PolicyPal: CoT, Zero-Shot Automated Expense Policy Compliance Analyzer

Mateo Quiros Bloch
Department of Symbolic Systems
Stanford University
mquiros@stanford.edu

Nicholas Wolf Reisner
Department of Computer Science
Stanford University
nreisner@stanford.edu

Abstract

Manual expense auditing remains a significant challenge for institutions, requiring extensive human effort to verify policy compliance across large volumes of receipts. Automated solutions struggle with complex policy interpretation, diverse document formats, and transparent decision-making. We present PolicyPal, a chain of thought (CoT), zero-shot system for automated expense compliance verification that addresses these challenges through a novel combination of policy pre-processing, receipt extraction, policy content retrieval, and compliance evaluation techniques that generalize to any organization's policy structure and maintain interpretable decision paths.

Through evaluation on Stanford University's *Business and Travel Expense* policy, our system demonstrates strong content retrieval and evaluation abilities compared with other RAG solutions. We further gathered feedback from users, who deemed the system highly transparent (4.5/5), and useful (3.8/5) compared to manual compliance analysis. While retrieval recall performs well, evaluation on real-world receipts and user experiences reveal current limitations in retrieval precision, which makes evaluation slow and costly, suggesting further pre-processing and a conversational agent to gather further expense information as a promising direction for future work.

1 Introduction

Expense auditing is the process of verifying that submitted expenses comply with organizational policies through review of receipts, documentation, and other supporting materials or user-provided information. Currently, expense auditing relies on manual review processes, where human auditors must evaluate each submission against complex institutional policies - a process that is both expensive and tedious. Traditional automation approaches to this problem might employ rigid rule-based systems, but such approaches are ineffective for expense auditing due to the semantic nuance of policies.

Large Language Models (LLMs) seem poised for this task, as they have demonstrated impressive reasoning capabilities across diverse domains (Zhang et al., 2024b; Luo et al., 2024). In financial applications specifically, LLMs have shown promise in regulatory compliance and auditing tasks (Nie et al., 2024), with recent work demonstrating their effectiveness in interpreting complex financial regulations (Cao and Feinstein, 2024) and analyzing tax audit processes (Choi and Kim, 2024). However, while these advances have primarily focused on standardized regulatory frameworks, the challenge of automating compliance verification for institution-specific policies remains unaddressed.

Recent work in other auditing contexts has made progress by incorporating LLMs into the intermediate reasoning steps of larger, multi-stage system (Cao and Feinstein, 2024; Choi and Kim, 2024). These advances mirror the wider promise chain of thought prompting (CoT; Wei et al., 2022) has demonstrated in promoting symbolic reasoning capabilities of LLMs and the potential of agential expert systems. Moreover, the reasoning transparency provided by CoT and, ideally, well-crafted expert system at large are of particular importance for expense auditing: institutional partners need clear justification for each decision in a process where errors could result in significant financial liability. Therefore, they need interpretable and deterministic results that keep track of the retrieval and compliance evaluation steps made along the way.

In this context, we propose PolicyPal, a transparent system for automated expense compliance verification that generalizes to any institutional expense policy with a sectional hierarchy. PolicyPal employs a novel prompt-based architecture to pre-processes institutional policies into sectional blocks

with metadata, standardize extracted receipt PDF data, retrieve relevant policy sections, and evaluate expenses against policy requirements identified during pre-processing. Finally, it generates structured reports detailing compliance status, further actions and recommendations.

We evaluated PolicyPal using our own expense dataset, which consists of diverse expense types and their corresponding policy requirements from Stanford University's *Business and Travel Expense* policy (Stanford University Financial Management Services, 2024). We define metrics for assessing policy rules labeling, policy retrieval, and compliance classification quality against human auditor decisions. Going further, we conducted a user study to understand the auditor experience.

Our main contributions include:

- A generalizable framework that maintains interpretable decision paths through policy preprocessing, extraction, retrieval, and evaluation.
- An information retrieval system that does not rely on RAG vector embeddings and ranking models but rather on zero-shot prompted LLM reasoning for applications where recall is the priority.
- Comprehensive evaluation metrics and user assessment that demonstrate the comparative performance of our approach while highlighting critical limitations and areas for future work.

2 Related Work

2.1 LLMs In Financial Auditing

LLMs have shown increasing promise in financial auditing and compliance tasks, with recent work demonstrating their effectiveness across multiple domains (Nie et al., 2024). A key emerging pattern is the success of multi-stage, LLM-based systems that combine zero-shot capabilities with specialized processing steps. Hillebrand et al. (2023) demonstrated this through ZeroShotALI, which paired LLMs with domain-specific BERT models for financial document auditing, while Deußer et al. (2023) extended this approach by introducing embedding-based paragraph clustering to detect inconsistencies in financial reports. In parallel, research in smart contract auditing has validated this multi-stage approach, with systems like GPTScan (Sun et al., 2024) and LLM-SmartAudit (Wei et al., 2024) showing that breaking down complex auditing tasks into smaller, LLM-manageable components improves performance. Our work builds on these insights, applying a multi-stage architecture to the novel domain of expense policy compliance.

