Proceedings of the 4th International Conference on Signal Processing and Machine Learning
DOI: 10.54254/2755-2721/49/20241065

Federated learning algorithm-based skin cancer detection

Jingyue Hu

The Department of Computer and Software, Chengdu Jincheng College, Chengdu, 611731, China

hujing@cdjcc.edu.cn

Abstract. Owing to the oversight regarding training data privacy within the realm of Deep Learning (DL), there have been inadvertent data leaks containing personal information, resulting in consequential impacts on data providers. Consequently, safeguarding data privacy throughout the deep learning process emerges as a paramount concern. In this paper, the author suggests the integration of FedAvg into the training procedure as a measure to ensure data security and privacy. In the experiments, the author first applied data augmentation to equalize the various samples in the dataset, then simulated four users using a Central Processing Unit (CPU) with four cores and established a network architecture starting with DenseNet201. Each user cloned all parameters of global model and received an equal portion of the dataset. After updating the parameters locally, the weights were aggregated by averaging and passed back to the global model. Additionally, the author introduced learning rate annealer to help the model converge better. The experimental results demonstrate that incorporating FedAvg indeed saves training time and achieves excellent performance in skin cancer classification. Despite a slight loss in accuracy, the algorithm can address privacy concerns, making the use of FedAvg highly valuable.

Keywords: Federated Learning, Skin Cancer Detection, FedAvg, Privacy.

1. Introduction

In global medicine, skin cancer has emerged as a prominent global health concern affecting individuals worldwide. Melanoma and nonmelanoma skin cancer are the two primary subtypes of skin cancer [1]. Meanwhile, nonmelanoma can be further classified as pigmented benign keratosis, vascular lesion and actinic keratosis etc. Out of 200 distinct forms of cancer, melanoma is the most deadly [2], but it is curable if recognized and addressed in early stage. However, if not promptly diagnosed, it exhibits a rapid tendency to metastasize to other regions of the body. Therefore, it is crucial to accurately determine the type of skin cancer in patients.

Currently, Artificial Intelligence (AI) and Deep Learning (DL) evolution have already revolutionized the diagnosis in the field of medical image analysis [3, 4], especially for skin cancer. For instance, Venugopal et al suggest a Deep Neural Network (DNN) model with superior learning performance on dermoscopic pictures and fine-tuned training for the recognition of skin cancer [5]. While Nahata et al created a Convolutional Neural Network (CNN) model in 2020 which can categorize different types of skin cancer disease, this model makes use of Transfer Learning methods for early convergence [6]. And in [7], a system was implemented by two traditional machine learning classifiers and together with a set of characteristics that define a skin lesion's borders, texture, and color. Experiments show that the three methodologies work together to produce the highest degree of accuracy, the experiments have indicated that integrating the three tactics results in the highest accuracy level. In the same vein, a seminal study in this area is the work for a simple, feature-based skin cancer identification model with fine-grained categorization as its foundation. They performed a semantic segmentation model which can segment the lesion area on the skin, and by using this method, they accomplished a higher accuracy [8]. Overall, these studies have been carried out on improving skin cancer diagnosis by using different models or some variants. Although they make significant contributions to the precision of skin cancer detection, the need of patient security and privacy has received very little attention in the medical industry, since there is a certain drawback associated with personal information leakage during the model training.

Regarding this problem, the author designs a Federated Learning (FL) algorithm to train the model for the purpose of protecting sensitive data from patients. This model is used as the global model for the Federated Averaging (FedAvg) algorithm, and the local models for each client are initialized as clones of the global model. The training loop updates the global model by averaging the weights from the local models. Therefore, this research allows for collaborative training of a global model using local models on distributed data while preserving data privacy. It iteratively updates the global model by aggregating the weights from the local models, resulting in a model that performs well on the overall dataset.

2. Method

2.1. Dataset description and preprocessing

The dataset utilized in this work, which was compiled by the International Skin Imaging Collaboration (ISIC), included 2357 photographs. According to the categorization made with ISIC, malignant and benign oncological diseases in this dataset were previously divided into 9 groups. The following is the distribution of data quantities in each class of the dataset shown in Figure 1 and sample images for each class shown in Figure 2.

