Exploring the Efficacy of Data-Decoupled Federated Learning for Image Classification and Medical Imaging Analysis

Muhammad Jahanzeb Khan
University of Nevada
Reno, USA
jahanzeb@nevada.unr.edu

Olamide Timothy Tawose
University of Nevada
Reno, USA
otawose@nevada.unr.edu

Rui Hu
University of Nevada
Reno, USA
ruihu@unr.edu

Dongfang Zhao
University of Nevada
Reno, USA
dzhao@unr.edu

ABSTRACT

The extensive use of medical imaging datasets, like Magnetic Resonance Imaging (MRI), in healthcare research and diagnosis, is often impeded by privacy concerns and computational costs. We address these challenges with a novel solution that integrates federated learning and data decoupling techniques, enabling efficient utilization of medical imaging datasets on high-performance computing (HPC) systems while ensuring data privacy and integrity. Our data decoupling federated learning framework allows hospitals to train a shared model on local MRI datasets without exposing raw data. By separating data management functionalities from the federated learning system, hospitals can utilize their existing HPC resources and maintain control over sensitive data. We extensively evaluated our approach using various image classification datasets, including SVHN, CIFAR10, and a specific medical imaging domain – Brain MRI datasets. Our results indicate improved model accuracy, reduced computational costs, and enhanced scalability while maintaining data privacy. Our work presents federated learning as a promising tool for healthcare, emphasizing the importance of data decoupling techniques in ensuring secure and cost-effective medical imaging data analysis.

KEYWORDS

Federated Learning, Image Classification, Medical Imaging, Data Decoupling

ACM Reference Format:
Muhammad Jahanzeb Khan, Olamide Timothy Tawose, Rui Hu, and Dongfang Zhao. 2023. Exploring the Efficacy of Data-Decoupled Federated Learning for Image Classification and Medical Imaging Analysis. In *KDD FL4Data-Mining '23, August 7, 2023, Long Beach, CA, USA.*. ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Federated learning (FL) has emerged as a distributed machine learning approach that allows multiple participants or clients, such as smart devices or hospitals, to collaboratively learn a shared model without the need to exchange their raw data [11]. This ability to train models while keeping data local can considerably enhance data privacy, decrease communication costs, and train an accurate model that represents the collective knowledge of all clients. As such, FL has attracted substantial attention in recent years and has found application in numerous fields such as health monitoring, autonomous driving, and more [9].

Despite the immense potential of FL systems [6, 14, 16, 19] in scalable privacy-preserving machine learning, their wider deployment necessitates the resolution of some existing challenges. One major issue is associated with the data management aspect of FL systems. For example in medical data analysis, the added ability to manage and query intermediate models could help promote reproducibility [18], enhance auditing [17] and optimize resource utilization [2].

However, in the current paradigm, clients lack an efficient means to manage and query the intermediate models during the training process or verification procedure of their FL applications. Broadly, the data management functionalities and computational aspects are tightly integrated, leading to difficulties in customizing or generalizing the entire FL system. A separate concern arises when certain applications or domains are unable to meet the hardware/platform requirements of modern FL systems; for example, existing FL systems do not fit well in large-scale high-performance computing (HPC) environments with limited local persistent storage [21].

To address these issues, we present a novel framework that decouples data management functionalities from FL systems. This innovative approach incorporates a loosely-coupled FL architecture where the global model and local models are managed by a specialized database. This separation allows clients to tailor their FL applications using specific data subsystems and facilitates the scalability of the FL system to accommodate a large number of clients. Through this approach, we aim to enhance the versatility, performance, and scalability of FL systems.

We integrated our Federated Learning (FL) prototype system with some mainstream databases such as MongoDB [5], Cassandra [7], Neo4j [12], and PostgreSQL [15]. These systems were selected due to their widespread use and proven efficacy in data management [20].

Additionally, we included the Secure Copy Protocol (SCP) [8] in our testing suite. Although primarily a secure file transfer tool, SCP was utilized similarly to a database system considering its capability for large file transfers between machines, analogous to data transfer operations essential in FL. We deploy this prototype system on an 11-node cluster on CloudLab [4]. Our experimental findings underline that data decoupling provides clients with increased options to optimize their FL applications in terms of performance, resilience, and usability.