2.2 CoT Reasoning

Chain of thought (CoT) prompting has emerged as a powerful technique for enhancing LLMs' reasoning capabilities on complex, multi-step tasks (Wei et al., 2022). While initial applications focused on mathematical and logical reasoning, recent work has demonstrated CoT's effectiveness in specialized domains requiring structured analysis. In legal applications, researchers have adapted CoT for rule-based reasoning by embedding established legal frameworks into the prompting process, improving zero-shot performance (Yu et al., 2022). Similarly, in financial contexts, CoT prompting has enabled LLMs to perform financial analysis comparable to professional analysts (Callanan et al., 2023). Our work builds on these advances by applying CoT to expense policy compliance verification, a unique blend of both domains.

2.3 Information Retrieval

Information retrieval has evolved from basic embedding-based approaches to more sophisticated methods like Retrieval-Augmented Generation (RAG) and reranking systems. However, in legal and regulatory contexts, where missing relevant information can have serious consequences, novel approaches have emerged to address domain-specific challenges. de Oliveira Lima (2024) demonstrated that breaking down legal texts into multiple granularities of embeddings, from individual clauses to structural groupings, can capture complex hierarchical relationships and improve retrieval performance. Similarly, Nguyen et al. (2024) observed that traditional retrieval systems struggle with real-world scenarios that do not use domain-specific vocabulary, leading them to leverage LLMs' reasoning capabilities for query expansion and reformulation. Our work combines insights from both approaches: like de Oliveira Lima (2024), we pre-process policies into structured, hierarchical components with clear relationships, and like Nguyen et al. (2024), we employ LLM reasoning to bridge the gap between everyday expense scenarios and formal policy language.

3 Core Ideas

PolicyPal was developed and validated using Stanford University's *Business and Travel Expense* policy. In particular, the "Business and Travel Expenses" overview page, as well as the eight expense category pages ("Airfare", "Lodging", "Ground Transportation", "Meals", "Other Reimbursable Business Expenses", "Employee Gifts", "Business Meals", and "Travel Meals") were included in the modeling set. While some methods were adapted to the specifics of this policy, our framework is generalizable to any institutional policy through its flexible architecture that built around zero-shot prompting. The PolicyPal framework operates in five main stages implemented in this instance using different sizes of Meta Llama 3.1 models (8B, 70B, 405B) (Grattafiori et al., 2024) based on task complexity: pre-processing (§3.1), extraction (§3.2), retrieval (§3.3), evaluation (§3.4), and report generation (§3.5).

3.1 Policy Pre-Processing

The pre-processing stage (Figure 1 ①) structures an institutional policy $\mathcal{P}$ for efficient runtime processing. Overall, pre-processing required both automatic and manual procedures. We first break the content-holding sections of $\mathcal{P}$ into blocks $\mathcal{B} = \{b_1, b_2, ..., b_n\}$ using turndown (Christie, 2017), an automated text-to-markdown converter. Following the blocking, we manual cleaned and formatted the blocks as a JSONL file to ensure clear indexing, retrieval, and data quality. Each block $b_i \in \mathcal{B}$ is then processed using Llama 3.1 70B to generate a summary s_i specialized to assist in later retrieval tasks.

A key component of pre-processing is using the advanced reasoning of Llama 3.1 405B to parse and label each block's content into clauses $\mathcal{C}_i = \{c_1, c_2, ..., c_n\}$ of specific clause types. We categorize clauses into three main types:

1. "Rule needing user detail to determine if valid expense": These clauses contain requirements that need specific information for verification. They typically include words like "must", "required", "only", or "should" and could invalidate an expense if not met. For example, requirements about meals falling under a certain dollar amount would fall into this category.

2. "Required Action": These clauses specify additional steps needed for compliance, such as submitting documentation, obtaining approvals, completing forms, or providing comparisons. Examples include requirements for uploading comparative quotes or obtaining manager signatures.

3. "Keep In Mind": These clauses provide contextual information without creating requirements. They include process descriptions and helpful context that do not affect expense validity, such as providing contacts in the case of confusion.

3.2 Receipt Information Extraction

The extraction stage (Figure 1 ②) standardizes diverse receipt formats into a consistent JSON structure for evaluation. Our system extracts the content of the receipt $\mathcal{R}$ using the Llama 3.1 70B and formats its data into a JSON $\mathcal{D}$ with a standardized structure.

The extraction process captures key fields including:

- General information (receipt ID, date, time, vendor details)
- Transaction details (payment method, currency)
- Expense details (total amount, subtotal, taxes, tips)
- Category-specific information based on receipt type

This standardization ensures consistent downstream processing regardless of original receipt format or complexity.

3.3 Policy Retrieval

Instead of relying on embedding-based retrieval or re-ranking, we employ a purely prompt-based, zero-shot retrieval paradigm (Figure 1 ③) augmented by chain-of-thought reasoning. We prioritize recall over precision, accepting a larger candidate set to avoid omitting crucial information.

