example uses a local Excel file to represent the user's internal system, which cannot be connected to the LLM through an API. Users can replace the Excel file with their own internal systems, such as an ERP system.

such as performing checks to avoid duplicate processing and recording any relevant error messages without disrupting the overall workflow. Additionally, if the GPT-4 model reaches its prompt limit, the bot is programmed to automatically pause for 30 seconds and then resume once the limit has been reset.[27]

The example of the API-RPA approach is demonstrated in Figure 4 and provided in the supplementary material. Compared to the UI-RPA method, this method is considerably simpler. This is because the bot does not need to recognize graphical elements on the GPT-4 UI and is not required to handle the many exceptions and prompt limits that typically arise when using the UI approach. Before running the RPA process, the bot will prompt the user to establish a connection between UiPath and OpenAI by inputting the API credentials in the web browser.[28] Once this connection is established, the user can execute the RPA with customized LLM settings, such as selecting models, setting the "Temperature," and limiting token output. Although the example RPA provided here is relatively straightforward, it highlights the fundamental steps involved in connecting UiPath with the LLM API. This connection enables users to develop future activities that can be integrated into the RPA workflow. For instance, users can input the results into an ERP system by incorporating the "Use Application" activities within UiPath and specifying the location within the application for data entry.

[27] The bot will scan the UI for keywords that indicate the reset of the limit. Typically, the three-hour countdown starts from the beginning of processing the first statement.

[28] Users should first install the package named "OpenAI" version 6.1.0 using the "Manage Packages" function in the menu ribbon of UiPath. Afterward, they should click on "Add new connection" within the "Generate Chat Completion" activity to input their OpenAI API key.

VII. DISCUSSION AND CONCLUSION

The application of LLMs in accounting is bringing significant changes in both academia and practice. Without a clear understanding of the advantages, limitations, and costs associated with each method of applying LLMs in current working processes, users in the accounting domain may struggle to identify the approach that best suits their tasks, processes, and budgets. This research addresses this by summarizing the mainstream methods of implementing LLMs in accounting literature, namely through UI and API. It further proposes two novel methods for their implementation via RPA integration. Comprehensive discussions are provided on the advantages and limitations of each approach, as well as the time and monetary costs associated with them. To enhance practical implications, this study also provides three off-the-shelf examples that researchers and practitioners can utilize.

This study contributes to the literature by reviewing the current methods of applying LLMs in the accounting domain. It elaborates on the advantages, limitations, and cost considerations of each implementation method, shedding light on the increasing demand for applying LLMs in the accounting field and responding to the challenges raised by researchers and practitioners (Dext 2024; Essex 2020; O'Leary 2023). The provided ready-to-use examples offer practical contributions to both researchers and practitioners.

However, this research is not without its limitations. The discussions of each method are based on the current level of technological development and cost. Some limitations might be overcome in the future with the adoption of new models. Additionally, the costs associated with each approach might change based on computing costs and market demand. Further research is needed in this literature to discuss additional application methods and cost-benefit models based on future developments of LLMs.

In light of the great potential of incorporating LLM into complex accounting practices, future research could further explore the integration of LLMs with RPA to enhance data transfer and processing efficiencies based on the method discussed in this study. By leveraging RPA's capability to automate the movement of data across various systems, researchers can develop innovative applications that streamline accounting workflows, reduce manual errors, and improve overall productivity. Some real-data examples to illustrate the impact of API-RPA integration could provide concrete evidence of these technologies' benefits. For instance, leveraging LLMs alongside RPA to automate complex financial reporting processes, such as extracting data from invoices or consolidating sales data for real-time reporting, could highlight notable improvements in accuracy, speed, and reliability. Future research could also bridge the gap between theoretical potential and practical application, demonstrating that the pros discussed in the study could translate into tangible benefits for the accounting profession.

REFERENCES

Alberts, I. L., L. Mercolli, T. Pyka, G. Prenosil, K. Shi, A. Rominger, and A. Afshar-Oromieh. 2023. Large language models (LLM) and ChatGPT: What will the impact on nuclear medicine be? *European Journal of Nuclear Medicine and Molecular Imaging* 50 (6): 1549–1552.

Bai, J. (Jianqiu), N. M. Boyson, Y. Cao, M. Liu, and C. Wan. 2023. Executives vs. Chatbots: Unmasking insights through human-ai differences in earnings conference Q&A. SSRN Scholarly Paper. Rochester, NY.

