Informative Initialization and Kernel Selection Improves t-SNE for Biological Sequences

Authors

Prakash Chourasia

Sarwan Ali

Dr. Murray Patterson

Georgia State University Atlanta USA

Presented by: Prakash Chourasia

GSU

Content

- Introduction
- Problem Formulation
- Idea
- Related Work
- Proposed Approach
- Evaluation
- Conclusion
- Future Work

Introduction

t-distributed stochastic neighbor embedding (t-SNE) :

- It is a method for interpreting high dimensional (HD) data by mapping each point to a low dimensional (LD) space (usually two-dimensional)
- Used for better visualization
- Dimensionality reduction.

Multi –access Edge Computing (MEC):

- bring compute capacity to networks edge
- It comes into picture for a new component - mobility

This paper: Propose for personalised FL Model, and faster convergence while using a new proposed metric

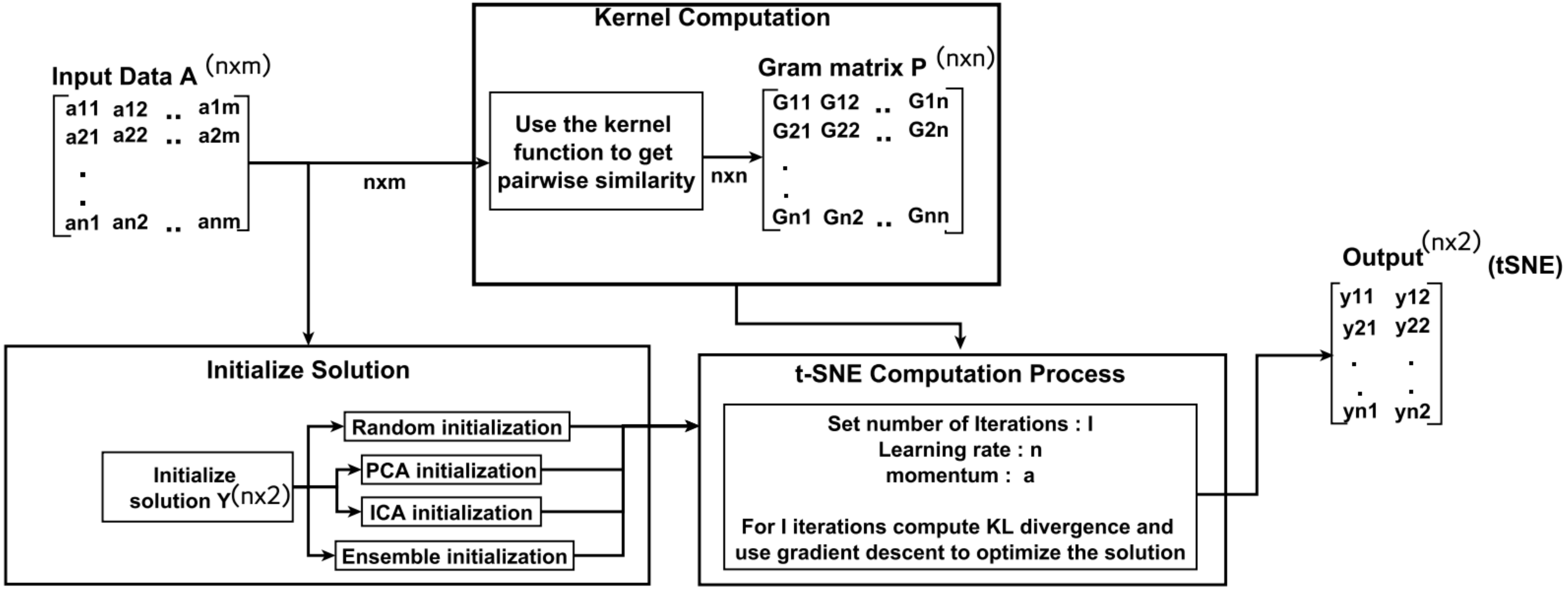
Federated Learning – ML with Keeping Data Privacy and Security

