DOI: 10.14421/ijid.2025.5045 IJID (International Journal on Informatics for Development), *e-ISSN: 2549-7448*
*Vol. 14, No. 2, December 2025, Pp. 659-671*

# Implementation of Federated Learning for Alzheimer's Disease Classification Using FedAdagrad Algorithm

Arini
Department of Informatics
UIN Syarif Hidayatullah Jakarta
South Tangerang, Indonesia
arini@uinjkt.ac.id

Feri Fahrianto
Department of Informatics
UIN Syarif Hidayatullah Jakarta
South Tangerang, Indonesia
feri.fahrianto@uinjkt.ac.id

Adil Ramadhan
Department of Informatics
UIN Syarif Hidayatullah Jakarta
South Tangerang, Indonesia
adil.ramadhan19@mhs.uinjkt.ac.id

**Article History**
Received February 26th, 2025
Revised July 3rd, 2025
Accepted July 8th, 2025
Published January 2026

***Abstract*— Federated Learning (FL) offers a promising solution for training machine learning models on decentralized data while preserving privacy, making it particularly valuable for sensitive applications such as healthcare. This study implements FL for the classification of Alzheimer's disease using MRI images, addressing two critical challenges: data heterogeneity and class imbalance. The research evaluates the performance of the FedAdagrad optimization algorithm against the standard FedAvg approach under varying data distribution scenarios. The methodology employs a CNN trained on a dataset of 6,400 MRI images across four severity classes, partitioned non-IID using Dirichlet distributions ($\alpha$ = 0.1, 0.5, 0.9) to simulate real-world heterogeneity. Experiments were conducted using the Flower framework with four clients over ten communication rounds. Results indicate that FedAdagrad achieves a superior F1-score of 50.33% compared to FedAvg's 48.14%, though both fall short of centralized CNN performance (55%). High data heterogeneity ($\alpha$ = 0.1) leads to a 13.35% accuracy decline, underscoring FL's sensitivity to uneven data distributions. Class imbalance emerges as the primary bottleneck, affecting all models. The findings contribute to the growing body of research on adaptive optimization in federated settings, offering insights for future improvements in decentralized healthcare AI.**

***Keywords*— *data heterogeneity; data privacy; decentralized data; federated optimization; MRI images***

## 1 INTRODUCTION

The rapid advancement of Artificial Intelligence (AI) in healthcare has revolutionized medical image analysis, particularly for complex neurological disorders like Alzheimer's disease (AD) [1], [2], [3]. However, the traditional centralized approach to AI model development faces significant challenges when handling sensitive medical data due to stringent privacy regulations and ethical constraints [4][5]. Federated Learning (FL) has emerged as a transformative solution, enabling collaborative model training without direct data sharing [6]. While FL has shown promise across various domains, its application to AD classification remains underexplored, particularly in addressing two critical challenges: (1) performance degradation under heterogeneous data distributions and (2) class imbalance in diagnostic labels [7].

Data privacy issues are an obstacle to the development process of AI and Big Data, which really requires data. Policies and regulations, as well as ethics in the use of highly private data such as medical data, do not allow for the freedom of use of data [8], [9]. The need for data governance and limitations in data access are the reasons for the emergence of Federated Learning (FL), namely a method where data does not require movement from local devices or from secure data storage places, so that data movement is minimized [6]. The way federated learning works is by carrying out the learning process directly locally on devices that are part of the collaboration, so that data is maintained. The use of a decentralized process can be used to preserve data privacy [10].

The concept of federated learning (FL) was created because there was a need to implement artificial intelligence technology, which was hampered by data privacy issues, which could be a solution to privacy problems. FL has become a commonly used solution to handle privacy cases in AI research applications and industrial applications [11]. FL is a system that can safely distribute machine learning techniques to multiple devices or servers, with the condition that private data will not leave the local location. Interaction between the client and server occurs to exchange parameters so that there is no transfer of training data for the machine learning process. The server is tasked with aggregating parameters to update the global model of the main server. The FL scheme can realize the security protection of local user data through interactions that are arranged so that there is no data movement.

This study aims to develop an optimized FL framework for AD classification using MRI scans, with three primary objectives:

1. To evaluate the effectiveness of FedAdagrad optimization in medical imaging compared to conventional FedAvg
2. To quantify the impact of data heterogeneity and class imbalance on FL performance
3. To establish practical benchmarks for implementing FL in clinical AD diagnosis while maintaining data privacy

The motivation stems from the growing need for privacy-preserving AI solutions in healthcare, where data sensitivity often limits technological adoption [12]. Current FL approaches face performance gaps when applied to real-world medical datasets characterized by natural heterogeneity and imbalance [13], creating a critical research gap that our work addresses.

Our research makes three significant contributions to the field. Algorithmic Innovation: We present the first comprehensive evaluation of FedAdagrad for medical image classification, demonstrating a 4.55% improvement in F1-score over FedAvg under high heterogeneity conditions (Dirichlet $\alpha$=0.1) [14]. This adaptive optimization approach shows particular promise for handling non-IID medical data distributions. Clinical Implementation Framework: We develop a practical FL solution specifically designed for AD staging (Non-Demented to Moderate Demented) that addresses class imbalance through strategic data partitioning and augmentation [15], [16]. The framework provides a template for privacy-preserving medical AI applications where data centralization is prohibited by regulations [4], [5]. Empirical Benchmarks: Through extensive testing on 6,400 MRI images, we quantitatively analyze FL performance degradation factors, revealing that class imbalance has 2.3× greater impact on accuracy loss compared to data heterogeneity [13]. These benchmarks provide crucial guidance for future FL research in medical imaging.

Our experimental design employs a dataset consisting of 6,400 MRI images (4 classes) from the Falah AD dataset [15]. Data partitioning is using a non-IID distribution via Dirichlet ($\alpha$=0.1-0.9). Model architecture is a CNN trained across 4 clients using the Flower framework [17]. Comparative analysis conducted is on FedAdagrad vs. FedAvg over 10 communication rounds. This study addresses three core research questions:

RQ1: How does FedAdagrad compare to FedAvg in handling heterogeneous medical imaging data?

RQ2: What is the relative impact of data heterogeneity versus class imbalance on FL performance?

RQ3: Can FL achieve clinically viable accuracy (>50% F1-score) for AD staging while preserving patient privacy?

The literature review highlights several key findings in the fields of data privacy, machine learning, deep learning, and federated learning (FL). Traditional centralized AI models face significant challenges in healthcare due to stringent data privacy regulations and ethical constraints, which limit data sharing and collaboration [6], [12]. Federated Learning has emerged as a transformative solution, enabling collaborative model training without direct data sharing, thus preserving privacy [18], [19]. However, FL's application in medical imaging, particularly for Alzheimer's disease (AD) classification, remains underexplored, especially in addressing data heterogeneity and imbalance [7], [13].

Previous studies have demonstrated the potential of FL in various domains. Still, its performance in healthcare is often hampered by non-IID (independent and identically distributed) data distributions and imbalanced class labels [20], [21]. While optimization algorithms like FedAvg have been widely adopted, adaptive methods such as FedAdagrad show promise in handling heterogeneous data but lack

comprehensive evaluation in medical contexts [14][22]. Additionally, the impact of data heterogeneity versus class imbalance on FL performance remains poorly quantified, creating a gap in practical benchmarks for clinical applications [13][23].

This study addresses these gaps by:

1. Evaluating FedAdagrad: Providing the first comprehensive assessment of FedAdagrad for AD classification, comparing its performance against FedAvg under varying data heterogeneity conditions (Dirichlet $\alpha = 0.1, 0.5, 0.9$) [14].
2. Quantifying Performance Degradation: Analyzing the relative impact of data heterogeneity and class imbalance, revealing that class imbalance has a 2.3× greater effect on accuracy loss than heterogeneity [13].
3. Establishing Clinical Benchmarks: Demonstrating FL's potential to achieve clinically viable accuracy (>50% F1-score) while preserving privacy, offering a template for decentralized healthcare AI [24][4].