PolicyPal employs a two-stage retrieval process to identify policy content relevant to the receipt data. Both filtering steps include the receipt data $\mathcal{D}$ and user context $\mathcal{U}$ (user status, organization, location, date) in the prompt fed to Llama 3.1 405B, and use CoT prompting to provoke reasoning. Each filtering step's final output is restricted to a predefined JSON structure that maps to the original

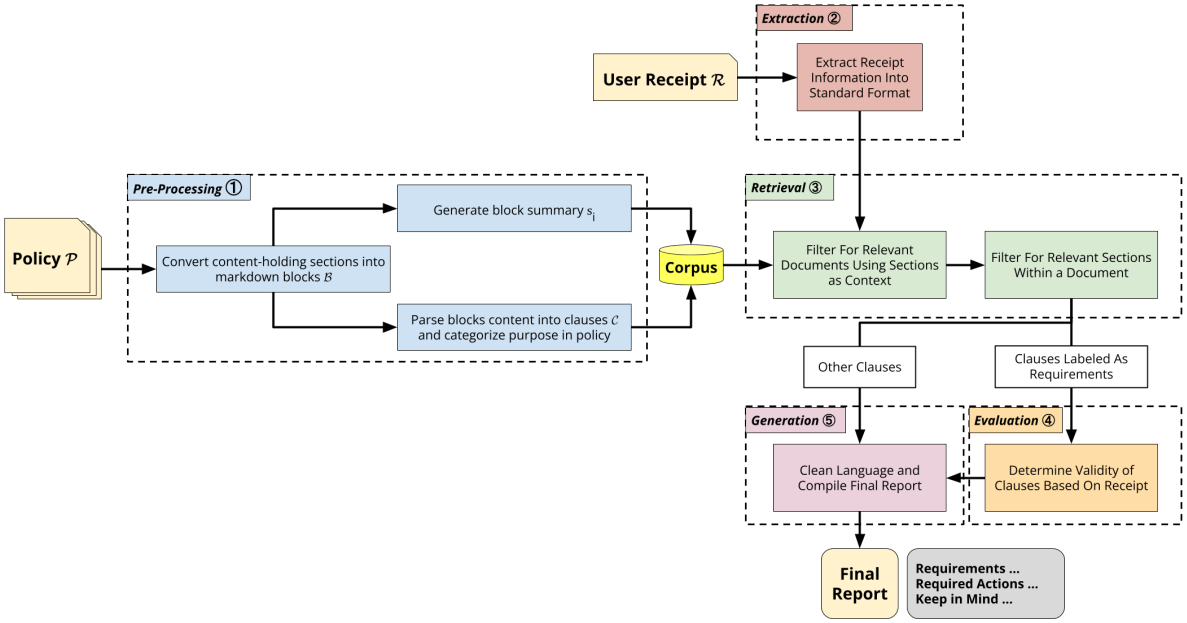

Figure 1: Overview of PolicyPal's automated expense compliance verification system. Starting with institutional policy documents, the system first pre-processes them into structured blocks with summaries and labeled clauses (①). When a user submits a receipt, the extraction stage standardizes it into a consistent format (②). The retrieval stage then identifies relevant policy sections through a two-stage filtering process (③). Finally, the system evaluates the receipt against retrieved requirements (④) and generates a structured compliance report (⑤).

policy JSON for smooth retrieval and keeps logs of the LLM's reasoning.

Our retrieval process comprises two filtering steps:

1. *Document-Level Filtering:* Given $\mathcal{D}$ and $\mathcal{U}$, we initially filter the sections blocks $\mathcal{B}$ by policy document $d_i \in \mathcal{P}$, using the titles of the block $b_j \in d_i$ and its summary s_j. This yields a coarse-grained subset of potentially relevant sections.

2. *Section-Level Filtering:* For each candidate block b_i from document-level filtering, b_i is ultimately included in the final retrieved sections $\mathcal{F}$ based on its title and s_i. This can be done in batches of blocks to accommodate the LLM's context window.

3.4 Requirement Evaluation

The evaluation stage (Figure 1 ④) processes each clause $c_j \in \mathcal{C}_i$ for all $b_i \in \mathcal{F}$ where c_j is labeled as a requirement during pre-processing. For c_j, the system determines validity based on receipt data $\mathcal{D}$ and user context $\mathcal{U}$ with four possible outcomes: "yes" (requirement met), "no" (requirement not met), "unclear" (insufficient information), or "not applicable" (the rule does not apply to the expense at hand). An explanation is provided for each determination to improve reasoning quality and provide transparency into decision making.

3.5 Report Generation

The final stage of PolicyPal is the generation of a structured report (Figure 1 ⑤) that synthesizes all c_j with the evaluation results, cumulating in a comprehensive yet accessible format. Rather than simply providing binary compliance decisions, the report organizes information into three key categories: Requirements Analysis, Required Actions, and Contextual Information. This structure ensures users receive not just compliance decisions, but also the guidance needed to resolve any issues.

The Requirements Analysis section forms the core of the report, grouping requirements by their source policy sections. For each requirement, the report provides an AI evaluation of "yes," "no," "unclear", or "not applicable," accompanied by a brief explanation of the reasoning. These evaluations are presented hierarchically under policy section headings, which are hyperlinked to their source documents using metadata from the pre-processing stage. This maintains clear traceability while allowing users to easily access the full policy context if needed.

