D-RAG: A Privacy-Preserving Framework for Decentralized RAG Using Blockchain

Tessa E Andersen*, Ayanna Marie Avalos*, Gaby G. Dagher, and Min Long

Department of Computer Science, Boise State University,
Brigham Young University, California State University, Fresno

Abstract. Retrieval Augmented Generation (RAG) has been a recent improvement in providing recent and accurate data to Large Language Models (LLMs). Although RAG has been successful in reducing hallucinations within LLMs, it remains susceptible to inaccurate and maliciously manipulated data. In this paper, we present Distributed-RAG (D-RAG), a novel blockchain-based framework designed to increase the integrity of the RAG system. D-RAG addresses the risks of malicious data by replacing the RAG's traditionally centralized database with communities, each consisting of a database and a permissioned blockchain. The communities are based on different subjects, each containing experts in the field who verify data through a privacy-preserving consensus protocol before it is added to the database. A Retrieval Blockchain is also designed to communicate between the multiple communities. The miners on this Retrieval Blockchain are responsible for retrieving documents from the database for each query and ranking them using an LLM. These rankings are agreed upon, and the top ranked documents are provided to the LLM with the query to generate a response. We perform experiments on our proposed D-RAG framework, and our results show that our Retrieval Blockchain is scalable and our privacy-preserving consensus protocol maintains efficiency as community members increase. These results demonstrate that in a real-world application setting D-RAG is scalable in maintaining data integrity.

Keywords: Blockchain, RAG, LLM, Privacy-Preserving.

1 Introduction

As technology continues to advance, there has been a noticeable improvement in modern Artificial Intelligent (AI) systems using Large Language Models (LLMs). LLMs have shown remarkable advancement in natural language processing by leveraging the computationally complex process of training neural networks with extremely large corpuses of text. With their much more realistic and conversational, human-like text output, LLMs have contributed to advancements in healthcare, education, financial services, and are continuing to improve various aspects of human life [1].

Although LLMs have demonstrated numerous improvements, they still have certain limitations. In order for LLMs to produce high-quality responses, they must be trained on enormous amounts of data, which takes significant time and resources. LLMs are unable to provide up-to-date responses (for example, to respond with information on a current event) without being retrained, unfortunately, this is computationally expensive and impractical to do on a daily basis. This has led to the development of Retrieval Augmented Generation (RAG). RAG pulls data directly from external, up-to-date databases to provide more recent and relevant information. As RAG becomes more prevalent among AI systems, it helps LLMs overcome limitations of its training to become more accurate.

While RAG has been successful in providing recent and relevant data to LLMs, little has been done to ensure that RAG is secure and trustworthy [2]. Our work is motivated by the increasing reliance on RAG within AI systems and the need to ensure the trustworthiness

* These authors contributed equally.

David C. Wyld et al. (Eds): SIPO, BDHI, NET, SOEA, CSML, AISCA, MLIOB, DNLP – 2025
pp. 183-198, 2025.CS & IT - CSCP 2025 DOI: 10.5121/csit.2025.150417

of the data being supplied to LLMs. RAG systems rely heavily on databases sourced from the internet, leaving it vulnerable to untrustworthy and biased data. This underscores a pressing challenge within RAG systems: maintaining the integrity and quality of data within the RAG database.

Introduced in 2020 by [3], RAG is a relatively recent advancement for AI, resulting in limited research on the integrity of RAG. Current strategies for improving RAG often do not focus on data integrity itself. These methods usually focus on updating the retrieval system of RAG to avoid retrieving irrelevant documents for queries. These solutions are designed to avoid noisy documents and retrieve higher relevant documents for the query, but are still not sufficient in protecting against malicious or unsafe data. In addition to improving the retrieval system, as in previous works [4][5], our framework improves the integrity and quality of the data used for RAG, therefore, increasing the trustworthiness of AI.

Data integrity is a problem that has become highly prevalent within the internet and AI systems. RAG's process of data retrieval from the internet can lead to biases, misinformation, and unreliable data. Attacks formulated against RAG have proven to be successful in altering an LLM's output, causing biases or harmful behavior [6][7]. Now, we pose the question of what is protecting the integrity and quality of the data that RAG uses.

In this paper, we strive to ensure the trustworthiness of the data used by RAG systems in order to improve the integrity of LLMs. To achieve this, we propose D-RAG (Distributed-RAG), a distributed blockchain framework that uses a privacy-preserving consensus protocol to create new databases for RAG. D-RAG consists of a permissioned public blockchain integrated into each database. These databases, referred to as communities, are divided based on certain subjects and only allow specialized users onto the blockchain associated with it. In these communities data is proposed and reviewed by community members before it is added to the RAG database, therefore improving the data integrity over current RAG systems. We also incorporate a public permissionless Retrieval Blockchain. The Retrieval Blockchain focuses on retrieving and ranking the documents for RAG to improve the relevance of the data given for the query. This framework increases the security of the RAG system and protects it against unsafe data.

1.1 Contributions

The key contributions of this paper are as follows:

1. D-RAG : We introduce a novel framework, named D-RAG, for building a distributed retrieval augmented generation system that ensures data integrity. This framework establishes communities of specialized and verified knowledge in order to facilitate generating an accurate response from multiple sources for any given user query.
2. We propose Privacy-Preserving Knowledge Incorporation (PPKI), a privacy-preserving consensus protocol that ensures the integrity of proposed data to be added to a community database. To ensure privacy and avoid biases, our consensus protocol uses zero-knowledge proofs to protect the identity of the proposer as well as the decision of the approving members, emulating a double-blind peer review. In these communities data is proposed and reviewed by community members before it is added to the database, therefore improving the data integrity over current internet based RAG systems.
3. A Retrieval Blockchain is used to retrieve and rank the documents for each query. It consists of a user submitting a transaction whenever there is a query. Then, quorum

members use the query to retrieve the relevant documents from community databases. After the documents are retrieved, the quorum members use LLMs to rank the documents within the context of the query. Quorum members reach consensus on the highest-ranking documents, which are later used to generate a final response.

4. We implemented D-RAG and our results demonstrated the accuracy of document ranking using LLMs, the scalability of our proposed Retrieval Blockchain, and the efficiency of our double-blind mechanism in PPKI.

