Internet Technology Letters

WILEY

LETTER OPEN ACCESS

Blockchain and NoSQL: Enhancing Throughput via a Deep Learning-Based Hybrid Architecture

Longo Antonello | Di Palma Antonio | Casamassima Luca | Fiore Marco | Mongiello Marina

Department of Electrical and Information Engineering, Polytechnic University of Bari, Bari, Italy

Correspondence: Longo Antonello (a.longo70@phd.poliba.it)

Received: 16 September 2024 | **Revised:** 15 October 2024 | **Accepted:** 19 October 2024

Keywords: architecture | Blockchain | deep learning | hybrid | NoSQL

ABSTRACT

Any data structure used to store information can be considered a database. Blockchain technology, at its core, is no more than a ledger to store information about transactions. In recent years, such technology has gained immense popularity in a wide range of applications. However, scalability and throughput are major challenges of the Blockchain technology since applications and APIs have much lower throughput compared to non-Blockchain ones. Several studies showed that the integration of the NoSQL paradigm along with a Blockchain pipeline enhances the overall throughput and scalability of the system, as well as the possibility of handling both on-chain and off-chain data. The goal of this paper is therefore to study how the advantages of both technologies can be capitalized on to enhance the system capabilities and to finally put forward a novel hybrid architecture that mixes NoSQL and Blockchain characteristics using Deep Learning techniques.

1 | Introduction

Blockchain is a peer-to-peer technology for developing distributed ledgers in which data can be stored in thousands of different servers and retrieved almost instantly. At its core, Blockchain is an immutable log of cryptographically signed transactions replicated and managed in a decentralized fashion through distributed consensus among a set of untrusted parties. It is particularly useful in different scenarios: from finance [1] to agri-food traceability [2], from smart cities [3] to e-voting systems [4]. The main advantages of Blockchain regard data immutability, transparency, and the possibility to execute transactions between two parties in an untrusted environment. However, scalability and throughput are considered major challenges when it comes to the use of this technology [5]. Centralized databases can achieve better results than Blockchain in terms of speed and latency. This work puts forward a novel hybrid architecture making use of both Blockchain and NoSQL characteristics to enhance the system's performance. In hybrid architectures that combine Blockchain and NoSQL databases, it is essential to ensure that the system operates efficiently while maintaining its core properties of decentralization, immutability, and performance. As shown in [6], runtime verification techniques can be employed to continuously monitor and validate the system's performance and compliance with functional requirements in dynamic environments. This is particularly relevant in scenarios where data storage decisions are made in real-time, as in the proposed hybrid system, which classifies data as sensitive or non-sensitive and stores it accordingly.

The presented idea is to ease the load of the Blockchain, thus reducing application bottlenecks, by using a parallel NoSQL database to host all data that do not require immutability constraints. To do so, a Deep Learning (DL) based pipeline that automatically decides whether data needs to be stored in the Blockchain ledger or in the NoSQL database is introduced. This process relies only on data structure and content, making the system context-agnostic.

Abbreviations: IoT, Internet of Things; DL, Deep Learning; DLP, Data Loss Prevention.

1.1 | Problem Statement

We focus on the coexistence of both a public Blockchain, in which any person can join and have equal right to read or write information in the platform, and a NoSQL database. Blockchain technology can guarantee a secure and robust workflow, and grant immutability to sensitive information. In contrast to all the above-listed benefits, Blockchain technology's throughput is much lower when compared to non-Blockchain technologies [7].

On one hand, NoSQL platforms are very often used by enterprises looking for distributed approaches and ways to reduce downtime of their internal systems. On the other hand, several industries choose to adopt Blockchain in their existing database structure because of its data immutability feature. This leads to the creation of hybrid architectures leveraging the advantages of both technologies. In such case, two database layers are used: (i) the first layer uses a lightweight-distributed consensus protocol that ensures some integrity level while providing good performance for querying, (ii) the second layer uses a Proof of Work-based Blockchain to store evidence of the first layer's database operations.

Using NoSQL in a Blockchain application allows to handle different types of data, namely on-chain data coming from transactions in the Blockchain and off-chain data related to the Blockchain but not stored directly within the blocks (NoSQL can provide additional security and privacy to off-chain data).

2 | Background and Related Work

In this section we review the state-of-the-art techniques to enhance the Blockchain throughput.

Authors of papers [8, 9] investigate performance bottlenecks in Blockchain and present an efficient high-performance system for caching the Blockchain data in the FPGA network interface controller (NIC) with the aim of improving scalability and throughput of Blockchain applications.

