Agent-Enhanced Large Language Models for Researching Political Institutions

Joseph R. Loffredo* Suyeol Yun†

Most Recent Draft: November 29, 2024

Abstract

The applications of Large Language Models (LLMs) in political science are rapidly expanding. This paper demonstrates how LLMs, when augmented with predefined functions and specialized tools, can serve as dynamic agents capable of streamlining tasks such as data collection, preprocessing, and analysis. Central to this approach is Agentic RAG, which equips LLMs with action-calling capabilities for interaction with external knowledge bases. Beyond information retrieval, LLM agents incorporate modular tools for tasks like document summarization, transcript coding, qualitative variable classification, and statistical modeling, enabling adaptability across diverse research contexts. To demonstrate the potential of this approach, we introduce `CongressRA`, an LLM agent designed to support scholars studying the U.S. Congress. Through this example, we highlight how LLM agents can reduce the costs of replicating, testing, and extending empirical research using the domain-specific data that drives the study of political institutions.

Keywords: large language models, retrieval-augmented generation, AI agent, Agentic RAG, political institutions

Paper prepared for presentation at the Symposium on AI and the Study of Political Institutions, University of Southern California, Los Angeles, CA, December 6, 2024. This is a working draft. Please do not share or circulate without author permission.

*PhD Candidate, Department of Political Science, MIT. `loffredo@mit.edu`

†Independent Researcher. `syyun@alum.mit.edu`

Introduction

In recent years, the applications of large language models (LLM) in political science have grown significantly. Recent surveys on the relevance of LLMs in political science (e.g., Linegar, Kocielnik, and Alvarez 2023; Ziems et al. 2024; Lee et al. 2024) highlight their utility for a range of tasks, including content generation (Bail 2024), measurement scale creation (Wu et al. 2023), simulating human behavior (Binz and Schulz 2023; Dillion et al. 2023; I. Grossmann et al. 2023; Baker and Azher 2024; Mei et al. 2024; Xie et al. 2024), and text analysis (Gilardi, Alizadeh, and Kubli 2023; Nay 2023; Halterman and Keith 2024; Heseltine and Clemm Von Hohenberg 2024; Liu and Shi 2024; Ornstein, Blasingame, and Truscott 2024; Törnberg 2024).

In this paper, we argue and demonstrate that LLMs, when enhanced with pre-defined functions and equipped with specialized tools, can serve as agents (Xi et al. 2023) capable of addressing many of the complex and time-intensive tasks faced by researchers. These agents streamline critical activities, such as data collection, preprocessing (e.g., classification and transforming unstructured data into structured formats), and analysis. By leveraging such agents, scholars can efficiently uncover overlooked variables, process large datasets, prepare data for analysis, and locate sources to measure key concepts.

While LLMs offer significant advantages, they also face well-documented limitations, such as generating incorrect, outdated, or irrelevant information and relying on non-credible sources (Ji et al. 2023; Zhang et al. 2024). These issues, often referred to as "hallucinations," stem from their reliance on static, pre-trained knowledge bases that cannot be easily updated or verified without extensive retraining. To address these challenges, we propose that political science scholars implement and utilize LLMs with the dynamic capabilities of agentic retrieval-augmented generation (Agentic RAG) (Ravuru, Sakhinana, and Runkana 2024; P. Lewis et al. 2020). By equipping LLMs with action-calling capabilities, Agentic RAG enables interaction with external knowledge bases—such as SQL databases, vector databases, or graph databases—as well as APIs and web-based resources. These

agents can autonomously determine the most relevant sources to query and help synthesize accurate, contextually appropriate responses (An et al. 2024; Gupta, Ranjan, and Singh 2024). This dynamic and adaptable system significantly reduces hallucinations and enhances reliability.

Beyond dynamic retrieval capabilities, these agents can package additional functionalities into a single modular system. By integrating tools such as document summarization, classification, and statistical analysis, researchers gain a flexible and transportable system. This means the same agent can be adapted to different research contexts by providing task-specific instructions, while leveraging a consistent set of shared tools that remain relevant across various studies. Such modularity and adaptability make these agents invaluable for addressing the diverse and evolving needs of political science research.

To demonstrate the potential of this approach, we introduce `CongressRA`, an agent designed to support scholars studying the U.S. Congress. `CongressRA` integrates dynamic retrieval capabilities with specialized tools, enabling it to streamline tasks such as replicating existing findings and generating new insights. We illustrate its utility by reconstructing the "legislative gridlock" index developed by Binder (1999). Through this example, we highlight how agents like `CongressRA` can reduce the costs of replicating, testing, and extending empirical research.

The Potential of LLM Agents

Large language models (LLM) have garnered significant attention from those aiming to leverage these tools for Natural Language Processing (NLP). In particular, LLMs are highly effective in Natural Language Generation (NLG), producing coherent and contextually relevant text from structured or unstructured data (Gatt and Krahmer 2018). Technically, LLMs are neural network-based and typically built on the transformer

architecture (Vaswani et al. 2023). By adopting the transformer architecture, LLMs output vectors as contextualized text embeddings that capture the underlying or semantic meaning of inputted text. These vector embeddings can be used for tasks like finding pieces of information by comparing their numerically computable similarities, such as cosine similarity and L_p norms. Furthermore, LLMs show a basic form of human-like intelligence, allowing them to handle a variety of tasks while understanding the context and intent of a user's prompt.

Because of these capabilities, LLMs have a range of potential applications useful to scholars of political science (Linegar, Kocielnik, and Alvarez 2023; Ziems et al. 2024; Lee et al. 2024). These applications include content generation (Bail 2024), measurement scale creation (Wu et al. 2023), simulating human behavior (Binz and Schulz 2023; Dillion et al. 2023; I. Grossmann et al. 2023; Baker and Azher 2024; Mei et al. 2024; Xie et al. 2024), and text analysis (Gilardi, Alizadeh, and Kubli 2023; Nay 2023; Halterman and Keith 2024; Heseltine and Clemm Von Hohenberg 2024; Liu and Shi 2024; Ornstein, Blasingame, and Truscott 2024; Törnberg 2024). The accessibility of these applications has grown due to the availability of open-source models and APIs, enabling scholars to utilize models trained on massive text corpora at relatively low monetary and computational costs. However, there are challenges associated with LLMs, particularly from biases observed in generated outputs (Feng et al. 2023; Gupta, Ranjan, and Singh 2024; Motoki, Pinho Neto, and Rodrigues 2024; Pit et al. 2024; Ranjan, Gupta, and Singh 2024).

Beyond these established applications, a largely overlooked potential of LLMs lies in their capacity to act as comprehensive research assistants (Lu et al. 2024). When equipped with predefined functions and tools, LLMs can perform a variety of tasks that extend beyond content generation or text analysis. For instance, LLMs can preprocess raw data, extract structured information from unstructured sources, and classify or summarize complex datasets. These capabilities make LLMs invaluable for handling repetitive, labor-intensive aspects of research, such as coding qualitative data, linking

datasets, or performing exploratory data analysis.