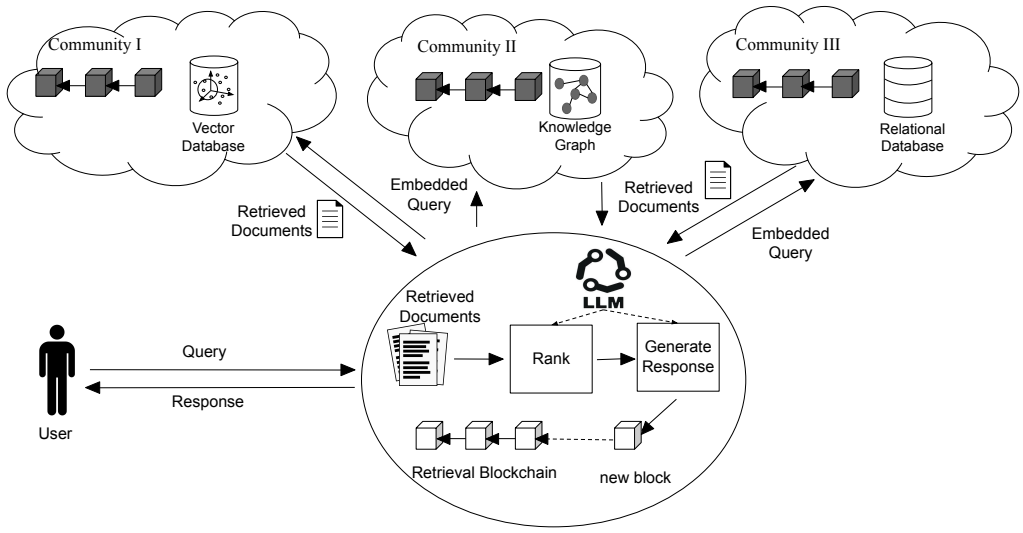

Fig. 1: Blockchain framework for D-RAG. This framework includes multiple communities each with a database and blockchain and a Retrieval Blockchain that takes a query as input and outputs a response. The interaction between the communities and Retrieval Blockchain is shown.

2 Related Work

RAG has been a useful method to improve the accuracy of LLM responses. However, it has also demonstrated its own issues that research is currently striving to address [8]. While work to decentralize RAG has received limited attention, research efforts have focused on improving the existing RAG system, specifically improving the evaluation of the retrieved documents from RAG.

Many proposed systems have utilized an LLM to improve the ranking process of the RAG system [4][5][9][10]. These systems utilize the LLM in two different approaches to improve RAG. These approaches include using an LLM to generate scores or labels for documents according to the query, or simply using an LLM to detect the noisy or irrelevant documents retrieved by RAG. Some of these systems specifically finetune or train an LLM for the processes of document evaluation, whereas some of the papers just use an existing LLM. In addition to using an LLM to help with ranking documents, Yan et al. [11] also proposed utilizing web searches for complementary knowledge sources if the LLM finds all retrieved documents to be incorrect.

These systems highlight a significant issue with RAG: the retrieval of irrelevant documents as answers to queries. While they have achieved some success in enhancing the accuracy

and trustworthiness of the documents retrieved by RAG, they do not directly address concerns regarding malicious or biased data prevalent within RAG databases. This limitation can lead to failures in their systems and underscores the need for further research to improve these systems.

Xiang et al. [2] introduced the first framework to defend against poisoning attacks, these attacks inject malicious or biased data into RAG's database. This work isolates each document retrieved and generates a response for each one and then performs a secure text aggregation over the responses. This defense guarantees that any outliers or poisoned data are detected, provided that the amount of poisoned data is less than the majority of retrieved passages. While this method showed high success rates against poisoned data, it still has some limitations. This system fails to prevent malicious data from entering the database, and if a significant amount of malicious data enters the database then this method is insufficient in preventing its retrieval.

Previous studies aimed at enhancing the current RAG system have focused on improving the retriever's ability to rank documents accurately, thereby enhancing AI accuracy. Because RAG is a recent advancement in AI, efforts to improve it have overlooked database enhancements. However, recent studies have begun addressing improvements specifically aimed at optimizing the RAG database. Work proposed by Edge et al. [12], focuses on implementing a knowledge graph that is split into communities. When knowledge is added to the database, LLMs are used to split the knowledge into chunks and to identify entities in the text, categorizing it into specific communities. Once entities are found, the LLM extracts descriptions of entities and their relations. Then, it forms communities and summaries of the communities are stored to help with retrieval. While this research does not decentralize RAG, it takes an important step forward in improving the database for RAG.

Research completed by Origin Trails, sought to decentralize the external databases used by RAG systems [13]. [13] introduced Verifiable Internet for AI, which uses a multiple blockchain framework to create a knowledge graph for RAG that stores non-AI generated data. The knowledge graph is split into communities, with each community having its own blockchain. Origin Trails focuses on non-AI generated data within the database, leaving the issues of data integrity unaddressed. Decentralizing RAGs database is the next step to make RAG more secure and trustworthy. In contrast, our D-RAG framework focuses on the integrity of the data within the database itself. We also ensure more security and trust by checking documents at the retrieval stage.

3 Preliminaries

3.1 RAG

RAG consists of an external database which is usually sourced from the internet or Wikapedia. Most RAG systems use vector databases, since it is easier to search for similarity with vector representations. The data in the RAG database is split into chunks and encoded in vector representations and stored into a vector database. When an LLM is given a query, RAG will search the database for related documents to give to the LLM. First, RAG will embed the query into vector representation so it can be used to search the documents, then it retrieves the top-k documents from its database. The documents retrieved are then ranked based on relevance to the query. The LLM then uses its pretrained or parametric knowledge and the top similar documents retrieved by RAG to formulate the final response to the original query.

3.2 Blockchain

Blockchain is an immutable decentralized digital ledger that records transactions. Within a blockchain there are 'blocks' that are cryptographically linked, each block records a transaction that has occurred. In order for transactions to occur, a required consensus between nodes must take place. This consensus is reached among the members of a blockchain without the need for any third party. The benefits of blockchain include increased security, immutability, and decentralization. Blockchain has been applied in many sectors such as health care, finance, government, and manufacturing [14].