In [10], authors propose a Blockchain relational database involving all features provided by existing Blockchain platforms and with better support for complex data types. After separating the concerns of concurrent transaction execution and decentralized ordering of blocks of transactions, they make use of serializable snapshot isolation (SSI) to concurrently execute transactions and validate each transaction in a block to obtain a serializable order that will be identical across all untrusted nodes.

Authors of paper [11] explain that hybrid Blockchain databases are decentralized systems that have three key components: (i) a shared database that uses a storage engine, (ii) a shared ledger replicated via a consensus mechanism, and (iii) a user interface that usually supports a simple key-value store. They also review all the existing frameworks already implementing hybrid architectures.

Different tools and online systems can be also taken into consideration:

- **BigchainDB** [12] is a decentralized and immutable data storage system which leverages MongoDB as storage engine and Tendermint for consensus among the nodes. BigchainDB introduces two optimizations: (i) Blockchain pipelining allows nodes to vote for a new block while the current block of the ledger is still undecided, and (ii) parallel validation on multiple CPU cores.
- **BlockchainDB** implements a distributed database on top of a classic Blockchain. It is different from the other systems because it partitions the database into few shards so that the overall storage overhead is reduced.
- **Couchbase** allows to implement both of the data handling components usually used in Hyperledger systems. The two handling components are: (i) operational transactions that verifies, creates and logs all transactions and (ii) world state, another data view that maintains the current account values.

The state-of-the-art review reveals that hybrid architectures mixing Blockchain properties (e.g., decentralization, immutability, owner-controlled assets) and database properties (e.g., high transaction rate, low latency, indexing, querying of structured data) are a valuable solution to overcome limitations of the Blockchain technology. However, to the best of the authors' knowledge, no existing framework is able to perform data management in a completely context-agnostic manner, thus deciding at runtime where to store data in an automatic way.

3 | System Description

The goal of this work is to design a hybrid architecture able to (i) automatically manage data in a distributed heterogeneous database and (ii) handle data belonging to different user types in a context-agnostic and unstructured manner. Such system allows greater data security, without losing in performance speed.

3.1 | System Architecture

This section explains how the overall architecture is organized, the roles of each component, and how the different parts are interconnected to create the workflow. The whole system consists of a three-layered architecture.

- **Data layer**

  Relies on a Firebase NoSQL database and an Ethereum Blockchain system. Firebase's key-value paradigm simplifies the process of working with unstructured data: each incoming information can be formatted into a key-value format. Moreover, to enter personal data into an Ethereum Blockchain, ad-hoc smart contracts are typically needed. If all data is represented using the same structure, a single general-purpose contract can be used instead.

- **Back-end layer**

  Holds the whole logic, ranging from basic functions allowing user-system interaction to the logic leading to how the system reacts to a specific data request. The content parser is the core element of the system because it bridges the gap between the logic layer and the data layer. Given a

set of information from a user, the content parser decides whether the set of information should be stored either in the NoSQL database (fast transaction but not immutable) or in the online Blockchain (slow transaction but immutable). To do so, the idea is to classify users' data as sensitive or non-sensitive. Nevertheless, training a DL model to perform such task would require a large dataset comprising samples from different use cases to account for generalization, but no appropriate dataset is currently available.

Data loss prevention (DLP) services are essential for ensuring the protection of sensitive data within organizations, adhering to privacy and trust policies. DLP solutions offer more granular control over sensitive data, allowing organizations to monitor and protect data in real-time. Various solutions can be analyzed in this topic. For instance, Forcepoint DLP offers advanced protection with features like data discovery, classification, and monitoring, but it can be complex to configure and manage. Spin, on the other hand, might not offer the same depth of protection and could have limitations in terms of scalability, such as the ability to handle a large volume of data and limited integration with other systems. Google Cloud DLP is designed to discover, classify, and protect sensitive data across various environments, including databases, text-based content, and images. It offers advanced de-identification techniques such as masking and tokenization, which help in maintaining data privacy while ensuring compliance with regulations like GDPR. Additionally, Google Cloud DLP provides real-time data protection and seamless integration within the Google ecosystem, making it easier to manage and enforce security policies. For these reasons, Google Cloud DLP has been chosen as reference model.