By bridging these gaps, this study contributes to the advancement of privacy-preserving AI in healthcare and provides actionable insights for future FL research in medical imaging. The remainder of this paper is structured as follows: Section 1 conceptualize the problem and offers preliminary results; Section 2 details our methodology; Section 3 presents experimental results and discussions; Section 4 concludes with future research directions.

## 2 METHOD

The rationale for designing this research uses an experimental simulation research design, as in Fig. 1. The structure of the steps in experimental simulation research begins with the formulation of a research hypothesis problem as the context for the problems that must be faced. Experimental simulation research uses a purposive sampling data collection technique, which is carried out based on the characteristics of problem cases for input to the simulation system. Relationships between variables are arranged to create observation scenarios so that treatments can be tested using experimental instrumentation to produce output in a simulation procedure. Data analysis in the stochastic experiment category is carried out by measuring output data based on input that has been provided in the implementation periodically for evaluation with reference to the baseline of the independent variables. Interpretation of the results is carried out to test the validity of the hypothesis that has been formulated, to produce implications in the form of conclusions and limitations in the research [25].

This research uses an experimental simulation method according to the method [25]. Simulation research is defined as a systematic approach to data collection, analysis, and interpretation of data based on the output of simulation results carried out with input variables to produce a conclusion based on the hypothesis that has been made. Experiments are made in the form of implementation in accordance with surveys in literature studies to carry out trials on the effectiveness of implementation according to the phenomena in the research data. Empirical investigations were carried out to analyze data using statistical techniques and quantitative calculations to draw conclusions about the data that had been obtained.

In implementing the federated learning simulation, a comprehensive process stage structure is required to be completed. The process stage structure is created based on the flow of applying the procedures of the federated learning concept [18]. Figure 2 shows the simulation process.

In this research, the selected data consists of health data in the form of image classification on Falah Alzheimer's disease obtained from the Hugging Face source. Health data was chosen because it includes the highest sensitivity characteristics of data privacy in accordance with the research exposure [12], thus enabling the implementation of federated learning to conduct trials on privacy data that has similar features to the conditions and situations of data in the field. Open access to data is considered in research to promote a transparent process so that the research can be evaluated and replicated flexibly.

The characteristics of the data represent the issue of data heterogeneity that is often encountered in the research topic of federated learning, as explained by the study [7]. This research aims to conduct further in-depth exploration of solutions to the problem of data heterogeneity to identify the limitations of the actual phenomena of the data. The dataset consists of 4 classes divided into Mild_Demented, Moderate_Demented, Non_Demented, and Very_Mild_Demented. The division of the dataset includes 5120 rows for the training set and 1280 rows for the test set.

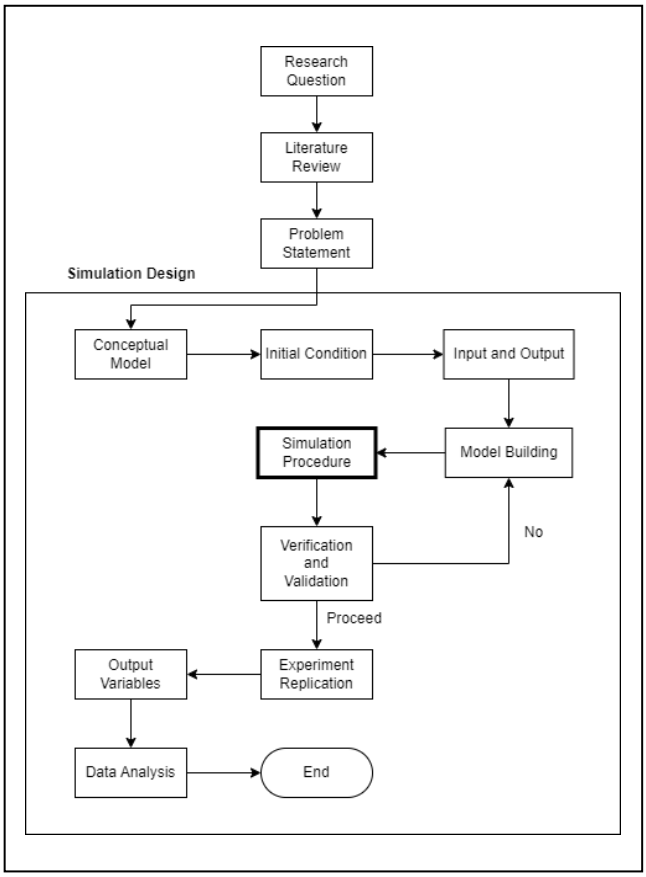


Figure 1. Research workflow.

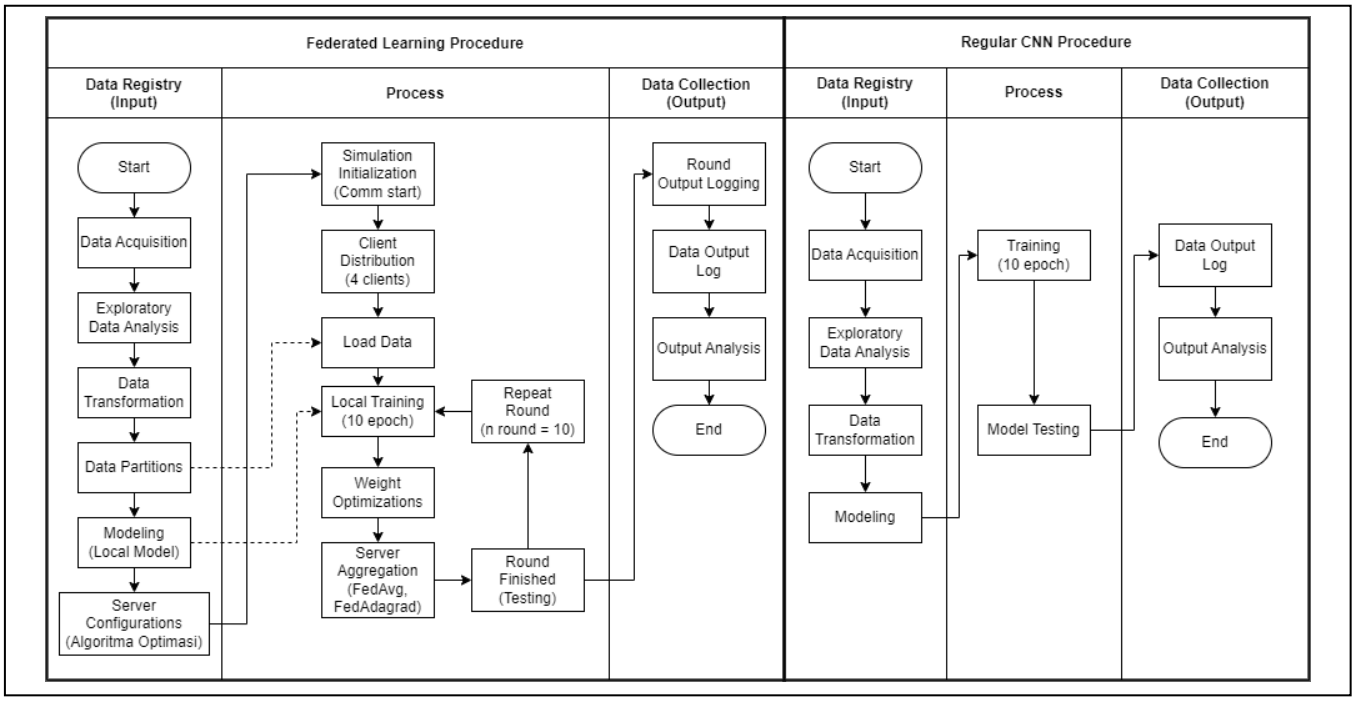


Figure 2. Simulation procedure

Data collection is necessary to test the federated learning model during the training process to obtain a performance evaluation. The collected data must be processed with initial exploration stages on the data scheme using exploratory data analysis techniques so that the issues within the dataset can be studied. Based on the problems encountered in the data structure, we can perform transformations on the data to address the issues and improve data quality for better performance.