Following the requirements analysis, the report presents Required Actions derived from clauses specifically labeled during pre-processing. These actions outline concrete steps needed for compliance, such as submitting additional documentation or obtaining approvals.

Finally, the report includes relevant Contextual Information drawn from "Keep in Mind" clauses, providing helpful process information and guidelines that, while not requirements themselves, aid in understanding and compliance.

This structured yet narrative approach to report generation ensures that users can quickly identify both compliance status and any needed actions while maintaining access to the broader policy context through section links. The hierarchical organization and clear labeling help users navigate complex policy requirements while maintaining the transparency of the system's decision-making process.

4 Experimental Results

We present the evaluation methodology and results of our agential policy retrieval and compliance classification framework applied to Stanford University's *Business and Travel Expenses* policy. Our primary objective is to assess the performance of the retrieval step—specifically, our novel zero-shot, CoT prompt-based retrieval approach—in comparison with baseline methods.

4.1 Data and Policy Corpus Setup

We used Stanford University's *Business and Travel Expenses* policy as our source document. Since the policy contained 117 sections, we selected 10 sections at random for evaluation of the Clause Type classification. For Section Retrieval and Rule Compliance, we also generated 6 synthetic expense receipts, varying in compliance status and category (e.g., air and ground transportation, business and travel meals, COVID-19 testing, conference expenses).

Two human evaluators, who read the complete policy beforehand, reviewed these receipts and identified which sections were relevant as well as how the rules applied. However, given that the evaluators are not trained auditing professionals, uncertainties may arise: ambiguities in interpreting certain clauses as rules or recommendations, as well as policy sections with broad relevance (e.g., IRS responsibilities sections that might apply indirectly), can influence the reliability of these results.

Evaluated Tasks:

1. *Clause Labeling (Ternary):* Whether the clause is a rule requiring user information to validate compliance, an action the user must take, or a "keep in mind" guideline.

2. *Section Retrieval (Binary):* Whether the clause's section should be considered relevant or irrelevant for a given expense submission.

3. *Rule Compliance (Quaternary):* For clauses identified as rules, whether an expense is compliant (*yes*), non-compliant (*no*), indeterminate (*unclear*) due to insufficient information, or simply (*not applicable*).

4.2 Evaluation Metrics

To evaluate the performance of our model, we examined accuracy, recall, precision, F1 score, and the macro-averaged F1 score.

4.3 Results

4.3.1 Clause Labeling Classification

We evaluated the clause type classification performance in terms of Precision, Recall, F1-score, and macro-average F1-score. Table 1 presents the results.

Clause Type	Precision	Recall	F1-Score
Requirements	0.85	0.78	0.81
Required Actions	0.73	0.70	0.71
Keep in Mind	0.87	0.55	0.67
Macro-Average	–	–	0.73

Table 1: Precision, Recall, and F1-Scores for Clause Type Classification. Macro-average F1 is computed across the three classes.

4.3.2 Section Retrieval Classification

As alternatives to our two-stage filtering process, we provided the same pre-processed Policy JSON to perform one-step retrieval with Llama 3.1 405B, LLamaRank and Genie Search (with adapted queries). For the LlamaRank and Genie Search alternatives, we picked the top-k sections, where k is the number of sections retrieved by our two-step retrieval system. Table 2 presents these findings.

1. *One-Step LLM Retrieval (Llama 3.1 405B):* A single-query retrieval method using a large language model without intermediate filtering steps.

2. *LlamaRank:* A retrieval re-ranker that uses embeddings and LLM scoring to return a ranked list of relevant sections.

3. *Genie Search:* A multilingual vector embedding-based retrieval system combined with an LLM-based ranker. Genie Search first identifies top-n documents using embeddings, then refines the ranking with an LLM re-ranker. We were not able to run the full evaluation for Genie Search, but this method was discarded after poor initial performance.

Method	Precision	Recall	F1-Score
PolicyPal (Two-step)	0.64	0.97	0.77
One-step (LLM)	0.40	0.82	0.54
LlamaRank	0.36	0.55	0.44

Table 2: Comparison of retrieval classification performance among different retrieval approaches.

4.3.3 Rule Compliance Classification

Table 3 presents the Precision, Recall, and F1-Score for each category, as well as the macro-average F1-score across all three classes.

Class	Precision	Recall	F1-Score
Yes	0.81	0.82	0.81
No	0.75	0.70	0.72
Maybe	0.30	0.93	0.45
Not applicable	0.92	0.79	0.85
Macro-Average	–	–	0.71

Table 3: Rule compliance classification results. Macro-average F1 is computed across the three classes.

4.4 User Study

Although we primarily focused on technical metrics to evaluate system performance, understanding how end-users perceive the quality, transparency, and utility of the system is also important. This user study mainly helps gauge whether our approach and final report can be trusted and understood by non-experts who would otherwise have to navigate complex organizational policies.

We recruited ten undergraduate participants, each tasked with reviewing an expense receipt. The users used our system and afterwards manually evaluated the compliance of the expense based on Stanford University's *Business and Travel Expenses* policy website. Participants rated the system on three dimensions using a 1-to-5 scale: Intelligence, Transparency, and Usefulness. They also provided open-ended feedback on their experience. Results from the rating are found in Table 4.