The DLP model comes with pretrained detectors that are specific for a given data type. An infoType is a type of sensitive data (e.g., name, email address, credit card number) that comes with its corresponding detector. For each detector, a minimum likelihood score can be fine-tuned. Although Google already provides more than a hundred detectors, the capabilities of the detection model can be further improved by defining custom detectors. The Google DLP model is therefore instantiated with a list of info types to define what to scan for and to load the corresponding detectors. During inference, the model parses input data and each detector returns all sensitive information. By slightly changing the approach, the Google DLP framework can therefore be exploited to solve the classification problem between sensitive data requiring immutability constraints and non-sensitive data. In the proposed architecture, all the information flows through the Content Parser, which relies on the Google DLP engine. If the DLP module detects some sensitive records needing to be masked, data is redirected toward the Blockchain ledger to guarantee immutability. If the parser does not detect sensitive data, all the information is stored in the NoSQL database.

- **Front-end layer**

By interacting with the remote server, the web application provides the users with all the data related functionalities: data storage, retrieval, and visualization. The realization of a web application allows to achieve portability, making the system independent from all sorts of platforms.

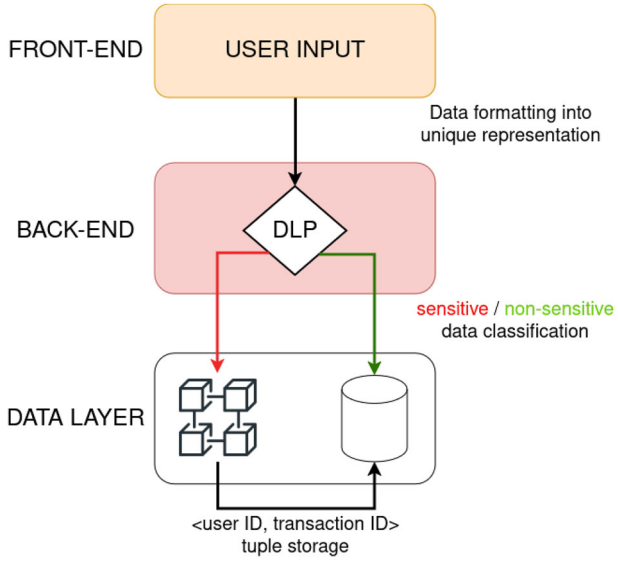

FIGURE 1 | Hybrid system layered architecture and information flow.

3.2 | Information Flow

Figure 1 shows the whole information flow within the system.

The DLP module analyzes the information and looks for sensitive data. At this point, if no critical record is found, the flow is redirected toward the NoSQL database. If sensitive data is detected, an online transaction is created in the Blockchain first and a tuple linking user ID and transaction ID is finally stored in the NoSQL database to keep track of every user's operation.

3.3 | Information Retrieval

To speed-up basic operations, the core application logic (login, registration, etc.) runs on top of the NoSQL database, where all user-related information (IDs, passwords, etc.) is stored. The same happens to non-sensitive data that is organized and retrieved via the user's unique ID. To retrieve user-specific data from the Blockchain, all user's transactions IDs are first gathered from the NoSQL database (via user's ID) and then used to query the Blockchain system.

4 | Experimental Results

To evaluate the performance of the proposed system architecture, a fully working prototype has been realized. The Python Flask library has been used to develop the front-end web application, as it allows to easily manage key-value structured data while rapidly develop and test the server logic without needing to deploy the server on a public infrastructure. The Ganache framework is used to build up the Blockchain architecture, while an online Firebase DB has been created to host the NoSQL data.

In this preliminary stage, the objective is to gather data about the performance of the overall system in order to assess the validity of the idea. The implemented architecture works therefore in an

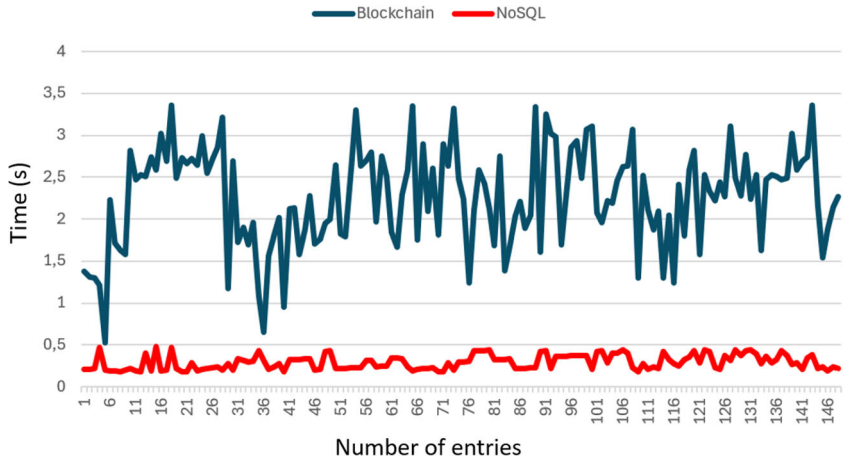

FIGURE 2 | Time comparison (in seconds) during data storing activities.