Bernard, D., E. Blankespoor, T. de Kok, and S. Toynbee. 2023. Confused readers: A modular measure of business complexity. SSRN Scholarly Paper. Rochester, NY.

Centric. 2023. Visual data modeling. http://www.centricdxb.com/insights/pim/the-challenges-of-ensuring-compatibility-and-integration-with-proprietary-software.

Dext. 2024. Challenges and Solutions in Adopting AI for Small and Medium Accounting Firms - Dext Blog | Dext USA. https://dext.com/us/resources/accounting-news/challenges-and-solutions-in-adopting-ai-for-small-and-medium-accounting-firms.

Emett, S. A., M. Eulerich, E. Lipinski, N. Prien, and D. A. Wood. 2023. Leveraging ChatGPT for enhancing the internal audit process – a real-world example from a large multinational company. SSRN Scholarly Paper. Rochester, NY.

Essex, D. 2020. 7 hurdles to using AI in accounting and finance | TechTarget. *ERP*. https://www.techtarget.com/searcherp/feature/7-hurdles-to-using-AI-in-accounting-and-finance.

Eulerich, M., A. Sanatizadeh, H. Vakilzadeh, and D. A. Wood. 2023. Can artificial intelligence pass accounting certification exams? ChatGPT: CPA, CMA, CIA, and EA? SSRN Scholarly Paper. Rochester, NY.

Eulerich, M., and D. A. Wood. 2023. A demonstration of how ChatGPT can be used in the internal auditing process. SSRN Scholarly Paper. Rochester, NY.

Föhr, T. L., M. Schreyer, T. A. Juppe, and K.-U. Marten. 2023. Assuring sustainable futures: Auditing sustainability reports using AI foundation models. SSRN Scholarly Paper. Rochester, NY.

Gu, H., M. Schreyer, K. Moffitt, and M. A. Vasarhelyi. 2023. Artificial intelligence co-piloted auditing. SSRN Scholarly Paper. Rochester, NY.

Hansen, A. L., and S. Kazinnik. 2023. Can ChatGPT decipher Fedspeak? SSRN Scholarly Paper. Rochester, NY.

Hong, B. (Yue), M. Ly, and H. Lin. 2023. Robotic process automation risk management: Points to consider. *Journal of Emerging Technologies in Accounting* 20 (1): 125–145.

Hu, N., P. Liang, and X. 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.

Huang, A. H., H. Wang, and Y. Yang. 2023. FinBert: A large language model for extracting information from financial text. *Contemporary Accounting Research* 40 (2): 806–841.

Huang, F., and M. A. Vasarhelyi. 2019. Applying robotic process automation (RPA) in auditing: A framework. *International Journal of Accounting Information Systems* 35: 100433.

Jha, M., J. Qian, M. Weber, and B. Yang. 2023. ChatGPT and corporate policies. SSRN Scholarly Paper. Rochester, NY.

Karatas, G. 2023. The ultimate guide to avoiding CAPTCHAs in web scraping. https://research.aimultiple.com/how-to-bypass-captcha/.

de Kok, T. 2023. Generative LLMs and textual analysis in accounting: (Chat)GPT as research assistant? SSRN Scholarly Paper. Rochester, NY.

Li, H., M. M. de Freitas, H. Lee, and M. Vasarhelyi. 2024. Enhancing Continuous Auditing with Large Language Models: A Framework for Cross-Verification Using Exogenous Textual Data. *Available at SSRN 4692960*.

Li, H., H. (Harry) Gao, C. Wu, and M. A. Vasarhelyi. 2023. Extracting financial data from unstructured sources: Leveraging large language models. SSRN Scholarly Paper. Rochester, NY.

Lopez-Lira, A., and Y. Tang. 2023. Can ChatGPT forecast stock price movements? Return predictability and large language models. SSRN Scholarly Paper. Rochester, NY.

MLTECHsoft. 2023. Addressing integration and compatibility Issues in legacy systems. https://mltechsoft.com/insights/blog/adressing-integration-compatibility-issues-in-legacy-systems.

Moffitt, K. C., A. M. Rozario, and M. A. Vasarhelyi. 2018. Robotic Process Automation for Auditing. *Journal of Emerging Technologies in Accounting* 15 (1): 1–10.

O'Leary, D. E. 2023. Enterprise large language models: Knowledge characteristics, risks, and organizational activities. *Intelligent Systems in Accounting, Finance and Management* 30 (3): 113–119.