Rating (1 to 5)	Intelligence	Transparency	Usefulness	Overall
Mean	3.4	4.5	3.8	3.9

Table 4: User study results from ten undergraduate participants.

5 Analysis

5.1 Clause labeling

The clause type classification results (Table 1) demonstrate that our system effectively identifies *Requirements* clauses with reasonably high precision and recall, leading to a strong F1-score of 0.81. Although performance on *Required Actions* is slightly lower, *Keep in Mind* clauses have a remarkably low recall, indicating the LLMs limitations in identifying ambiguous recommendations and tendency to label them as rules or required actions. Furthermore, a different classifications system could be considered given the similarities between "Required Actions" and "Keep in Mind" classes.

5.2 Section Retrieval

In terms of retrieval classification (Table 2), the two-step *PolicyPal* approach outperforms baseline methods. Its recall of 0.97 ensures that almost all relevant sections are identified. However, the low precision posses problems downstream at the time of rule compliance evaluation and final report generation due to the list of relevant sections to parse through being too long. While one-step LLM retrieval offers competitive recall, their precision remains lower. As for the two embedding-based and ranking methods (LlamaRank and Genie Search), overall performance indicates that these approaches are not suitable for the Expense Policy Auditing task, highlighting the limitations of embedding-ranking-based retrieval and the importance of reasoning for nuanced tasks like ours.

5.3 Rule Compliance

For rule compliance evaluation (Table 3), our system yields solid performance on definite compliance judgments (*Yes* and *No*). However, the *Maybe* category—representing cases where insufficient in-

formation prevents a clear decision—is challenging. Despite very high recall, the low precision here indicates that the model overestimates ambiguity, thus lowering the F1-score for this class. The challenge here is mainly the lack of information about what the data the expense is missing and what the user could provide if prompted, thus overestimating how much data is missing to make an assessment. As for the *Not applicable* classification, it performed really well in precision and recall, effectively aiding as a third filtering step of non-applicable rules within a given section's content. A slightly lower recall for this class indicates that some ambiguous clauses were treated as applicable and could have contributed to the large number of *unclear* predictions for clauses like "The user should be responsible when utilizing the university's funds."

5.4 User Study

For the user study, as shown in Table 4, the system received an average high rating for Usefulness and Transparency and a lower one for Intelligence, suggesting that while the underlying reasoning capabilities could be improved, the system effectively communicates how it arrives at its results. Participants found the system reasonably useful which is reflected in an average overall rating of 3.9.

In the open ended part, participants noted that the system at times provided irrelevant details and exhibited considerably slower response times than expected. Despite these drawbacks, they still perceived it as beneficial, highlighting the value of clear explanations and traceability.

These findings support our earlier assertions: while the system is not yet fully automated or free of superfluous information, its transparency and clear linkage back to the original policy content foster user trust and facilitate effective human-in-the-loop review. They also identified challenges for future research, including addressing cases where: (1) policy ambiguity requires human judgment for interpretation, and (2) LLMs make incorrect connections between unrelated policy sections.

6 Key Learnings

6.1 Value of Policy Pre-Processing

Our experiments demonstrated that policy pre-processing significantly improves system efficiency at runtime. Policies are written with humans as their intended audience. Therefore, there can be significant gains in adapting these policies to chunked and labeled sections of granular rules, required actions and recommendations that would facilitate LLM reasoning. Extracting requirements once during pre-processing rather than at runtime showed substantial computational advantages, and given that institutions process thousands of receipts daily, this approach would seemingly only scale in practice. Pre-processing would also enable expert oversight of the pre-processing output, allowing for manual correction of any LLM-generated errors before deployment.

6.2 Challenges in Policy Retrieval

Policy retrieval proved particularly challenging due to the nuanced nature of policy text. Traditional approaches using embedding models (e.g., Genie Search) (Semnani et al., 2023; Zhang et al., 2024a) and reranking models (e.g., Llama Rank) (Ginart and Kodali, 2024) demonstrated insufficient recall, even for seemingly obvious relevant sections. LLMs showed superior performance in this task, leading to our prompt-based approach. While we settled on the two-stage filtering technique, alternative architectures were also explored. Most notably, a tree-based structure for navigating policy sections showed promise but proved computationally expensive without significant performance gains (see §7 for further discussions).

The fundamental challenge in policy retrieval stems from inherent ambiguity in relevance determination. For instance, while certain policy sections (such as "Receipts and Supporting Documentation" in Stanford's Business and Travel Policy) technically apply to all submissions, their retrieval may not provide additional value to users already familiar with basic documentation requirements. An approach that merges RAG embeddings with our prompt-based CoT approach for ranking or filtering could be used to exploit the strengths of each technique.

6.3 LLM Reasoning Limitations

While LLMs demonstrated strong reasoning capabilities throughout our system, they also exhibited notable limitations. Some challenges were anticipated, such as difficulty in cleanly categorizing policy clauses during pre-processing. However, other limitations were surprising and revealed the potential limit of an LLM-based approach to expense auditing at this time.