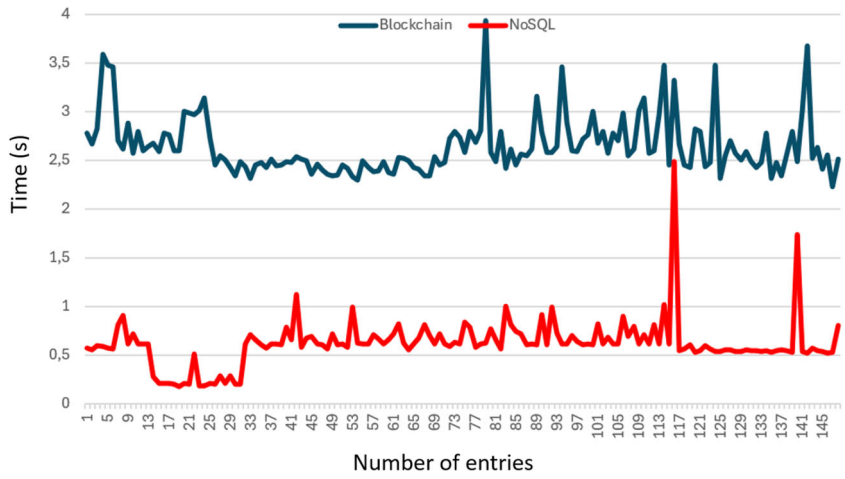

FIGURE 3 | Time comparison (in seconds) when performing data retrieving.

offline setting. To show the capabilities of the framework, four use cases have been targeted: (i) sensors' data from IoT networks, (ii) medical records, (iii) bank transactions, (iv) school records from different universities.

The goals are: (i) to check whether the Google DLP engine is able to perform the above-mentioned data classification by capturing sensitive information and (ii) to measure the data management time difference between the NoSQL and the Blockchain systems. To carry out the first test, fac-simile data adherent to real-world structures were submitted to the prototype application. No specific dataset has been used in this stage. A custom dataset has been realized instead by generating, for each scenario, 200 samples. For instance, in the case of a bank transaction, the application requires to insert the bank account number of both the payer and the payee. Starting from the structure of a classical bank account number, the tool is then able to generate real-world data with plausible bank account numbers, while fake records contain random strings. The same happens for Social Security Numbers and so on. The generated dataset has finally been used to test the DLP module.

The DLP basic module allows to handle the selected case scenarios because it already provides built-in detectors for all the sensitive information potentially linked to each case. Even though no additional training has been performed on top of the model, some custom detectors based on regular expressions have been developed to extend the model detection capabilities. Before deploying the application, the likelihood threshold of each detector has been fine-tuned so as to reduce the number of both false positives and missed detections.

All the tests carried out show that Google DLP is able to perform the task of discriminating sensitive and non-sensitive information, thus redirecting the first toward the Blockchain and saving the latter in the NoSQL database. To ensure the validity of the tests, some constraints were fixed for each use case (e.g., a doctor was asked to insert the Social Security Number of the patient when saving a medical record or the bank coordinates had to be specified for each transaction).

The capabilities of the platform have been extended to automatically measure and store the time needed to carry out each task. This study particularly focuses on comparing the speed of both

TABLE 1 | Retrieving and saving time comparison.

	Retrieving time			Saving time		
	Max (s)	**Min (s)**	**Avg (s)**	**Max (s)**	**Min (s)**	**Avg (s)**
Blockchain	3.93	2.23	2.63	3.36	0.53	2.14
NoSQL	2.48	0.18	0.47	0.48	0.18	0.25

architectures during data storage and data retrieving. These execution times are therefore measured while a user is interacting with the platform and saved in the NoSQL database for further analysis.

Figure 2 shows the performance comparison when new data is being permanently saved. Due to the creation of an immutable transaction in the ledger, the Blockchain average memorization time is much higher when compared to that of the NoSQL.