To keep the same number of images for each class, this study performed data augmentation by generating additional images using various transformations e.g., rotation, zooming, and flipping, therefore, the quantity of every class was added to 2500. Subsequently, the dataset for this research was divided into training and testing sets. Normalization operations were applied to all images. After that, the experiment performed one-hot encoding on the labels and reshaped the images in 3 dimensions (75×100×3) to match the input shape expected by the model.

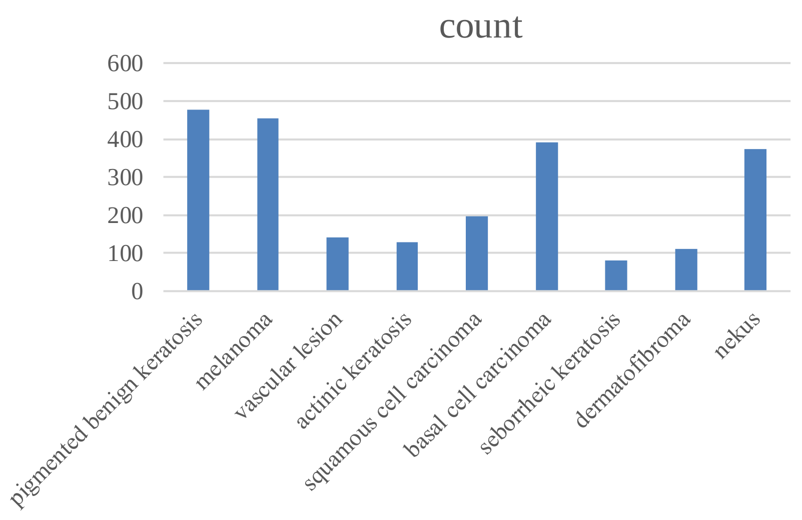

Figure 1. The distribution of data quantities in each class of the dataset.

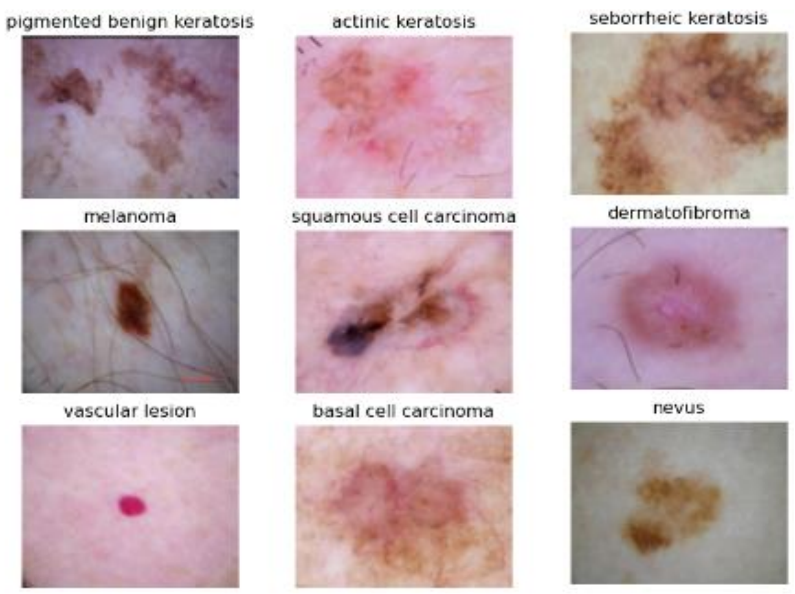

Figure 2. The sample images for each class.

2.2. Proposed approach

In this article, the author used FedAvg to protect the sensitive data in skin cancer detection. Since there are petabytes of data produced every day by several independent computing devices and only a small portion of them can be gathered and used for Deep Learning (DL) due to concerns about data security and privacy breaches. Nevertheless, in this situation, FedAvg was suggested to accomplish model training using pooled data from various clients without dataset sharing inside the cluster [9]. The fundamental process of FedAvg is shown in Figure 3.