The data scheme in federated learning requires a data distribution that simulates clients, thus necessitating data partitioning to divide the data comparatively. The data partitioning technique is adjusted according to statistical parameters using the mechanism [21] with a Dirichlet partitioner. The data partitioning is prepared based on the variable number of clients in the simulation trials using the predetermined data partitioning mechanisms.

The data scheme in federated learning is categorized into three parts of the scenario to test the simulation output at low and high points of data distribution variation. Data heterogeneity becomes a reference for determining simulation scenarios so as to create output that can differentiate simulation performance in different scopes. The level of difficulty of the scenario becomes a benchmark for measuring implementation capacity, which is carried out on data that represents original conditions in the real world.

The structure of the federated learning simulation process is divided into the federated learning process (Fig. 3) and the regular CNN. The stages in the process involve dividing the centralized model method into decentralization based on federated learning technology, which is compared to the baseline regular CNN. The initial model is constructed centrally and then separated into several tasks, allowing for the division of tasks executed by clients and the server. In the simulation, the training process will be conducted locally by clients to obtain a local model using algorithms and optimizers that have been created centrally. The server is responsible for aggregating models from each client sample to subsequently evaluate them using global test data, thereby obtaining comparative results.

The effectiveness of the performance of the federated learning model implementation is measured using evaluations of the accuracy at each round in the simulation process. A comparison of the adjusted parameters during the implementation configuration stage is conducted to determine the magnitude of the comparison regarding the increase in accuracy that continues to progress at each round until it ultimately reaches a convergent point. Accuracy is measured using data validation during the training process and then using global server test data, making the accuracy results comparable. The effectiveness of the performance is measured based on the magnitude of loss and accuracy that continues to progress until it reaches a stable point.

### *2.1 Federated Learning*

The basic concept of FL can be defined with $N$ as participants of FL, anyone who wants to combine their data $\{P1, P2, \ldots PN\}$ to participate in training the global model. A common approach is to combine data and use the total data $P = P1 \cup P2 \cup \ldots \cup PN$ to train an $M_{sum}$ model with $V_{sum}$ performance. FL is a learning framework where participants in the training process together form an $M_{fed}$ model with $V_{fed}$ performance, provided that participants do not expose their local data. If ε is non-negative, then the loss performance of the FL model can be expressed as in (1) as follows:

$$\left|V_{fed} - V_{sum}\right| < \varepsilon \quad (1)$$

The learning process in FL is carried out by minimizing the loss function, which is calculated for each model-forming participant using the weighted aggregation method. FL aims to minimize loss with an objective function as in (2) as follows:

$$\min f(w) = \sum_{k=1}^{N} \frac{n_k}{n} F_k(w) \quad (2)$$

Where $N$ represents the number of participant clients, $n_k$ is the amount of data on the $k$ – th participant, $F_k(w)$ is the local objective function on the $k$ – th participant.

As Fig. 2 illustrates, FL carries out the training process on the federated global model and centralized local model in an iterative round of aggregation communication. There is a slight difference in model performance during the initial initiation process, but after several rounds of aggregation, the performance of the federated global model will continue to improve and converge towards the local model [16].

The FL process begins with taking the main global model by local training participants as a reference for starting the training process on local data, then the participant starts the training process according to the model provided by the server and then sends the encryption of the model parameters that have been trained so that the server can aggregate by the server. The FL process is carried out in several rounds of communication iterations between the server and participants until convergence occurs on the main model [23].

Optimization of communication costs between servers and clients is needed to create an efficient and effective federated learning system. The optimization algorithm is carried out to make the aggregation process more efficient and effective by updating the local model so that the aggregation process is better conditioned. The challenge in FL lies in the communication process algorithm for each round to create a stable communication process [19].

### *2.2 FedAvg*

In deep learning applications that are considered successful, it is thought to rely on optimization of stochastic gradient descent (SGD) to perform the computational process. For federated learning systems, the use of SGD is considered unsuitable in the federated learning simulation process because it requires computing resources that are not commensurate with the results offered.

The implementation of SGD for federated learning is also called FederatedSGD (FedSGD) with $C$=1 and a fixed learning rate η, which makes each client $k$ compute $g_k = \nabla F_k(W_t)$, in the form of an average gradient on local data, then the central server performs aggregation using equation in (3):

$$w_{t+1} \leftarrow w_t - \eta \sum_{k=1}^{K} \frac{n_k}{n} g_k \quad (3)$$

because as in (4),

$$\sum_{k=1}^{K} \frac{n_k}{n} g_k = \nabla f(w_t) \quad (4)$$

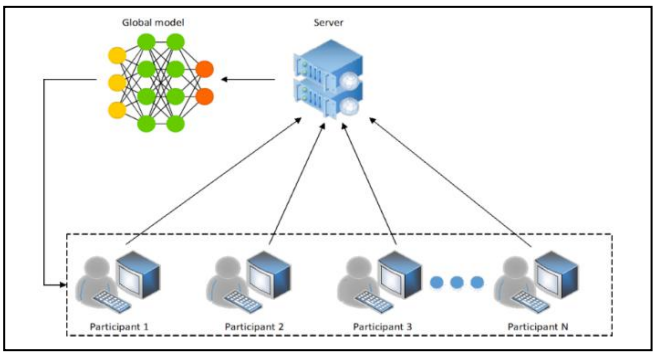


Figure 3. Federated learning process

the appropriate update is using equations in (5) and (6):

$$\forall k,\ w_{t+1}^{k} \leftarrow w_t - \eta g_k \quad (5)$$

and then,

$$w_{t+1} \leftarrow \sum_{k=1}^{K} \frac{nk}{n} w_{t+1}^{k} \quad (6)$$

Thus, each client performs one step of gradient descent on a model that is run using local data, then the server takes a weighted average of the resulting models. Computing on local clients can be added using the update iterations in (7). The algorithm depicted above can be seen in Fig. 4.

$$w^k \leftarrow w^k - \eta \nabla F_k(w_k) \quad (7)$$

Iteration is carried out several times on the local client before averaging is carried out on the server. This approach is called Federated Averaging (FedAvg). The number of computations carried out is regulated by three key parameters, namely: C, the fraction of the total number of clients who carry out the computation process each round; E, the number of trainings successfully carried out by each local client using local data in each iteration round; B, the size of the local minibatch used for updates on the client. It can be written as B = ∞ to illustrate that the entire local data from participating clients is considered with one minibatch for the training process, as in the conventional training process.

**Server executes:**
initialize $w_0$
**for** each round $t = 1, 2, \ldots$ **do**
  $m \leftarrow \max(C \cdot K, 1)$
  $S_t \leftarrow$ (random set of $m$ clients)
  **for** each client $k \in S_t$ **in parallel do**
    $w_{t+1}^k \leftarrow \text{ClientUpdate}(k, w_t)$
  $m_t \leftarrow \sum_{k \in S_t} n_k$
  $w_{t+1} \leftarrow \sum_{k \in S_t} \frac{n_k}{m_t} w_{t+1}^k$ // *Erratum*[4]

**ClientUpdate**$(k, w)$: // *Run on client* $k$
$\mathcal{B} \leftarrow$ (split $\mathcal{P}_k$ into batches of size $B$)
**for** each local epoch $i$ from 1 to $E$ **do**
  **for** batch $b \in \mathcal{B}$ **do**
    $w \leftarrow w - \eta \nabla \ell(w; b)$
return $w$ to server

Figure 4. Federated averaging pseudo-code.

