**International Journal of Applied Mathematics**

**Volume 38 No. 1s, 2025**

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

# PRIVACY-AWARE FEDERATED LEARNING ALGORITHM FOR HEALTHCARE IOT USING OPTIMIZED VGG16 WITH NOISE ADJUSTMENT

**[1]Jyoti L. Bangare, [2]Nilesh P. Sable, [3]Parikshit N. Mahalle, [4]Gitanjali R. Shinde**

[1]Ph.D. Scholar, Department of Computer Engineering, Vishwakarma Institute of Information Technology, Pune - 411048. Savitribai Phule Pune University, Pune, India.

[2]Associate Professor, Department of Computer Science & Engineering (Artificial Intelligence), BRACT's, Vishwakarma Institute of Information Technology, Pune, India. Savitribai Phule Pune University, Pune, India.

[3]Dean R&D, Professor and Head, Department of Artificial Intelligence and Data Science, Vishwakarma Institute of Technology ,Pune, India. Savitribai Phule Pune University, Pune, India.

[4]Associate Professor, BRACT's Vishwakarma Institute of Information Technology, Pune, India. Savitribai Phule Pune University, Pune, India.

jyoti.bangare@cumminscollege.in, drsablenilesh@gmail.com, aalborg.pnm@gmail.com, gr83gita@gmail.com

**Abstract**

As healthcare technologies quickly improve, data protection and security have become very important issues. This is especially true as more and more Internet of Things (IoT) devices are added to healthcare systems. Federated learning (FL) seems like a good way to deal with these issues because it allows for decentralised model training while keeping private patient data kept locally. Existing shared learning algorithms, on the other hand, often have problems like models that don't work well together, data leaks, and threats from other computers. This article suggests a Privacy-Aware Federated Learning (PAFL) algorithm for Internet of Things (IoT) systems in healthcare. It uses an improved VGG16 deep learning design and a noise adjustment method. The suggested method improves the safety and performance of healthcare IoT apps by making sure that private data is never sent between devices. This way, models can still be trained on decentralised datasets successfully. Our method uses a noise adjustment system to hide gradients while the model is being updated. This lowers the risk of private information getting out and makes the model more resistant to possible hostile attacks. Because VGG16 has been optimised, model convergence is more accurate and efficient. This makes it a good choice for healthcare settings where making decisions quickly and in real time is important. The suggested Privacy-Aware Federated Learning algorithm

Received: July 20, 2025

International Journal of Applied Mathematics

Volume 38 No. 1s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

does better than other federated learning methods in protecting privacy, making accurate models, and using computers more efficiently. This is especially true in healthcare IoT devices that don't have a lot of resources.

**Keywords:** Federated Learning, Healthcare IoT, VGG16 Optimization, Data Privacy, Adversarial Resistance

## I. Introduction

The Internet of Things (IoT) has opened up a new era of medical progress by letting doctors keep an eye on patients in real time, give them more personalised care, and improve their health. Wearable tech, sensors, and mobile health apps are all examples of IoT devices that collect huge amounts of data that is essential for identifying and treating a wide range of health problems. But as IoT devices become more common, privacy and security issues have become very important. Because healthcare data is private, hackers love to target it. The need to send huge amounts of patient data to central computers for analysis and model training makes the risk of data breaches even higher. To meet this challenge, we need new ways to protect patient privacy while still using the insights that data-driven machine learning models can give us. Federated learning (FL) has come up as a possible way to deal with these privacy issues in decentralised systems. FL lets many Internet of Things (IoT) devices or healthcare facilities train machine learning models together without sharing raw data. In shared learning, model updates are done directly on each device or school. Only the changes that are added together are sent to a central computer. This method keeps patient data close to home, protecting privacy and lowering the risks that come with sending data. Federated learning seems like a good idea, but it still has problems, like making sure that models stay accurate, protecting data from different types of attacks, and dealing with hostile attacks. One big worry about shared learning is that privacy could be broken when model changes are sent. The model changes (i.e., gradients) can still show private data about the underlying data, even though raw data is never sent back and forth [1]. These changes can be used by adversarial attacks, like gradient inversion or model inversion, to get private details about patients, like their health states or treatment records.

Also, IoT devices used in healthcare settings often don't have a lot of computing power, which can affect how well and quickly pooled learning algorithms work. This study suggests a Privacy-Aware Federated Learning (PAFL) approach that combines a noise-adjustment method with an optimised VGG16 deep learning model to get around these problems. The VGG16 model is a well-known convolutional neural network (CNN) design that has done very well at sorting images into different categories [2]. However, it can also be changed to work in healthcare settings, where it can look at medical pictures like X-rays, MRIs, or tissue slides. In Figure 1, see how privacy-aware collaborative learning and safe decentralised models work together to make healthcare IoT work better.

Received: July 20, 2025

International Journal of Applied Mathematics
Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