A specific example of the research assistance capabilities of LLMs is information retrieval. LLMs can refine human-written queries, search through vast amounts of text, and retrieve contextually relevant results (Petroni et al. 2019; Hu et al. 2024; Zhu et al. 2023; Dai et al. 2024; Labruna, Campos, and Azkune 2024; Zhang et al. 2024). However, traditional LLMs face limitations when it comes to handling field-specific contexts or addressing nuanced research questions. These issues often stem from the reliance of models like GPT (Radford et al. 2019) and BERT (Devlin et al. 2018) on static, pre-trained knowledge bases. Without the ability to update their internal knowledge effectively, these models are prone to hallucinations, producing plausible but inaccurate or outdated information (Burger et al. 2023; Ji et al. 2023). For example, Finn et al. (2024) found that ChatGPT struggled to provide accurate and up-to-date descriptions of political conditions across U.S. states, often relying on vague or misleading references.

Two approaches currently exist to address these limitations: fine-tuning (Ouyang et al. 2022; Schick et al. 2023) and retrieval-augmented generation (RAG) (P. Lewis et al. 2020). Fine-tuning involves retraining a model on domain-specific data to update its internal knowledge, but this process is resource-intensive and risks overwriting existing capabilities. RAG, on the other hand, dynamically retrieves external information in response to a user query, allowing the model to access up-to-date and contextually relevant content without model retraining. By incorporating external resources such as SQL databases, vector embeddings, or APIs, RAG enhances the precision, reliability, and transparency of LLM outputs (Thakur et al. 2021).

While traditional RAG is an important enhancement, Agentic RAG builds on this capability by enabling LLMs to act as autonomous agents that dynamically decide when, where, and how to retrieve external information. Unlike traditional implementations of RAG that rely on predefined processes for data retrieval, agents equipped with an

agentic interface can independently select the most appropriate data sources based on the task at hand. For example, within a single workflow, an LLM agent might query a vector database for semantic similarity, switch to a SQL database for structured information, or query APIs for real-time data. Moreover, an agent can iteratively refine its retrieval process, repeatedly querying different databases or the same source multiple times until it identifies plausible and persuasive information to address the user's query.

The flexibility of Agentic RAG lies in its ability to enable discretion in determining which resources to query and how to integrate the retrieved data into the response given to the user. This dynamic approach ensures that the agent can adapt its strategy based on context, using a combination of data sources to provide comprehensive and contextually accurate answers. This capability is particularly powerful in scenarios where the information required is fragmented across multiple sources or when a single data source alone cannot provide sufficient depth or breadth of knowledge. While Agentic RAG is a key feature of LLM agents, it represents just one of many functionalities that can be integrated into these LLM agents. By combining dynamic retrieval with advanced processing and analytical capabilities, LLM-based agents transform from static models into versatile systems capable of tackling complex, interdisciplinary challenges in political science and beyond.

This modular design not only enhances the flexibility and adaptability of LLM agents but also streamlines the research process. The same agent can be repurposed for different research contexts by leveraging shared tools and providing task-specific instructions. Whether retrieving information, analyzing data, or synthesizing insights, these agents can significantly reduce manual effort required for scholarly work while maintaining consistency, reliability, and transparency in their outputs.

The Case for LLM Agents and the Study of Political Institutions

Theoretical models are foundational to the study of political institutions. These models aim to be a reflection of the way the world operates, with a focus on how individuals, context, and organizational structures produce outcomes. As Clarke and Primo (2012) note, models are tools that scholars strive to ensure are useful for framing lines of inquiry, grouping empirical patterns under one framework, speculating on causal mechanisms, or deriving predictions. In constructing theoretical models, scholars strive to showcase their parsimony and boundedness, yet broad applicability. It is through empirical research designs that scholars probe the implications of theoretical models on particular actors, organizational structures, and behavioral constraints. As Ashworth, Berry, and Bueno de Mesquita (2021) explain, with the right empirical design, scholars can adjudicate whether theoretical models are similar to what they aim to reflect or whether the implications of those models should be elaborated on, reinterpreted, or disentangled to better describe relevant mechanisms. To more thoroughly assess a theoretical model, we want to gather data on as many of its observable implications as possible. This involves collecting data across a variety of contexts. The more instances in which a model's implications are consistent with empirical realities, the more useful that model is (King, Keohane, and Verba 1994).

For scholars of political science, we argue that LLM agents—intelligent systems equipped with dynamic retrieval capabilities like Agentic RAG, as well as tools for processing and analyzing data—are key to addressing the challenges of empirical research. These agents go beyond being mere search engines; they act as comprehensive research assistants (Lu et al. 2024), capable of automating and streamlining tasks that are otherwise time-intensive and complex. Scholars can leverage these agents to efficiently identify relevant case examples to test theoretical expectations, uncover overlooked variables to address research questions, and locate or preprocess data sources for measuring key concepts.

The study of political institutions often involves working with context-specific and evolving data, making traditional methods both arduous and resource-intensive. For instance, identifying relevant historical case studies or compiling datasets often requires specialized knowledge and manual effort, such as reading and coding archival materials or synthesizing disparate data sources. LLM agents alleviate these burdens by integrating information retrieval, preprocessing, and synthesis into a cohesive, iterative workflow. They can dynamically select the appropriate tools and data sources, refine their outputs, and adapt their methods to the specific demands of the task at hand.

This adaptability makes LLM agents particularly valuable for political science research, where the upstream tasks of data collection and preparation often require repeated effort for different empirical designs or research contexts. By automating these processes, agents not only improve efficiency but also enable scholars to revisit existing theories and replicate foundational steps of data measurement and analysis with greater scalability and precision. Importantly, LLM agents are not limited to retrieval; their modular design allows them to be equipped with additional functionalities, such as data transformation, classification, and even statistical modeling, making them indispensable tools for advancing the examination of the robustness and generalizability of theoretical models with empirical research designs.

In the following section we describe how LLM agents can be useful for those conducting research on the U.S. Congress. We then apply our own LLM agent—which we call `CongressRA`—to the work of Binder (1999) examining the dynamics of legislative gridlock. Binder questions the measure of legislative productivity used by Mayhew (2005) to argue that incentives of individual members—in combination with prevailing political conditions—means that the enactment of important legislation is always possible, regardless of whether there is unified or divided government. Binder reevaluates this theoretical argument by suggesting that to measure gridlock in the U.S. Congress, all

possible policies that could have been enacted, not just those that were enacted, needed to be factored into a measure of legislative gridlock. There are several steps required to create such a measure. First, Binder identifies all policy issues mentioned in unsigned editorials appearing in the *New York Times* between 1947 and 1996. This required a team of research assistants to read nearly 15,000 editorials. Second, Binder identifies proposed bills related to each of these policy issues using a variety of sources from *Congressional Quarterly Almanac* to the Library of Congress's website. Lastly, Binder identifies whether these bills were enacted into law. From policy issue identification to bill legislative history acquisition, this process involves information retrieval from a range of sources. By enabling an LLM to interact with external data sources, we show that scholars can accurately and efficiently perform such a data collection and preparation.