In order to thoroughly assess the practical implications of our decoupling framework, we have performed extensive experiments on standard image classification datasets such as SVHN [13], and CIFAR10 [10]. Additionally, recognizing the relevance and potential of our framework in healthcare settings, we extend our evaluations to a more specific, medical imaging domain – brain MRI dataset [1]. This enables us to gauge the performance metrics in a setting closely related to image classification, providing a realistic picture of our framework's applicability and advantages in the healthcare sector. In summary, this research and its evaluation make the following contributions:

- **(i)** We propose a loosely-coupled FL architecture that separates the data management functionalities from the FL system. This approach allows clients to customize their FL applications using specific data subsystems, enhancing the interoperability and scalability of the system.
- **(ii)** We extend the proposed architecture to support heterogeneous data models under the same framework through a database-agnostic approach. By designing our system to be agnostic to specific database technologies used by clients, we enable seamless integration and communication between clients using different data models, thereby further improving the adaptability and flexibility of the system.
- **(iii)** We explore the possibility of recommending a specific data model for the user's workload based on the characteristics of the data. Specifically, we investigate the performance of several mainstream database systems, including MongoDB, Cassandra, Neo4j, and PostgreSQL, and provide insights into the strengths and weaknesses of each system for FL workloads.
- **(iv)** We implement our proposed approach by building a prototype FL system that utilizes several types of mainstream database systems. To the best of our knowledge, this is the first work to conduct a thorough evaluation of comparing different database services in the context of federated learning systems.
- **(v)** We carry out an extensive evaluation of the performance of the prototype system by conducting experiments on an 11-node cluster on CloudLab [4]. Our experimental results demonstrate that the use of data decoupling provides clients with more options to optimize their FL applications in terms of performance, resilience, and usability.

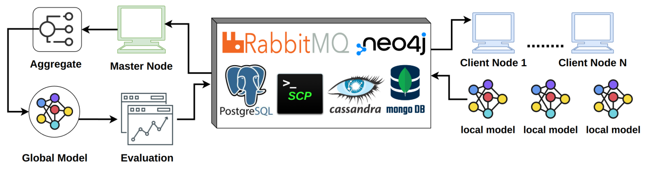

Figure 1: Proposed Design of Data-Decoupling FL (DDFL) framework.

By proposing a loosely-coupled FL architecture, supporting heterogeneous data models, and exploring the possibility of recommending a specific data model for the user's workload, our work provides a comprehensive and flexible framework for data management in FL systems. The insights gained from our evaluation of different database technologies and our proposed recommendations can help clients make informed decisions when selecting a database system for their FL application. Overall, our work aims to enhance the interoperability, performance, and scalability of FL systems, enabling clients to optimize their FL applications for their specific use cases.

2 SYSTEM DESIGN AND IMPLEMENTATION

The concept of data decoupling in federated learning offers distinct benefits but actualizing it presents a non-trivial set of challenges. In contrast to conventional FL systems where the data management is inherently tied to the FL workflow, forming a cohesive unit where computations and data transmissions occur simultaneously. The extraction of data management functions from this intertwined setting of the proposed data decoupling approach necessitates the redefinition of existing computational and communication protocols. Notably, the data management service needs to seamlessly interface with diverse data sources, catering to a heterogeneous mix of local datasets. It must handle a variety of data types and formats due to the nature of the machine-learning models used in FL. Hence, the design and implementation of such a service require careful consideration of compatibility, interoperability, and scalability aspects.

Furthermore, the implementation must prioritize the security of data transfers. Given the sensitive nature of the data involved (e.g., in the healthcare sector), any decoupling strategy must not compromise the robustness of the encryption protocols used. Therefore, the challenge also lies in devising a system that ensures secure transmission and storage of data throughout the FL process while maintaining the system's performance and efficiency. In the subsequent sections, we present our approach to overcoming these challenges by providing a detailed explanation of the design and implementation of our Data-Decoupling Federated Learning (DDFL) framework.

Our Data-Decoupling Federated Learning (DDFL) framework is devised to facilitate the training of a comprehensive model, with a dedicated data management service handling intermediate and final results. The architecture of the proposed framework is outlined in Figure 1, which includes a master node, a set of N client nodes, and a dedicated data management service.

In the context of an FL system, N represents the number of clients that retain their datasets locally and aim to learn a shared global model G through iterative collaboration, coordinated by the master node. Each training round involves each client node updating the downloaded global model using its local training data and sharing these updated local model parameters with the master node. The master node then aggregates all received local models to update the global model for the next training round.