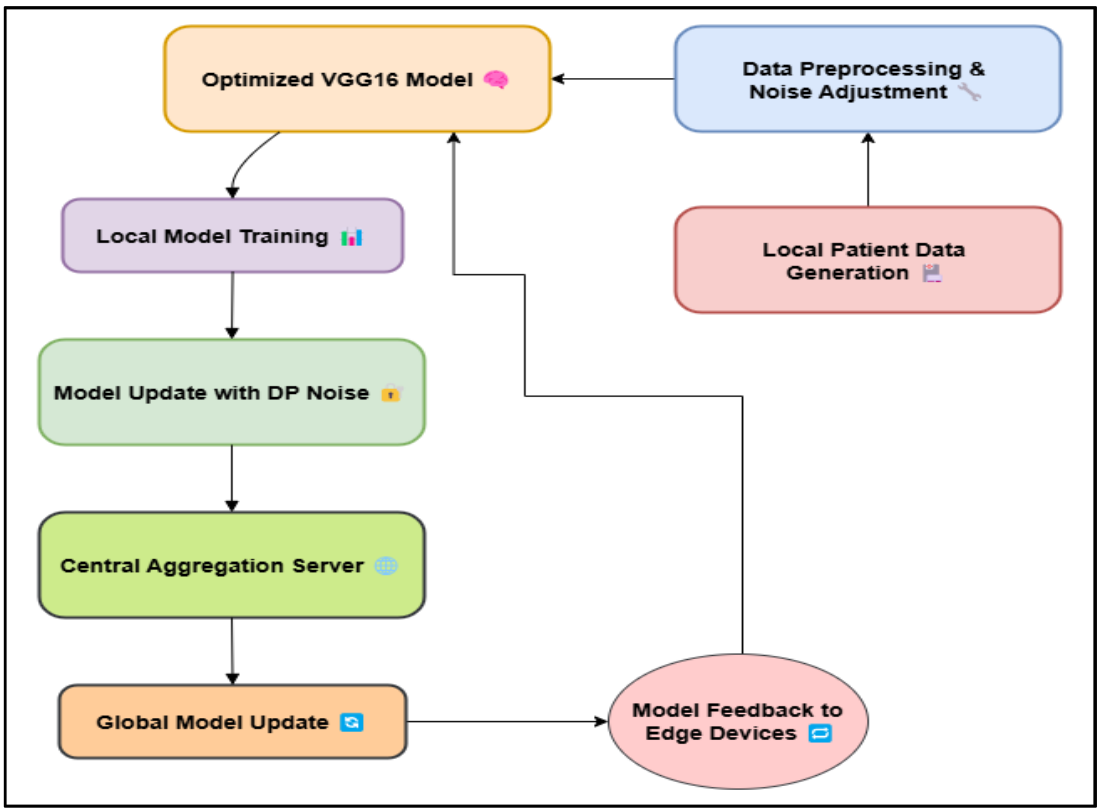


Figure 1: Privacy-Aware Federated Learning Algorithm for Healthcare IoT

This makes it desirable for healthcare settings in which decisions need to be made proper away. At some stage in education, the noise adjustment approach adds random noise to the model adjustments. This makes it tougher for attackers to get personal facts from the gradients. By using adding controlled noise to the gradient changes, the system improves privateness safety at the same time as nevertheless letting the fashions be trained nicely [3]. This method makes the collaborative learning version more stable; because of this it can manage threats from other fashions higher barring affecting its overall performance. The safety-conscious Federated learning technique for healthcare IoT is supposed to strike a balance among 2 very essential elements: the safety of the facts and the overall performance of the fashions. The program protects patient privateness through keeping non-public affected person records domestically and using a noise adjustment technique that doesn't display the patient's identification. at the identical time, VGG16's optimisation makes positive that the shared model stays accurate and effective, which makes it perfect for healthcare IoT structures that need to work in actual time and with restricted sources [4]. This paper offers a complete assessment of the recommended algorithm and shows that it may do better than not unusual collaborative mastering techniques in protecting privacy, making accurate fashions, and the use of less computing electricity [5]. The effects show that the recommended technique can work well with healthcare IoT systems, permitting secure and personal system learning to happen in decentralised settings.

Received: July 20, 2025

International Journal of Applied Mathematics

Volume 38 No. 1s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

## II. Related Work

A lot of attention has been paid to federated learning (FL) in healthcare systems because it can protect data privacy while allowing joint machine learning. Several studies have looked into collaborative learning methods that can be used in healthcare applications. However, there are still problems with data leaks, model accuracy, and computational efficiency, especially in IoT settings with limited resources. Protecting privacy is an important area of study in cooperative learning for healthcare. Traditional shared learning keeps basic data on the devices that use it, but sharing model changes (called gradients) can still make private data public. Several privacy-protecting methods have been suggested as ways to lower this risk [6]. One popular method is differential privacy (DP), in which noise is added to the slopes when the model is updated to hide private data. The idea of federated average was first put forward by McMahan et al. It has since been the basis of many federated learning algorithms. Recently, DP and shared learning have been used together in studies to improve privacy. For example, Geyer et al. suggested adding differential privacy to collaborative learning to make sure that privacy is better when updating models [7][8]. These methods do a good job of lowering privacy risks, but because they add noise, they often mean giving up privacy security for model success. Researchers have also looked into tactics that try to stop collaborative learning. Techniques like gradient inversion, in which criminals use slopes to figure out private data, are very dangerous for healthcare apps [9].

Researchers have come up with safe combination methods and security techniques to deal with these kinds of flaws. Secure multi-party computation (SMC) is an interesting method because it lets many people work together to compute the model parameters without showing each other their own changes. Shokri et al. created a safe aggregation method that makes sure gradients are combined in a way that keeps each person from seeing other participants' data. This makes collaborative learning even safer for privacy [10]. Because IoT devices don't have a lot of resources, computing speed is a big deal in hospital IoT. To get both high accuracy and low processing waste, it is important to make deep learning models work best for collaborative learning. A deep convolutional neural network (CNN) design called VGG16 has been used a lot in picture processing jobs because it is reliable and accurate. Several studies have shown that it could be used in healthcare for things like classifying medical images. However, VGG16's big number of factors can make it hard to run on some computers, especially when used on Internet of Things (IoT) devices [11]. To fix this, new work has been done to make VGG16 better for group learning. This is done by either making the model smaller or changing the design to make it more efficient while keeping the same speed. Zhang et al. offered a way to trim models for VGG16 that would require less computing power without reducing accuracy too much.

Received: July 20, 2025

**International Journal of Applied Mathematics**

**Volume 38 No. 1s, 2025**

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Table 1: Summary of Related Work

| Approach/Method | Privacy Mechanism | Application Domain | Key Finding |
|---|---|---|---|
| Federated Averaging | No Privacy | General ML | Introduced federated learning framework |
| Differential Privacy in FL | Differential Privacy | General ML | Introduced DP in FL for privacy |
| Secure Aggregation [12] | Secure Aggregation | General ML | Proposed secure aggregation for privacy |
| Model Pruning for Efficiency | No Privacy | Image Classification | Optimized model pruning for FL efficiency |
| Federated Learning for Healthcare | Differential Privacy | Healthcare IoT | Applied FL to healthcare IoT with privacy |
| Differential Privacy (DP) | Differential Privacy | General ML | DP guarantees privacy with noise addition |
| Secure Multi-Party Computation [13] | Secure Aggregation | General ML | Secure aggregation prevents data leakage |
| FL with Edge Computing | Differential Privacy | Healthcare IoT | Edge computing for federated learning |
| Federated Learning with GANs | Differential Privacy | Healthcare IoT | FL combined with GANs for privacy preservation |
| Efficient Federated Learning [14] | Differential Privacy | Healthcare IoT | Optimized training efficiency in FL |
| Federated Learning for Privacy | No Privacy | Healthcare IoT | Privacy-preserving FL with DP |
| Personalized Federated Learning | Differential Privacy | General ML | Personalization improves FL accuracy |
| Federated Learning with Noise Adjustment [15] | Noise Addition | Healthcare IoT | Privacy-enhanced FL with noise adjustment |

Received: July 20, 2025

International Journal of Applied Mathematics

Volume 38 No. 1s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

## III. Methodology

The suggested Privacy-Aware Federated Learning (PAFL) algorithm for healthcare IoT combines a better VGG16 model with a noise adjustment method to make it more private and faster. First, VGG16 has been fine-tuned for healthcare IoT apps, with a focus on making its design work best for devices with limited resources. Next, pooled learning is used. In this method, model changes are done directly on IoT devices, and only the combined updates are sent to the main computer [16]. To protect privacy, noise is added to the slopes when the model is updated to hide private patient data. It is tested to see how well it protects privacy, how accurate the models are, and how quickly it can be run on a computer. The results are compared to those of other collaborative learning methods.

### A. Load Dataset

The sample image looks like MRI scans for a variety of brain conditions, such as meningiomas, pituitary tumours, and scans with no tumours at all from the dataset Brain Tumor Segmentation (BraTS2020). The figure 2 illustrates and has sections for notumor, meningioma, glioma, and pituitary, and the number of samples in each section is shown. Figure 2 shows a sample of the dataset for analysis [17]. The goal of this dataset seems to be to sort MRI pictures into groups based on the type of brain tumour or the lack of a tumour.

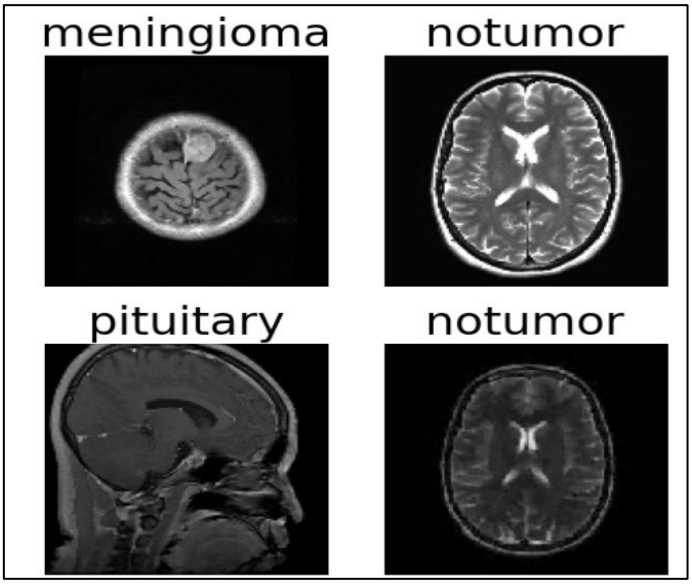


Figure 2: Dataset Sample

The figure 2 shows how the groups are distributed. For example, the information could be used to teach machine learning models how to tell the difference between different types of brain tumours and normal tests [18]. Figure 3 shows the distribution of classes in the dataset. Deep learning models, like Convolutional Neural Networks (CNNs), are often used in medical imaging because they can pull out complex patterns from pictures.