In Figure 3, the time required to retrieve data is compared. Retrieving data directly from a NoSQL database is faster due to the direct access nature of key-value stores, where the data is retrieved using a unique key. On the other hand, Blockchain retrieval is a multi-step process that involves first obtaining the transaction ID from the NoSQL database, then using this ID to query the Blockchain. Although this process is slower, it offers the added advantage of transparency and traceability, which are crucial in environments where data integrity and auditability are essential. Table 1 compares saving and retrieving times in both approaches.

5 | Conclusion and Future Works

In conclusion, results show that the proposed architecture successfully enhances the throughput of a Blockchain-based application. While state-of-the-art architectures enhance the performance speed by using parallel and distributed data sources, the proposed architecture achieves the objective by saving only data actually requiring immutability constraints in the distributed ledger. This is done by leveraging a parallel NoSQL database and Blockchain platform, and a DL-based decision pipeline to manage all data.

The DL system relies on Google DLP model to perform a binary classification between sensitive and non-sensitive information. After classification, critical data is stored in the Blockchain to satisfy immutability constraints, while the NoSQL hosts all non-sensitive information.

Future works will investigate the performance comparison when the system is deployed on an online Blockchain architecture. In this setting, an even greater gap between times is expected, further motivating the choice to use a hybrid system.

Conflicts of Interest

The authors declare no conflicts of interest.

Data Availability Statement

Data sharing is not applicable to this article as no new data were created or analyzed in this study.

Peer Review

The peer review history for this article is available at https://www.webofscience.com/api/gateway/wos/peer-review/10.1002/itl2.604.

References

1. H. Wu, Q. Yao, Z. Liu, et al., "Blockchain for Finance: A Survey," *IET Blockchain* 4 (2024): 101–123.

2. M. Fiore, M. Frem, M. Mongiello, et al., "Blockchain-Based Food Traceability in Apulian Marketplace: Improving Sustainable Agri-Food Consumers Perception and Trust," *Internet Technology Letters* (2024): e503, https://doi.org/10.1002/itl2.503.

3. ARS Khan, HJ Syed, J Shuja, A Khan, F Qamar, QN Nguyen. *Advancing Secure and Efficient Sensor Data Integration in Smart Cities: A Blockchain-Based Approach* (2024).

4. S. Tanwar, N. Gupta, P. Kumar, and Y. C. Hu, "Implementation of Blockchain-Based e-Voting System," *Multimedia Tools and Applications* 83, no. 1 (2024): 1449–1480.

5. M. Fiore and M. Mongiello, "Blockchain Technology to Support Agri-Food Supply Chains: A Comprehensive Review," *IEEE Access* 11 (2023): 75311–75324, https://doi.org/10.1109/ACCESS.2023.3296849.

6. M. Mongiello, P. Pelliccione, and M. Sciancalepore, "AC-Contract: Run-Time Verification of Context-Aware Applications," *In: 2015 IEEE/ACM 10th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*, Florence, Italy (New York, NY: ACM, 2015), 24–34.

7. A. A. Monrat, O. Schelén, and K. Andersson, "A Survey of Blockchain From the Perspectives of Applications, Challenges, and Opportunities," *IEEE Access* 7 (2019): 117134–117151, https://doi.org/10.1109/ACCESS.2019.2936094.

8. A. I. Sanka and R. C. Cheung, "Efficient High Performance FPGA Based NoSQL Caching System for Blockchain Scalability and Throughput Improvement," in *2018 26th International Conference on Systems Engineering (ICSEng)* (New York, NY: IEEE, 2018), 1–8.

9. A. I. Sanka, M. H. Chowdhury, and R. C. Cheung, "Efficient High-Performance FPGA-Redis Hybrid NoSQL Caching System for Blockchain Scalability," *Computer Communications* 169 (2021): 81–91.

10. S. Nathan, C. Govindarajan, A. Saraf, M. Sethi, and P. Jayachandran, "Blockchain meets database: Design and Implementation of a Blockchain Relational Database," 2019 arXiv preprint arXiv:1903.01919.

11. Z. Ge, D. Loghin, B. C. Ooi, P. Ruan, and T. Wang, "Hybrid Blockchain Database Systems: Design and Performance," *Proceedings of the VLDB Endowment* 15, no. 5 (2022): 1092–1104.

12. T. McConaghy, R. Marques, A. Müller, et al., "Bigchaindb: A Scalable Blockchain Database," *White Paper, BigChainDB* (2016): 53–72.