3.3 Zero Knowledge Proofs

Zero Knowledge Proofs (ZKP) are a method used to prove that a statement is true without revealing any other information about the statement. Schnorr's protocol is an example of a sigma protocol that is used for ZKP [15]. In Schnorr's protocol, a prover will compute a random number, then calculate a value a using the random number and their public key. The prover will give a to the verifier, then the verifier will send a challenge value c which the prover uses with their secret and random value to calculate z. Once the verifier receives z from the prover, they can validate with a and z that the prover knows the secret with low probability of fraud.

3.4 Exponential ElGamal Encryption

In exponential ElGamal encryption, users use a shared public key in order to encrypt their message. The users can then collectively compute the total sum of the encrypted cipher texts, then decrypt the sum of the cipher texts. The users then perform a discrete log algorithm on the decrypted value in order to compute the value of the message [16]. Exponential ElGamal is effective for voting because it allows users to submit encrypted values of 0 or 1 and then add them homomorphically without revealing who submitted which values.

4 SOLUTION: D-RAG

4.1 Solution Overview

Our proposed system introduces a novel framework which leverages blockchain to decentralize RAG. D-RAG is designed to mitigate concerns of data integrity within RAG systems, which leads to an increase in trustworthiness of LLMs utilizing RAG.

D-RAG utilizes a multiple blockchain approach to improve the current RAG system. This involves using separate blockchains for various databases, each dedicated to specific specialized subjects, thereby creating distinct communities. Along with the communities, there is a primary blockchain or Retrieval Blockchain which retrieves documents and ranks them. This framework is illustrated in Figure 1. In this system, the blockchains need to be able to communicate with each other for the process of document retrieval. Protocols such as those proposed by [17] could be utilized for this process of inter-communication between the blockchains.

All blockchains utilized by the communities are permissioned public blockchains, meaning they only allow experts of the certain fields within the blockchain. This ensures that the data is verified and validated by experts within the field before being integrated into the system. To further increase the integrity of the system, a privacy-preserving consensus

Decentralized Response Generation Protocol

Input: Query Q, user's public key P_k, list of quorum members
Output: A new block on Retrieval Blockchain

---

1. Each miner within the quorum executes a smart contract that performs the following
   (a) Embeds Q to Q'
   (b) Performs a semantic search within each community database with Q' to retrieve top-k relevant documents, $D = \{d_1, d_2, \ldots, d_m\}$
2. Each miner within the quorum executes the following steps individually
   (a) Create a prompt to request an LLM to rank the documents. As will be shown in Section 5, we experimented with four different methods: *single-step prompt*, a *multi-query prompt*, a *summarization prompt*, and an *answer ranking prompt.*
   (b) Prompt LLM to rank retrieved documents D on the relevance to query Q.
3. The quorum members perform the following
   (a) Collectively reach consensus on the most relevant documents within D.
   (b) Prompt LLM using the most relevant documents as context in order to generate a response R for query Q.
   (c) Encrypt R with P_k and store $[\![R]\!]_{P_k}$ in a separate database.
   (d) Construct and sign a new block on the Retrieval Blockchain containing the link to $[\![R]\!]_{P_k}$ and a hash $H(R)$

Protocol 1.1: Steps of the Retrieval Blockchain takes in order to generate a response for a query

protocol called Privacy-Preserving Knowledge Incorporation (PPKI) is implemented during the verification process. This protocol uses a double-blind mechanism, in which the proposer's identity and members' votes are hidden. This mechanism mitigates the risks of biases influencing the validation of information. The communities and PPKI serve to increase the trustworthiness and security of RAG.

In this framework, the main Retrieval Blockchain is crucial for detecting malicious data that may still be within the database. This blockchain is utilized to replace the process of retrieval and document ranking within RAG. The Retrieval Blockchain is a public permissionless blockchain which uses its' consensus protocol to retrieve information from the database and rank the documents. Figure 1 shows how the Retrieval Blockchain interacts with the communities. A user submits a query and it is later embedded by members in a quorum on the Retrieval Blockchain. The embedded query is used to retrieve documents which are then ranked by an LLM. The quorum members generate a response with the top ranked documents, then a new block is created with the response returned to the original user. This system of ranking and retrieving is used to detect any outliers of malicious or biased data to further enhance the integrity of the RAG system. The Retrieval Blockchain, along with the databases, increases the trustworthiness of LLMs by using blockchain to increase the security and output of RAG.

4.2 Adversarial Model

Adversaries in the D-RAG system include (1) malicious users attempting to inject unsafe or biased data, (2) colluding participants manipulating consensus or rankings, (3) external attackers seeking unauthorized access or disruption, and (4) insider threats exploiting permissions to compromise data integrity. To counter these threats, D-RAG employs