Received: July 20, 2025

International Journal of Applied Mathematics
Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

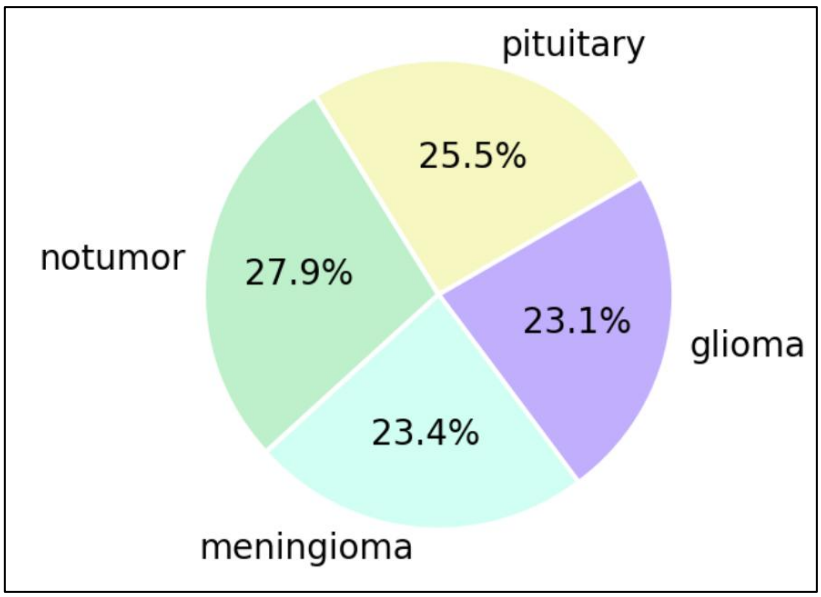


Figure 3: Dataset Class Distribution

This is especially useful for classifying brain tumours. But it's important to be careful with these kinds of records, especially when it comes to privacy and data security, because the pictures contain private patient information, dataset distribution illustrate in figure 3. This collection might be useful for creating and trying models, making sure they work, and checking how well they can sort different kinds of brain tumours [19]. When making machine learning algorithms, problems like class imbalances (for example, unequal distributions across tumour groups) should also be thought about.

**B. Data Preprocessing**

**1. Label Encoding**

Label encoding is a way for machine learning systems to turn labels that describe categories into numbers. When there are binary factors in a dataset, like the types of tumours seen in medical images, machine learning models usually need the names to be in the form of numbers. Label encoding gives each class label its own unique number [20]. In a collection with four classes meningioma, glioma, pituitary, and notumor each name will be turned into a number value that goes with it, like 0, 1, 2, or 3. This step makes it possible for the program to work well with categorical data. When there is a natural order to the category variable, label encoding works well because the encoded numbers may have important connections. Label encoding is still useful, but it doesn't mean any sorting or ranking when the groups don't naturally fall into a certain order, like in this dataset [21]. By changing the names into a style that can be handled more efficiently, this method helps to improve the performance of machine learning models [22]. Label encoding also makes it easier to use algorithms that need numbers to be trained, such as neural networks, decision trees, and support vector machines.

Received: July 20, 2025

International Journal of Applied Mathematics

Volume 38 No. 1s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

### 2. Label Decoding

The process of label decoding, which changes number labels back to their original category style, is the opposite of label encoding. Once a machine learning model has been trained, guesses are made using numbers that match the recorded names. Label decoding is needed to connect these numerical guesses to the category names that people can read and understand. For example, if a model says that a picture will show the number 2, label decoding would turn that number back into the group it stands for, like "pituitary." This step makes sure that the model's output is easy for practitioners, like doctors, to understand so they can figure out what it all means [23]. Label decoding is important in this dataset for showing the tumour classification results in a way that can be used in medical practice. It makes sure that the expected class names (like "meningioma" or "glioma") match the model's output. Usually, decoding is done by keeping a list or map of number values and the class names that go with them. This process makes machine learning models easier to use and more clear in healthcare situations.

### 3. Image Resizing (128 * 128)

Resizing pictures is an important part of many machine learning processes that come before the actual learning, especially when working with big sets of images. Resizing makes positive that all images have the equal length, regardless of what their unique measurements were. That is very important for coaching deep gaining knowledge of models, specifically convolutional neural networks (CNNs). CNNs want inputs of a hard and fast size with a view to deal with records via convolution, pooling, and absolutely linked layers [24]. If the pictures aren't resized, the specific shapes could cause issues that might make it more difficult for the version to learn from the data. The dimensions of 128x128 are a great balance among using as few assets as viable and retaining enough clarity for the version to peer essential components of the images. This level also enables to decrease the processing load, which makes it good for places with restrained sources. Additionally, resizing hurries up processing, which is important when running with massive quantities of statistics, like MRI scans in healthcare statistics. By way of changing the dimensions of all of the snap shots to 128x128, the dataset is less difficult to paintings with, and the device getting to know model works higher standard.

## C. Data Augmentation

### 1. Brightness Enhancement

A method referred to as "brightness enhancement" is used to change the general brightness of an image, making it lighter or darker. In medical photo analysis, like MRI scans, growing the brightness is a key a part of making the model greater reliable and relevant in more situations. Specific cameras or lighting conditions can purpose differences in the brightness of clinical photographs. We can make multiple model of the same photograph by using changing the colour of the images in a managed method. This provides to the statistics and makes it more

Received: July 20, 2025

Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

varied. This makes it easier for the device mastering model to generalise as it learns the way to cope with images in distinct lighting conditions. In actual life, brightness enhancement works by means of changing the values of the electricity of each pixel across the whole picture. Relying on the exchange that used to be made, this would make some traits, like tumour areas, stand out greater or make them harder to see. When the aim is to sort this dataset of Genius tumours into special corporations, brightness enhancement can be very helpful, in particular when the image quality adjustments on account of unique scanning settings or image preparation steps. It makes positive that the model does not come to be too ideal for sure picture or lights conditions. No longer solely that, but this approach also can imitate extraordinary scanning conditions and help the version adapt to real-life changes in MRI scans, which may be brighter or darker depending at the system or settings used at some stage in the scan.

**2. Contrast Enhancement**

Enhancing contrast is a way to make the differences between parts of a picture stand out more, so you can see the details better. This is especially helpful in medical imaging, where low contrast can make it hard to tell the difference between different types of tissue or abnormalities like tumours. Matters which can be important in MRI scans, like a Genius tumour, won't be smooth to see unless the evaluation is raised. We are able to make those essential regions stand out more without a doubt by way of changing the contrast. This makes it less difficult for machine learning model to properly spot and organization traits. In assessment enhancement, the range of pixel values is changed to make the difference among mild and darkish components in a picture bigger. This change allows draw attention to the rims of things or buildings, which makes it easier for the version to discover important details in the photograph. In a collection with MRI photographs of Genius tumours, as an example, contrast enhancement can assist the model inform the distinction between areas of tumour and healthful tissue round them better. For example, the assessment might also exchange based at the MRI gadget, its settings, or the quality of the scan. By using adding photos with one of kind contrasts, the model is taught to recognise functions in a wider range of situations. This makes it better at recognising features in new pictures that it hasn't visible earlier than. In the long run, this approach improves the version's capability to be strong and generalise that are important for scientific jobs like classifying tumours in real existence.