Pal, S., M. Bhattacharya, S.-S. Lee, and C. Chakraborty. 2023. A domain-specific next-generation large language model (LLM) or ChatGPT is required for biomedical engineering and research. *Annals of Biomedical Engineering*.

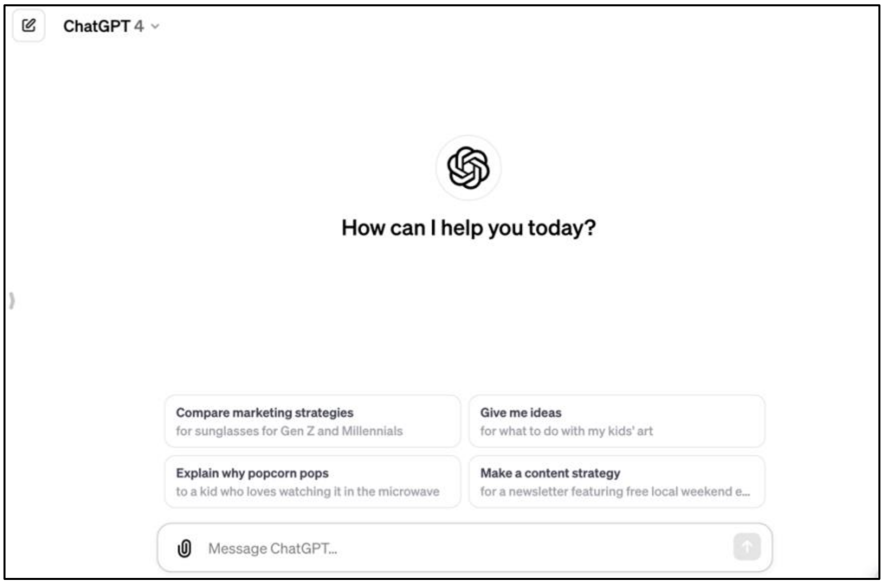
Sallam, M. 2023. ChatGPT utility in healthcare education, research, and practice: Systematic review on the promising perspectives and valid concerns. *Healthcare* 11 (6): 887.

Vasarhelyi, M. A., K. C. Moffitt, T. Stewart, and D. Sunderland. 2023. Large language models: An emerging technology in accounting. *Journal of Emerging Technologies in Accounting* 20 (2): 1–10.

Zhang, C. (Abigail), H. Issa, A. Rozario, and J. S. Soegaard. 2023. Robotic process automation (RPA) implementation case studies in accounting: A beginning to end perspective. *Accounting Horizons* 37 (1): 193–217.