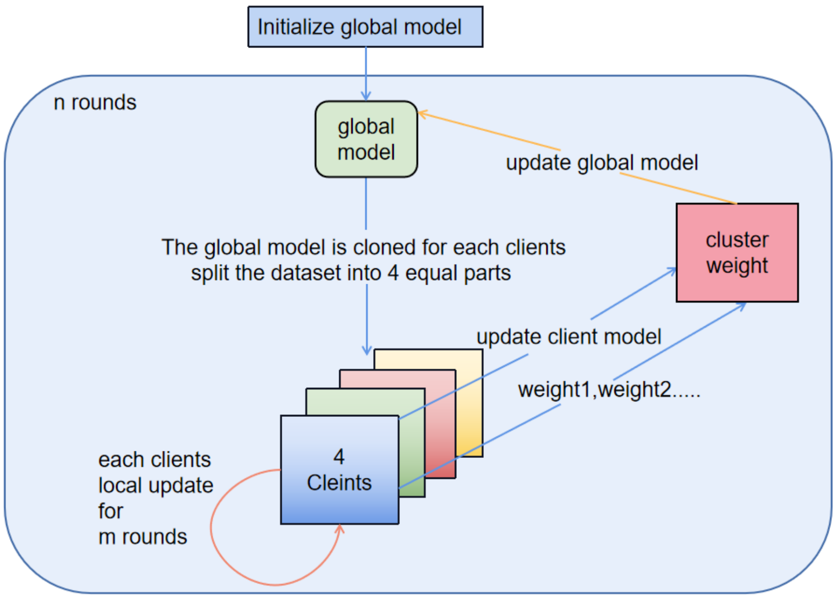

Figure 3. The fundamental process of FedAvg.

Meanwhile, DenseNet201 architecture shown in Figure 4 was used in model training. DenseNet201 is a network with a huge number of parameters since it is made up of 201 layers of convolutional and fully connected layers. The construction of DenseNet201 network improves gradient propagation and information transfer, ultimately boosting network accuracy and stability [10].

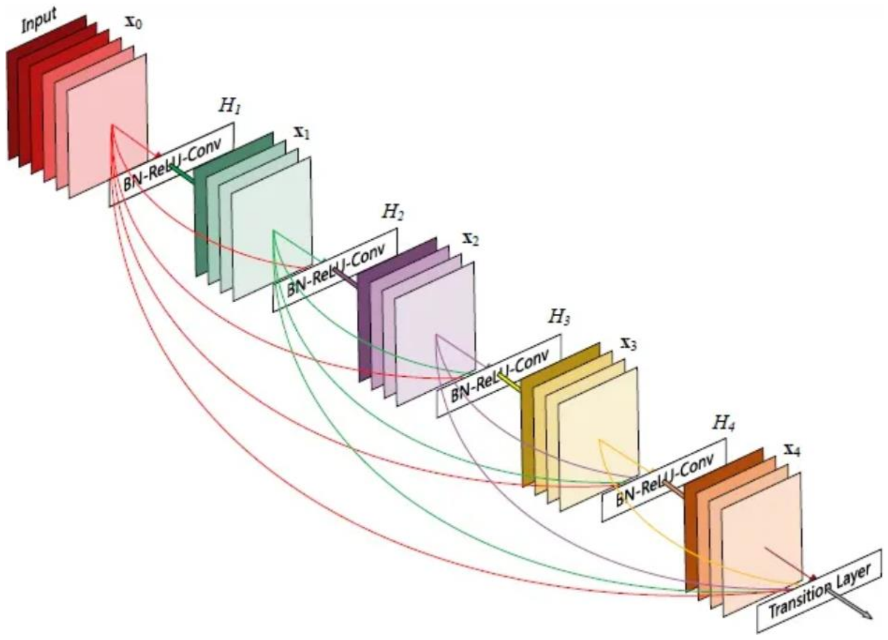

Figure 4. The architecture of DenseNet201 [11].

In detail, the experiment used 4 cores of CPU as 4 clients, In the first round, the global model is initialized. The model started with a DenseNet201 base, and A layer that flattens the output of the base model into a one-dimensional tensor is applied after the basic model. To avoid overfitting, a dropout layer was introduced, and two Dense layers were added, with the first layer having 512 units and using ReLU. In the end, class probabilities were produced by the final Dense layer by using the softmax activation function.