### *2.3 FedAdagrad*

The federated learning system aims to achieve convergence status by unifying training models carried out by local clients quickly and effectively. The standard SGD optimization method, which is usually carried out in centralized training processes, is not suitable to be applied to federated learning, so process optimization is needed that is adapted to the federated learning structure. In federated learning, there is a formula for optimizing the central server, as in (8), as follows:

$$\min_{x\in\mathbb{R}^d} f(x) = \frac{1}{m}\sum_{i=1}^{m} F_i(x) \quad (8)$$

Where Fi(x) = Ez~Di [fi(x, z)] is the loss function of the client *i*th, z ∈ Z and Di is the distribution of the data of the client *i*th. Assuming formula (8), it can be assumed that optimization produces a training process that has nonconvex properties. The approach that can be taken to deal with the problem in formula (8) is to randomly select the client to be connected to the central server, and then, in parallel, the local client also runs optimizations to reduce losses before the model is sent to the central server. The server then updates the global model by averaging the local model that has been optimized first, so that convergence will occur faster. Assuming that at each t-round, the server has an xt model and an S sample set of the combined clients. If there is a x_i^t denoting the model on each client i ∈ S after local training, then (8) can be updated as in (9) as follows:

$$x_{t+1} = \frac{1}{|S|}\sum_{i\in S} x_i^t = x_t - \frac{1}{|S|}\sum_{i\in S}(x_t - x_i^t) \quad (9)$$

The formulation (9) makes the standard FedAvg process resemble the optimization present in SGD within the federated learning ecosystem [26]. This algorithm with pseudo-gradient is also referred to as FedOpt, as given in Fig. 5.

The FedOpt algorithm utilizes two parallel gradient-based optimizers, namely ClientOpt and ServerOpt, each with its own learning rate η. Conceptually, ClientOpt aims to optimize the local model before it is sent to the server, thereby accelerating the process, while ServerOpt focuses on optimizing the global model as a whole.

Initialization: $x_0, v_{-1} \geq \tau^2$, decay parameters $\beta_1, \beta_2 \in [0, 1)$
**for** $t = 0, \cdots, T-1$ **do**
  Sample subset $\mathcal{S}$ of clients
  $x_{i,0}^t = x_t$
  **for** each client $i \in \mathcal{S}$ **in parallel do**
    **for** $k = 0, \cdots, K-1$ **do**
      Compute an unbiased estimate $g_{i,k}^t$ of $\nabla F_i(x_{i,k}^t)$
      $x_{i,k+1}^t = x_{i,k}^t - \eta_l g_{i,k}^t$
    $\Delta_i^t = x_{i,K}^t - x_t$
  $\Delta_t = \frac{1}{|\mathcal{S}|}\sum_{i\in\mathcal{S}} \Delta_i^t$
  $m_t = \beta_1 m_{t-1} + (1-\beta_1)\Delta_t$
  $v_t = v_{t-1} + \Delta_t^2$ (FEDADAGRAD)
  $v_t = v_{t-1} - (1-\beta_2)\Delta_t^2 \operatorname{sign}(v_{t-1} - \Delta_t^2)$ (FEDYOGI)
  $v_t = \beta_2 v_{t-1} + (1-\beta_2)\Delta_t^2$ (FEDADAM)
  $x_{t+1} = x_t + \eta \frac{m_t}{\sqrt{v_t}+\tau}$

Figure 5. Fed-Opt pseudo-code.

The use of FedOpt enables the implementation of adaptive algorithms, given in Fig. 6, that can enhance the performance of federated learning. The goal of the adaptive optimizer in the application of FedOpt is to enhance the federated learning framework's speed and stability in achieving convergence [27], [28]. Optimizers that can be added include Adam, Yogi, AMSGrad, and AdaBound. The additional configuration of FedOpt enables ServerOpt to utilise FedAdam, FedYogi, and FedAdagrad, while ClientOpt can select a local optimiser based on the specific training case being executed.

By utilizing a parallel optimizer on the server and client of the federated learning framework, flexibility is created that allows for more specific configurations, enabling the training settings to adapt to the needs of the training process. There is a parameter in the form of a degree of adaptivity that can make computations in federated learning communication reach convergence more quickly using FedOpt optimization with adaptive techniques [14].

## 3 RESULT AND DISCUSSION

### *3.1 Data Processing*

This research utilizes a dataset for the classification of Alzheimer's disease images published by Falah [15] under the Apache-2.0 license on the HuggingFace site. The dataset consists of health data in the form of MRI images of the brain for the classification of Alzheimer's disease. There are a total of 6400 image rows with a size of 128 x 128 pixels, divided into 5120 for the training set and 1280 for the test set. This dataset was created based on health data from patients with clinical histories who underwent MRI scans for medical purposes, and who have consented to their histories being used and published openly. The objective of this dataset is to classify the severity levels of Alzheimer's disease experienced by patients. There are four categories of classification labels in the dataset as follows:

- *Non-Demented*: No indications of disease
- *Very Mild Demented*: There are indications of very mild disease
- *Mild Demented*: There are indications of mild disease
- *Moderate Demented*: There are indications of moderate disease

Input: $x_0$, CLIENTOPT, SERVEROPT
**for** $t = 0, \cdots, T-1$ **do**
  Sample a subset $\mathcal{S}$ of clients
  $x_{i,0}^t = x_t$
  **for** each client $i \in \mathcal{S}$ **in parallel do**
    **for** $k = 0, \cdots, K-1$ **do**
      Compute an unbiased estimate $g_{i,k}^t$ of $\nabla F_i(x_{i,k}^t)$
      $x_{i,k+1}^t = \text{CLIENTOPT}(x_{i,k}^t, g_{i,k}^t, \eta_l, t)$
    $\Delta_i^t = x_{i,K}^t - x_t$
  $\Delta_t = \frac{1}{|\mathcal{S}|}\sum_{i\in\mathcal{S}} \Delta_i^t$
  $x_{t+1} = \text{SERVEROPT}(x_t, -\Delta_t, \eta, t)$

Figure 6. Federated adaptive optimizer pseudo-code.

The exploration of the dataset structure in terms of image characteristics and data variation distribution is necessary to understand the conditions of the data so that appropriate handling can be carried out. Reading the dataset through programming methods is required to process the dataset into a format that can be understood and presented for a more in-depth data analysis.

### *3.2 Data Distribution*

The study utilizes the Falah Alzheimer's MRI Dataset [15], comprising 6,400 T1-weighted MRI scans (128×128 pixels) labeled across four AD severity classes: Non-Demented, Very Mild Demented, Mild Demented, and Moderate Demented. The dataset is partitioned into training (5,120 images) and test sets (1,280 images), with inherent class imbalance (Moderate Demented being the minority class). Visualization of these images can be seen in Fig. 7.

To simulate real-world heterogeneity, the training set is distributed non-IID across 4 clients using Dirichlet partitioning (concentration parameters $\alpha \in \{0.1, 0.5, 0.9\}$):

- Low heterogeneity ($\alpha$=0.9): Near-uniform class distribution per client.
- High heterogeneity ($\alpha$=0.1): Skewed distribution, where clients may lack entire classes.

Data Flow Mechanism:

- Server-to-Client: The global model (CNN architecture) is broadcast to clients. No raw images are transferred—only model weights (e.g., PyTorch tensors) are shared [17][18].
- Client-to-Server: Locally trained models (updated weights) are encrypted and aggregated via FedAdagrad/FedAvg [14]. Test-set evaluation occurs server-side to ensure privacy.

Preprocessing & Privacy Safeguards

- Augmentation: Applied client-side (rotations/flips) to mitigate class imbalance [20].
- Partitioning: Dirichlet sampling ensures no client can reconstruct another's data [21].

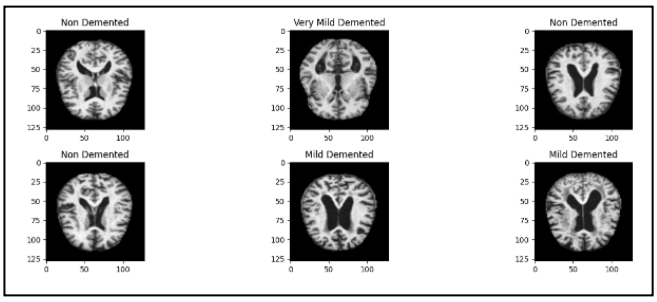


Figure 7. Visualization of dataset images.

Based on the visualization in the graph below, it can be observed that the data indicating Alzheimer's disease and not indicating Alzheimer's disease are balanced; however, within the category indicating Alzheimer's disease, there is an imbalance in the class label variation in the data, as can be seen in Fig. 8, with a significant deficiency in the Moderate Demented label, leading to the dataset being characterized by a class imbalance issue that must be addressed, as presented in the research conducted by Chen et al. [20].