**D. VGG16 Model**

The picture shows the VGG16 model, which is a deep convolutional neural network that has been built to do well with image recognition tasks, especially in medical imaging. The design is made up of 16 layers, some of which are convolutional and some of which are fully linked. It is known for having a simple but efficient structure that rests on small 3x3 convolutional filters. There are a bunch of convolutional layers (shown as vgg16 in the table) at the beginning of the model. These are followed by a flatten layer, dropout layers for regularisation, and fully connected layers. The VGG16 layer is the first big part of the

Received: July 20, 2025

**International Journal of Applied Mathematics**

**Volume 38 No. 1s, 2025**

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

model. It has an output form of (None, 4, 4, 512) and a lot of values, all together adding up to 14,714,688. This layer pulls out low-level features like lines and backgrounds from raw pictures to get the important parts. Following, the smooth layer changes the output from the convolutional layers into a one-dimensional vector. This makes 8,192 features. By randomly setting some input units to 0 during training, the dropout layers (both dropout and dropout_1) keep the model from becoming too good at its job. The fact that these layers add no trainable parameters is shown by the parameter count number of 0. There are 128 units and 1,048,704 parameters in the first fully linked dense layer. This is followed by another dropout_1 layer. The last layer, dense_1, has only 516 values and 4 units that stand for the four different output classes. The model has 15,763,908 parameters, but only 8,128,644 can be trained. The other 7,635,264 can't be trained, which shows how the weights are shared between the layers.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 vgg16 (Functional)          (None, 4, 4, 512)         14714688