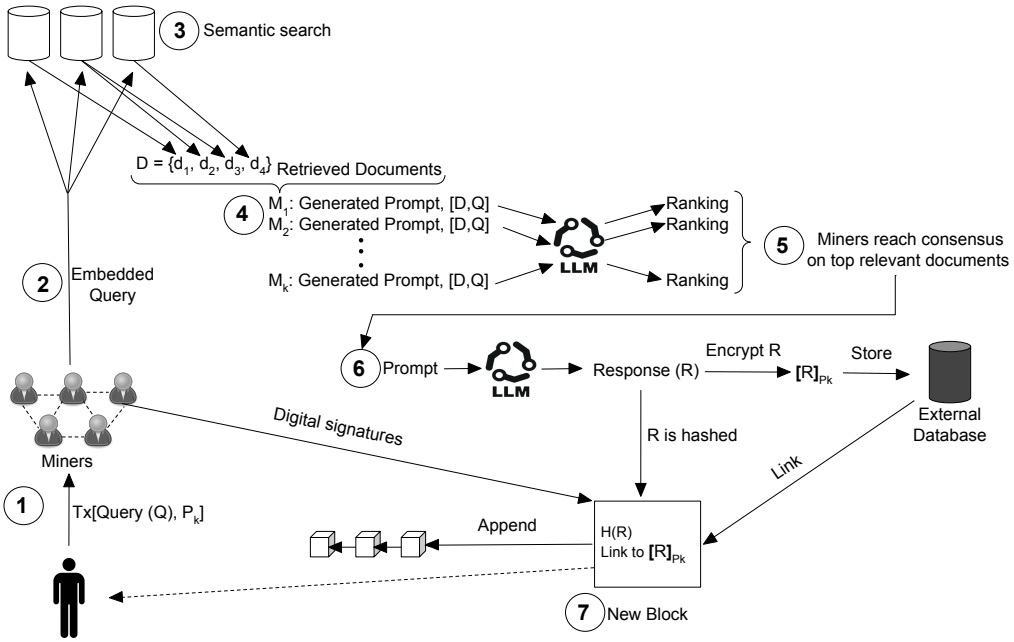

Fig. 2: The steps of the Decentralized Response Generation Protocol for the Retrieval Blockchain. Each number represents a different step in the process.

privacy-preserving protocols to prevent unsafe data injection and collusion, permissioned blockchains to restrict access and maintain data integrity, and a consensus-driven retrieval process to ensure query results are unbiased and relevant. Additionally, its distributed architecture enhances resilience against service disruptions and unauthorized tampering.

4.3 Retrieval Blockchain

The Retrieval Blockchain replaces the retriever of RAG in our proposed solution to bring more accuracy to the documents retrieved as an answer to a query. For this blockchain system, it uses a consensus protocol, or Decentralized Response Generation Protocol, to retrieve documents related to a query from the databases and then rank the retrieved documents to reach a common answer of the top-k relevant documents. The top-k relevant documents are then used as context for the answer to the query given.

In Protocol 1.1, it explains the steps of the consensus protocol for the Retrieval Blockchain and Figure 2 also illustrates these steps. It begins with a user submitting a query transaction, by inputting the query Q and their public key P_k. A quorum of 10% of the miners within the blockchain is selected for the consensus process. The selection of quorum members is a needed process for our consensus, there have been many literature works that study the selection of quorums within blockchains. Our method will use algorithms like the following to select our quorum [18][19]. Each miner in the quorum is given a smart contract which ensures the retrieved documents are the same for each miner. The smart contract directs the quorum members to embed Q and use the embedded query Q' to perform a semantic search on the D-RAG databases. The quorum members retrieve documents within a certain similarity threshold. The similarity between a query and document is determined through a cosine similarity measurement, which measures the angles between vector embeddings to find the most relevant documents. Only documents with the highest cosine similarity are retrieved.

After the quorum members retrieve the top similar documents D, they each generate their own prompt to provide an LLM with so that it can rank the documents regarding their relevance to the query. In our implementation, we tested four different methods of prompts that can be used to rank documents using Chat GPT-4o. The different types of prompts included a single-step prompt, a summarization prompt, multi-query prompt and multi-answer prompt.

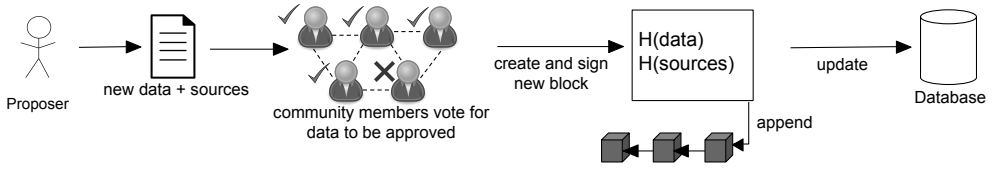

Fig. 3: The process of Privacy-Preserving Knowledge Incorporation within the communities

For the quorum members, they each design their own prompt for the LLM to rank the documents and use an LLM of their choosing. When submitting the scores of the ranking, the quorum members give the documents scores based off of where the LLM ranked them. If only 10 documents are retrieved, then the document ranked as the most relevant by the LLM will be given a score of 10, with the next document a score of 9 and so on. Once the quorum members receive their scores for the documents, they submit them to the blockchain to reach consensus. The scores for each document are added up and the documents with the highest scores are identified as the top documents.

Once consensus is achieved for the top documents, then the quorum members are given a smart contract which requires them to collectively feed the LLM a prompt, the query Q, and the top documents to generate a response to the Q. Once the response R is received, it is encrypted with the users public key P_k, which was submitted in the beginning of the transaction. Then the encrypted response $[\![R]\!]$ is stored in an external database and the quorum members construct a new block on the Retrieval Blockchain. This block contains a link to $[\![R]\!]$ on the external database, and a hash of the original response. The encryption of the response ensures privacy of the user's transaction history. The hash of the response is so that the user can ensure the response wasn't tampered with by miners on the blockchain. The user can use the link and collect the encrypted response from the external database. Then, the user can decrypt and hash the response from the database and compare it to the hashed response on the block. If the hash of the responses both match, then the user can be sure that the response is valid.

The Retrieval Blockchain ensures RAG to be more secure and trustworthy in what it outputs to LLMs. Using blockchain improves the original system of RAG through the ranking and the consensus protocol. RAG uses semantic search and cosine similarity calculations to determine the top rated document. The Retrieval Blockchain updates this by utilizing the miners on the blockchain to rank the documents using an LLM. Since the documents are ranked with various methods and the top documents are agreed upon, it ensures more integrity in the eventual response that is generated.