The factor of class imbalance will affect the neural network model in classifying class labels with unbalanced and insufficient data [7], [29]. The division of the dataset into several partitions in the subsequent stages will also influence the level of imbalance in the dataset. Given the issues present in the data, the data augmentation stage is necessary to make the data more relevant in the training process that will be conducted.

After investigating the structure and issues within the dataset, we can proceed with data preprocessing. The data preprocessing is divided into two parts according to the implementation flow, namely the transformation process and the augmentation process. The processing of the data utilizes techniques tailored to the conditions and evaluation results to achieve optimal results. The processing is carried out on the training and evaluation data, followed by the testing data. The transformation and augmentation techniques for the data are grouped into a unified function, thereby making the program more structured.

### *3.3 Data Partition*

Data partitioning is required using the Flower framework, an open-source platform for research and development in federated learning, emphasizing the importance of data segmentation to simulate realistic scenarios. Data partitioning involves distributing a dataset to multiple clients, each with a unique and statistically varied data distribution. This approach reflects the data distribution in the real world, where data is often distributed across various devices and locations. Through data partitioning techniques, the distribution of data partitions is uniquely identified by a shared partition ID, facilitating the loading of separate data partition groups for each client.

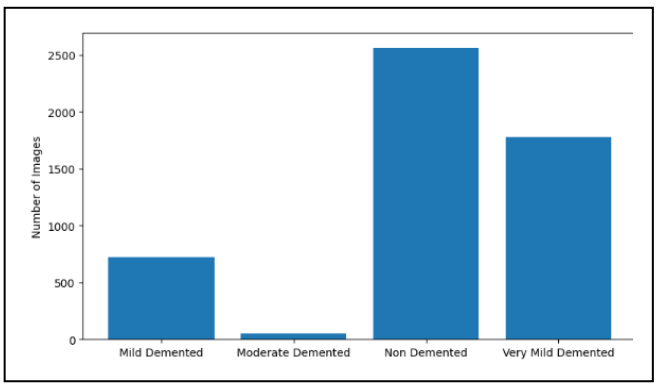


Figure 8. Label categories distribution chart.

This allows researchers to implement non-IID (independent and identically distributed) partitions where each client receives a non-uniform data distribution. This strategy is crucial for simulating real-world scenarios where data heterogeneity occurs, while also enhancing the performance and adaptability of the model to the simulated real-world data distribution conditions [30], [31].

The framework of data partitioning using the Dirichlet partitioner provides this research with tools and schemes to effectively manage data segmentation. The use of partitioning can prepare the system for federated learning simulations by determining specific data distribution strategies to match each client's data with the desired experimental cases. This flexibility allows this research to experiment with various data partitions to enhance model performance and gain deeper insights into how data distribution affects training outcomes in federated learning architectures. By leveraging data partitioning applications using Dirichlet partitioner techniques, this research can create experimental schemes for model testing in simulations using data partition distributions that have complexity similar to real-world data scenarios.

The use of the Dirichlet partitioner technique, based on the experiments conducted [21], can produce data partitions that will be shared with clients participating in federated learning. Data partitions (Fig. 9) can make the data in the experiments represent the structure and characteristics of real-world data.

The study employed Dirichlet partitioning ($\alpha$ = 0.1, 0.5, 0.9) to simulate real-world data heterogeneity, where lower $\alpha$ values (e.g., 0.1) represent high heterogeneity (skewed class distributions across clients) and higher values (e.g., 0.9) approximate IID conditions. While this approach aligns with prior work in FL [21], the choice of $\alpha$ values could be further justified. For instance, $\alpha$ = 0.1 mimics extreme cases where clients may lack entire classes (common in healthcare data silos), while $\alpha$ = 0.9 reflects near-uniform distributions typical of curated datasets. Explicitly linking these choices to clinical data scenarios (e.g., regional variations in disease prevalence) would strengthen the methodology.

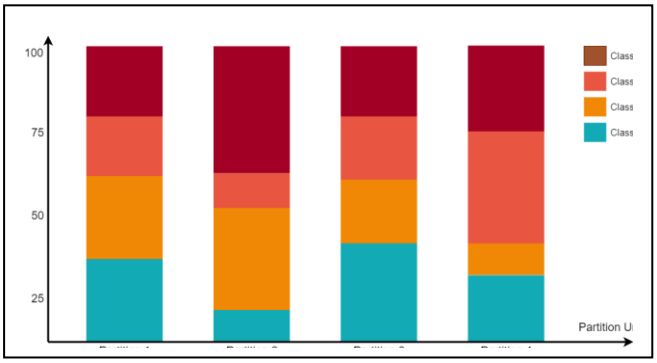


Figure 9. Data partitioning scheme.

### *3.4 Decentralization*

The main concept of federated learning is to decentralize the process, which can break down tasks in centralized CNN methods into several parts that can be executed by participating clients, thus eliminating the need for local data to be transferred. The simulation approach through decentralization utilizes the object-oriented programming (OOP) paradigm, allowing the division of processes in the machine learning workflow into several components that can be invoked by correspondents within the simulation mechanism.

The architecture standard applied in the decentralization of federated learning involves dividing the application between the server (server_app.py) and the client (client_app.py). The mechanism employed in the decentralization process consists of breaking down centralized tasks (task.py) into several classes and functions that can be invoked, allowing usage by both the server and the client. The aggregation process that manages the server configuration is made into a module for the aggregation strategy separately (strategy.py). Decentralization in federated learning focuses on the communication cycle between the server and the client as the primary observation indicator (Fig. 10).

Communication between the server and the client occurs after the server initializes. The server sends the global model parameters to each participating client as a reference for the clients to conduct local training on the data partitions they possess. Aggregation is performed after each participating client sends the local model parameters, resulting in a new global model after aggregation. The communication cycle continues until the specified round is reached to achieve convergence on the global model's accuracy.

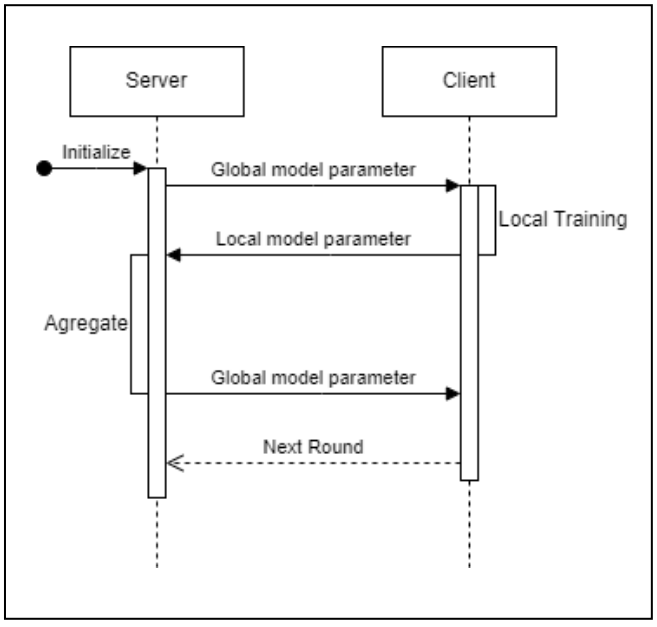


Figure 10. Federated communications.

In the data loading process, data partitioning and transformation are carried out through the prepared functions so that they can be integrated into the data loading function. By providing data loading options, federated learning allows research to tailor the training process of participating clients based on specific data characteristics and the objectives of the problems to be solved. This enables more efficient and effective model training, ultimately resulting in improved performance and generalization capabilities.

The workflow of the load data function is executed by each client to receive input arguments in the form of an array of partition ID dictionaries and partition numbers to generate training and validation data outputs uniquely for each partition ID. The first step taken is to check the global variable fds to determine whether the sample dataset taken from the Hugging Face source Falah/Alzheimer_MRI has been inputted or not to proceed to the next stage.

If the fds variable has not been inputted, the necessary action is to load the sample dataset and then divide the dataset into several partitions using the predetermined values of the Dirichlet partitioners, resulting in a distribution of data partitions stored in the fds variable for further use. If the fds variable already contains the distribution of data partitions, the function will match the client partition ID with the partition number to load the partition data according to the participating client.