Designing LLM Agents

Definition and Core Components of an LLM Agent

An LLM agent is a system that combines a Large Language Model (LLM) with predefined tools, defined as well-defined functions with input-output arguments, enabling the LLM to autonomously execute those tools and functions based on its understanding of the context of a user's prompt. These tools are implemented as callable functions with clearly defined inputs and outputs, which an LLM can interpret and utilize effectively. Once integrated, an LLM gains the ability to determine when and how to use these tools, selecting appropriate functions and parameters to address specific problems.[1]

Implementing such agents can be achieved through frameworks like OpenAI's

1. To enhance controllability and effectiveness for specific use cases, fine-tuning is often required. By providing explicit system prompts or guidelines that instruct the agent on how to call functions in various scenarios, a user can help the agent better understand the context of a problem and identify the most relevant functions. This process—known as few-shot learning or in-context learning—improves the agent's performance by augmenting its capabilities with successful examples or strategies.

Assistant API or open-source solutions such as LangChain. These frameworks simplify the development of LLM agents by providing built-in mechanisms for defining functions, executing them, transmitting output, and handling the re-execution of functions when necessary. We recommend that those in our field wishing to develop an LLM agent for their purposes rely on these frameworks. This will allow their focus to be on key decisions, such as selecting the most suitable LLM, identifying which function tools to integrate, and providing the necessary code for the agent's functionality.

Design Principles for Creating LLM Agent Functions

When designing an LLM agent, one of the most critical tasks is selecting which functions to integrate. These functions—which can range from querying data sources to generating descriptive statistics—determine the agent's capabilities and how effectively it can address research tasks. The following principles provide a structured approach to designing and selecting functions:

- **Modularity**: Functions should be designed as standalone components to enable their seamless integration, removal, or updating, thus ensuring the overall system remains adaptable as research needs evolve over time.
- **Reusability Across Research Contexts**: Prioritize functions that are broadly applicable across multiple projects within a specific area of research (e.g., functions for semantic search, database querying, and generating descriptive statistics).
- **Synergistic Workflow**: The strength of an LLM agent is its ability to combine multiple functions to address complex research questions. For instance, an agent with both data retrieval and statistical analysis capabilities can dynamically retrieve data and immediately analyze it within a single workflow.

Retrieval-Augmented Generation (RAG) as a Functional Capability

Retrieval-Augmented Generation (RAG) is a cornerstone capability for LLM agents, enabling them to access external knowledge bases and augment their responses with up-to-date and relevant information. Implementing RAG as a functional tool involves careful planning of both data structure and querying methods, which is largely dependent on the nature of the data and specific research task at hand. Below are three key types of RAG functionality tailored to institutional research in political science:

1. **Semantic Search with Vector Databases**: Text-based data, such as news articles or legislative summaries, which cannot be effectively retrieved through traditional filtering or keyword-based querying, can be stored as vector embeddings (e.g., OpenAI embeddings). These embeddings capture the semantic meaning of the data, enabling an LLM agent to retrieve entries that are contextually relevant to a user's query, even when exact matches are not present.[2]

2. **Structured Data Retrieval with SQL Databases**: For tasks requiring exact, structured outputs, relational databases are more appropriate. Important data sources from a researcher's field of study that are often available in tabular form should be stored in SQL databases, for example, to allow for easy retrieval.[3]

3. **Entity Relationships with Graph Databases**: Data depicting relationships between entities (e.g., connections between bills, lobbying groups, sponsors, and committee memberships) can be stored in graph databases in the form of nodes, edges, and properties.[4] By storing network-based data in this manner, researchers are

2. For example, in the study on legislative gridlock by Binder (1999), semantic search could replace manual keyword searches used to locate relevant *New York Times* articles discussing policy issues. This would require storing article details in a vector database, but allow an agent to identify broader themes or debates in public discourse that would otherwise require extensive manual review.

3. For example, if researchers need to identify members of the Senate Banking Committee during the 108th Congress, an SQL database can store committee membership information in a tabular format. The agent can dynamically generate SQL queries to retrieve precise information.

4. For instance, an agent could use Cypher queries in Neo4j to analyze which legislators sponsor bills associated with specific lobbying groups and identify patterns of influence within Congress.

equipped to easily map relationships and conduct network analyses.

Guidelines for Using RAG and Custom Functions in Research Workflows

LLM agents are highly versatile systems that extend beyond their core Retrieval-Augmented Generation (RAG) capabilities. By incorporating additional custom functions, researchers can tailor agents to address a wide variety of research tasks. This flexibility allows agents to not only retrieve data but also transform, analyze, and synthesize it within an integrated workflow. Because of their reasoning capacity, LLM agents can dynamically chain function calls together and accomplish multi-step tasks without user intervention.

While the capabilities of LLM agents are extensive, their effectiveness depends on thoughtful design and usage. Below are two guidelines to maximize the robustness and reliability of LLM agents:

- **Limit Query Complexity for Better Iteration**: Agents often have constraints on the number of function calls they can make within a single query. For example, OpenAI's function call limits typically allow up to five iterations per query. Complex tasks that require extensive iterations to complete may exceed these limits. Instead of overloading a single query, design workflows that split tasks into smaller, manageable subtasks.

- **Design Agents for Specific Subtasks**: For robustness, structure agents to handle one row or unit of a task at a time. Then, coordinate multiple calls to the agent programmatically using external tools like R or Python, varying the prompts so that each prompt targets a specific subtask. This approach allows the agent to work more robustly without becoming overloaded, ensuring reliability and scalability.