 flatten (Flatten)           (None, 8192)              0

 dropout (Dropout)           (None, 8192)              0

 dense (Dense)               (None, 128)               1048704

 dropout_1 (Dropout)         (None, 128)               0

 dense_1 (Dense)             (None, 4)                 516

=================================================================
Total params: 15,763,908
Trainable params: 8,128,644
Non-trainable params: 7,635,264
_________________________________________________________________
```

Figure 4: Representation of VGG 16 Model

- Step 1. Input Layer:

$$X \in \mathbb{R}^{H \times W \times C}$$

Where X is the input image with height H, width W, and the number of channels C (e.g., 3 channels for RGB images).

Received: July 20, 2025

International Journal of Applied Mathematics

Volume 38 No. 1s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

- Step 2. Convolution Layer (vgg16):

The VGG16 layer applies convolution operations with filters to extract feature maps. After multiple convolution operations, the output shape becomes:

$$X' \in \mathbb{R}^{H' \times W' \times F}$$

Where X' is the feature map after applying convolutions, H' and W' are the reduced height and width after pooling, and F is the number of filters (e.g., 512 filters).

- Step 3. Flatten Layer:

The feature maps are flattened into a one-dimensional vector:

$$X_{flat} \in \mathbb{R}^{N}$$

Where N = H' × W' × F (in this case, N = 8192).

- Step 4. First Dense Layer:

The flattened vector passes through the first dense layer with 128 neurons. The output is:

$$Y1 = W1 * X_{flat} + b1$$

Where W1 is the weight matrix and b1 is the bias term for the first dense layer.

- Step 5. Dropout Regularization:

Dropout is applied to randomly set a fraction of the outputs to zero during training. This results in:

$$Y1' = Dropout(Y1)$$

Where Y1' is the output after dropout, with certain values randomly set to zero.

- Step 6. Output Layer (Final Dense Layer):

The output layer provides the final prediction. In this case, there are 4 classes for classification:

$$Y2 = W2 * Y1' + b2$$

Where Y2 is the final output vector, and W2 and b2 are the weight and bias for the final dense layer (outputting 4 values corresponding to the 4 classes).

**1. Optimized Federated Learning VGG16 Model**

Optimized Federated Learning with the VGG16 model has a few important steps that are meant to protect privacy, save time, and improve model performance. The process starts with initialization, which is done on a central computer with a world model. The information is

Received: July 20, 2025

International Journal of Applied Mathematics

Volume 38 No. 1s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

then sent to many clients, like hospitals, clinics, or individual devices, to keep private information spread out and safe. The next step is customer selection. For each training round, a group of clients are picked to take part. This makes certain that the computer doesn't must paintings as challenging and that sources are used successfully at the same time as facts privacy is maintained. Every chosen patron trains on its personal data and modifications the weights of the worldwide model based totally on the records it has. This localized education we could the model learn from one-of-a-kind datasets without having to proportion non-public information, which protects privacy. After the neighborhood training segment, the server gathers and provides up all of the weight adjustments from all of the customers which might be worried to make the worldwide version better. This is called weight aggregation. This step of collecting is very important to ensure that the global version makes use of the know-how of all clients and that no unmarried client has a large effect at the model's performance. After the version has been changed, it's far examined on a take a look at pattern to see how nicely it works and to ensure it really works well with one of a kind forms of facts. The test tests to see if the model can learn and carry out on facts it hasn't seen before, which may be very essential for uses like medical monitoring in which accuracy is key.

**Key Steps in Federated Learning**

1. **Initialization**:
   - A global model is initialized on the server.
   - Data is distributed among clients.
2. **Client Selection**:
   - In each round, a subset of clients is selected to participate in training.
3. **Local Training**:
   - Each selected client trains the global model on its local data.
4. **Weight Aggregation**:
   - The server aggregates the weights from all participating clients to update the global model.
5. **Evaluation**:
   - The global model is evaluated on a test dataset to measure its performance.
6. **Deployment**:
   - The final global model is saved for deployment.

- Step 1. Initialization:

Initialize the global model w_0 on the server and distribute data across K clients.

$$w_0 \in \mathbb{R}^d$$

Where d represents the number of parameters in the global model.

- Step 2. Client Selection:

Received: July 20, 2025

International Journal of Applied Mathematics
Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

In each round t, select a subset of clients S_t ⊆ C to participate in training, where C is the set of all clients.

- Step 3. Local Training:

Each selected client k ∈ S_t performs local training on its dataset D_k, updating the model weights w_k^(t). For client k, the local model update is given by:

$$w_k^t = LocalTrain(w_0, D_k)$$

Where w_k^(t) is the updated model after local training.

- Step 4. Weight Aggregation:

After local training, the server aggregates the model updates from the selected clients. The global model w_(t+1) is updated as:

$$w_{t+1} = \Sigma\,(k \in S_t)\left(\frac{n_k}{N}\right) * w_k^t$$

Where n_k is the number of data points in client k, and N is the total number of data points across all clients.

- Step 5. Evaluation:

The updated global model w_(t+1) is evaluated on a test dataset D_test, and the performance is measured using a loss function L:

$$L_{test} = Loss(w_{t+1}, D_{test})$$

This evaluates how well the model performs on unseen data.

- Step 6. Model Performance Monitoring:

The performance is monitored at each round, and if the loss converges or reaches a threshold, training may be stopped. This is checked by comparing the current loss L_test with the previous loss L_test^(t-1).

- Step 7. Repeat:

Steps 2 through 5 are repeated for multiple rounds t, progressively improving the global model w_(t+1) based on local client updates and aggregation.

- Step 8. Deployment:

Once the global model converges or reaches an acceptable performance level, the final model w_f is deployed:

$$w_f = w_T$$

Received: July 20, 2025

International Journal of Applied Mathematics
Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