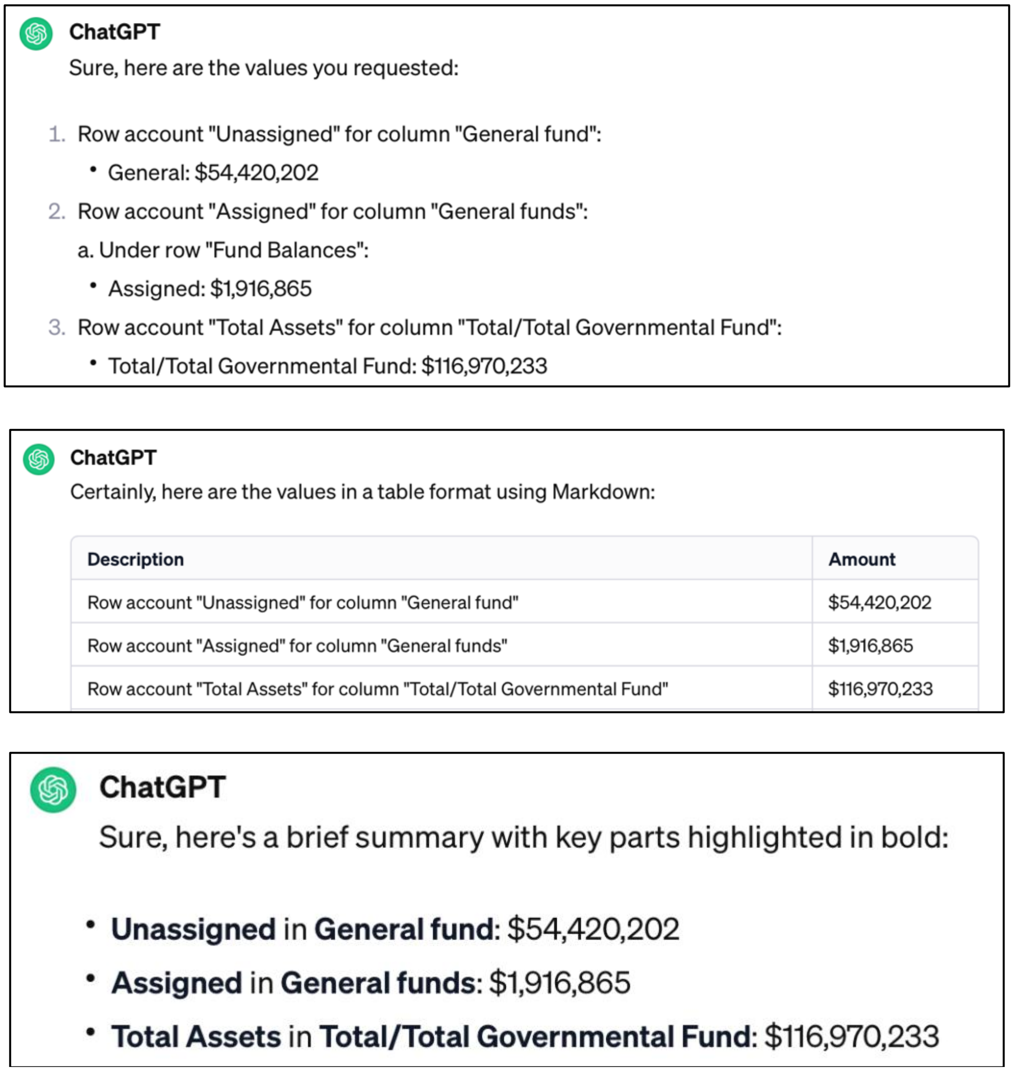
Zhang, C. (Abigail), C. Thomas, and M. A. Vasarhelyi. 2022. Attended Process Automation in Audit: A Framework and A Demonstration. *Journal of Information Systems* 36 (2): 101–124.

Figure 1 ChatGPT 4 UI



Notes: This figure shows the ChatGPT 4's user interface.

Figure 2 Examples of UI Output Formats



ChatGPT

Sure, here are the values you requested:

1. Row account "Unassigned" for column "General fund":
   - General: $54,420,202
2. Row account "Assigned" for column "General funds":

   a. Under row "Fund Balances":
   - Assigned: $1,916,865
3. Row account "Total Assets" for column "Total/Total Governmental Fund":
   - Total/Total Governmental Fund: $116,970,233

ChatGPT

Certainly, here are the values in a table format using Markdown:

Description	Amount
Row account "Unassigned" for column "General fund"	$54,420,202
Row account "Assigned" for column "General funds"	$1,916,865
Row account "Total Assets" for column "Total/Total Governmental Fund"	$116,970,233