DDFL introduces a dedicated data management service to handle all intermediate models during the training process, decoupling data management from the FL system. This service utilizes one of six options: RabbitMQ Queues, MongoDB Collection, Secure Copy Protocol (SCP), Cassandra, Neo4j, and Postgres Relational Database Management System (RDBMS).

These options were chosen based on their unique capabilities for buffering, delivering, storing, and sharing intermediate and fully trained models. For example, in the SCP, the global model is saved on the persistent disk for secure and reliable storage, while in MongoDB and Cassandra systems e.t.c, the global model is stored in a model variable, housed in memory for swift retrieval.

- **(i) RabbitMQ Queues**: RabbitMQ [3] is an open-source message broker used in DDFL to buffer and deliver intermediate and trained models via message queues.
- **(ii) MongoDB Collection**: MongoDB [5] is a popular open-source NoSQL database employed in DDFL for storing and sharing intermediate and trained models using a flexible BSON data format.
- **(iii) Secure File Transfer (SCP)**: SCP [8] is a secure network protocol used in DDFL to transfer intermediate and trained models between the master node and client nodes.
- **(iv) Cassandra**: Apache Cassandra [7] is a distributed NoSQL database suitable for managing and storing intermediate and trained models in the FL framework.
- **(v) Neo4j**: Neo4j [12] is a graph database used in DDFL to store and query intermediate and trained models represented as nodes and edges.
- **(vi) Postgres RDBMS**: PostgreSQL [15] is a reliable and robust open-source relational database management system used in DDFL for storing intermediate and trained models while supporting complex queries.

2.1 Local Model Computation

The data management solution S is selected from the deployed databases. Each training round involves each client $i \in [N]$ performing the following steps:

- (i) Initializing the local model M_i as the global model G;
- (ii) Loading local data D_i into its local storage;
- (iii) Training the local model M_i using local data D_i;
- (iv) Encrypting the local model M_i using a secret key k_i, and
- (v) Storing the encrypted local model E_{M_i} in S.

2.2 Model Aggregation

After local model computation, all local models are aggregated to update the global model on the server for the next round of training. First, each client $i \in [N]$ executes the following steps:

- (i) Retrieves encrypted local model updates E_{M_i} from S;
- (ii) Decrypts local model update E_{M_i} using the secret key k_i: $M_i = \text{Decrypt}(E_{M_i}, k_i)$;
- (iii) Encrypts the local model update M_i and transmits it to S.

Subsequently, the master node executes the steps:

- (i) Loads all model updates $M_1, M_2, \ldots, M_N$ from memory;
- (ii) Aggregates model updates in the global database to generate a new global model, i.e., $G = \text{Aggregate}(M_1, M_2, \ldots, M_N)$;
- (iii) Encrypts the new global model G using the secret key k: $E_G = \text{Encrypt}(G, k)$;
- (iv) Stores the encrypted global model E_G in S, and
- (v) Shares the new global model G with client nodes.

2.3 Model Evaluation

During training, both client nodes and the master node are permitted to evaluate a model using their testing dataset. The master node, for example, can execute the following steps to evaluate the performance of the latest global model:

- (i) Retrieves the encrypted latest global model E_G from S;
- (ii) Decrypts the global model E_G using the secret key k: $G = \text{Decrypt}(E_G, k)$;
- (iii) Evaluates the global model G on the test dataset T; and
- (iv) Obtains the testing accuracy of the trained global model $\text{Accuracy}(G, T)$.

Each model is stored in the database as a collection of the model parameters and other features. Specifically, each collection consists of the following columns:

1. **Round:** This column stores the round number of the federated training process for each model. As the federated learning process consists of several rounds of training, this column serves as a unique identifier for each model and helps to distinguish it from others in the database.
2. **Model:** The model column stores the parameters of the model, which are essential for reusing and updating the model in future rounds of training. This column enables the retrieval of a particular model, and its parameters from the database, which can then be used to resume the training process or update the model with the new training data.
3. **Accuracy:** This column stores the testing accuracy of the model, which provides an evaluation metric for the model. This metric can be used to compare the effectiveness of different models or identify the best-performing model among all the stored models in the database.
4. **Time:** This column stores the time taken to complete the corresponding round of training. This column serves as a performance metric for the training process and can be used to identify the training rounds that took more time and optimize them to achieve better performance.