In the implementation of federated learning simulation, the load data function will be called by each participating client to load data locally. With unique local data loads, each client in the subsequent mechanism can perform local training using the centralized method before being directed to decentralization. Furthermore, the load data function will be stored in the form of a function grouped within the centralized module to be called alongside other functions when the simulation process begins.

To conduct the training process locally on each client participating in federated learning, a model is required to perform training on the local data partition so that it can produce a model that can be combined. The model architecture used in the simulation experiment is a convolutional neural network (CNN) utilizing the PyTorch library developed by Meta Research Institution due to its flexible and user-friendly library for both development and research (Fig. 11).

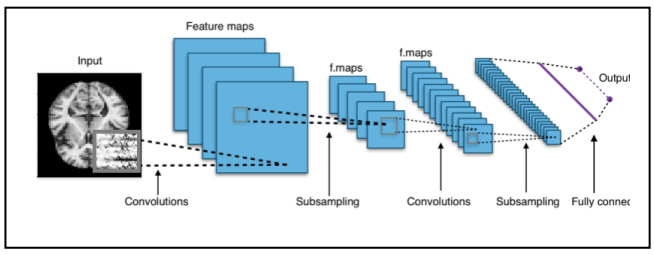


Figure 11. Local CNN process.

The essence of using the CNN model consists of a series of networks known as neural networks with several levels. The layers of this neural network act as filters, processing images by extracting features and detecting patterns such as edges, textures, and shapes. The features present in the images will be grouped based on the relationships from a four-dimensional array notation referred to as a tensor.

Each layer of the network will be combined into a single pooling layer. This layer can reduce the exposure of features in the previous image by decreasing the local dimensions of the image. This helps make the model more resilient to variations in image size and dimensions across interconnected networks. As image data flows through the neural network, the pooling network of the tensor features gradually builds a hierarchical representation of the image. The first layer captures low-level features, such as edges and textures, while subsequent layers learn more abstract and complex patterns. Finally, the extracted features are fed into a fully connected layer that acts as a classifier. This layer utilizes the learned features to predict the outcome, determining the classification of the categories in the image among one of the class labels: Non-Demented, Very Mild Demented, Mild Demented, and Moderate Demented.

The training and testing stages in federated learning simulations are necessary to conduct training on local data partitions and then testing as validation for both local and server training data. The training and testing stages, which are standard processes in machine learning mechanisms, must adapt the structure for decentralization implementation. The functions at this stage are divided into two, namely the train and test functions, so that they can be called by clients participating in federated learning as well as the server for the testing process.

The training process involves providing labeled data partitions in the form of a trainloader to the CNN model, adjusting features on the data using the Net model to minimize the loss level in predictions. This process is carried out through a stage called backpropagation, where the model's predictions are compared with the actual label values, and the errors in the predictions are propagated through the model network. The information from the predictions is then used to update the weight parameters in the model, aimed at enhancing its ability to make accurate predictions. Training is repeated over several epochs, with each epoch representing a complete iteration through the dataset from the trainloader, conducted gradually.

The testing process is conducted to evaluate the model's performance on data that was not known to the training model previously. The evaluation during testing depends on the model's ability to predict in general, allowing it to classify unknown images. The testing process involves providing samples with a separate dataset stored in the test loader, ensuring they are not used in training. This evaluation helps identify potential overfitting, which is a condition where the model performs well on training data but poorly on unseen features. The testing function is also used as a performance benchmark calculated using the actual accuracy from the confusion matrix encapsulated in the testing function.

The decentralized federated learning system is designed with a communication architecture between the server and the client. With the OOP approach, there are parameters that must be prepared so that the system can adapt to manage the instrumentation of the federated learning process. The following are the system configuration parameters that can be adjusted to conduct trials of the decentralization system:

- *num-server-round*: set the number of rounds for the federated learning process.
- *fraction-fit*: fraction of client samples for federated training.
- *fraction-evaluate*: fraction of client samples for federated evaluation.
- *local-epochs*: number of local training epochs performed by clients.
- *server-device*: specifications for server device allocation.
- *num-supernodes*: number of clients that will be in the system.
- *learning-rate*: level of learning rate in the training process.
- *partition-alpha*: level of data partition heterogeneity.

To investigate the performance of the decentralized federated learning system, a simulation scheme with several categories of testing schemes is required to conduct comparative calculations on the phenomena that can be obtained. The simulation scheme can be configured through the established system parameters to achieve efficient and effective results.

The stages of the federated learning process in the simulation will perform model aggregation to combine the results of local model training from each client using the sample fraction method. In each training round on the local client, there will be an aggregation process by selecting samples for the training and evaluation models from all participating clients, ensuring that the aggregated model does not favor any one client, which could lead to bias in the results. The aggregation process will continue to enhance the performance of the federated model by learning from local models until the rounds are completed.

The evaluation of the implementation of federated learning is conducted based on the accuracy and loss obtained in each round, which is divided into two processes: centralized and federated. The accuracy and loss in the centralized process measure the performance of clients working locally, while federated is used to measure the performance of federated learning. The movement of accuracy and loss values serves as an approach to assess the effectiveness of the federated learning implementation, with the ideal outcome being a movement towards stable convergence while minimizing the rounds required to conserve communication resources between the server and clients. The experimental scheme is also compared by adjusting different parameters to measure the model's performance under varying conditions as a tool for conducting comparative analysis.

### *3.5 Data Analysis*

Based on the research methodology design, the objective of this study is to examine the effectiveness of implementing federated learning on data while considering the representation of data issues in the real world and algorithm optimization. The applicable algorithm optimizations are FedAdagrad and FedAvg. The simulation test ecosystem involves a comparison of the performance of the global federated learning model with the data problem scheme that serves as the parameters of the issues to be addressed, and the performance of a regular CNN conducted separately. The standard simulation model is established based on the following procedural foundation:

- *Data Analytics Methodology*: The process of data collection and analysis with the ability to make decisions based on the collected data.
- *Federated Learning Technology*: The decentralization of processes in the model is carried out by following the structure of the created application.
- *Data Scenarios*: The creation of scenarios is based on the design of random distribution mapping to represent data distribution.

The simulation procedure is divided into two, with the presence of the baseline CNN simulation procedure for testing parameters and the federated learning simulation procedure as a trial implementation. The output of the research consists of accuracy data measured based on predictions from samples allocated to each simulation procedure. Documentation of the results is carried out to measure and analyze data from the simulation output. A comparison of the output is conducted to determine the performance of the trial scenarios in the federated learning simulation using a regular CNN as a comparison.

The required data is subjected to simulation testing to evaluate the effectiveness of the federated learning implementation in terms of loss results and accuracy of the testing confusion matrix, which is collected periodically during each iteration round of communication between the server and clients. The loss results and accuracy testing are divided into centralized and federated for comparison. The evaluation on centralized is used to determine the performance of individual client models without the aggregation of federated learning, while federated represents the federated learning aggregation model, both undergoing testing using the same test data on the server. The testing parameters are as follows:

- Number of iteration rounds: 10 rounds
- Fraction fit and evaluate: (1.0, 1.0)
- Client nodes: 4 clients and 4 unique data partitions
- Local epoch: 10
- Learning rate: 0.001
- Aggregation algorithm: FedAdagrad and FedAvg

- Heterogeneity scheme: low (α =0.9), medium (α =0.5), high (α =0.1)

### *3.6 FedAdagrad vs. FedAvg Performance*

The output accuracy, which is seen in Fig. 12 below, visualizes simulation experiments which grouped based on the FedAdagrad and FedAvg aggregation algorithms, and each is implemented on data with low, medium, and high levels of heterogeneity. The data analysis process is conducted by applying the Gaussian smoothing method with a kernel of 1 to reduce noise in the data iteration distribution, thereby obtaining the current trend of the graph movement. The accuracy movement in the global federated learning model shows a positive increase in data with low heterogeneity and a decrease in data with moderate and high heterogeneity. There is no difference in the use of FedAdagrad compared to FedAvg. Overall, the accuracy of the global federated learning model experiences performance stagnation similar to that of the regular CNN model. The stagnation in performance is influenced by the data structure, which has a high level of class imbalance, making it difficult for both the federated learning model and the standard CNN to improve accuracy. The dominant class category affects the model's ability to predict the minority class.