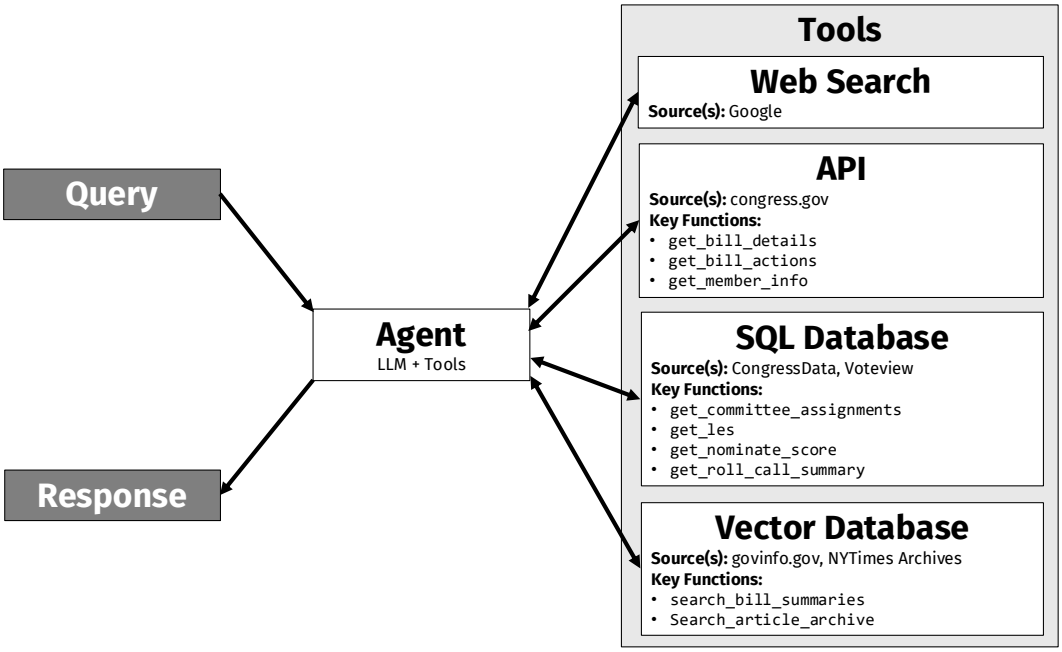

Figure 1: **CongressRA, an LLM agent for the study of the U.S. Congress**. Figure outlines the four main sets of tools—along with the related external data sources—available to our LLM agent: web search, API access, SQL database querying, and vector database querying.

Example LLM Agent: `CongressRA`

To showcase the potential of LLM agents in advancing research on political institutions, we present the design of `CongressRA`, a specialized agent tailored for Congressional research. Based on the design principles discussed earlier, `CongressRA` integrates an array of functions that interact with external data sources, retrieve structured and unstructured information, and process this data to address complex research questions.

Two questions guided our design. First, which data sources do scholars of the U.S. Congress rely on across most of the questions they aim to answer with empirical research designs? Second, from these data sources, what information do scholars typically extract? Answers to these questions helped frame the way we made certain data sources accessible to our LLM agent and the input and output of each pre-defined action we set up for our LLM agent.

While `CongressRA` is still under development, its basic architecture is outlined in

Figure 1. A more complete description of the data sources `CongressRA` interacts with and its pre-defined functions are described in Appendix A.

The data sources `CongressRA` interacts with are ones regularly used among scholars in the field and aim to provide the latest information on both members and legislative action within the U.S. Congress. To ensure our LLM agent can retrieve information about individual bills and members, we implemented three functions that interact with the `congress.gov` API.[5] We chose to have our LLM agent acquire data on bills and members via API calls because reliance on static CSV files from commonly used sources opens up the possibility of retrieving outdated information.

Because the `congress.gov` API does not provide comprehensive coverage across all data points scholars of Congress would want, we then obtain CSV files from two data sources: `CongressData` (M. Grossmann et al. 2022) and `Voteview` (J. B. Lewis et al. 2024). We store these data in a SQL database that is remotely accessible to ensure they can be queried as needed through the pre-defined functions we have implemented for the LLM agent. While the capabilities of `CongressRA` are still under development, these pre-defined functions currently include the ability to retrieve committee assignments, legislative effectiveness scores, and DW-NOMINATE scores for individual members.[6] Additionally, we defined functions that query both vote summary and complete roll vote records from `VoteView` for an individual bill.

Lastly, we store text summaries of bills introduced in Congress and articles from the *New York Times* Archive API in a vector database and allow our LLM agent to perform semantic searches with these data. In doing so, we can use human-language queries to retrieve bills and articles rather than relying on keyword searches, thereby increasing

5. Specifically, we define functions to call the `GET /bill/:congress/:billType/:billNumber`, `GET /bill/:congress/:billType/:billNumber/actions`, and `GET /member/:bioguideId` endpoints. These enable the LLM agent to retrieve details related to individual bills typically required by researchers, a bill's legislative history, and basic demographic information about individuals.

6. These data all retrieved from M. Grossmann et al. (2022). Committee assignment data are originally collected by Stewart and Woon (2017). Legislative effectiveness scores (LES) were originally calculated by Volden and Wiseman (2014). DW-NOMINATE scores were originally made available by J. B. Lewis et al. (2024).

the ability to return contextually-relevant results.

By integrating these functions together, `CongressRA` offers a powerful tool for addressing research questions. By integrating semantic search (via vector databases) and precise querying (via SQL databases), the agent ensures both breadth and depth in its data retrieval capabilities. By leveraging multiple data sources, `CongressRA` effectively transforms labor-intensive research workflows into cohesive unit tasks. These design choices are consistent with the broader guidelines we have outlined in this paper for the successful use of Agentic RAG. In particular, our LLM agent interacts with data sources that are relevant for all those who study Congress and relies on pre-defined functions that are intended for specific subtasks and can be iterated on using external tools.

In its current form, `CongressRA` is able to assist researchers with a variety of questions that are essential to study of Congress, including: (1) tracing the legislative history of individual bills including their sponsors, any roll call votes taken, and how members of different types engaged in the legislative process; (2) assessing the level of media coverage associated with particular bills or policy debates; or (3) building a full dataset of member-level covariates from demographics to legislative participation. In the next section, we show additional pre-defined actions can be added to calculate a measure of legislative gridlock similar to that of Binder (1999).

Application: Measuring Legislative Gridlock

In this section, we replicate and extend the measure of legislative gridlock introduced by Binder (1999). Binder's measure captures the proportion of salient policy issues that Congress fails to address through legislation. This is a tweak to the measure used by Mayhew (2005) to capture legislative productivity as a function of the number of important bills enacted into law. Constructing Binder's measure involves several labor-intensive steps: (1) identify all salient policy issues by examining unsigned editorials appearing in the *New York Times*; (2) identify proposed bills related to each policy

issues; and (3) determine whether these bills were enacted into law. Using `CongressRA`, we replicate this measure for the year 2023, showcasing the agent's ability to automate and streamline the data collection and analysis process.

To identify salient policy issues, we utilize the *New York Times* Article Search API to collect all articles (not just editorials) published in 2023. To enable `CongressRA` to search through these articles, we convert their headlines and summaries into Open AI embeddings and store them in a vector database. This enables our LLM agent to search across more than 50,000 articles that were available in 2023 using human-language queries rather than keyword searches. To extract articles related to the legislative process in the U.S. Congress and cluster them into common topics, we provided the agent with the following multi-step instruction:

> "Find news articles about policy issues related to the legislative process in the U.S. Congress, and then cluster them into common topics."

This instruction required the agent to perform multiple actions sequentially: (1) find relevant articles and (2) cluster or classify them by topic. To complete this process, the agent begins by transforming the query into a vector embedding to search the vector database for relevant articles. The agent then reads the entries returned from the vector database and organizes them into clusters based on common themes and topics within their content. Notably, this step does not involve unsupervised machine learning algorithms. Overall, this process follows a common flow in Agentic RAG where an agents runs functions, obtains outputs, and generates answers based on those outputs.