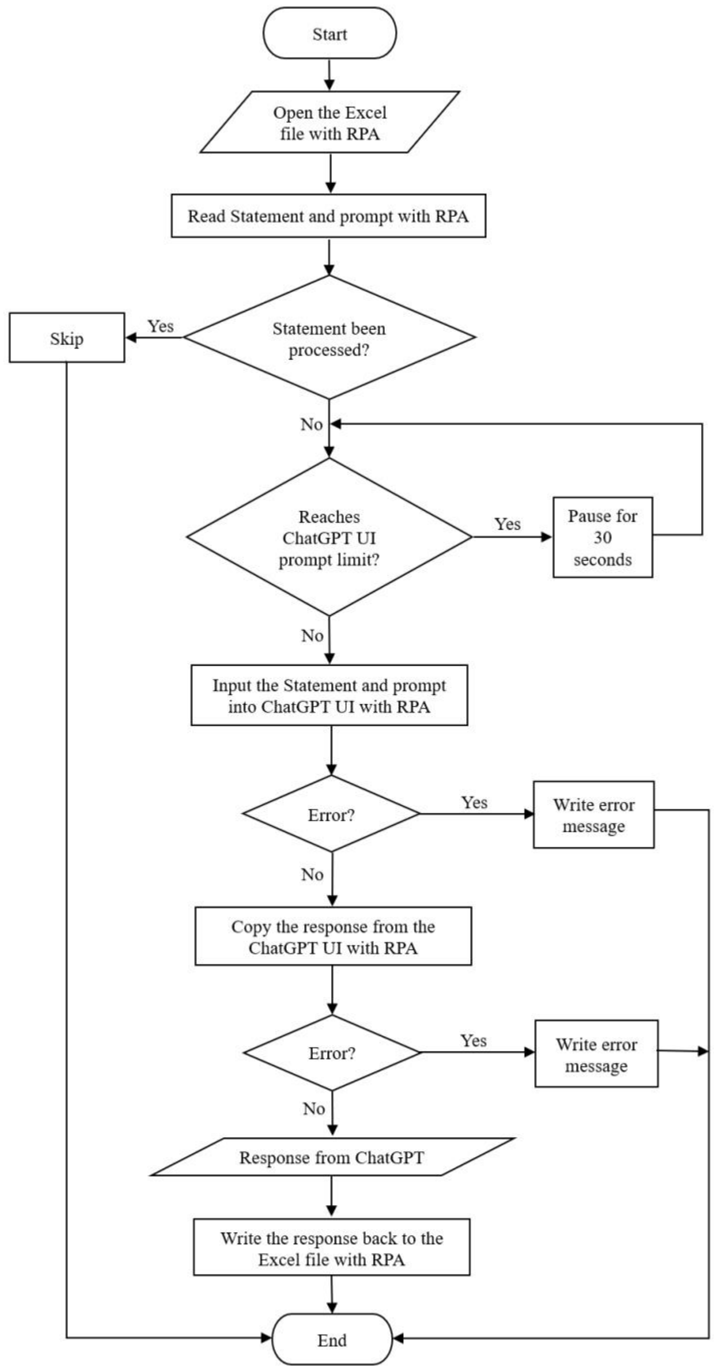
ChatGPT

Sure, here's a brief summary with key parts highlighted in bold:

- **Unassigned** in **General fund**: $54,420,202
- **Assigned** in **General funds**: $1,916,865
- **Total Assets** in **Total/Total Governmental Fund**: $116,970,233

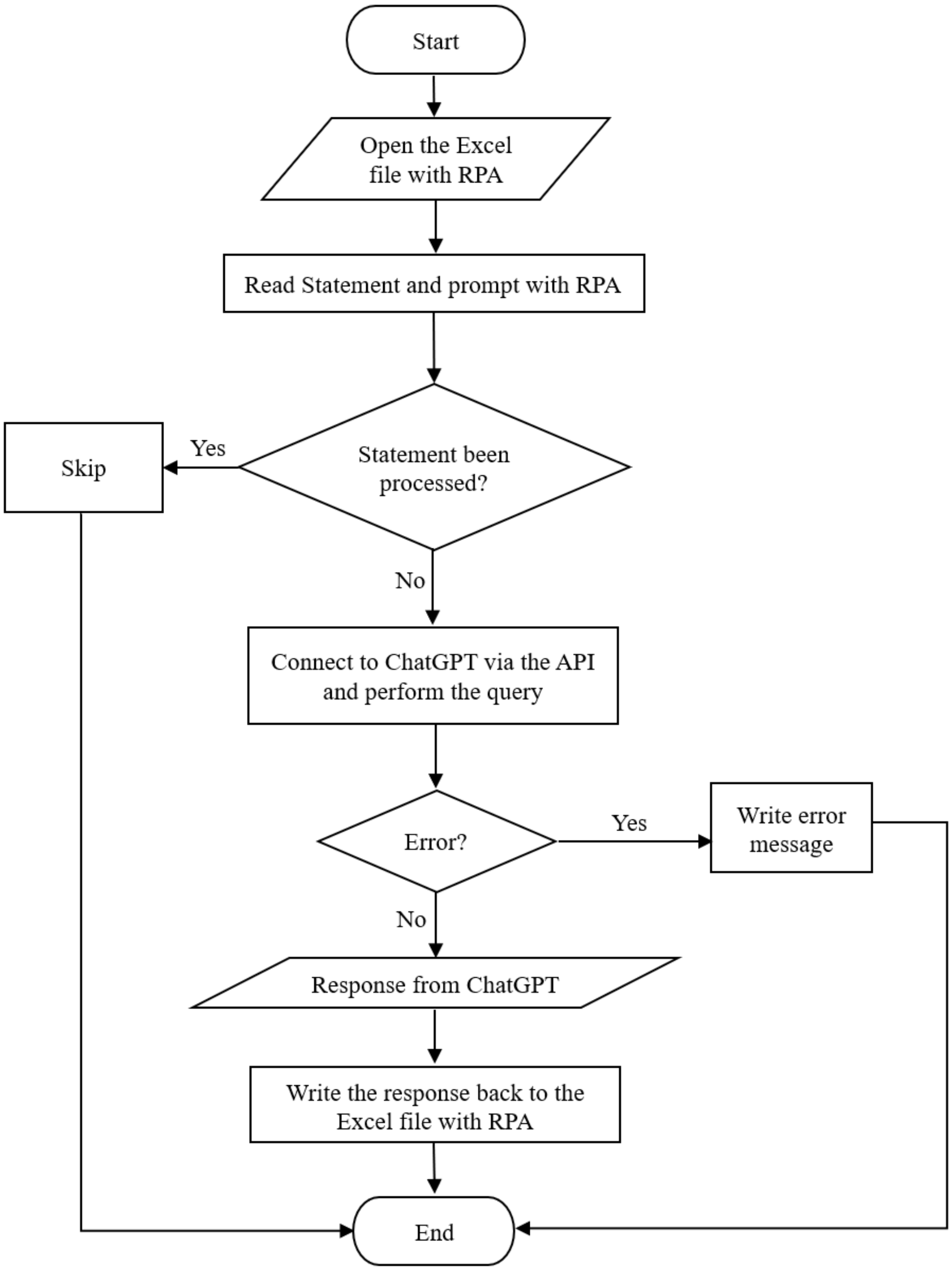
Notes: This figure shows the various types of UI output from the ChatGPT.