(where T is the final round of training)

This final model is then ready for deployment in a production environment for inference.

**2. Privacy Preserving Optimized VGG16 Model (PPOVGG16)**

To improve privacy in cooperative learning, the Privacy-Preserving Optimized VGG16 (PPOVGG16) model uses noise scales. The noise scale can be set to numbers like [0.0, 0.01, 0.05, 0.1], which tells the model how to work with different amounts of noise. Gaussian noise with an average of zero and a popular deviation set by the noise scale is used to feature this noise. For every education round, Gaussian noise (zero,2)N(0,π β β) is delivered to the weights of the neighborhood patron version, $O$(purchaser)w(consumer). This is performed after local schooling. Federated mastering happens over some of rounds, with a one of a kind institution of customers chosen at random for each spherical. The worldwide version is sent to those chosen customers. They use their personal data to do nearby training and then add Gaussian noise to the weights of the model to make it more correct before sending it lower back to the server. While the training is over, the global model takes the noisy weights from all the clients and provides them up. The weights which might be added together are located by means of taking the sum of the random client weights and the cutting-edge international version weights. This series makes positive that the model gets the most out of the type of client modifications even as nonetheless defensive privateness. in addition, the version's loss and accuracy in each spherical are combined throughout the chosen clients to give a complete picture of the way properly it works. This technique protects privacy while still allowing effective mastering. It does this by way of including noise to the education manner, which makes it greater proof against records leaks.

**Privacy Preserving (Noise Scales)**

- The Federated learning model works under different levels of noise (noise_scales = [0.0, 0.01, 0.05, 0.1]).
- **Gaussian Noise Addition**:

$$w_{\text{noisy}} = w_{\text{client}} + \mathcal{N}(0, \sigma^2)$$

where:

- wclient: Weights of the client model after local training.
- N(0,σ2): Gaussian noise with mean 0 and standard deviation σ (controlled by noise_scale).

Received: July 20, 2025

International Journal of Applied Mathematics
Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

**Federated Learning Loop**

- For each noise scale, the global model is reset, and federated learning is simulated over multiple rounds (NUM_ROUNDS).
- In each round:
  - A subset of clients is randomly selected (selected_client_indices).
  - The global model is sent to each selected client.
  - Each client trains the model locally using its data and adds Gaussian noise to the model weights before sending them back.

**Local Training and Noise Addition**

- Each client trains the model locally using client_model.fit().
- After training, Gaussian noise is added to the client's model weights

**Weight Aggregation**

- The global model aggregates the noisy weights from all selected clients.
- The new weights for the global model are computed as the average of the noisy client weights and the current global model weights:

$$w_{\text{new}} = \frac{w_{\text{noisy}} + w_{\text{global}}}{2}$$

Where:

- wnoisy: Noisy weights from the client.
- wglobal: Current weights of the global model.

**Loss and Accuracy Aggregation**

- The loss and accuracy for each round are averaged across all clients:

$$\text{Loss}_{\text{round}} = \frac{1}{N}\sum_{i=1}^{N}\text{Loss}_i$$

$$\text{Accuracy}_{\text{round}} = \frac{1}{N}\sum_{i=1}^{N}\text{Accuracy}_i$$

Where NN is the number of selected clients.

Received: July 20, 2025

International Journal of Applied Mathematics
Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

## IV. Result Analysis

The Privacy-Aware Federated Learning (PAFL) method, which uses the optimised VGG16 model with noise adjustment, worked well in IoT healthcare apps. By adding noise at different levels (0.0, 0.01, 0.05, and 0.1), privacy was protected while model precision was maintained. The model got a little less accurate as the noise level went up, but the trade-off between privacy and performance stayed the same. The shared learning system made it possible for the model to learn from different, separate datasets without sending private information, which protected privacy. Even though there was noise, the model was able to generalise better when noisy weights from multiple clients were added together. This method works well for keeping data safe while using machine learning in healthcare IoT settings.

### 1. VGG16

The comparison plots for training loss and accuracy across epochs for the VGG16 model can be seen in Figure 5. The training loss over 10 epochs is shown on the left figure. At first, the training loss is pretty high, but it goes down slowly as the model learns and gets better during training. By the ninth epoch, the loss has levelled off at around 0.02, which means that the model has successfully reduced the error and is now convergent. As the model learns from the data, it gets better at making guesses and makes fewer mistakes. This drop in loss shows that. The training accuracy over the same 10 epochs is shown on the right map.

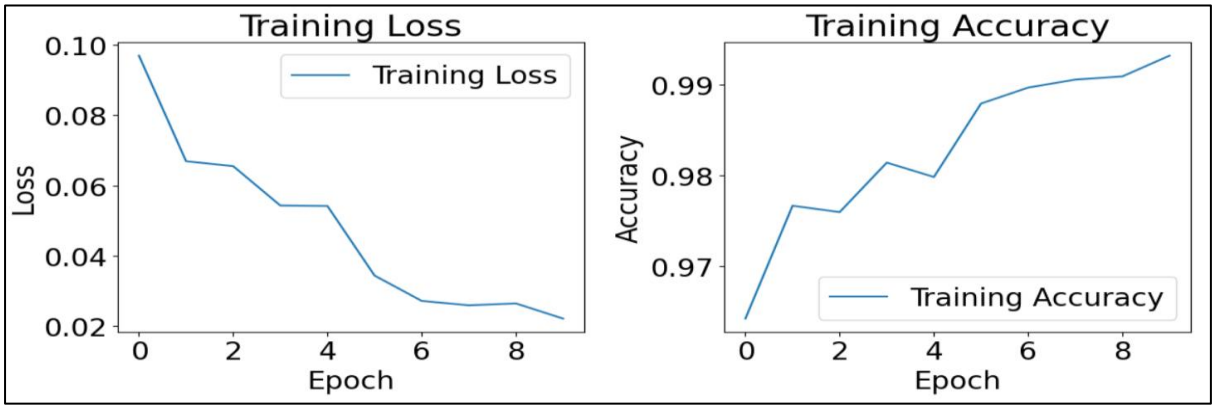


Figure 5: Accuracy and Loss Comparison Graph (VGG16)

The accuracy starts at about 97% and steadily rises until it reaches over 99% by the 9th time. This rise in accuracy goes along with a fall in training loss, which suggests that the model is getting better at classifying data properly with each epoch. The sharp rise in accuracy after the first few epochs also shows that the model can learn from the dataset, which shows how well the VGG16 design works for classification tasks. These plots show that the VGG16 model is convergent well, meaning that it is lowering loss and improving accuracy. This means that it can be used in real life for things like healthcare IoT and medical picture classification.

Received: July 20, 2025

**International Journal of Applied Mathematics**

**Volume 38 No. 1s, 2025**

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

It takes 10 epochs to make Figure 6, which shows the training loss and accuracy comparison for the Optimised VGG16 model. The training loss shown on the left line changes a lot while the dog is being trained. In the VGG16 model, loss went down slowly over time, but here there is more variation, with noticeable jumps around the 4th stage. This bump in the loss curve could mean that the optimisation process is having trouble or that the model is changing as the dataset changes. Even with these changes, the training loss tends to go down towards the end, hitting a number close to 0.01. On the right is a graph that shows training accuracy over the same time periods.