For example, while experimenting with conversationality, LLMs consistently struggled to differentiate between questions with seemingly obvious differences in specificity (e.g., conflating "Is this expense for transportation?" with "Is this expense specifically for ground transportation?") despite explicit prompting to be aware of qualifiers. These challenges suggest potential opportunities for hybrid approaches combining LLM reasoning with traditional NLP techniques that remove some of the decision making burden from the LLM by introduce deterministic steps such as keyword extraction, rule-based pattern matching, or some other relevant method.

7 Future Work

7.1 Exploration of Conversational Approaches

We experimented extensively with introducing conversational capabilities to enhance the system's retrieval and evaluation accuracy through targeted user interaction. The primary goal was to develop a mechanism for identifying and resolving ambiguities in receipt interpretation through user dialogue. The most promising architecture implemented a tree-structured traversal of policy sections, where each node represented a decision point for further policy exploration. Initially, we attempted dynamic question generation at each node based on receipt content and conversational history. However, this approach revealed significant limitations in LLMs' ability to generate contextually appropriate, non-redundant questions that captured policy nuances within a single prompt.

These limitations led to a decomposed approach: generating questions for each node during preprocessing, followed by runtime processing where an LLM would attempt to answer questions using receipt data, deferring ambiguous cases to user interaction. While this structured approach showed initial promise, it encountered several challenges. First, LLM-based question answering demonstrated poor reliability, leading us to implement conservative answering thresholds that favored user interaction. However, this conservative approach resulted in excessive user questioning, often with redundant or overlapping queries. Our attempts to mitigate redundancy through LLM-based question grouping proved unreliable, with groupings frequently failing to capture semantic similarities or differences accurately.

Furthermore, the hybrid approach of interweaving LLM and user interactions introduced significant computational overhead. While potential solutions through multi-threading were identified, they exceeded both our computational resources and the scope of this initial investigation.

This experience suggests that while conversational approaches show theoretical promise for improving system certainty during retrieval and rule evaluation, their practical implementation remains an active line of study. One possibility could be the dynamically generate a Genie Worksheets to handle in conversation. No matter which approach is taken, the requirement would be to address fundamental challenges in question generation, redundancy detection, and computational efficiency.

7.2 Towards a More Deterministic and Generalizable System

Our current approach relies on multiple calls to various large language models (LLMs) for policy pre-processing, retrieval, and compliance evaluation. While this strategy demonstrates flexibility, it introduces complexity and potential inconsistencies by increasing the chances of hallucinations.

To address these challenges, future work should focus on introducing deterministic methods where possible. These could include using rule-based patterns, regular expressions, or specialized NLP techniques for clause extraction and classification. Another promising avenue involves additional LLM-powered stages that could improve determinism, such as the generation of satisfiability modulo theories (SMT) formulae that could then be solved by a theorem prover for compliance evaluation. These deterministic strategies can effectively complement LLM outputs, ensuring consistency in areas where language models are prone to variability.

Furthermore, integrating parameter-efficient tuning techniques can help align LLMs more closely with domain-specific tasks. This approach has the potential to reduce the number of necessary model calls while simultaneously improving the system's overall reliability. By combining these various strategies—deterministic methods, specialized validation, and targeted model tuning—we can create a more robust and consistent system while preserving the advantages of LLM-based processing.

8 Conclusion

Our experimental results highlight both the promise and the current limitations of a prompt-based LLM-driven approach to organizational expense policy auditing. While the two-step retrieval method offers high recall, achieving a fully automated and error-free solution remains challenging. For a system to operate without human oversight, it would need to reliably attain near-perfect retrieval recall, ensuring that no relevant policy clauses are overlooked. Likewise, compliance evaluation still poses difficulties, particularly in ambiguous or information-sparse scenarios. Integrating a conversational agent capable of querying the user for missing details could help bridge these gaps, allowing the system to gather additional information and improve decision confidence.

Despite these hurdles, the approach delivers tangible benefits. By providing a detailed reasoning trail for each retrieval step, explaining which rules were applied and why, and linking back to the original policy content, the system fosters transparency and trust. This auditability makes it easier for human reviewers to understand the AI's logic, confirm its conclusions, and intervene when necessary.

Ultimately, while the current system is not yet ready for fully automated policy compliance auditing, it lays a foundation for more robust, explainable, and human-in-the-loop solutions that can mature into highly reliable decision-support tools in the future.

References

Ethan Callanan, Amarachi Mbakwe, Antony Papadimitriou, Yulong Pei, Mathieu Sibue, Xiaodan Zhu, Zhiqiang Ma, Xiaomo Liu, and Sameena Shah. 2023. Can gpt models be financial analysts? an evaluation of chatgpt and gpt-4 on mock cfa exams. *Preprint*, arXiv:2310.08678.

Zhiyu Cao and Zachary Feinstein. 2024. Large language model in financial regulatory interpretation. *arXiv preprint arXiv:2405.06808*. Https://doi.org/10.48550/arXiv.2405.06808.

Ga-Young Choi and Alex Kim. 2024. Firm-level tax audits: A generative ai-based measurement. *Chicago Booth Research Paper No. 23-23*.