After processing, the agent identifies 13 distinct policy issues: "Debt Ceiling and Government Shutdowns", "Immigration Policy and Border Security", "Regulation of Artificial Intelligence (A.I.)", "Abortion Rights and Legislation", "Defense Policy and Military Funding", "Investigations and Oversight", "Voting Rights and Election Laws", "Transgender Rights and Legislation", "Supreme Court Ethics and Judicial Reforms",

"Gun Control Legislation", "Health Care Legislation", "Surveillance Laws and Privacy", and "Climate and Energy Policy". For each cluster, the agent provides a summary and lists the associated articles.[7]

To find the bills introduced in the 118th Congress during 2023 that correspond to each policy issue, we store bill data within a vector database with summaries converted into OpenAI embeddings to enable semantic similarity searches. For each policy issue, we then instruct `CongressRA`:

> For the policy issue identified in the given articles, find all bills introduced in the 118th Congress that are relevant to this issue. Use semantic similarity to match the policy issue description and article summaries to bill summaries, ensuring that only bills from the 118th Congress are considered.

Because the agent already identified distinct policy issues and had a record of the outputs—including the articles and summaries—it understood which policy issue the query referred to and performed the semantic search. In return, the agent retrieved a list of bills for each policy issue, ranked in descending order of cosine similarity scores.[8]

The last step for collecting the necessary to measure legislative gridlock is to determine whether any of these bills were enacted into law. To do so, `CongressRA` autonomously calls the `get_bill_status` function to access a SQL database containing legislative history information. For each bill, the agent used the bill ID (comprising the Congress number, bill type, and bill number) to query the database and retrieve the bill's status. The agent automatically performs this function multiple times to check the enactment status of all the bills related to each policy issue. By efficiently iterating through the list of retrieved bills, the agent compiles the legislative outcomes needed for our analysis.

Based on the agent's findings and our manual review, among the 13 policy issues

7. See Appendix A.2 for an example of what this output looks like.
8. See Appendix A.3 for an example of what this output looks like.

identified, 5 had at least one corresponding bill enacted into law. These issues included: "Debt Ceiling and Government Shutdowns", "Defense Policy and Military Funding", "Investigations and Oversight", "Health Care Legislation", and "Surveillance Laws and Privacy". Following Binder's methodology, the legislative gridlock score is calculated as the proportion of salient policy issues that were not addressed through enacted legislation. In our case, with 13 policy issues identified and 5 having at least one enacted bill, the gridlock score for 2023 is 0.615 (8/13). This indicates that approximately 61.5% of salient policy issues were not addressed by Congress through enacted legislation in 2023.

While our approach is not identical to Binder's original method, it follows the same spirit by identifying salient policy issues and assessing legislative action on them. However, there are some notable differences. First, Binder's original method focused on unsigned editorials to capture policy salience, whereas we included all articles, potentially capturing a broader set of issues. Additionally, our analysis is limited to a single year (2023), serving as a proof of concept rather than a comprehensive replication. One key advantage of using `CongressRA` is the scalability and efficiency it offers. Binder's process required a team of researchers to manually read and code thousands of articles, whereas our method automates much of this process, enabling rapid analysis and the potential for extending the measure across multiple years with minimal additional effort.

In using an LLM agent like `CongressRA` for data collection, processing, and measurement like we do here, there are a few key technical points to consider. First, there will still be points in which researchers need to make substantive decisions. When performing semantic searches using vector embeddings, it is crucial to set appropriate thresholds for similarity scores to ensure relevant results. Cosine similarity scores range from -1 to 1, with higher scores indicating greater similarity. In our bill search, the agent retrieved bills in descending order of similarity scores. However, we found that the

agent did not always set satisfactory thresholds to filter out unrelated bills. To maintain the accuracy of the measure, we manually reviewed the lists of retrieved bills for each policy issue and set appropriate thresholds to include only those bills directly related to the policy area.

Second, commercial APIs used to implement LLMs often have token[9] limits for inputs and outputs. When developers set up predefined actions for the agent to take, they must do so in a way that accounts for these limits. In our initial attempt to process articles retrieved from the *New York Times*, the agent encountered a token limit error due to the large number of articles (over 550) it was trying to process. Recognizing the error, the agent autonomously adjusted its parameters by reducing the number of articles retrieved (i.e., lowering the *top K* value) to 330 articles, which could be processed without exceeding token limits.

Third, agents should be equipped with error-handling logic to execute functions effectively. In designing `CongressRA`, we ensured the agent could identify which part of the process caused an error by analyzing error logs and either automatically adjusting its parameters or explaining the issue. This self-correcting mechanism is a notable advantage of the recent surge in LLM agents across various domains, making them more flexible and less error-prone. The ability to perform multi-step actions and manage errors autonomously highlights the potential of LLM agents in research tasks.

Overall, this application of `CongressRA` illustrates how LLM agents can replicate complex data collection and analysis tasks in political science research. The agent's ability to self-correct and handle errors autonomously not only enhances the robustness of the process but also exemplifies the recent advancements in LLM agents across various domains. By automating the identification of salient policy issues, retrieval of relevant legislation, and assessment of legislative outcomes, we can efficiently measure

9. A token is a unit of text used by language models for processing and generation. Tokens can be as short as a single character or as long as an entire word or phrase, depending on the tokenization scheme. Token limits refer to the maximum number of tokens that can be processed in a single input or output by the model.

legislative gridlock. While our method is not an exact replication of Binder's, it demonstrates a scalable and efficient approach that can be extended over time and adapted to various research contexts.

Discussion

As scholars of political institutions, the data we work with is specific to the context we study. As theories are developed and then tested with empirical research designs, we collect data on the individuals and the structures we study. Yet, the theories we develop are intended to be generalizable. While these theories aim to be generalizable, their utility often hinges on their ability to adapt to new contexts and remain robust when confronted with additional variables. It is through the iterative process of collecting and analyzing data that we evaluate the transportability of these models and refine their applicability to broader political phenomena.

An impetus for working with these data, however, is that these datasets are frequently maintained across disparate sources, some of which are outdated or inconsistently structured, making integration labor-intensive and prone to error. While LLMs have entered the research toolkit primarily for content generation and text analysis, this paper has argued and demonstrated that their potential extends far beyond these functions. When enhanced with pre-defined functions and integrated with domain-specific data sources, LLM agents are uniquely positioned to assist researchers in the complex and time-intensive tasks of data collection, preprocessing, and analysis. The ability of `CongressRA`—our example LLM agent—to mirror the process of constructing Binder (1999)'s legislative gridlock index with a fraction of the effort traditionally required highlights the efficiency gains these tools can offer. By automating data collection processes and integrating diverse data sources, `CongressRA` demonstrated that LLM agents could extend the temporal and substantive scope of foundational studies in a

scalable manner.

The utility of LLM agents, however, depends on thoughtful design and implementation. Developers must make deliberate decisions about both the structure of the data and the actions the agent will perform. Two critical considerations emerge for developers aiming to create tools that benefit a broad community of scholars.