The global model was then cloned for each client, and the local models were compiled with the same optimizer, loss function, and metrics. After dividing the total number of training samples by the number of clients, it was able to obtain the number of samples per client. It then splits the training data and labels into equal subsets for each client. Similarly, the experiment calculates the number of samples per client for the validation set and splits the validation data and labels into equal subsets for each client. After training each local model, the code collected the local model weights and loss metrics. In addition, the study calculated the average weights of the local models. The average weights derived from the local models were added to the global model. This process was repeated for the specified number of training rounds. The learning rate annealer, which the author implements during training, essentially enables adaptive change of learning rate according to the model's performance. Reducing the learning rate can help the model converge to better solutions and potentially avoid getting stuck in suboptimal local minima.

2.3. *Implementation details*

Algorithm 1 Server Update

```
Initialize the model $x_t$, and for each communication round t= 1, .... T. At the t-th round, do the following:
    I. Select a set St of m out of the K clients, uniformly at random
    II. Perform ClientUpdate(i, $x_t$) at the chosen clients, and receive $x_{t+1}^{(i)}$ from client $i \in S_t$
    III. Aggregate the updates: $x_{t+1} = \Sigma_{i \in St} p_i x_{t+1}(i)$
```

Algorithm 2 Client Updates: ClientUpdate(i, x_t)

```
I. Initialize the local model $x_{t,0}^{(i)} \leftarrow x_t$ for $T_i$= $En_i$/B local updates
II. For local iteration index j= 0, ....$T_i$-1 do the following:
    Sample minibatch $\xi_j$ from the local dataset $D_i$, and make the local update
    $x_{t,j+1}^{(i)} = x_{t,j}^{(i)} - \eta g(x_{t,j}^{(i)}, \xi_j)$
    Return $x_{t+1}^{(i)} \leftarrow x_{t,T_i}^{(i)}$ to the server
```

3. Results and discussion

To validate the effective implementation of the FedAvg algorithm while preserving privacy, this study conducted two sets of experiments. One experiment involved training a pre-designed model using the FedAvg algorithm, while the other experiment trained the model without using the FedAvg algorithm. The final results were compared in terms of the evaluation metrics including accuracy, loss, and training time. The experiments were conducted with epoch=20 and batch size=32, with 450 training steps in each epoch. The time comparison was presented in Table1, and the comparison of loss and accuracy for both models was shown in Figure 5, Figure 6, Figure 7 and Figure 8.

Table 1. The comparison between a model trained using the FedAvg algorithm and a model trained without FedAvg.

	Amount of steps in each epoch	The time takes for each step (s/step)	Total time for 20 epoch
FedAvg (4 Clients)	450 (113steps for each client)	3	1h54min36s
Without FedAvg	450	2	5h4min12s

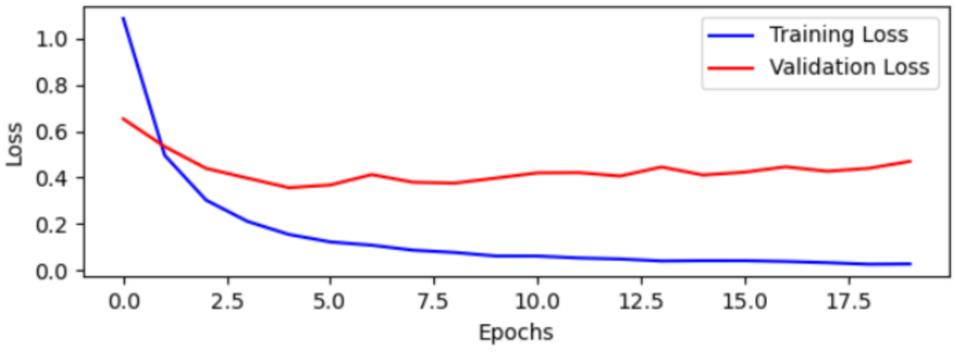

Figure 5. The variation of loss in train and validation sets without using FedAvg.