4.4 Communities

Within the D-RAG system, we include multiple communities which each have a database and a permissioned public blockchain integrated within them. The blockchain is there to

Privacy-Preserving Knowledge Incorporation
Input: Identity (public keys) of community members $y_1, y_2, \ldots, y_m$
Output: A new block of the proposed data

1. The proposer, who is supposed to be a community member, proposes new data and related sources.
2. The proposer also produces a proof that they belong to the community without disclosing their identity. To do that, the proposer combines Schnorr's identity protocol with Zero-Knowledge OR protocol:
$$ZK\_OR(Schnorr(y_1), Schnorr(y_2), \ldots, Schnorr(y_m))$$
3. Each community member selects a large random private key $a_i : 2 \leq i \leq p-1$ and computes public key $g^{a_i} \; mod \, p$
4. All community members collectively compute the group public key $A_i \equiv g^{\sum a_i} \; mod \, p$
5. Each member individually provides:
   (a) An encrypted vote $[\![v]\!]$ of 0 or 1 to accept or reject the block. To encrypt their vote v, they choose random value r and then encrypt their vote:
   $$[\![v]\!] = (c_1, c_2) = (g^r mod \, p, g^v \cdot A^r mod \, p)$$
   (b) A Zero-Knowledge Proof that proves [\![v]\!] encrypts either 0 or 1
   (c) A signature of their encrypted vote $\sigma_{[\![v]\!]}$
6. Each member homomorphically tally all the votes
$$[\![V]\!] = \prod_{i=1}^{m} [\![v]\!] = (A^{\sum r} \cdot g^{\sum v} mod \, p, g^{\sum r} mod \, p)$$
7. Community members jointly decrypt $[\![v]\!]$:
$$g^V = (A^{\sum r} \cdot g^{\sum v}) / \prod_{i=1}^{m} (g^{\sum r})^{a_i} = g^{\sum v} mod \, p$$
8. Apply discrete log algorithm on g^V to compute $V = v_1 + v_2 + ...v_m$
9. If V is above the agreed upon threshold (the new data is accepted):
   (a) The database is updated with the new information
   (b) A new block is created containing the hash of the new data and the hash of the corresponding resources

Protocol 1.2: Algorithm for Privacy-Preserving Knowledge Incorporation

validate the information before it is updated to the database to increase the integrity of the data. The communities are split based on different subjects, for instance, molecular biology or calculus, with the blockchains only allowing experts of the community to join. Each of these communities utilize the experts on the blockchains to validate and verify the information being proposed. While traditional RAG uses a vector database, D-RAG can use different forms of databases within the communities. Figure 1 illustrates a vector database, a knowledge graph, and a relational database as examples of different databases for the communities. This allows more flexibility for the D-RAG system rather then relying on one type of database.

While our proposed solution involves each community operating on its own blockchain, it is also feasible to implement all communities on a single blockchain. In this setup, the single blockchain would utilize smart contracts to manage community specific processes such as member registration, proposing data, and voting. Each community would function independently within this framework, with smart contracts maintaining the necessary sep-

aration between them. This approach retains the integrity of the community-based structure while simplifying the overall system architecture by consolidating operations onto a single blockchain.

4.5 Privacy-Preserving Knowledge Incorporation

Each community within the framework includes a database and a blockchain. The consensus protocol for the blockchains is Privacy-Preserving Knowledge Incorporation, this protocol is used to verify the information entering the database. Each member on the blockchain is an expert in the specific community, therefore, increasing the relevance and integrity of the data being proposed. PPKI ensures that the integrity is maintained within the databases while also protecting the privacy of the miners in the system. This protocol is shown in Figure 3 and Protocol 1.2. It begins with either a miner on the blockchain or a proposer proposing new information to be added to the database. Along with any proposed information, sources are also provided by the proposer to help verify that the information is correct.

The other nodes on the blockchain then receive the new proposed information and compare it to the provided sources to verify that the data is trustworthy and correct. While this system utilizes human opinion as the ground truth for data, it still maintains security since experts serve as validators. For information to be approved and added to the database, the information must receive 70% of approving votes from the members of the community. Once proposed information receives enough votes, it is then added to the database and a new block is created with a hash of the new information and hash of the sources associated with it.

In this consensus protocol, we also create a privacy-preserving aspect to it that uses Zero Knowledge Proofs (ZKP) and a process of voting with Exponential ElGamal encryption in order to perform a double-blind consensus. In the scientific community, there is concern of biases when scientific papers are peer reviewed. The scientific community implements a double-blind system, wherein a paper is peer reviewed but the author's identity is kept secret from the reviewer and the reviewer's identity is kept secret from the author, with the overall goal of preventing biases. In our consensus protocol, we design a similar system which the identity of the proposer and the votes of the members are hidden to mitigate the problem of biases within scientific communities. The steps of this process are outlined below.

- *ZKP for proposer*: The first part of Privacy-Preserving Knowledge Incorporation is outlined in steps 1 and 2 of Protocol 1.2. This portion uses ZKP to hide the identity of the proposer when new information is proposed. When a proposer provides new information, they use ZKP to prove they are a member of the community without revealing who they are. For this system, we will be using Schnorr's protocol with Zero-Knowledge OR. Schnorr's protocol uses the private and public key pair with an algorithm to prove that someone is a valid member without revealing who they are. Zero-knowledge OR is used to prove one statement of a set of statements is true, without revealing which one is true. Thus, the proposer's identity is proven to be true but not revealed. The ZKP is provided as part of the transaction of the proposed information so the other members know that the proposer is a member of the community.
- *Computing Group Key with Exponential ElGamal:* This part of the consensus protocol involves steps 3 and 4 in Protocol 1.2. In these steps, the members of the community

each compute their own public and private key pair. Then, they collectively work together to compute a group public key to use for encryption.

- *Providing vote, ZKP, and signature:* Once the members of the community have the group public key, they then use it to encrypt their vote of either 0 or 1, resulting in 2 cipher texts. If the member votes 1, they approve the knowledge being proposed. If a member votes 0, they do not approve it. The members then provide ZKP that their vote is either a 0 or 1 and also provide a signature of the encrypted vote with their own set of keys. Since each public key is posted with the signature, it ensures that a member within the community does not vote more than once. This part of the consensus is shown in step 5 of Protocol 1.2.
- *Homomorphically compute total sum of the votes:* In step 6 of Protocol 1.2, the members homomorphically compute the total sum of votes V using exponential ElGamal. The equation illustrates that the members compute the total sum of the random values of r and the votes v within the ElGamal equation from step 5a. At this point of the process V by itself is still unknown.
- *Decrypting and computing number of votes:* In steps 7 through 9 in Protocol 1.2, the members of the community decrypt the total sum. The members compute from step 5a in protocol 2, c_2 divided by the product of c_1. This outputs the value of $g^V mod\, p$. Once the members decrypt the votes, V by itself is still unknown to them and they need to apply the discrete log algorithm in order to calculate the total value of V. Once the members of the community calculate V, if V is at least 70% of the members in the community, then the proposed knowledge is valid. The knowledge is added to the database and a new block is created.