First, data accessibility must be prioritized. Researchers rely on timely, structured, and contextually relevant data, and the way data is stored and retrieved significantly impacts the agent's utility. For instance, `CongressRA` demonstrated how SQL databases can efficiently handle structured, tabular data, while vector databases excel in semantic search for unstructured text. The ability to integrate APIs and conduct web searches further ensures that LLM agents can access the most up-to-date and comprehensive information available. This flexibility is essential for navigating the diverse and dynamic data landscapes typical of political science research.

Second, to best assist researchers in all phases of the empirical process—data collection, processing, and analysis—developers should be intentional in creating pre-defined functions that are parsimonious, targeted to specific subtasks, can be iterated on, and most importantly, reusable across research contexts. For `CongressRA`, we implemented functions with broad use cases that retrieve specific important data points as well as specialized functions to process data and produce measurements. This modular approach not only streamlines the development process but also ensures that agents can evolve with changing research needs, maintaining their relevance over time.

To be clear, successfully creating an LLM agent that can assist scholars sharing common research interests is technically-intensive. The integration of LLM agents into political science research, however, represents a transformative opportunity to address long-standing challenges in the field. As the field embraces these technologies, fostering a community-driven approach to the development and refinement of LLM agents will be critical in ensuring their impact.

References

An, Zhiyu, Xianzhong Ding, Yen-Chun Fu, Cheng-Chung Chu, Yan Li, and Wan Du. 2024. *Golden-Retriever: High-Fidelity Agentic Retrieval Augmented Generation for Industrial Knowledge Base.* https://arxiv.org/abs/2408.00798.

Ashworth, Scott, Christopher R. Berry, and Ethan Bueno de Mesquita. 2021. *Theory and Credibility: Integrating Theoretical and Empirical Social Science.* Princeton: Princeton University Press.

Bail, Christopher A. 2024. "Can Generative AI improve social science?" *Proceedings of the National Academy of Sciences* 121, no. 21 (May): e2314021121. https://pnas.org/doi/10.1073/pnas.2314021121.

Baker, Zachary R., and Zarif L. Azher. 2024. *Simulating The U.S. Senate: An LLM-Driven Agent Approach to Modeling Legislative Behavior and Bipartisanship.* https://arxiv.org/abs/2406.18702.

Binder, Sarah A. 1999. "The Dynamics of Legislative Gridlock, 1947–96." *American Political Science Review* 93 (3): 519–533.

Binz, Marcel, and Eric Schulz. 2023. "Using cognitive psychology to understand GPT-3." *Proceedings of the National Academy of Sciences* 120, no. 6 (February): e2218523120. https://pnas.org/doi/10.1073/pnas.2218523120.

Burger, Bastian, Dominik K. Kanbach, Sascha Kraus, Matthias Breier, and Vincenzo Corvello. 2023. "On the use of AI-based tools like ChatGPT to support management research." *European Journal of Innovation Management* 26, no. 7 (December): 233–241. https://www.emerald.com/insight/content/doi/10.1108/EJIM-02-2023-0156/full/html.

Clarke, Kevin A., and David M. Primo. 2012. *A Model Discipline: Political Science and the Logic of Representations.* Oxford: Oxford University Press.

Dai, Sunhao, Chen Xu, Shicheng Xu, Liang Pang, Zhenhua Dong, and Jun Xu. 2024. "Bias and Unfairness in Information Retrieval Systems: New Challenges in the LLM Era." In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 6437–6447. Barcelona Spain: ACM, August. https://dl.acm.org/doi/10.1145/3637528.3671458.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.* https://arxiv.org/abs/1810.04805.

Dillion, Danica, Niket Tandon, Yuling Gu, and Kurt Gray. 2023. "Can AI language models replace human participants?" *Trends in Cognitive Sciences* 27 (7): 597–600.

Feng, Shangbin, Chan Young Park, Yuhan Liu, and Yulia Tsvetkov. 2023. *From Pretraining Data to Language Models to Downstream Tasks: Tracking the Trails of Political Biases Leading to Unfair NLP Models.* https://arxiv.org/abs/2305.08283.

Finn, Peter, Lauren C. Bell, Amy Tatum, and Caroline Victoria Leicht. 2024. "Assessing ChatGPT as a tool for research on US state and territory politics." *Political Studies Review* (July). https://eprints.soton.ac.uk/492639/.

Gatt, Albert, and Emiel Krahmer. 2018. "Survey of the State of the Art in Natural Language Generation: Core tasks, applications and evaluation." *Journal of Artificial Intelligence Research* 61 (January): 65–170. https://jair.org/index.php/jair/article/view/11173.

Gilardi, Fabrizio, Meysam Alizadeh, and Maël Kubli. 2023. "ChatGPT outperforms crowd workers for text-annotation tasks." *Proceedings of the National Academy of Sciences* 120, no. 30 (July): e2305016120. https://pnas.org/doi/10.1073/pnas.2305016120.

Grossmann, Igor, Matthew Feinberg, Dawn C. Parker, Nicholas A. Christakis, Philip E. Tetlock, and William A. Cunningham. 2023. "AI and the transformation of social science research." *Science* 380, no. 6650 (June): 1108–1109. https://www.science.org/doi/10.1126/science.adi1778.

Grossmann, Matt, Caleb Lucas, Josh McCrain, and Ian Ostrander. 2022. *CongressData.* East Lansing, MI.

Gupta, Shailja, Rajesh Ranjan, and Surya Narayan Singh. 2024. *A Comprehensive Survey of Retrieval-Augmented Generation (RAG): Evolution, Current Landscape and Future Directions.* https://arxiv.org/abs/2410.12837.

Halterman, Andrew, and Katherine A. Keith. 2024. *Codebook LLMs: Adapting Political Science Codebooks for LLM Use and Adapting LLMs to Follow Codebooks.* https://arxiv.org/abs/2407.10747.

Heseltine, Michael, and Bernhard Clemm Von Hohenberg. 2024. "Large language models as a substitute for human experts in annotating political text." *Research & Politics* 11, no. 1 (January): 20531680241236239. http://journals.sagepub.com/doi/10.1177/20531680241236239.

Hu, Linmei, Zeyi Liu, Ziwang Zhao, Lei Hou, Liqiang Nie, and Juanzi Li. 2024. "A Survey of Knowledge Enhanced Pre-Trained Language Models." *IEEE Transactions on Knowledge and Data Engineering* 36, no. 4 (April): 1413–1430. https://ieeexplore.ieee.org/document/10234662/.

Ji, Ziwei, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. "Survey of Hallucination in Natural Language Generation." *ACM Computing Surveys* 55, no. 12 (December): 1–38. https://dl.acm.org/doi/10.1145/3571730.

King, Gary, Robert O. Keohane, and Sidney Verba. 1994. *Designing Social Inquiry: Scientific Inference in Qualitative Research.* Princeton: Princeton University Press.