By using this schema to store and organize all the models in the database system, the client nodes and the master node can efficiently manage and retrieve the models during the federated learning process. This enables us to efficiently track the performance of the models, identify the best-performing models, and use the stored models for resuming or updating the training process.

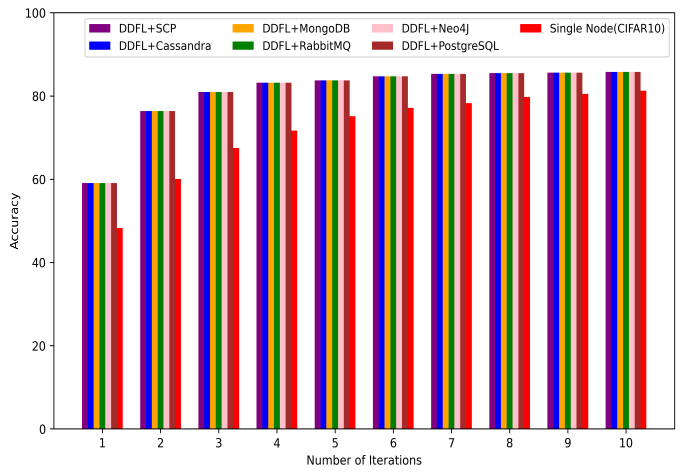

Figure 2: DDFL: CIFAR10 Testing Accuracy by Database.

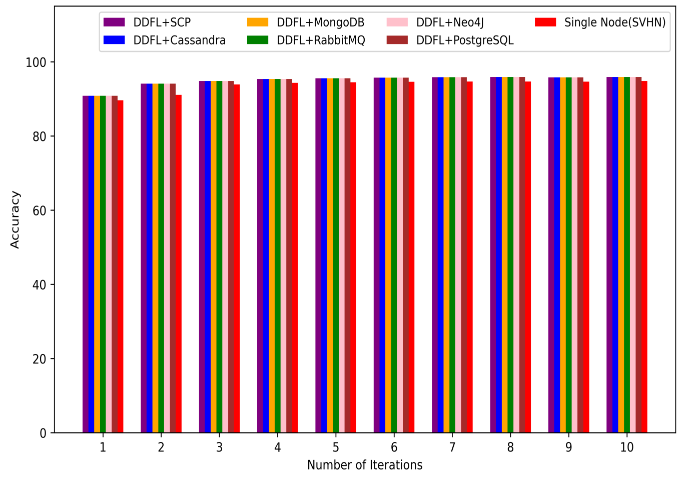

Figure 3: DDFL: SVHN Testing Accuracy by Database.

2.4 Experimental Setup

Our assessment of the Data Decoupled Federated Learning (DDFL) framework employed two standard datasets: CIFAR10 [10], and SVHN [13], spanning a variety of image classification tasks. Furthermore, we gauged the DDFL framework's capabilities on a complex Medical Imaging Dataset, comprising multi-modal brain scans labeled with different regions of interest. The Medical Imaging Dataset [1], a set of NIfTI files detailing varied brain scan modalities provides a unique challenge due to its complexity and scale, which is suitable as a benchmark to further demonstrate the applicability of our proposed framework.

For this assessment, we trained a federated learning model using the Unet3d architecture [22], with multiple clients each holding a data subset. The goal was to collaboratively train a model capable of accurately classifying pixels in brain images. Experiments were conducted on an 11-node CloudLab [4] cluster, with one master node and the remaining as client nodes. Each node was equipped with an Intel Xeon CPU E5-2690 v4 @ 2.60GHz processor and 377GB of RAM. Due to hardware limitations, Deep Convolutional Neural Net Model Training was CPU exclusive.

2.5 Performance Metrics

The performance evaluation of the DDFL framework considered various metrics to assess its effectiveness in training and model management. The following metrics were utilized:

- **Training Time:** This metric measures the time taken for the DDFL framework to train the model on the provided datasets. It provides insights into the efficiency and speed of the training process.
- **Accuracy:** The accuracy of the global model was evaluated to assess the quality of the predictions made on the validation set. Accuracy is defined as the ratio of correct predictions to the total number of predictions made.