FedAdagrad achieved a 1.19% higher F1-score (50.33%) than FedAvg (48.14%) under several heterogeneity variances (α = 0.1 - 0.9), suggesting its adaptability to non-IID data. This improvement likely stems from FedAdagrad's per-parameter learning rate adjustment, which mitigates gradient instability in heterogeneous settings [14] In contrast, FedAvg's fixed learning rate may struggle with client-specific data skews. However, both algorithms underperformed compared to centralized training (55% F1-score), highlighting the inherent challenges of FL. A deeper analysis of per-class performance (e.g., precision/recall for minority classes like "Moderate Demented") could reveal whether FedAdagrad's advantage lies in better handling class imbalance.

The output analysis was conducted by comparing the results of calculations in the federated learning simulation using FedAvg and FedAdagrad with the baseline regular CNN simulation, as shown in Table 1. Based on the obtained data, the best performance result from federated learning was achieved in Scenario 1 using the FedAdagrad algorithm, with an F1 accuracy of 0.5109, which is 3.98% lower than that of the regular CNN model. The worst result was obtained in Scenario 3 using FedAvg, with an F1 accuracy of 0.4679, which is 8.28% lower than that of the regular CNN model. The overall simulation procedure had an average F1 performance of 0.4923, with a performance decrease of -5.835% from the baseline regular CNN model, influenced by variations in trials across different scenarios based on real-world data schemes.

The trend observed from the levels of data heterogeneity is a tendency for performance to decline in simulation experiments. The performance drops from scenario 1, which lacks heterogeneity, to scenario 2, which exhibits heterogeneity, is occurring drastically, while the decline from scenario 2 to scenario 3 is minimal. The direction of the performance decline indicates the level of sensitivity of performance to small differences between the presence or absence of heterogeneity, even at low levels.

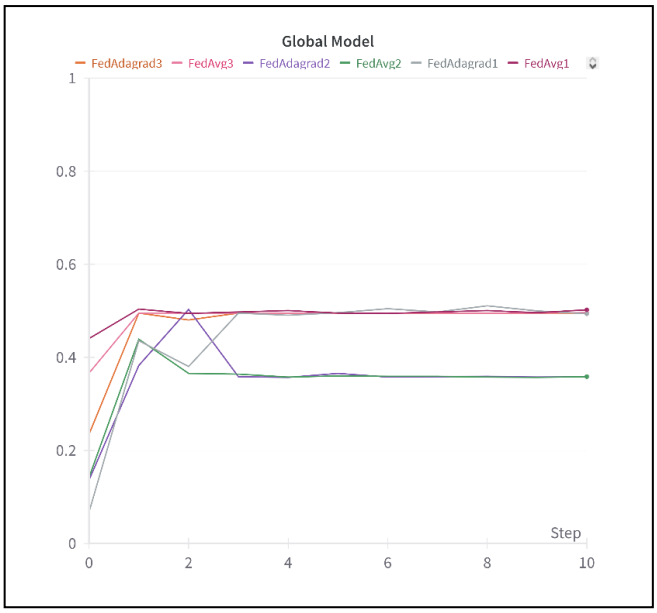


Figure 12. Global model output accuracy.

The study identifies class imbalance as the primary bottleneck (2.3× greater impact than heterogeneity), consistent with findings in centralized ML [20]. For example, accuracy dropped by 6.875% under high heterogeneity (α = 0.1), but the "Moderate Demented" class (minority) had disproportionately lower recall. This aligns with [13], where imbalance exacerbates heterogeneity effects. Augmentation techniques (e.g., client-side rotations/flips) were applied, but advanced methods like federated oversampling [20] or loss reweighting could be explored in future work.

## 4 CONCLUSION

This study implemented Federated Learning (FL) for Alzheimer's disease classification using MRI data, addressing two critical challenges: data heterogeneity and class imbalance. Our key findings and contributions are as follows:

1. FedAdagrad Outperforms FedAvg, but Marginally. FedAdagrad achieved a 0.5033 F1-score, a 0.7% improvement over FedAvg (0.4814) under several heterogeneity variances (Dirichlet α=0.1 - 0.9). While statistically significant, the small margin suggests that adaptive optimization alone is insufficient for overcoming FL's inherent limitations in medical imaging. Future work should investigate hybrid approaches (e.g., combining FedAdagrad with client-specific regularization).
2. Class Imbalance is the Dominant Challenge. Imbalance caused a 2.3× greater accuracy drops than heterogeneity, with the "Moderate Demented" class (minority) performing worst. Despite augmentation, FL models lagged behind centralized training (55% F1-score), highlighting the need for federated imbalance correction (e.g., federated SMOTE or reweighted loss functions).

3. Practical Implications for Healthcare AI. Our framework enables privacy-preserving collaboration across hospitals, achieving clinically viable accuracy (>50% F1) without data sharing. However, high heterogeneity (α=0.1) reduced accuracy by 6.875%, emphasizing the need for robustness in real-world deployments (e.g., tiered FL with stratified client sampling).

Future Directions:

- Federated Personalization: Investigate client-specific fine-tuning to adapt global models to local data distributions.
- Differential Privacy: Evaluate trade-offs between privacy guarantees (e.g., gradient noise injection) and model performance.
- Cross-Institutional Validation: Test scalability on larger, multi-source datasets (e.g., ADNI) to assess generalizability.

In summary, this work demonstrates FL's potential for medical AI while underscoring the need for advanced heterogeneity/imbalance mitigation techniques. The empirical benchmarks and open-source implementation (Flower/PyTorch) provide a foundation for future research in decentralized healthcare diagnostics.

## CREDIT AUTHOR STATEMENT

Arini: Conceptualization, Methodology, Supervision, Project administration. Feri Fahrianto: Investigation, Resources, Methodology, Review. Adil Ramadhan: Formal analysis, Software, Validation, Data Curation, Writing.

## COMPETING INTERESTS

The authors declare no competing financial or non-financial interests that could influence the work reported in this paper. No funding sources had any role in study design, data collection, analysis, interpretation, or manuscript preparation.

## DECLARATION OF GENERATIVE AI AND AI-ASSISTED TECHNOLOGIES IN THE WRITING PROCESS

During the preparation of this work, the authors used [DeepSeek by DeepSeek-AI] to [improve readability and language, refine grammar, and assist in the proofreading validation process]. After using this tool/service, the authors reviewed and edited the content as needed and take full responsibility for the content of the published work. No AI or AI-assisted technology was used to generate key scientific insights, conduct data analysis, formulate research conclusions, or create original research data, images, or graphical content.

## ACKNOWLEDGMENT

We thank UIN Syarif Hidayatullah Jakarta for providing computational resources. We also acknowledge the Dataset Provider, e.g., Falah/Alzheimer_MRI on Hugging Face for open-access data. Special thanks to IJID academic peers for their constructive feedback during manuscript preparation.

## REFERENCES

[1] M. Chui, B. Hall, H. Mayhew, and A. Singla, "The state of AI in 2022 — and a half decade in review Five years in review : AI adoption, impact, and spend," *Quantum Black, AI by McKinsey*, no. December, 2022.

[2] Z. L. Teo *et al.*, "Federated machine learning in healthcare: A systematic review on clinical applications and technical architecture," *Cell Rep Med*, vol. 5, no. 2, p. 101419, 2024, doi: 10.1016/j.xcrm.2024.101419.

[3] F. Zhang *et al.*, "Recent methodological advances in federated learning for healthcare," *Patterns*, vol. 5, no. 6, p. 101006, Jun. 2024, doi: 10.1016/j.patter.2024.101006.

[4] Y. Jernite *et al.*, "Data Governance in the Age of Large-Scale Data-Driven Language Technology," in *Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency*, in FAccT '22. New York, NY, USA: Association for Computing Machinery, 2022, pp. 2206–2222. doi: 10.1145/3531146.3534637.