5 Experimental Evaluation

For our experiments, we first performed tests using Chat GPT-4o with OpenAI [20] to test the accuracy of ranking documents with different types of prompts. We also tested the scalability of the consensus protocol for the Retrieval Blockchain with different numbers of nodes. Finally, we tested the efficiency of our proposed double-blind review in Privacy-Preserving Knowledge Incorporation.

The three experiments of our proposed framework were implemented using both Python and Java, as well as libraries such as LangChain, OpenAI, BlueChain, ZKProver, and Bouncy Castle. The temperature for Chat GPT was not explicitly set, however, OpenAI's documentation suggests that the default temperature is 1, on the scale of 0 to 2 [21][22]. All experiments were conducted on an HP Spectre x360 laptop with the following specifications: 11th Gen Intel Core i5-1135G7 processor (2.40 GHz), 8 GB of RAM (7.65 GB usable), running Windows 11 Home.

For ranking documents with an LLM, we implemented our experiments in Python and used the Cranfield Collection [23], which is a dataset containing 1400 documents and 225 queries. The Cranfield collection includes information related to aerodynamics. With regards to specific queries, documents are given a relevance score of 1 to 4 with 1 being the most relevant and 4 being the least. In our assessment, we consider a relevant document to be a document with a relevance score of 1 or 2, while irrelevant documents have scores of 3 and 4. We utilized Chat GPT 4o with different prompts on all of the queries and documents.

5.1 Accuracy of document ranking with LLM

We included four different types of prompts in our experiments, all of which are designed to ask Chat GPT 4o to rank the documents. The first prompt was a single-step prompt that asked the LLM to rank the documents with relevance to the query in one prompt. The next prompt we tested was a summarization prompt, first it asked the LLM to summarize each document and then asked the LLM to rank the summarizations with relevance to the query. We also included a multi-query prompt that asked the LLM to rephrase the query into multiple queries and then rank the documents based off of the rephrased queries. The last prompt asked the LLM to generate answers with the documents and query and rank the answers based off of the query.

Our first round of experiments used four documents from the Cranfield database with

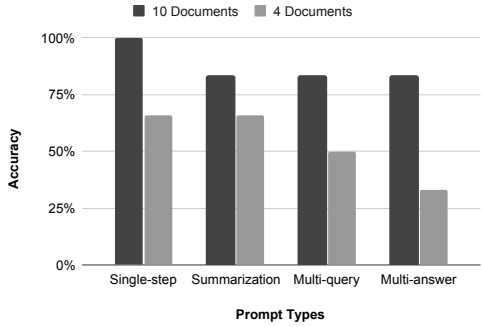

Fig. 4: The overall accuracy for the LLM ranking a relevant document as the most relevant document according to prompt and experiments

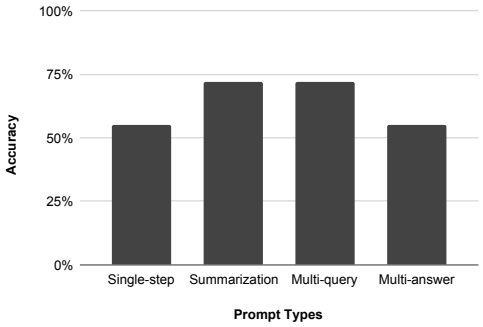

Fig. 5: The overall accuracy for the LLM ranking relevant documents in the first 3 spots of relevancy depending on the prompt used

the related query. Each document had a different relevancy score between 1 to 4 and we sampled 6 different queries for our tests. We calculated the accuracy of the LLM ranking a relevant document in the top spot of its ranking. These calculations are shown in Figure 4. We observe that the single-step and summarization prompts demonstrate the most accuracy of 66% with four documents, but are still below the accuracy from the second round of experiments. In our review of the data, with four documents the LLM performed well in ranking a document with a score of 1 in relevancy in the top spot, but the last 3 rankings varied of what relevancy score was ranked. The performance for the multi-query and multi-answer prompts decreased leading to less trust in the LLM ranking the right documents with these prompts.

For our next round of experiments, we sampled the same queries but with ten documents each instead of four. Since there were ten documents, this meant that multiple documents had the same relevancy score. Figure 4 and Figure 5 demonstrate the overview of our results. Once again, the LLM did better with ranking relevant documents in the top spots but became more confused with lower rankings. In Figure 4, it demonstrates that the single-step prompt achieved a 100% accuracy with ranking a relevant document in the first spot while the other prompts had an 83% accuracy. We also analyzed the accuracy of the ranked relevant documents in the first 3 spots. In these results, the single-step and multi-answer prompts returned an accuracy of 55% while the summarization and multi-query prompts had an accuracy of 72%. As the rankings increased, the accuracy of the

LLM decreased, but it generally ranked the relevant documents in the top 3 rankings. Figure 4 also demonstrated that the LLM performance improves when more documents are given to rank.

The results demonstrate that while Chat GPT 4o is not perfect with ranking documents, it still produces the more relevant documents in the top rankings, leading to more accuracy in the context used by an LLM for a query. The experiment overall demonstrated more accuracy in certain prompts than others. The summarization prompt was overall the most accurate while the multi-answer prompt did not perform well, it caused the LLM to become confused and even refused to rank the answers once. While the experimental results showed some general patterns for different prompts, it was not perfect since Chat GPT 4o may not be well versed on aerodynamics and may therefore fail more in ranking with this subject.

While these findings are based on a specific dataset and LLM model, they are not intended to be generalized across all datasets or models. Instead, the goal was to demonstrate the potential accuracy of LLMs in ranking documents. Further testing on different datasets may improve the results to confirm whether LLMs really can be trusted to rank relevance and detect malicious data. ChatGPT 4o was selected for this study due to its popularity and status as a state-of-the-art language model, thus, making it a relevant choice for assessing how well it performs document ranking tasks. The Cranfield Collection was chosen because it was one of the few datasets that provided queries, documents, and a specific ranking of said documents according to the query. This allowed for a direct and unbiased comparison between the LLM's rankings and the dataset's relevance judgments. This issue of establishing ground truth is discussed in greater detail in our future work section.