3 PRELIMINARY RESULTS

3.1 DDFL: Model Accuracy and Analysis

Our Data-Decoupling Federated Learning (DDFL) system refines the Federated Learning approach by decoupling data and model training across nodes. To evaluate this, we tested the accuracy of DDFL against a single-node model—optimized via hyperparameter tuning—across various databases, using CIFAR-10, SVHN, and Brain MRI (B-MRI) datasets. The results of our experiment are presented in Figures 2, 3, and 4. These graphs illustrate the testing accuracy of our DDFL models compared to the single-node baseline model.

Across all databases, our DDFL models demonstrated a modest improvement in average accuracy for CIFAR-10 and SVHN datasets, at 2.23% and 1.87% respectively. While these figures may not appear substantial, it's important to note that they are indicative of the promise our data-decoupling and distributed training approach holds in federated learning settings. The presented data offers a perspective on potential future directions of improvement and optimization for our approach.

3.1.1 ***MRI Accuracy Analysis.*** When it comes to the Brain MRI dataset, our federated learning model delivered an impressive average accuracy across all regions of interest as shown in Figure 4. It demonstrates the potential of the DDFL framework in medical imaging applications. Despite these promising results, we faced some challenges. For instance, ensuring the reliability and consistency of the performance across different database systems was a complex task, as each system has its own unique characteristics and potential bottlenecks.

In the future, we plan to delve deeper into optimizing the DDFL framework for different database systems and exploring ways to further enhance the accuracy of our models. We also aim to apply the DDFL framework to more diverse datasets and problems and investigate the influence of different deep learning architectures on system scalability and performance. Overall, our study shows that data-decoupling in federated learning can enhance testing accuracy. The consistent performance across different database systems provides strong evidence of the efficacy of the DDFL framework in distributed deep-learning scenarios.

3.2 DDFL: Training Time Comparison

Our investigation also scrutinizes the impact of different databases on the training time of our Data-Decoupling Federated Learning (DDFL) system. We also examined a variety of databases - MongoDB,

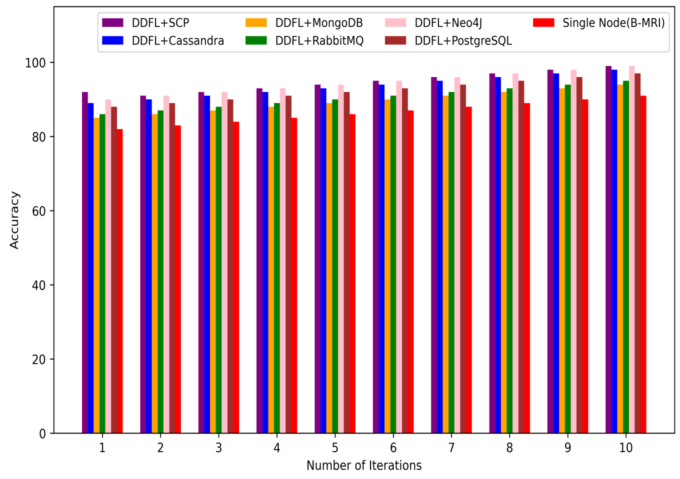

Figure 4: DDFL: B-MRI Testing Accuracy by Database.

SCP, PostgreSQL, Neo4j, RabbitMQ, and Cassandra. Figures 5 and 6 present the variation in training time across these databases for the image datasets. Notably, each database system manifests distinct performance characteristics during the training phase due to its inherent architectures and operational principles.

While the time taken for training was generally increased in the DDFL setting compared to a single-node model, it is important to consider the benefits of distributed training and data privacy preservation that outweigh this overhead. The DDFL framework enables efficient collaboration among multiple nodes while ensuring the privacy of local data, which is crucial in scenarios where data sharing is restricted. Interestingly, our evaluation revealed that certain database systems within the DDFL framework exhibited shorter training times compared to the single-node baseline.

These database systems, such as MongoDB[5], Postgres [15], and Neo4j [12], demonstrated improved efficiency in handling the distributed training process. This suggests that the choice of the database system can significantly impact the overall training time in the DDFL setup. However, it is worth noting that the slight increase in training time observed in the DDFL setting is justified by the significant advantages it offers, such as improved model accuracy and enhanced privacy protection.

The trade-off between training time and these benefits should be carefully considered when deploying DDFL systems in practical applications. Future work will aim at optimization strategies to reduce the training times further. The findings highlight the importance of selecting an appropriate database system when deploying a DDFL setup, balancing performance, complexity, and training time.