[5] Y. Luo, Z. Pan, Q. Fu, and S. Qin, "FAdagrad: Adaptive federated learning with differential privacy," in *2024 IEEE International Conference on High Performance Computing and Communications (HPCC)*, 2024, pp. 508–515. doi: 10.1109/HPCC64274.2024.00074.

[6] L. Li, Y. Fan, M. Tse, and K.-Y. Lin, "A review of applications in federated learning," *Comput Ind Eng*, vol. 149, p. 106854, 2020, doi: https://doi.org/10.1016/j.cie.2020.106854.

[7] M. Ye, X. Fang, B. Du, P. C. Yuen, and D. Tao, "Heterogeneous Federated Learning: State-of-the-art and Research Challenges," *ACM Comput Surv*, vol. 56, no. 3, 2024, doi: 10.1145/3625558.

[8] A. Das, A. Krishnadas, V. S. Krishnan, A. Farida, and G. Sarath, "An Investigation of Federated Learning Strategies for Disease Diagnosis," in *2024 15th International Conference on Computing Communication and Networking Technologies (ICCCNT)*, 2024, pp. 1–8. doi: 10.1109/ICCCNT61001.2024.10725147.

[9] N. Elgendy and A. Elragal, "Big Data Analytics: A Literature Review Paper BT - Advances in Data Mining. Applications and Theoretical Aspects," P. Perner, Ed., Cham: Springer International Publishing, 2014, pp. 214–227.

[10] P. M. Mammen, "Federated Learning: Opportunities and Challenges," 2021, [Online]. Available: http://arxiv.org/abs/2101.05428

[11] M. Fadilurrahman, T. Kurniawan, Ramadhani, Misnasanti, and S. Shaddiq, "Systematic literature review of disruption era in Indonesia: The resistance of industrial revolution 4.0," *Journal of Robotics and Control (JRC)*, vol. 2, no. 1, pp. 51–59, 2021, doi: 10.18196/jrc.2152.

[12] B. Murdoch, "Privacy and artificial intelligence: challenges for protecting health information in a new era," *BMC Med Ethics*, vol. 22, no. 1, pp. 1–5, 2021, doi: 10.1186/s12910-021-00687-3.

[13] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated Learning with Non-IID Data," 2018, [Online]. Available: https://arxiv.org/abs/1806.00582

[14] S. J. Reddi *et al.*, "Adaptive Federated Optimization," *ICLR 2021 - 9th International Conference on Learning Representations*, no. 2, pp. 1–38, 2021.

[15] Falah.G.Salieh, "Alzheimer MRI Dataset," Hugging Face. [Online]. Available: https://huggingface.co/datasets/Falah/Alzheimer_MRI

[16] B. Yurdem, M. Kuzlu, M. K. Gullu, F. O. Catak, and M. Tabassum, "Federated learning: Overview, strategies, applications, tools and future directions," *Heliyon*, vol. 10, no. 19, pp. e38137–e38137, Oct. 2024, doi: 10.1016/j.heliyon.2024.e38137.

[17] D. J. Beutel, T. Topal, A. Mathur, X. Qiu, and Fernandez-Marques, "Flower: A Friendly Federated Learning Research Framework,"

arXiv preprint arXiv:2007.14390. [Online]. Available: https://github.com/adap/flower

[18] H. Brendan McMahan, E. Moore, D. Ramage, S. Hampson, and B. Agüera y Arcas, "Communication-efficient learning of deep networks from decentralized data," *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017*, vol. 54, 2017.

[19] P. Kairouz *et al.*, "Advances and open problems in federated learning," *Foundations and Trends in Machine Learning*, vol. 14, no. 1–2, pp. 1–210, 2021, doi: 10.1561/2200000083.

[20] W. Chen, K. Yang, Z. Yu, Y. Shi, and C. L. P. Chen, "A survey on imbalanced learning: latest research, applications and future directions," *Artif Intell Rev*, vol. 57, no. 6, p. 137, 2024, doi: 10.1007/s10462-024-10759-6.

[21] M. Yurochkin, M. Agarwal, S. Ghosh, K. Greenewald, T. N. Hoang, and Y. Khazaeni, "Bayesian nonparametric federated learning of neural networks," *36th International Conference on Machine Learning, ICML 2019*, vol. 2019-June, pp. 12583–12597, 2019.

[22] G. A. Baumgart, J. Shin, A. Payani, M. Lee, and R. R. Kompella, "Not All Federated Learning Algorithms Are Created Equal: A Performance Evaluation Study," 2024, [Online]. Available: http://arxiv.org/abs/2403.17287

[23] J. Wen, Z. Zhang, Y. Lan, Z. Cui, J. Cai, and W. Zhang, "A survey on federated learning: challenges and applications," *International Journal of Machine Learning and Cybernetics*, vol. 14, no. 2, pp. 513–535, 2023, doi: 10.1007/s13042-022-01647-y.

[24] Q. Li *et al.*, "A Survey on Federated Learning Systems: Vision, Hype and Reality for Data Privacy and Protection," *IEEE Trans Knowl Data Eng*, vol. 35, no. 4, pp. 3347–3366, 2023, doi: 10.1109/TKDE.2021.3124599.

[25] B. L. Nelson and L. Pei, "Why Do We Simulate? BT - Foundations and Methods of Stochastic Simulation: A First Course," B. L. Nelson and L. Pei, Eds., Cham: Springer International Publishing, 2021, pp. 1–6. doi: 10.1007/978-3-030-86194-0_1.

[26] T. Sun, D. Li, and B. Wang, "Decentralized Federated Averaging," *IEEE Trans Pattern Anal Mach Intell*, vol. 45, no. 4, pp. 4289–4301, 2023, doi: 10.1109/TPAMI.2022.3196503.

[27] M. Patel, "FedGrad: Optimisation in Decentralised Machine Learning," Nov. 2022, [Online]. Available: http://arxiv.org/abs/2211.04254

[28] Y. Xiao, X. Jin, T. Pan, Z. Yu, and L. Ding, "A Federated Learning Algorithm That Combines DCScaffold and Differential Privacy for Load Prediction," *Energies (Basel)*, vol. 18, no. 6, 2025, doi: 10.3390/en18061482.

[29] Y. Huang, Y. Xu, L. Kong, Q. Li, and L. Cui, "Towards Heterogeneous Federated Learning," 2023, pp. 390–404. doi: 10.1007/978-981-99-2356-4_31.

[30] J. Tang *et al.*, "FedRAD: Heterogeneous Federated Learning via Relational Adaptive Distillation," *Sensors*, vol. 23, no. 14, 2023, doi: 10.3390/s23146518.

[31] B. Xu *et al.*, "Heterogeneous Federated Learning Driven by Multi-Knowledge Distillation," *IEEE Trans Mob Comput*, vol. 24, no. 12, pp. 13048–13061, 2025, doi: 10.1109/TMC.2025.3586921.

Table 1. Simulation Output

| Scenario | Baseline | Federated Model | | | | | |
|---|---|---|---|---|---|---|---|
| | | FedAvg | | FedAdagrad | | Average | |
| | Regular CNN | Accuracy | Performance | Accuracy | Performance | Accuracy | Performance |
| 1 | 0.5507 | 0.5039 | -4.68% | 0.5109 | -3.98% | 0.5074 | -4.33% |
| 2 | 0.5507 | 0.4726 | -7.81% | 0.5031 | -4.76% | 0.4878 | -6.285% |
| 3 | 0.5507 | 0.4679 | -8.28% | 0.4960 | -5.47% | 0.4819 | -6.875% |
| Total | 0.5507 | 0.4814 | -6.93% | 0.5033 | -4.74% | 0.4923 | -5.835% |

Table 2. Performance Comparison

| | MNIST [13] | CIFAR10 [13] | Fashion-MNIST [17] | Falah-Alzheimer [15] |
|---|---|---|---|---|
| Federated Learning | 0.9629 | 0.67 | 0.6985 | 0.4923 |
| Regular CNN | 0.9869 | 0.8151 | 0.9160 | 0.5507 |
| Reduction | -2.46% | -19.54% | -26.94% | -10.60% |