Labruna, Tiziano, Jon Ander Campos, and Gorka Azkune. 2024. *When to Retrieve: Teaching LLMs to Utilize Information Retrieval Effectively,* May. http://arxiv.org/abs/2404.19705.

Lee, Kyuwon, Simone Paci, Jeongmin Park, Hye Young You, and Sylvan Zheng. 2024. "Applications of GPT in Political Science Research." May. https://hyeyoungyou.com/wp-content/uploads/2024/05/gpt_polisci.pdf.

Lewis, Jeffrey B., Keith Poole, Howard Rosenthal, Adam Boche, Aaron Rudkin, and Luke Sonnet. 2024. *Voteview: Congressional Roll-Call Votes Database.* https://voteview.com/.

Lewis, Patrick, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, et al. 2020. *Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks.* https://arxiv.org/abs/2005.11401.

Linegar, Mitchell, Rafal Kocielnik, and R. Michael Alvarez. 2023. "Large language models and political science." *Frontiers in Political Science* 5 (October): 1257092.

Liu, Menglin, and Ge Shi. 2024. *PoliPrompt: A High-Performance Cost-Effective LLM-Based Text Classification Framework for Political Science.* https://arxiv.org/abs/2409.01466.

Lu, Chris, Cong Lu, Robert Tjarko Lange, Jakob Foerster, Jeff Clune, and David Ha. 2024. *The AI Scientist: Towards Fully Automated Open-Ended Scientific Discovery,* September. http://arxiv.org/abs/2408.06292.

Mayhew, David R. 2005. *Divided We Govern: Party Control, Lawmaking, and Investigations, 1946-2002.* 2nd ed. New Haven: Yale University Press.

Mei, Qiaozhu, Yutong Xie, Walter Yuan, and Matthew O. Jackson. 2024. "A Turing test of whether AI chatbots are behaviorally similar to humans." *Proceedings of the National Academy of Sciences* 121, no. 9 (February): e2313925121. https://pnas.org/doi/10.1073/pnas.2313925121.

Motoki, Fabio, Valdemar Pinho Neto, and Victor Rodrigues. 2024. "More human than human: measuring ChatGPT political bias." *Public Choice* 198, nos. 1-2 (January): 3–23. https://link.springer.com/10.1007/s11127-023-01097-2.

Nay, John J. 2023. *Large Language Models as Corporate Lobbyists,* January. http://arxiv.org/abs/2301.01181.

Ornstein, Joseph T., Elise N. Blasingame, and Jake S. Truscott. 2024. "How to Train Your Stochastic Parrot: Large Language Models for Political Texts." July. https://joeornstein.github.io/publications/ornstein-blasingame-truscott.pdf.

Ouyang, Long, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, et al. 2022. "Training language models to follow instructions with human feedback." In *Advances in neural information processing systems,* edited by S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, 35:27730–27744. Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2022/file/b1efde53be364a73914f58805a001731-Paper-Conference.pdf.

Petroni, Fabio, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H. Miller, and Sebastian Riedel. 2019. *Language Models as Knowledge Bases?* https://arxiv.org/abs/1909.01066.

Pit, Pagnarasmey, Xingjun Ma, Mike Conway, Qingyu Chen, James Bailey, Henry Pit, Putrasmey Keo, Watey Diep, and Yu-Gang Jiang. 2024. *Whose Side Are You On? Investigating the Political Stance of Large Language Models.* https://arxiv.org/abs/2403.13840.

Radford, Alec, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. "Language models are unsupervised multitask learners." *OpenAI blog* 1 (8): 9.

Ranjan, Rajesh, Shailja Gupta, and Surya Narayan Singh. 2024. *A Comprehensive Survey of Bias in LLMs: Current Landscape and Future Directions,* September. http://arxiv.org/abs/2409.16430.

Ravuru, Chidaksh, Sagar Srinivas Sakhinana, and Venkataramana Runkana. 2024. *Agentic Retrieval-Augmented Generation for Time Series Analysis,* August. http://arxiv.org/abs/2408.14484.

Schick, Timo, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. *Toolformer: Language Models Can Teach Themselves to Use Tools.* https://arxiv.org/abs/2302.04761.

Stewart, Charles, III, and Jonathan Woon. 2017. *Congressional Committee Assignments, 103rd to 114th Congresses, 1993–2017.*

Thakur, Nandan, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. *BEIR: A Heterogenous Benchmark for Zero-shot Evaluation of Information Retrieval Models,* October. http://arxiv.org/abs/2104.08663.

Törnberg, Petter. 2024. "Large Language Models Outperform Expert Coders and Supervised Classifiers at Annotating Political Social Media Messages." *Social Science Computer Review* (September): 08944393241286471. https://journals.sagepub.com/doi/10.1177/08944393241286471.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. *Attention Is All You Need,* August. http://arxiv.org/abs/1706.03762.

Volden, Craig, and Alan Wiseman. 2014. *Legislative Effectiveness in the United States Congress: The Lawmakers.* Cambridge: Cambridge University Press.

Wu, Patrick Y., Jonathan Nagler, Joshua A. Tucker, and Solomon Messing. 2023. *Large Language Models Can Be Used to Estimate the Latent Positions of Politicians.* https://arxiv.org/abs/2303.12057.

Xi, Zhiheng, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, et al. 2023. *The Rise and Potential of Large Language Model Based Agents: A Survey.* https://arxiv.org/abs/2309.07864.

Xie, Chengxing, Canyu Chen, Feiran Jia, Ziyu Ye, Shiyang Lai, Kai Shu, Jindong Gu, et al. 2024. *Can Large Language Model Agents Simulate Human Trust Behavior?*, November. http://arxiv.org/abs/2402.04559.

Zhang, Weinan, Junwei Liao, Ning Li, and Kounianhua Du. 2024. *Agentic Information Retrieval.* https://arxiv.org/abs/2410.09713.

Zhu, Yutao, Huaying Yuan, Shuting Wang, Jiongnan Liu, Wenhan Liu, Chenlong Deng, Haonan Chen, Zheng Liu, Zhicheng Dou, and Ji-Rong Wen. 2023. *Large Language Models for Information Retrieval: A Survey.* https://arxiv.org/abs/2308.07107.

Ziems, Caleb, William Held, Omar Shaikh, Jiaao Chen, Zhehao Zhang, and Diyi Yang. 2024. "Can Large Language Models Transform Computational Social Science?" *Computational Linguistics* 50, no. 1 (March): 237–291. https://direct.mit.edu/coli/article/50/1/237/118498/Can-Large-Language-Models-Transform-Computational.

Appendix

Contents

A Design of `CongressRA`

A.1 Data Sources and Key Functions

The following outlines the primary data sources integrated into `CongressRA`, the functions implemented to interact with these sources, and specific research questions these functions are designed to address.