3.2.1 ***MRI- Training Time***. One of the key aspects of our study was to investigate the training time requirements of the Data-Decoupled Federated Learning (DDFL) model when applied to the Medical Imaging Dataset. The DDFL framework is designed to prioritize localized computation and minimize data transfer, resulting in a notable reduction in training time compared to conventional centralized learning models, which typically require approximately 8 hours to complete the training process. Figure 7 provides a comparative analysis of the training time across different database systems for the B-MRI dataset. The evaluation reveals notable variations in performance among the various database systems.

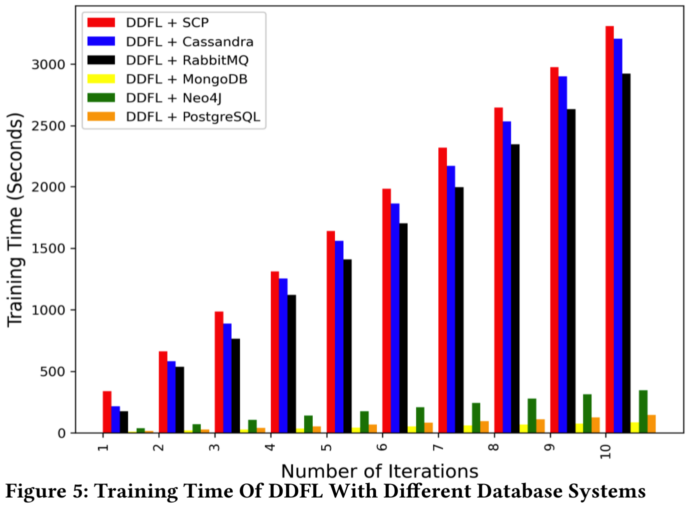

Figure 5: Training Time Of DDFL With Different Database Systems On CIFAR-10 Dataset.

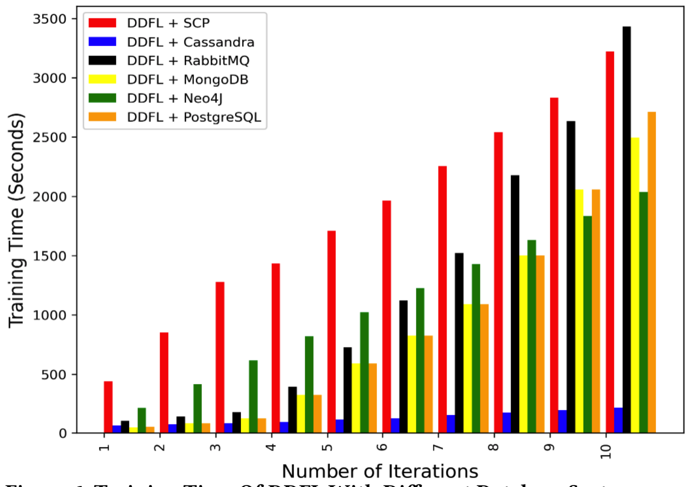

Figure 6: Training Time Of DDFL With Different Database Systems On SVHN Dataset.

MongoDB emerges as the most time-efficient system, closely followed by Neo4j and PostgreSQL. On the other hand, the SCP, RabbitMQ, and Cassandra systems exhibit longer training times in comparison. These findings highlight the significance of the choice of a database system in the DDFL framework, as it can significantly influence the overall training time. MongoDB, with its efficient handling of data management and processing capabilities, demonstrates superior performance in terms of reduced training time.

Neo4j and PostgreSQL also offer favorable performance, although they have slightly higher training time than MongoDB. In contrast, the SCP, RabbitMQ, and Cassandra systems exhibit relatively longer training times, indicating areas for potential optimization in future work.

The observed reduction in training time achieved by the DDFL model in processing the Medical Imaging Dataset holds promise for applications requiring timely analysis of medical images. By leveraging localized computation and minimal data transfer, the DDFL