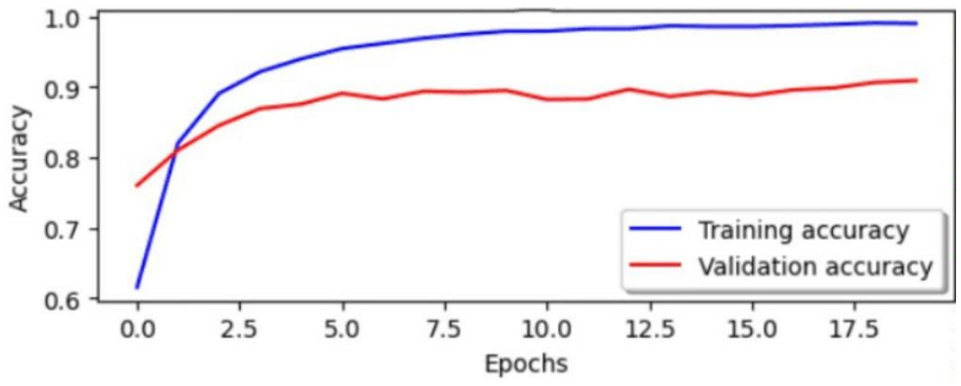

Figure 6. The variation of accuracy in train and validation sets without using FedAvg.

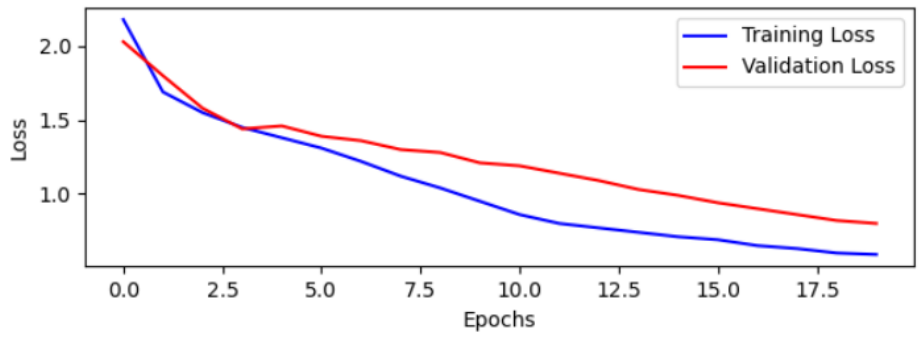

Figure 7. The variation of loss in train and validation sets in FedAvg algorithm.

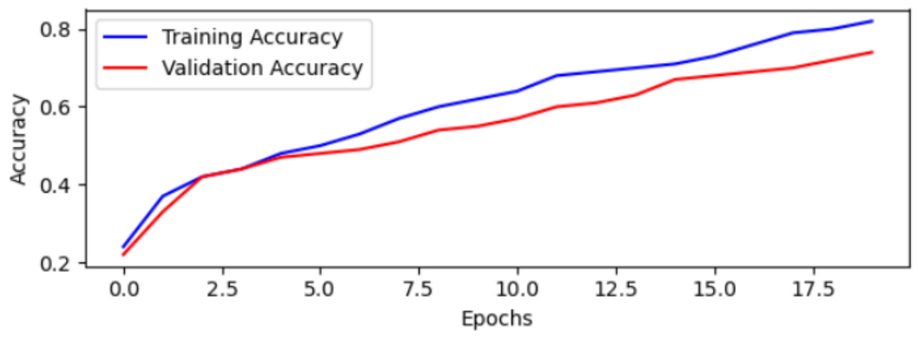

Figure 8. The variation of accuracy in train and validation sets in FedAvg algorithm.

Table 1 demonstrates the total training time for the model trained with the FedAvg algorithm was only 1 hour, while the model trained without the FedAvg algorithm took 5 hours. Although the FedAvg algorithm achieved a relatively shorter training time, there was a trade-off of 0.17 accuracy. In Figure 6, it can be observed that the algorithm models without using FedAvg achieved an accuracy increase from 0.6 to 0.98 over 20 rounds. However, in Figure 8, when FedAvg was employed for model training, the accuracy improved from 0.2 to 0.81. Compared to the algorithm models without using FedAvg, this algorithm demonstrates a significant improvement in accuracy.