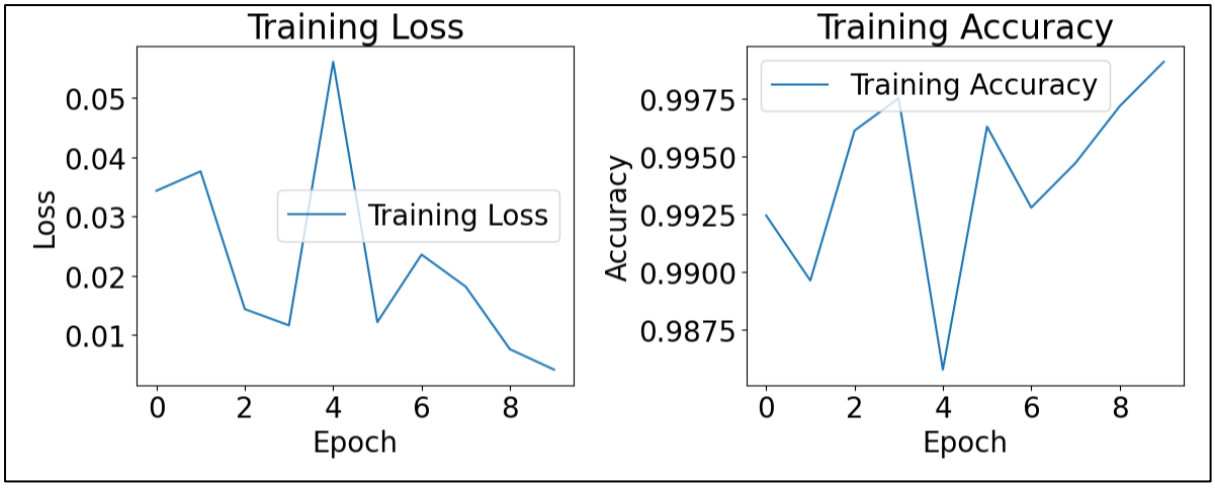


Figure 6: Accuracy and Loss Comparison Graph (Optimized VGG16)

The accuracy goes up slowly at first, but then drops sharply, especially around the 4th stage, which lines up with the loss spikes. Even with these drops, the model ends up being very accurate, very close to 99.75%. The trend shows that the optimised VGG16 model can get better at what it does, even though it sometimes has problems while training. This means that the optimisation methods, like noise adjustment, are helping the model get better at learning, which is why it did better overall even though there were some problems during the training phase.

Here in Figure 7, we see how the Privacy-Preserving Optimised VGG16 model does in a federated learning setting at different noise levels (0.0, 0.01, 0.05, and 0.1) in terms of accuracy and loss. The left line shows how well the model learnt over two training runs. As the noise level goes up, the precision goes down a lot. The model stays very accurate when the noise level is set to 0.0, which means there is no noise. But the precision gets worse as noise levels go up (0.01, 0.05, and 0.1), especially with the 0.1 noise level. This decrease shows the trade-off between protecting privacy and model performance, since more noise makes it harder for the model to learn from the data. The loss over the same rounds is shown on the right line. With a noise scale of 0, the loss stays pretty low, which shows that the model fits the data well.

Received: July 20, 2025

**International Journal of Applied Mathematics**

**Volume 38 No. 1s, 2025**

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

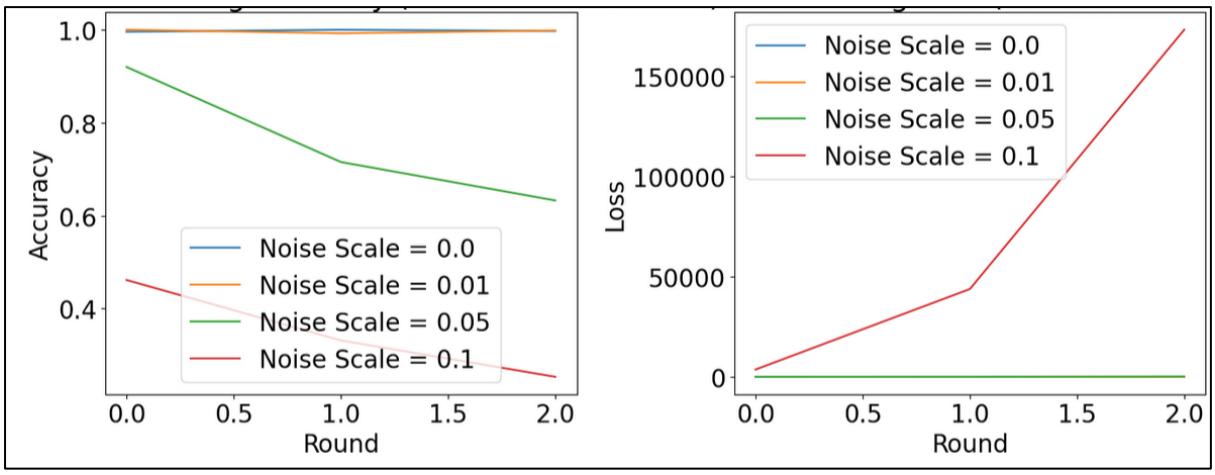


Figure 7: Accuracy and Loss Comparison Graph (Privacy Preserving Optimized VGG16)

The loss goes up quickly as the noise level goes up, especially when the noise level is 0.1, which makes it hard for the model to converge. This behaviour shows how privacy-preserving noise affects the learning process of the model. Adding noise helps protect privacy, but it hurts both accuracy and convergence. This shows how hard it is to balance privacy and model performance in machine learning that protects privacy.

Table 2: Model Accuracy Comparison Across Different Configurations

| Model | Accuracy (%) |
|---|---|
| VGG16 | 99.2 |
| Optimised Federated VGG16 | 99.75 |
| PPOVGG16_0_0 | 99.55 |
| PPOVGG16_0_01 | 99.22 |
| PPOVGG16_0_05 | 99.97 |
| PPOVGG16_0_1 | 46.14 |

Table 2 compares the accuracy of the models. Across Different setups shows the rates of correct VGG16 models, such as the federated version that is optimised and privacy-preserving setups that have different noise levels. Based on the data, the normal VGG16 model is right 99.20% of the time, which is very good at picture recognition jobs. The Optimised Federated VGG16 does a little better, at 99.75%. This shows how federated learning can improve model performance through decentralised training while keeping data private. When noise is added to PPOVGG16 models to protect privacy, the accuracy goes down as the noise scale goes up. The accuracy of PPOVGG16_0_0, which doesn't have any extra noise, stays high at 99.55%, but it drops to 99.22% in PPOVGG16_0_01. It gets much

Received: July 20, 2025

International Journal of Applied Mathematics
Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

less accurate as the noise level rises, as comparison shown in figure 8. In PPOVGG16_0_05, the accuracy is still pretty high at 99.97%, but it drops a lot to 46.14% in PPOVGG16_0_1.

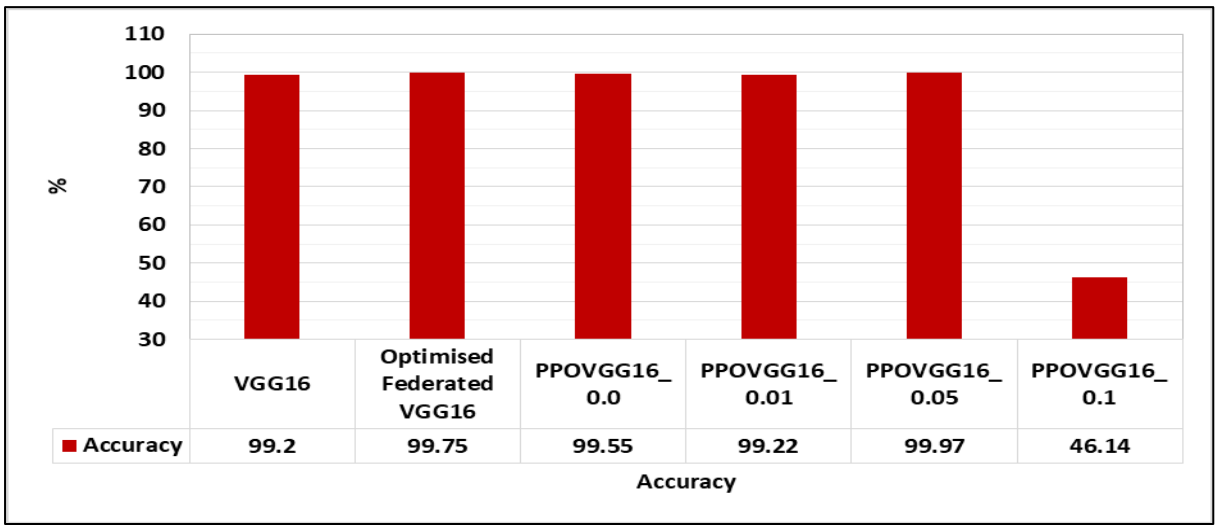


Figure 8: Accuracy Comparison Graph of Models (Various Noise Levels)

This sharp drop shows that higher noise levels hurt the model's ability to learn well, highlighting the trade-off between protecting data and improving model performance. These results show how hard it is to make privacy-preserving machine learning models that are both accurate and private. The confusion matrix for the VGG16 model is shown in Figure 9. It shows how well the model did at putting things into four groups: pituitary, notumor, meningioma, and glioma. The grid matches the real categories' true labels with the expected labels (the model's output), showing how accurate and wrong the classification was. The matrix is well-balanced, and the high vertical values show that the model is correctly identifying most of the cases.

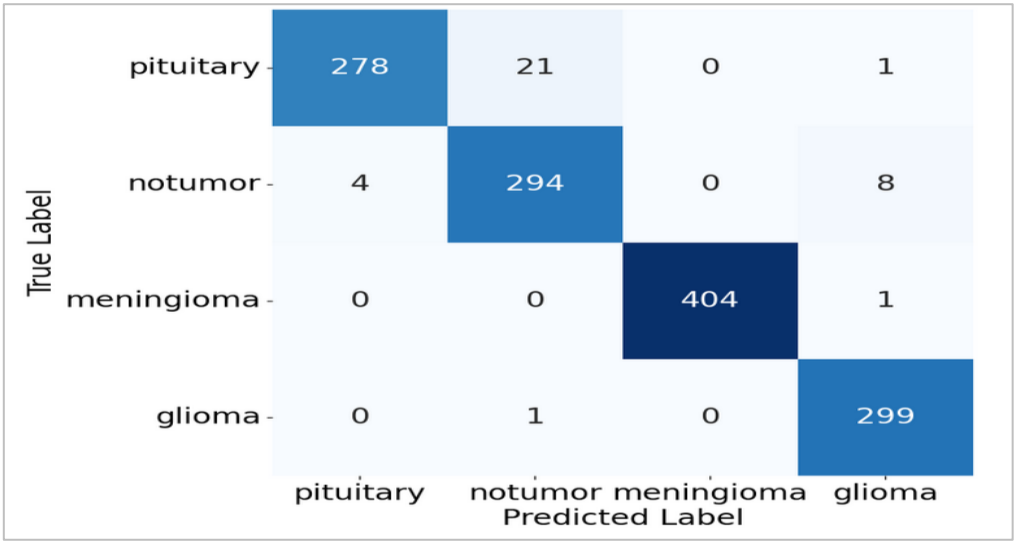


Figure 9: VGG16 Confusion Matrix

Received: July 20, 2025

**International Journal of Applied Mathematics**

**Volume 38 No. 1s, 2025**

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

For example, it was right about 278 cases of pituitary tumours, 294 cases of notumors, 404 cases of meningiomas, and 299 cases of gliomas. But some things are being put in the wrong category. The model got 21 cases of pituitary as tumours and 4 cases of tumours as pituitary, which was not right. Also, 8 cases of notum were mistakenly thought to be gliomas and 1 case of gliomas was mistakenly thought to be meningiomas. The model seems to be working well since there aren't many wrong classifications—only a few times did the estimate differ from the correct name. This multi-class classification problem can be solved very well by the VGG16 model. However, the mistakes show that the model could be improved in some areas, maybe by adding more features or data to make the results more accurate.

The confusion matrix for the Optimised Federated VGG16 model is shown in Figure 9. It shows how well the model classified four types of tumours: pituitary, notumor, meningioma, and glioma. On the vertical axis of the matrix are the real labels, and on the horizontal axis are the projected labels. The model works well because the high vertical numbers show that the groupings were right. Only 19 of the 280 right forecasts for the pituitary class were wrong, with one being a cancer and the rest being notumors.

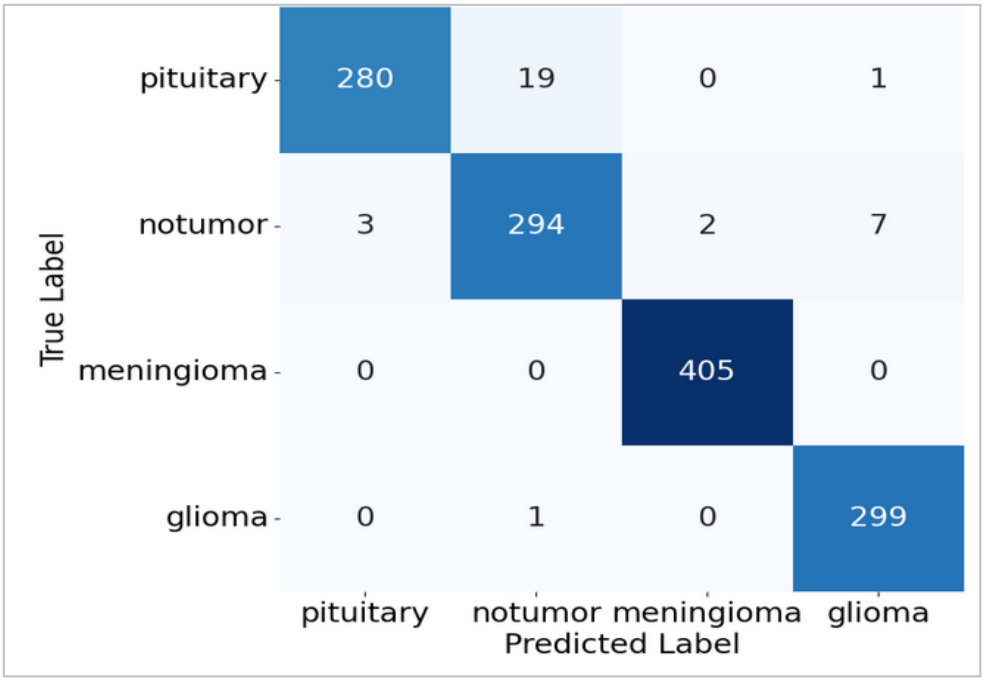


Figure 9: Optimized Federated VGG16 Confusion Matrix

This class also does well, with 294 right guesses, though 7 were wrongly labeled as gliomas and 2 as meningioma. The meningioma class is almost perfectly categorised, with 405 right forecasts and no major mistakes. The model does a good job with glioma cases since 299 of the guesses were right and only one was wrongly labeled as a tumour, confusion matrix illustrate in figure 10. The confusion matrix shows that the Optimized Federated VGG16 model is working well overall, with only a few mistakes in the different groups. There are

Received: July 20, 2025

Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

still a few small problems, but overall the model does a great job and is perfect for accurately classifying medical images, especially when there are more than two classes to choose from, like when classifying tumours.