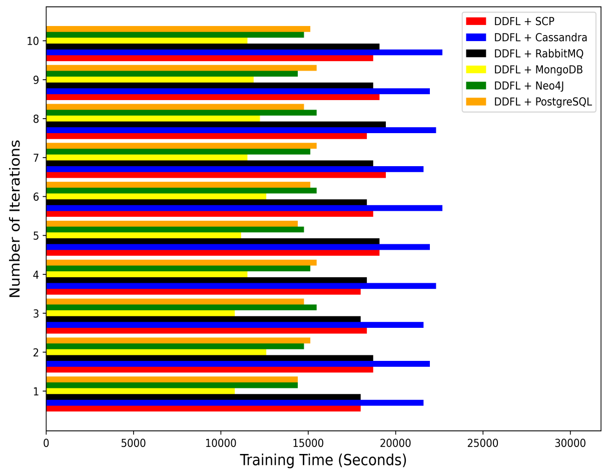

Figure 7: Training Time of DDFL With Different Database Systems On B-MRI Dataset.

framework presents an effective solution for efficient and time-sensitive medical imaging analysis. Further research and optimization efforts can focus on enhancing the performance of the DDFL framework, particularly for database systems exhibiting longer training times, ultimately improving the overall efficiency and effectiveness of medical image analysis in federated learning settings.

These findings underscore the crucial role of choosing an appropriate database system in federated learning contexts. MongoDB's relatively lower training times can provide a significant advantage in scenarios requiring expedient model training and deployment. However, the chosen database system should also cater to the specific needs and constraints of the data, infrastructure, and application.

Our results illustrate the inherent advantage of DDFL when dealing with large-scale datasets, like the Medical Imaging Dataset, in a federated learning setting. By offering a balance between data privacy, efficient computation, and reduced data transfer, DDFL holds promising potential for revolutionizing data-intensive fields like medical imaging.

4 INFLUENCE OF DIFFERENT DATABASES ON TRAINING TIME

In this study, we assess how different database systems - MongoDB, SCP, PostgreSQL, Neo4j, RabbitMQ, and Apache Cassandra - affect deep learning model training times. The performance of these databases is largely determined by their respective data retrieval speeds, storage efficiencies, and scalability capacities. MongoDB, a NoSQL database, offers flexible data storage, potentially speeding up training times. SCP, a secure file transfer protocol, while convenient, may lag in data retrieval speed. PostgreSQL, an enterprise-class RDBMS, might have increased training times due to the overhead of managing relational databases.

Neo4j, a graph database, has variable performance depending on the data structure and model needs. RabbitMQ, a message broker, may underperform in training time as it's not tailored for database operations. Lastly, Apache Cassandra, a scalable, distributed NoSQL database, can have its training time influenced by data distribution and replication. Therefore, database selection can have a significant impact on the efficiency of deep learning model training and must be carefully considered based on specific application requirements and data nature.

5 CONCLUSION

Our study presented the Data Decoupled Federated Learning (DDFL) framework, a new toolset to enhance Federated Learning (FL) system capabilities. It allows for custom client applications and efficient model querying. Acting as a benchmarking tool, DDFL assesses diverse database subsystems used in FL, leading the way to a novel FL environment fortified with various data management services.

The investigation involved implementing DDFL with multiple database systems including MongoDB, SCP, Cassandra, RabbitMQ, Neo4j, and Postgres. Rigorous testing on CIFAR-10 and SVHN datasets revealed comparable accuracy levels to traditional FL systems, validating their functionality. However, the inspection of key metrics like training time and accuracy showed significant differences due to the unique attributes of each system.

DDFL was also applied to a complex Medical Imaging Dataset, showcasing its adaptability and effectiveness for complex and large-scale tasks. The evaluation of FL model performance on this dataset, in terms of accuracy and training time served as a robust proof-of-concept for DDFL's potential in improving efficiency and performance in medical imaging analysis. Notably, our study emphasized that the choice of the database system can significantly influence the efficiency of FL systems. This insight, along with the data collected on the strengths and weaknesses of existing methods, paves the way for new research in this field.

The DDFL framework prompts system-centric research focusing on data subsystems within FL ecosystems. This research also underlines the need for a recommendation system to guide users in selecting the most suitable data subsystems for FL based on their specific use case. In conclusion, the Data-Decoupled Federated Learning (DDFL) framework presents a novel and innovative approach at the intersection of database systems and federated learning. It offers a fresh perspective to address the challenges associated with traditional federated learning systems by decoupling the data management aspects from the federated learning workflow. This decoupling introduces a higher degree of flexibility, efficiency, and adaptability to federated learning systems.