Dom Christie. 2017. turndown. Copyright © 2017+ Dom Christie. Released under the MIT license. Accessed: December 7, 2024.

João Alberto de Oliveira Lima. 2024. Unlocking legal knowledge with multi-layered embedding-based retrieval. *Preprint*, arXiv:2411.07739.

Tobias Deußer, David Leonhard, Lars Hillebrand, Armin Berger, Mohamed Khaled, Sarah Heiden, Tim Dilmaghani, Bernd Kliem, Rüdiger Loitz, Christian Bauckhage, and Rafet Sifa. 2023. Uncovering inconsistencies and contradictions in financial reports using large language models. In *2023 IEEE International Conference on Big Data (BigData)*, pages 2814–2822.

Antonio Ginart and Naveen Kodali. 2024. Introducing llamarank: A state-of-the-art reranker for trusted ai. Accessed: December 8, 2024.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*. Https://doi.org/10.48550/arXiv.2407.21783.

Lars Hillebrand, Armin Berger, Tobias Deußer, Tim Dilmaghani, Mohamed Khaled, Bernd Kliem, Rüdiger Loitz, Maren Pielka, David Leonhard, Christian Bauckhage, and Rafet Sifa. 2023. Improving zero-shot text matching for financial auditing with large language models. In *Proceedings of the ACM Symposium on Document Engineering 2023*, DocEng '23, New York, NY, USA. Association for Computing Machinery.

Xiaoliang Luo, Akilles Rechardt, Guangzhi Sun, Kevin K. Nejad, Felipe Yáñez, Bati Yilmaz, Kangjoo Lee, Alexandra O. Cohen, Valentina Borghesani, Anton Pashkov, Daniele Marinazzo, Jonathan Nicholas, Alessandro Salatiello, Ilia Sucholutsky, Pasquale Minervini, Sepehr Razavi, Roberta Rocca, Elkhan Yusifov, Tereza Okalova, Nianlong Gu, Martin Ferianc, Mikail Khona, Kaustubh R. Patil, Pui-Shee Lee, Rui Mata, Nicholas E. Myers, Jennifer K. Bizley, Sebastian Musslick, Isil Poyraz Bilgin, Guiomar Niso, Justin M. Ales, Michael Gaebler, N. Apurva Ratan Murty, Leyla Loued-Khenissi, Anna Behler, Chloe M. Hall, Jessica Dafflon, Sherry Dongqi Bao, and Bradley C. Love. 2024. Large language models surpass human experts in predicting neuroscience results. *Nature Human Behaviour*.

Hai-Long Nguyen, Tan-Minh Nguyen, Duc-Minh Nguyen, Thi-Hai-Yen Vuong, Ha-Thanh Nguyen, and Xuan-Hieu Phan. 2024. Exploiting llms' reasoning capability to infer implicit concepts in legal information retrieval. *Preprint*, arXiv:2410.12154.

Yuqi Nie, Yaxuan Kong, Xiaowen Dong, John M. Mulvey, H. Vincent Poor, Qingsong Wen, and Stefan Zohren. 2024. A survey of large language models for financial applications: Progress, prospects and challenges. *Preprint*, arXiv:2406.11903.

Sina Semnani, Violet Yao, Heidi Zhang, and Monica Lam. 2023. WikiChat: Stopping the hallucination of large language model chatbots by few-shot grounding on Wikipedia. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 2387–2413, Singapore. Association for Computational Linguistics.

Stanford University Financial Management Services. 2024. Business & travel expenses. Accessed: December 6, 2024.

Yuqiang Sun, Daoyuan Wu, Yue Xue, Han Liu, Haijun Wang, Zhengzi Xu, Xiaofei Xie, and Yang Liu. 2024. Gptscan: Detecting logic vulnerabilities in smart contracts by combining gpt with program analysis. In *Proceedings of the IEEE/ACM 46th International Conference on Software Engineering*, ICSE '24, page 1–13. ACM.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*. Https://doi.org/10.48550/arXiv.2201.11903.

Zhiyuan Wei, Jing Sun, Zijiang Zhang, Xianhao Zhang, Meng Li, and Zhe Hou. 2024. Llm-smartaudit: Advanced smart contract vulnerability detection. *Preprint*, arXiv:2410.09381.

Fangyi Yu, Lee Quartey, and Frank Schilder. 2022. Legal prompting: Teaching a language model to think like a lawyer. *Preprint*, arXiv:2212.01326.

Heidi Zhang, Sina Semnani, Farhad Ghassemi, Jialiang Xu, Shicheng Liu, and Monica Lam. 2024a. SPAGHETTI: Open-domain question answering from heterogeneous data sources with retrieval and semantic parsing. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 1663–1678, Bangkok, Thailand. Association for Computational Linguistics.

Tianhua Zhang, Jiaxin Ge, Hongyin Luo, Yung-Sung Chuang, Mingye Gao, Yuan Gong, Yoon Kim, Xixin Wu, Helen Meng, and James Glass. 2024b. Natural language embedded programs for hybrid language symbolic reasoning. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 4131–4155, Mexico City, Mexico. Association for Computational Linguistics.