- **Data Source:** `congress.gov API`
  - **Purpose:** Retrieve comprehensive information about Congressional bills, their legislative histories, sponsors, and related subjects.
  - **Impact:** Questions that previously required consulting multiple datasets or manually collating information, such as analyzing legislative progress or understanding sponsor demographics, can now be addressed in minutes with these automated functions.
  - **Key Functions**:
    * `get_bill_details`: Fetches essential details about a bill, such as its title, sponsors, and introduction date.
      · **Input:** `bill_id`, a unique identifier for each bill, consisting of Congress number (e.g., 118), bill type (e.g., H.R.), and bill number (e.g., 5376).
      · **Output:** Bill title, sponsor information (e.g., bioguide IDs), introduction date.
      · **Example Query:** *"What are the key details of the Postal Service Reform Act of 2022?"*
    * `get_bill_actions`: Retrieves the legislative history of a bill, including major actions taken and their dates.
      · **Input:** `bill_id`.
      · **Output:** List of legislative actions (e.g., "Passed House on 2022-02-08").
      · **Example Query:** *"What steps were taken to pass the Affordable Care Act in Congress?"*
    * `get_member_info`: Provides biographical and Congressional role information for individual members of Congress.
      · **Input:** `bioguide_id`, a unique identifier for each member of Congress.
      · **Output:** Member name, state, chamber, leadership positions, terms served.
      · **Example Query:** *"What leadership roles has Senator Elizabeth Warren held in Congress?"*
- **Data Source:** `CongressData` (M. Grossmann et al. 2022)
  - **Purpose:** Retrieve supplementary member information, such as committee assignments and legislative effectiveness scores.

  - **Impact:** Tasks like linking committee membership to legislative productivity, previously requiring manual data cleaning and joining, are automated, enabling faster exploration of key questions about institutional influence.
  - **Key Functions**:
    * `get_committee_assignments`: Identifies the committees a member served on during a specific Congress and whether they held a chair position.
      · **Input:** `bioguide_id`, Congress.
      · **Output:** Committees served on, indicators for chair roles.
      · **Example Query:** *"Which committees did Senator John McCain chair during the 108th Congress?"*
    * `get_les`: Retrieves the Legislative Effectiveness Score (LES) for a member, reflecting their legislative productivity.
      · **Input:** `bioguide_id`, Congress.
      · **Output:** LES value.
      · **Example Query:** *"How effective was Representative Alexandria Ocasio-Cortez during the 117th Congress?"*

- **Data Source:** `Voteview` (J. B. Lewis et al. 2024)
  - **Purpose:** Retrieve NOMINATE scores for ideological analysis and detailed roll-call vote information.
  - **Impact:** Researchers analyzing voting patterns and ideological trends can automate data retrieval and focus on analysis, avoiding time-consuming manual data extraction.
  - **Key Functions:**
    * `get_nominate_score`: Fetches NOMINATE scores for ideological placement on key dimensions.
      · **Input:** `bioguide_id`, Congress.
      · **Output:** First and second dimension NOMINATE scores.
      · **Example Query:** *"How has Senator Mitt Romney's ideological position shifted over his Congressional career?"*
    * `get_roll_call_summary`: Provides summaries of roll-call votes, including yea/nay counts and vote outcomes.
      · **Input:** `bill_id`.
      · **Output:** Roll number, yea/nay counts, vote result.
      · **Example Query:** *"What were the vote totals for the Infrastructure Investment and Jobs Act?"*

- **Data Source: Vector and SQL Databases**
  - **Purpose:** Enhance Retrieval-Augmented Generation (RAG) with tailored querying capabilities.

- **Impact:** Semantic and structured retrieval capabilities enable the agent to handle diverse query types, from finding thematically related text to retrieving precise, structured data.
- **Key Functionalities**:
  - **Vector Database for Semantic Search:** Suitable for unstructured data. By embedding text into vectors, the agent can retrieve semantically similar content efficiently. As an example, we create a vector database with summaries of articles published in the *New York Times* and summaries of legislative bills introduced in the U.S. Congress.
    - **Example Query:** *"Find editorials in the New York Times discussing Congressional health care reform from 2010. Then find bills introduced in Congress related to this news coverage"*
  - **SQL Database for Structured Queries:** We store data from the `CongressData` and `Voteview` in a SQL database. The functions mentioned above run the relevant queries to extract the desired data. This is ideal for retrieving exact information, such as committee memberships or roll-call vote records.
    - **Example Query:** *"Who were the members of the Senate Banking Committee during the 117th Congress?"*

A.2 Example of Policy Issue Identification

After `CongressRA` identifies distinct policy issues from articles published in the *New York Times*, it provides information like the following for each cluster.

Regulation of Artificial Intelligence (A.I.)

Articles (10):

- *I'm a Congressman Who Codes. A.I. Freaks Me Out.*
- *As A.I. Booms, Lawmakers Struggle to Understand the Technology*
- *Schumer Lays Out Process to Tackle A.I., Without Endorsing Specific Plans*
- *In U.S., Regulating A.I. Is in Its 'Early Days'*
- *2 Senators Propose Bipartisan Framework for A.I. Laws*
- *The U.S. Regulates Cars, Radio and TV. When Will It Regulate A.I.?*
- *Congress Has a 'Zombie' Problem, and A.I. Is to Blame*
- *Lawmakers and Former Officials Press for Answers on U.F.O.s*
- *Senators Seek to Address A.I. Risks with New Legislation*
- *Schumer's A.I. Insight Forums Aim to Educate Lawmakers*

Summary:

These articles highlight the growing concern among lawmakers about the rapid advancement of artificial intelligence and the need for regulation. Members of Congress acknowledge a steep learning curve in understanding A.I. technologies. Bipartisan efforts are underway to establish frameworks and legislation to address issues such as ethics, privacy, security, and the impact of A.I. on jobs and society. The challenge lies in crafting effective regulations without stifling innovation.

A.3 Example of Relevant Legislation Identification

For the policy issue identified in the given articles, find all bills introduced in the 118th Congress that are relevant to this issue. Use semantic similarity to match the policy issue description and article summaries to bill summaries, ensuring that only bills from the 118th Congress are considered.

- **ID: 118-hres-66** — *Expressing support for Congress to focus on artificial intelligence.*
  - **Cosine Similarity Score:** 0.636
  - **Summary:** This resolution supports a congressional focus on artificial intelligence (AI) to ensure (1) it is developed in a safe and ethical manner that respects the rights and privacy of Americans, and (2) a wide distribution of AI benefits and minimization of risks.
- **ID: 118-hr-4223** — *National AI Commission Act*
  - **Cosine Similarity Score:** 0.608
  - **Summary:** This bill establishes a National Artificial Intelligence (AI) Commission in the legislative branch. The purpose of the commission is to support the role of the United States in mitigating the risks of AI through appropriate regulation while also supporting U.S. innovation and opportunities with respect to AI.
- **ID: 118-s-1356** — *ASSESS AI Act*
  - **Cosine Similarity Score:** 0.603
  - **Summary:** This bill directs the President to appoint a task force to assess the privacy, civil rights, and civil liberties implications of artificial intelligence (AI).