5.2 Scalability of the Decentralized Response Generation Protocol

Our implementation for these experiments include four different steps of the Retrieval Blockchains consensus protocol (Decentralized Response Generation Protocol). The first step involves the miners retrieving the relevant information for a query. To implement this step, we used LangChain [24][25] as the retriever. LangChain takes the query in the form of plain text, embeds it [22], and then searches for the relevant documents. We used the Cranfield collection and split the dataset into four databases to represent our framework, which uses multiple databases. Nodes in this step each retrieve one document from the four datasets to test the scalability of retrieving from multiple databases.

The next step involves ranking documents, we used OpenAI's Chat GPT versions 3.5, 4

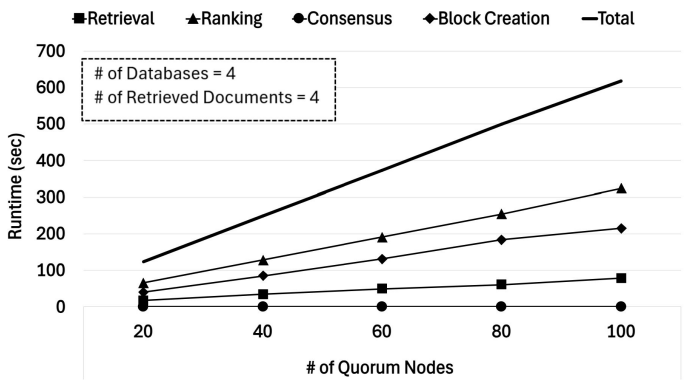

Fig. 6: Scalability of the Decentralized Response Generation Protocol

and 4o [20] for this process to mimic nodes using different LLMs. We cycled through the four different types of prompts each node could use, then using the prompts, the LLM would be asked to rank the doucments. Next, we implemented the consensus process for the document ranking, this involved nodes assigning each document with a score based on the document's ranking. Consensus is reached by summing all the scores and returning the document that has the highest score. The last step is block creation for the Retrieval Blockchain. For this step, we built upon BlueChain [26]. Our block creation included a total of 150 nodes with different quorum sizes. It is assumed that the response to the query has already been generated. The purpose of this implementation is to measure the runtime, starting when a transaction is submitted, forming a block, and ending when all the nodes in the network have added this block. At each step, we implemented with 20, 40, 60, 80, and 100 nodes, then summed all the components to calculate the total runtime for this protocol. Our findings show the scalability of our architecture and offer insightful information about components that account for the largest portion of the total runtime.

We observe from Figure 6 that the four components are indeed scalable, as the total runtime shows a clear linear increase as the nodes in the quorum increase linearly. We also observe that the ranking step dominates all the other components, accounting for roughly half of the total runtime. The retrieval process shows the second highest runtime due to the constraint of each node having to retrieve documents from four different datasets. These results calculate the runtime for each node performing each process one at a time. While the ranking process dominates the runtime calculations, we acknowledge that in a real world setting that multiple nodes would retrieve and rank their documents at the same time, thus taking less time.

The total runtime is shown to start at a total of 2 minutes with 20 nodes and increase to a total of 5 minutes with 100 nodes. The total runtime provides many insights on improvements that could be made by increasing the efficiency of the protocol's process. This efficiency could be increased through changes made in the retrieval and ranking process, this will be critical for increasing the usability of our proposed RAG system.

5.3 Efficiency of the Privacy-Preserving Knowledge Incorporation Protocol

This implementation tests the efficiency of the double-blind component of Privacy-Preserving Knowledge Incorporation. Our experiment tested the efficiency of two steps in this process. The first step, the proposition, occurs when the proposer provides a ZKP as part of their transaction for new proposed knowledge. The second step, the verification, is the process of the members voting and encrypting their vote using Exponential ElGamal and ZKP.

For the proposition, the proposer must provide a ZKP. We implemented Schnorr's protocol within a Zero-Knowledge OR. This process begins by simulating Schnorr's protocol which entails generating one secret key and then creating a corresponding public key. We then generate $m - 1$ random public keys for $m - 1$ nodes with m representing the total amount of nodes. With this setup in place, the implementation then proceeds to construct m Schnorr proofs. Since there is only one valid pair of matching secret and public keys, only one of these proofs will be identified as true. As long as one of the Schnorr's proof returns true, the single Zero-Knowledge OR will always return true.

The next portion included implementing the votes and requires ZKPs provided by the voting members of the community. For this process, we implemented exponential ElGamal which is shown in steps 3 through 8 in Protocol 2. We utilize exponential ElGamal in order to encrypt the votes. Then, we used a ZKP to prove that the vote provided was

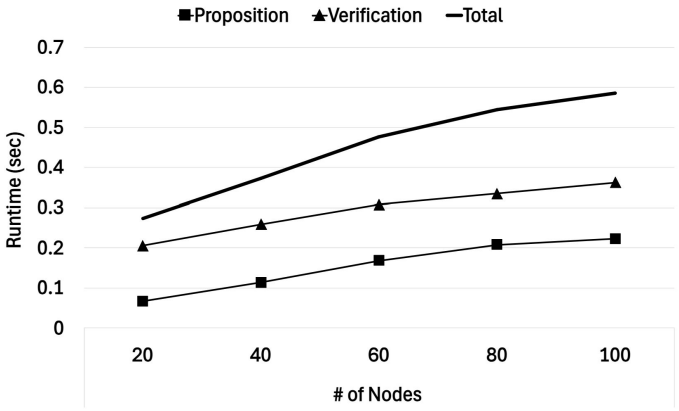

Fig. 7: Efficiency of Privacy-Preserving Knowledge Incorporation protocol

either a 0 or 1. To provide the ZKP for the vote we took the values of c_1, c_2, g, p, A_i, and r as input. The prover solves for c_2 assuming the vote is either 0 or 1, if it succeeds then the prover returns true. In each of these steps, we tested efficiency of the protocol with 20, 40, 60, 80, and 100 nodes.