In Table 1, when comparing, it can be observed that the algorithm models utilizing FedAvg consume more time for each training step, nearly three seconds per step. Therefore, one possible reason why the local updates in FedAvg take longer compared to training with a normal algorithm is the communication overhead involved in the federated learning process. In FedAvg, the model is trained collaboratively across multiple client devices, and each client performs local training on its own data. After local training, the client uploads its updated model to the server for aggregation with other client models and the update by each client consists of a large gradient vector [12]. This communication step adds additional time compared to a traditional centralized training approach where all data is available on a central server. The communication overhead in FedAvg arises from factors such as network latency, limited bandwidth, and synchronization between clients and the server. Each client needs to send its model updates, typically in the form of model weights to the server, and the server needs to wait until it receives updates from all participating clients before aggregating them. This synchronization and communication process can introduce delays, especially if the network conditions are suboptimal or if the number of clients is large. Although this will make a severe bottleneck to the communication, the model performs similarly on both the training and validation sets when using the FedAvg algorithm than without FedAvg, which indicates that the model is able to generalize well to unseen data. The validation set acts as a proxy for evaluating the model's performance on new, unseen examples that are similar to the training data.

4. Conclusion

In this experiment, this article employed federated learning to train a skin cancer classification system with the aim of protecting patients' personal privacy information. The author established the FedAvg algorithm and trained a DenseNet model to validate the training performance of federated learning in skin cancer classification. The experimental results demonstrate that federated learning, while preserving patient privacy, reduces training time and achieves excellent performance in skin cancer classification with high accuracy. The experiments also indicate that federated learning enables the model to achieve more consistent performance on both the training and validation sets. In future work, the authors desire to re-create actual data distribution scenarios using Non-IID data and evaluate the effectiveness of various federated learning algorithms in this project, such as FedProx and FedNova. Additionally, this study will intend to optimize the communication aspect of federated learning in order to reduce time loss caused by communication.

References

[1] Dildar M et al 2021 Skin Cancer Detection: A Review Using Deep Learning Techniques (International Journal of Environmental Research and Public Health) vol 18 p 5479
[2] Rashid J et al 2022 Skin Cancer Disease Detection Using Transfer Learning Technique (Sciences) vol 12
[3] Qiu Y et al 2022 Pose-guided matching based on deep learning for assessing quality of action on rehabilitation training. Biomedical Signal Processing and Control, 72, 103323.
[4] Faruqui N et al 2021 LungNet: A hybrid deep-CNN model for lung cancer diagnosis using CT and wearable sensor-based medical IoT data. Computers in Biology and Medicine, 139, 104961.
[5] Vipin V et al 2023 A deep neural network using modified EfficientNet for skin cancer detection in dermoscopic images (Decision Analytics Journal) vol 8 pp 2772-6622
[6] Nahata H et al 2020 Deep Learning Solutions for Skin Cancer Detection and Diagnosis (Machine Learning with Health Care Perspective vol 13) ed Jain V and Chatterjee J
[7] Daghrir J et al 2020 Melanoma skin cancer detection using deep learning and classical machine learning techniques: A hybrid approach (ATSIP: International Conference on Advanced Technologies for Signal and Image Processing) pp 1-5
[8] Wei L et al 2020 Automatic Skin Cancer Detection in Dermoscopy Images Based on Ensemble Lightweight Deep Learning Network vol 8 pp 99633-99647
[9] Zhou Y Ye Q and Lv J 2021 Communication-Efficient Federated Learning With Compensated Overlap-FedAvg (Transactions on Parallel and Distributed Systems) vol 33 pp 192-205
[10] Godlin Jasil S P and Ulagamuthalvi V 2021 Skin Lesion Classification Using Pre-Trained DenseNet201 Deep Neural Network (ICPSC: International Conference on Signal Processing and Communication) pp 393-396

[11] Adam K Mohamed I I and Ibrahim Y 2021 A Selective Mitigation Technique of Soft Errors for DNN Models Used in Healthcare Applications: DenseNet201 Case Study vol 9 pp 65803-65823
[12] Wang L Wang W and Li B 2019 CMFL: Mitigating Communication Overhead for Federated Learning (ICDCS: International Conference on Distributed Computing Systems) pp 954-964