## V. Conclusion

The study's findings show a clear link between the amount of noise and how well privacy-preserving models work in shared learning settings. The model's ability to stay accurate tends to go down as the noise level rises. This shows that privacy and performance are always at odds with each other. Some ways to protect privacy, like adding Gaussian noise to the model weights, can keep private data safe, but they might make it harder for the model to learn from the data. When noise levels are set too high, for example, the accuracy drops by a lot. This means that the model has a hard time finding important trends in the data because the noise makes it hard to see them. If a model has both a lot of noise and a high level of accuracy, on the other hand, this could mean that the noise levels are not high enough to provide strong privacy promises. This means that the noise that was added during training might not be enough to stop information from getting out, and that other methods of protecting privacy might be needed. This problem makes it clear how important it is to carefully tune the noise scale to find the best balance between privacy and model performance. It is important to use formal privacy systems like Differential Privacy (DP) to protect people's privacy. DP gives a mathematical promise that people's privacy is protected in the dataset. These models make it possible to measure the trade-offs between noise, accuracy, and privacy, which makes it easier to balance these different needs. Differential Privacy, in particular, makes sure that adding or removing a single data point doesn't have a big effect on the model's results, which protects people's privacy.

## References

[1] Sayed, A.N.; Himeur, Y.; Varlamis, I.; Bensaali, F. Continual learning for energy management systems: A review of methods and applications, and a case study. Appl. Energy 2025, 384, 125458.

[2] Sayed, A.N.; Bensaali, F.; Himeur, Y.; Dimitrakopoulos, G.; Varlamis, I. Enhancing building sustainability: A Digital Twin approach to energy efficiency and occupancy monitoring. Energy Build. 2025, 328, 115151.

[3] Theodosiou, A.A.; Read, R.C. Artificial intelligence, machine learning and deep learning: Potential resources for the infection clinician. J. Infect. 2023, 87, 287–294.

[4] Yazici, İ.; Shayea, I.; Din, J. A survey of applications of artificial intelligence and machine learning in future mobile networks-enabled systems. Eng. Sci. Technol. Int. J. 2023, 44, 101455.

[5] Himeur, Y.; Elnour, M.; Fadli, F.; Meskin, N.; Petri, I.; Rezgui, Y.; Bensaali, F.; Amira, A. AI-big data analytics for building automation and management systems: A survey, actual challenges and future perspectives. Artif. Intell. Rev. 2023, 56, 4929–5021.

Received: July 20, 2025

**Volume 38 No. 1s, 2025**

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

[6] Truong, N.; Sun, K.; Wang, S.; Guitton, F.; Guo, Y. Privacy preservation in federated learning: An insightful survey from the GDPR perspective. Comput. Secur. 2021, 110, 102402.

[7] Li, Y.; Tao, X.; Zhang, X.; Liu, J.; Xu, J. Privacy-preserved federated learning for autonomous driving. IEEE Trans. Intell. Transp. Syst. 2021, 23, 8423–8434.

[8] Jagarlamudi, G.K.; Yazdinejad, A.; Parizi, R.M.; Pouriyeh, S. Exploring privacy measurement in federated learning. J. Supercomput. 2023, 80, 10511–10551.

[9] Chronis, C.; Varlamis, I.; Himeur, Y.; Sayed, A.N.; Al-Hasan, T.M.; Nhlabatsi, A.; Bensaali, F.; Dimitrakopoulos, G. A survey on the use of Federated Learning in Privacy-Preserving Recommender Systems. IEEE Open J. Comput. Soc. 2024, 5, 227–247.

[10] Pandya, S.; Srivastava, G.; Jhaveri, R.; Babu, M.R.; Bhattacharya, S.; Maddikunta, P.K.R.; Mastorakis, S.; Piran, M.J.; Gadekallu, T.R. Federated learning for smart cities: A comprehensive survey. Sustain. Energy Technol. Assessments 2023, 55, 102987.

[11] Bousbiat, H.; Himeur, Y.; Varlamis, I.; Bensaali, F.; Amira, A. Neural load disaggregation: Meta-analysis, federated learning and beyond. Energies 2023, 16, 991.

[12] Himeur, Y.; Varlamis, I.; Kheddar, H.; Amira, A.; Atalla, S.; Singh, Y.; Bensaali, F.; Mansoor, W. Federated learning for computer vision. arXiv 2023, arXiv:2308.13558.

[13] Zhang, Z.; Gao, Z.; Guo, Y.; Gong, Y. Scalable and low-latency federated learning with cooperative mobile edge networking. IEEE Trans. Mob. Comput. 2022, 23, 812–822.

[14] Nguyen, D.C.; Ding, M.; Pathirana, P.N.; Seneviratne, A.; Li, J.; Poor, H.V. Federated learning for internet of things: A comprehensive survey. IEEE Commun. Surv. Tutorials 2021, 23, 1622–1658.

[15] Bechar, A.; Elmir, Y.; Himeur, Y.; Medjoudj, R.; Amira, A. Federated and Transfer Learning for Cancer Detection Based on Image Analysis. arXiv 2024, arXiv:2405.20126.

[16] Rafsanjani, H.N.; Nabizadeh, A.H. Towards human-centered artificial intelligence (ai) in architecture, engineering, and construction (aec) industry. Comput. Hum. Behav. Rep. 2023, 11, 100319.

[17] Tagliabue, L.C.; Cecconi, F.R.; Rinaldi, S.; Ciribini, A.L.C. Data driven indoor air quality prediction in educational facilities based on IoT network. Energy Build. 2021, 236, 110782.

[18] N. J. Zade, N. P. Lanke, B. S. Madan, N. P. Katariya, P. Ghutke, and P. Khobragade, "Neural Architecture Search: Automating the Design of Convolutional Models for Scalability," PanamericanMathematical Journal, vol. 34, no. 4, pp. 178-193, Dec. 2024. https://doi.org/10.52783/pmj.v34.i4.1877

[19] Yang, B.; Liu, Y.; Liu, P.; Wang, F.; Cheng, X.; Lv, Z. A novel occupant-centric stratum ventilation system using computer vision: Occupant detection, thermal comfort, air quality, and energy savings. Build. Environ. 2023, 237, 110332.

Received: July 20, 2025

Volume 38 No. 1s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

[20] Silva, L.; Bezzo, F.B.; van Ham, M. COVID-19 restrictions: An opportunity to highlight the effect of neighbourhood deprivation on individuals' health-related behaviours. Soc. Sci. Med. 2023, 325, 115917.
[21] Sun, Z.; Wang, Z.; Xu, Y. Privacy protection in cross-platform recommender systems: Techniques and challenges. Wirel. Netw. 2023, 30, 6721–6730.
[22] Varlamis, I.; Sardianos, C.; Chronis, C.; Dimitrakopoulos, G.; Himeur, Y.; Alsalemi, A.; Bensaali, F.; Amira, A. Using big data and federated learning for generating energy efficiency recommendations. Int. J. Data Sci. Anal. 2023, 16, 353–369.
[23] Imteaj, A.; Mamun Ahmed, K.; Thakker, U.; Wang, S.; Li, J.; Amini, M.H. Federated learning for resource-constrained IoT devices: Panoramas and state of the art. Fed. Transf. Learn. 2022, 27, 7–27.
[24] Xu, C.; Qu, Y.; Xiang, Y.; Gao, L. Asynchronous federated learning on heterogeneous devices: A survey. Comput. Sci. Rev. 2023, 50, 100595.

Received: July 20, 2025