We observe in Figure 7 that the total runtime stays under 0.6 seconds for the total amount of nodes that were tested on. The proposer providing a Zero-Knowledge Proof is shown to have a linear increase as the number of nodes increase, taking the time between 0.1 to 0.2 seconds. The runtime for the voting members of the community shows a complete linear increase over time, with the process only taking about 0.2 to 0.3 seconds. Our results demonstrate the efficiency and feasibility of the double-blind mechanism within our consensus protocol for the varying amount of nodes.

6 Conclusion and Future Work

In this work, we introduce D-RAG, a novel blockchain based framework designed to increase the integrity of RAG systems. Our solution addresses the challenges of data integrity within RAG systems by leveraging blockchain to increase the trust of data utilized by LLMs. Our proposed framework incorporates communities that use a consensus protocol, PPKI, to verify data before it is introduced into the RAG database. Additionally, we include a double-blind mechanism in PPKI that conceals the identity of the proposer and decisions of the approving members. Our proposed Retrieval Blockchain is used to detect malicious data retrieved by the RAG system. Our testing with document ranking has demonstrated that LLMs are effective at ranking relevant documents for queries, therefore, increasing the trust of our proposed system. The experimentation on our framework demonstrates the scalability of our proposed Retrieval Blockchain and the efficiency of the double-blind mechanism in PPKI.

For future research, there are several key areas that can help improve the integrity of RAG systems. One of the most prominent, is that of ground truth, finding a more robust and secure way to determine ground truth in data that can help prevent malicious data being used. Another possible area of future work could involve training an LLM for the specific purpose of ranking the documents retrieved by a RAG system. This could vastly improve the experiments that were performed with OpenAI's Chat-GPT models. Other future directions could focus on improving the retrieval process in this proposed framework, ensuring it is efficient and robust. Our work on the RAG system is an important step forward in the research for more secure and trustworthy AI.

References

1. F. D. Felice, A. Petrillo, C. D. Luca, and I. Baffo. Artificial Intelligence or Augmented Intelligence? Impact on our lives, rights and ethics. In Procedia Computer Science, pp. 1846-1856, 2022.
2. C. Xiang, T. Wu, Z. Zhong, D. Wagner, D. Chen, and P. Mittal. Certifiably Robust RAG against Retrieval Corruption. In arXiv, 2024.
3. P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W. Yih, T. Rocktäschel, S. Riedel, and D. Kiela. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. 2020.
4. K. Kim and J.-Y. Lee. RE-RAG: Improving Open-Domain QA Performance and Interpretability with Relevance Estimator in Retrieval-Augmented Generation. In arXiv, 2024.
5. O. Yoran, T. Wolfson, O. Ram, and J. Berant. Making Retrieval-Augmented Language Models Robust to Irrelevant Context. In arXiv, 2024.
6. H. Chaudhari, G. Severi, J. Abascal, M. Jagielski, C. A. Choquette-Choo, M. Nasr, C. Nita-Rotaru, and A. Oprea. Phantom: General Trigger Attacks on Retrieval Augmented Language Generation. In arXiv, 2024.
7. W. Zou, R. Geng, B. Wang, and J. Jia. Knowledge Poisoning Attacks to Retrieval-Augmented Generation of Large Language Models. In arXiv, 2024.
8. S. Setty, K. Jijo, E. Chung, and N. Vidra. Improving retrieval for rag based question answering models on financial documents. In arXiv, 2024.
9. T. Zhang, S. G. Patil, N. Jain, S. Shen, M. Zaharia, I. Stoica, and J. E. Gonzalez. RAFT: Adapting Language Model to Domain Specific RAG. In arXiv, 2024.
10. A. Asai, Z. Wu, Y. Wang, A. Sil, and H. Hajishirzi. Self-RAG: Learning to Retrieve, Generate, and Critique through Self-Reflection. In arXiv, 2023.
11. S.-Q. Yan, J.-C. Gu, Y. Zhu, and Z.-H. Ling. Corrective Retrieval Augmented Generation. In arXiv, 2024.
12. D. Edge, H. Trinh, N. Cheng, J. Bradley, A. Chao, A. Mody, S. Truitt, and J. Larson. From Local to Global: A Graph RAG Approach to Query-Focused Summarization. In arXiv, 2024.
13. Origin Core Developers. Verifiable Internet for Artificial Intelligence: The Convergence of Crypto, Internet and AI. 2023.
14. O. Ali, A. Jardat, A. Kulaki, and A. Abuhalimeh. A Comparative Study: Blockchain Technology Utilization Benefits, Challenges and Functionalities. IEEE Access, Vol 9, p. 12730–12749, 2021.
15. J. Holmes and G. G. Dagher. Variance: Secure two-party protocol for efficient asset comparison in bitcoin. In IEEE International Conference on Blockchain, p. 62–71, 2020.
16. T. Elgamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In IEEE TRANSACTIONS ON INFORMATION THEORY, 1985.
17. J. Kwon and E. Buchman. Cosmos whitepaper. A Netw. Distrib. Ledgers, vol. 27, pp. 1–32, 2019.
18. G. D. Bashar, J. Holmes, and G. G. Dagher. "Accord:A scalable multileader consensus protocol for healthcare blockchain". In IEEE Transactions on Information Forensics and Security, vol. 17, p. 2990–3005, 2022.
19. Z. Zhang, Y. Rao, H. Xiao, X. Xiao, and Y. Yang. Proof of quality: A costless paradigm for trustless generative ai model inference on blockchains. 2024.
20. OpenAI, "Chatgpt," OpenAI, 2024.
21. T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, Language models are few-shot learners. 2020.
22. OpenAI, "Embeddings," OpenAI, 2024
23. E. A. Fox and D. Harman. Cranfield collection. University of Glasglow, 2022.
24. LangChain, "Get started with langsmith", LangChain, 2024.
25. L. Martin. "rag-from-scratch", 2024. [Online]. Available: https://github.com/langchain-ai/rag-from-scratch/blob/main/rag from scratch 1 to 4.ipynb
26. P. Lundquist and G. G. Dagher, "Bluechain," ISPM Research Lab, 2023. [Online]. Available: https://www.boisestate.edu/bluechain/