Figure 3 Example Workflow of UI-RPA Approach



Notes: This figure illustrates the process of using UI-RPA to extract key indicators from unstructured financial statements in text format and then inputting these indicators into the company's internal system, which lacks an API connection. Here, the Excel file represents the user's internal system that cannot be connected to the LLM through an API.

Figure 4 Example Workflow of API-RPA Approach



Notes: This figure describes the process of using API-RPA to extract key indicators from unstructured financial statements in text format and then inputting these indicators into the company's internal system, which lacks an API connection. Here, the Excel file is used to represent the user's internal system that cannot be connected to the LLM through an API.

Table 1 Summary of the Pros and Cons of LLM Implementation Method in Accounting

Method	Pro or Con	Description	Accounting Tasks
UI	Pro	• No-code environment • Ease of demonstration • Interactive experience	• Client engagement and consultation • Basic financial analysis and reporting • Basic compliance checking
	Con	• Low scalability • Limited customization • Input and output token limitations	
API	Pro	• Ease of integration into the current working process • Scalable processing • Allows parameter setting and customization	• Basic financial data extraction • Transaction classification and verification • Basic fraud detection
	Con	• Complexity in setup and maintenance • Incompatibility with certain legacy systems • Make the processes more opaque to users	
UI-RPA	Pro	• No-code environment • Ease of demonstration • Interactive experience • Achieves batch querying with LLMs while maintaining ease of integration with existing workflows • Enhance the accuracy and consistency of querying results while also maintaining a high level of verifiability • Supports the combination of manual processes	• Expense management and audit • Asset management and depreciation scheduling • Budgeting and forecasting

		• Caters to a broader range of users with diverse interests and needs	
	Con	• Limited customization • Input and output token limitations • Complexity in setup and maintenance • Not compatible with systems require CAPTCHA • Creates additional privacy and cybersecurity concerns	
API-RPA	Pro	• Scalable processing • Allows parameter setting and customization • Caters to a broader range of users with diverse interests and needs • Achieve the most in-depth integration with existing workflows	• Systematic financial data extraction and analysis • Systematic regulatory compliance and reporting • Systematic audit trail analysis and fraud detection
	Con	• Complexity in setup and maintenance • Incompatibility with certain legacy systems • Not compatible with systems requiring CAPTCHA • Creates additional privacy and cybersecurity concerns • Make the processes more opaque to users	

Notes: This table summarizes the pros and cons of each LLM implementation method in the accounting domain and some related accounting tasks.