REFERENCES

[1] Sartaj Bhuvaji, Ankita Kadam, Prajakta Bhumkar, Sameer Dedge, and Swati Kanchan. 2020. Brain Tumor Classification (MRI). https://www.kaggle.com/dsv/1183165
[2] Jeffrey Dean and Sanjay Ghemawat. 2008. MapReduce: simplified data processing on large clusters. *Commun. ACM* 51, 1 (2008), 107–113.
[3] Chris Duncan. n.d.. *RabbitMQ: A messaging broker.* Pivotal Software, Inc. https://www.rabbitmq.com/
[4] Dmitry Duplyakin, Robert Ricci, Aleksander Maricq, Gary Wong, Jonathon Duerig, Eric Eide, Leigh Stoller, Mike Hibler, David Johnson, Kirk Webb, Aditya Akella, Kuangching Wang, Glenn Ricart, Larry Landweber, Chip Elliott, Michael Zink, Emmanuel Cecchet, Snigdhaswin Kar, and Prabodh Mishra. 2019. The Design and Operation of CloudLab. In *Proceedings of the USENIX Annual Technical Conference (ATC).* 1–14. https://www.flux.utah.edu/paper/duplyakin-atc19
[5] Eliot Horowitz Dwight Merriman and Kevin Ryan. n.d.. *MongoDB: The database for modern applications.* MongoDB Inc. https://www.mongodb.com/
[6] FedML. Accessed 2023. https://fedml.ai/.
[7] The Apache Software Foundation. 2010. *Apache Cassandra.* https://cassandra.apache.org/

[8] Simson Garfinkel and Gene Spafford. 2002. *Practical UNIX and Internet Security*. O'Reilly Media, Inc.
[9] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. 2021. Advances and Open Problems in Federated Learning. *Foundations and Trends® in Machine Learning* 14, 1–2 (2021), 1–210.
[10] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. 2019. CIFAR-10 (Canadian Institute for Advanced Research). http://www.cs.toronto.edu/~kriz/cifar.html.
[11] H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. 2023. Communication-Efficient Learning of Deep Networks from Decentralized Data. arXiv:1602.05629 [cs.LG]
[12] Neo4j, Inc. 2023. Neo4j. https://neo4j.com/.
[13] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A.Y. Ng. 2011. Reading Digits in Natural Images with Unsupervised Feature Learning.
[14] Takayuki Nishio and Ryo Yonetani. 2018. Client Selection for Federated Learning with Heterogeneous Resources in Mobile Edge. *CoRR* abs/1804.08333 (2018).
[15] PostgreSQL Global Development Group. n.d.. PostgreSQL. Accessed: 2023.
[16] PySyft. Accessed 2023. https://github.com/OpenMined/PySyft.
[17] Alvin Rajkomar, Eyal Oren, Kai Chen, Andrew M Dai, Nissan Hajaj, Michaela Hardt, Peter J Liu, Xiaobing Liu, Jake Marcus, Mimi Sun, et al. 2018. Scalable and accurate deep learning with electronic health records. *NPJ digital medicine* 1, 1 (2018), 18.
[18] Victoria Stodden, Marcia McNutt, David H Bailey, Ewa Deelman, Yolanda Gil, Brooks Hanson, Michael A Heroux, John PA Ioannidis, and Michela Taufer. 2016. Enhancing reproducibility for computational methods. *Science* 354, 6317 (2016), 1240–1241.
[19] TensorFlow Federated. Accessed 2023. https://github.com/tensorflow/federated.
[20] Chad Vicknair, Michael Macias, Zhendong Zhao, Xiaofei Nan, Yixin Chen, and Dawn Wilkins. 2010. A Comparison of a Graph Database and a Relational Database: A Data Provenance Perspective. In *Proceedings of the 48th Annual Southeast Regional Conference* (Oxford, Mississippi) *(ACM SE '10)*. Association for Computing Machinery, New York, NY, USA, Article 42, 6 pages.
[21] Lesi Wang and Dongfang Zhao. 2022. Practical Federated Learning Infrastructure for Privacy-Preserving Scientific Computing. In *2022 IEEE/ACM International Workshop on Artificial Intelligence and Machine Learning for Scientific Applications (AI4S)*. IEEE, 38–43.
[22] Özgün Çiçek, Ahmed Abdulkadir, Soeren S Lienkamp, Thomas Brox, and Olaf Ronneberger. 2016. 3D U-Net: Learning Dense Volumetric Segmentation from Sparse Annotation. *International Conference on Medical Image Computing and Computer-Assisted Intervention* (2016), 424–432.