A Appendix

A.1 Example Prompts

Prompt 1: Adding Policy Section Summary Metadata

""" You are given a segment of an expense policy. Your task is to produce a concise summary that highlights which types of expenses this section applies to, along with the conditions or contexts in which it is relevant. The summary should:

1. Identify the categories of expenses covered (e.g., travel, lodging, meals, conference fees, equipment, services).

2. Highlight any roles or stakeholders mentioned (e.g., employees, faculty, students, guests).

3. Mention any conditions or constraints (e.g., domestic vs. international travel, allowable amounts, required documentation).

4. Avoid irrelevant details and focus on information that would help determine whether this section is applicable to a given expense.

5. Be written as a standalone summary, without referencing the instruction text.

Policy:

$\{content\}$

Summary: """

Prompt 2: Labeling Clauses

""" # Instructions You are helping with expense auditing for $\{organization\}$. The goal is to preprocess the expense policy so that an automated system can later check if a receipt complies with the policy.

For the following section break it up into sentences or logically connected sentences and label each as one of the following:

1. "Need user detail to determine if valid expense" Use this label when: - The clause contains words like "must", "required", "only", "should" that create a requirement - The system needs specific information to verify if the requirement was met - The requirement could make the expense invalid if not met

Examples: - "Must adhere to policy X" -> Need user detail (system needs to verify adherence) - "Only economy flights allowed" -> Need user detail (system needs flight class) - "Should be used when X condition applies" -> Need user detail (system needs to verify condition)

2. "Required Action" Use this label when the clause requires specific additional steps like: - Submit extra documentation - Get approvals - Complete forms - Provide comparisons

Examples: - "Upload comparative quotes" - "Attach receipt copies" - "Get manager signature"

3. "Keep In Mind" Use this label ONLY for: - Pure information with no requirements - Process descriptions that do not affect validity - Helpful context that does not create any rules

Examples: - "Reimbursements are processed weekly" - "The university has preferred vendors" - "You can create a travel account"

Section: $\{sectioncontent\}$

Key points: - If the clause uses words like "must", "required", "should", "only" - it is usually "Need user detail" - If it requires new documents/approvals or the user - it is "Required Ac-

tion" - If it is purely informational with no requirements - it is "Keep In Mind"

Analyze the policy section and return a JSON array where each element has two fields: - "text": The exact sentence or logically connected group of sentences without any modifications, numbering, or formatting changes - "label": One of these three exact strings: - "Need user detail to determine if valid expense" - "Required Action" - "Keep In Mind"

Return nothing but the JSON array, with no additional text whatsoever. """

Prompt 3: Filtering Policy Sections

""" # Instructions You are tasked with identifying the relevant sections of an organization's expense policy based on the given receipt data, user information, and policy section titles with their summaries. Your goal is to decide for each policy section whether it should be included for further audit review or not. Follow these guidelines:

1. Analysis of Input: - Consider the Receipt data and User information as the primary reference. - Compare these details with the Policy Document Titles, Section Titles, and Summaries.

2. Criteria for Inclusion: - Include sections that directly relate to the expense type, category, or scenario indicated by the receipt data. - Consider aspects like expense category (meals, lodging, transportation, airfare, gifts), user's role and affiliation, submission date, and any other contextual information from the user profile. - If uncertain about a section's relevance, err on the side of inclusion.

3. Exclusion Criteria: - Exclude sections that are clearly unrelated to the receipt's details (e.g., no mentioned expense category, irrelevant policy guidance).

4. Output Format Requirements: - Do not produce code. - Return the results as a list of objects in valid JSON format. - Each object should contain: - "documenttitle": The exact document title given. - "sectiontitle": The exact section title given. - "reasoning": A brief explanation of why this section was included or excluded. - "include": "YES" or "NO" depending on whether the section is considered relevant.

5. Formatting: - The output must be enclosed in triple backticks as shown below. - Do not add extra commentary outside the JSON structure.

Receipt Data $\{receiptdata\}$

User Information The user that provided this receipt is a $\{status\}$ at $\{organization\}$, located in $\{location\}$, and the current date is $\{date\}$.

Policy Document Titles, Section Titles and Summaries $\{batchofsections\}$

JSON output example (Final Answer): "'json

[

"documenttitle": "",

"sectiontitle": "",

"reasoning": "",

"include": ""

,

"documenttitle": "",

"sectiontitle": "",

"reasoning": "",

"include": ""

] "' """

Prompt 4: Rule Compliance evaluation

""" # Instructions You are helping with expense auditing for organization. You need to evaluate if a receipt meets several policy requirements. Not all requirements are applicable to the expense type or category.

For each requirement, determine:

- 'yes' if the receipt directly meets it

- 'no' if the receipt directly does not meet it

- 'unclear' if you're not certain or missing information

- 'not applicable' if the requirement is not applicable to the expense type or category

Analyze this receipt against the following requirements.

Receipt: $\{receipt\}$

Requirements:

$\{json.dumps([clause['text'] for clause in clauses], indent = 2)$

Return a JSON array with an object for each requirement. Each object should have:

- 'verdict': exactly 'yes', 'no', 'unclear', or 'not applicable'

- 'explanation': brief explanation in 5-10 words

Return only the JSON array, no other text. """