Table 2 Costs of Different LLM Implementation Methods

	Time cost in minutes (Without GPT-4's prompt limitation)	Time cost in minutes (With GPT-4's prompt limitation)	Labor required	Monetary cost
	(1)	(2)	(3)	(4)
UI	208	1,800	Yes	$0.83
API	141	141	Yes	$15.36
UI-RPA	67	1,800	No	$0.83
API-RPA	41	41	No	$15.36

Notes: This table summarizes the estimated cost of implementing GPT-4 to extract key indicators from 500 unstructured financial statements in text format and then input these indicators into the company's internal system, which lacks an API connection. Column (1) calculates the time cost in minutes without GPT-4's current prompt limitations of 50 prompts every three hours. Column (2) calculates the time cost in minutes under GPT-4's current prompt limitations of 50 prompts every three hours. Column (3) indicates whether labor cost is required to perform the tasks. Column (4) estimates the monetary cost in dollar amounts for different implementation approaches. The time cost for the UI approach in column (1) is calculated by considering 10 seconds to search the statement for values and 15 seconds to fill the value into the system: (10s + 15s) * 500 / 60s = 208 minutes. The time cost for the API approach in column (1) is calculated by considering 2 seconds to search the statement for values and 15 seconds to fill the value into the system: (2s + 15s) * 500 / 60s = 141 minutes. The time cost for the UI-RPA approach in column (1) is calculated by considering 5 seconds to search the statement for values and 3 seconds to fill the value into the system: (5s + 3s) * 500 / 60s = 67 minutes. The time cost for the API-RPA approach in column (1) is calculated by considering 2 seconds to search the statement for values and 3 seconds to fill the value into the system: (2s + 3s) * 500 / 60s = 41 minutes. For column (2), the time cost for UI and UI-RPA is calculated by considering 50 prompts every three hours: (500 / 50) * 3h * 60m = 1,800 minutes. The monetary cost for the UI and UI-RPA approaches in column (4) is calculated by considering the current monthly price of $20 for GPT-4 UI and the 50 prompts limitation every three hours. The price per prompt would be $20 / (50 * (30d * 24h / 3h)) = $0.0017. Thus, extracting 500 statements would result in a cost of $0.0017 * 500 = $0.83. The monetary cost for the API and API-RPA approaches in column (4) is calculated by considering the current OpenAI's API charges of about $0.03 per 1,000 input tokens and $0.06 per 1,000 output tokens. Assuming an average token length for the statements of 1,024, the monetary cost of the API for 500 statements would be 1.024 * $0.03 * 500 = $15.36.

Appendix 1 API Approach Code Example

```
# Importing necessary libraries
import pandas as pd
import re
import openai
import os
import glob
import json

# Setting the OpenAI API key.
openai.api_key = os.getenv('OPENAI_API_KEY')

# Function to connect to OpenAI's language model
def openai_llm_connector(statement):
    """
    Connects to OpenAI's language model to process text data.
    Parameters:
    - statement: str. The textual content to be processed by the language model.
    Returns:
    - str: The processed output from the language model in JSON format.
    Exceptions:
    - Raises exceptions on API call failure or data processing issues.
    """
    try:
        completion = openai.ChatCompletion.create(
            model="gpt-4",
            temperature=0,
            messages=[
            {
            "role": "system",
            "content": '''You are an assistant who is good at extracting information from
            unstructured textual data. Strictly obey the following rules when extracting:
            Rule 1. Find each value by recognizing the relevant row and column names.
            Rule 2. If certain row and column names cannot exactly match, find the
            highly possible one based on fuzzy matching and surrounding information.
            Rule 3. If a certain value cannot be found, return '' for that value.
            Rule 4. Only output the above value in JSON Schema with the following
            format, without any other explanations:
            {
                "NPL": INSERT LIST HERE,
                "OPEB": INSERT LIST HERE,
            }'''
            },
            {
            "role": "user",
            "content": "Context:\n" + statement + '\n' +
            '''The context is a table. Find the following values from the table while
            following all the rules:
            1. Net pension liability for Total activities. Output as a list format.
            2. Other post-employment benefits (OPEB) liabilities for Total activities,
            do not consider deferred inflows or OPEB assets. Output as a list format. '''
            }
        ]
        )
        return completion.choices[0].message.content
    except Exception as e:
        raise Exception(f"Error in OpenAI API call: {e}")
```

```python
# Perform the data extraction
dir_statements = 'PATH_TO_THE_STATEMENTS_FOLDER' # Replace with actual folder path
files = glob.glob(os.path.join(dir_statements, '*.txt'))

for file in files:
    try:
        with open(file, 'r') as f:
            file_content = f.read()

        data = openai_llm_connector(file_content)
        print(os.path.basename(file).split('.')[0], ':', data)
    except Exception as e:
        print(f"Error processing file {file}: {e}")
```