Content

- Motivation
- Problem Formulation
- Idea
- Related Work
- Proposed Approach
- Evaluation
- Conclusion
- Future Work

tSNE Workflow

Flow Chart


Kernel Computation
Input Data A (nxm)
a11 a12 .. a1m
a21 a22 .. a2m
an1 an2 .. anm
nxm
Use the kernel function to get pairwise similarity
nxn
Gram matrix P (nxn)
G11 G12 .. G1n
G21 G22 .. G2n
Gn1 Gn2 .. Gnn
Output(nx2) (tSNE)
y11 y12
y21 y22
yn1 yn2
Initialize Solution
Initialize solution Y(nx2)
Random initialization
PCA initialization
ICA initialization
Ensemble initialization
t-SNE Computation Process
Set number of Iterations : I
Learning rate : n
momentum : a
For I iterations compute KL divergence and use gradient descent to optimize the solution

Motivation

Edge Computing :

- bring compute capacity closer to data creator

(install server on network close to business)

Multi –access Edge Computing (MEC):

- bring compute capacity to networks edge
- It comes into picture for a new component - mobility

This paper: Propose for personalised FL Model, and faster convergence while using a new proposed metric

GSU

Motivation by example

Edge Computing – Multi access Edge Computing (MEC) – Federated Learning

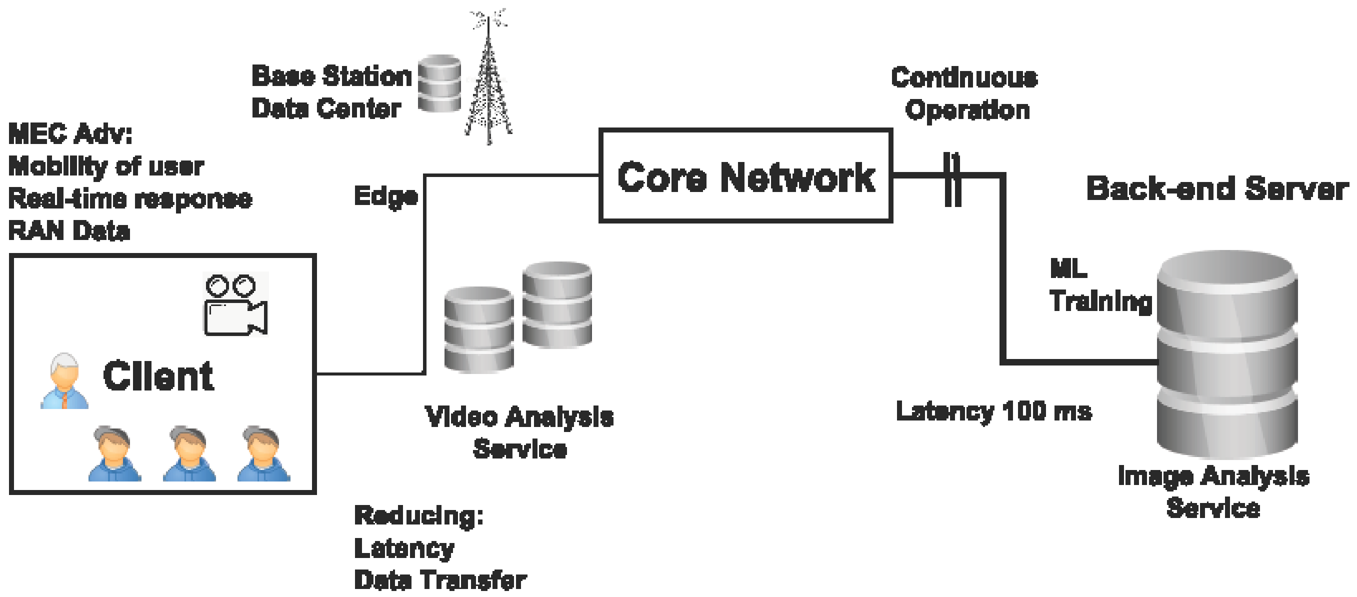


Fig 1.1 Edge Computing and Multi-access Edge Computing. (Some components are taken from external resources for this figure (dreamstime, pngkey)

This paper: Is logical next step to solve privacy concern.

Problem Formulation

Problem:

- Avail benefit of MEC – bring compute capacity closer to user
- Address privacy concern

Solution:

Federated Learning – collaboratively training Deep Neural Networks (DNNs) on mobile devices without transferring sensitive user data[Mills et al.].

GSU

Idea – Federated Learning

General Approach -

- A **global model** is initialized on a central computer known as a **federated server** and is given to (many) organization.
- Models are trained using the private data at each organization.
- A **global model** is modified using local updates (only weights) from local client.
- **Aggregation function** are used to aggregate response from local clients
- This process keeps going on **iteratively** to update regular and continuous data

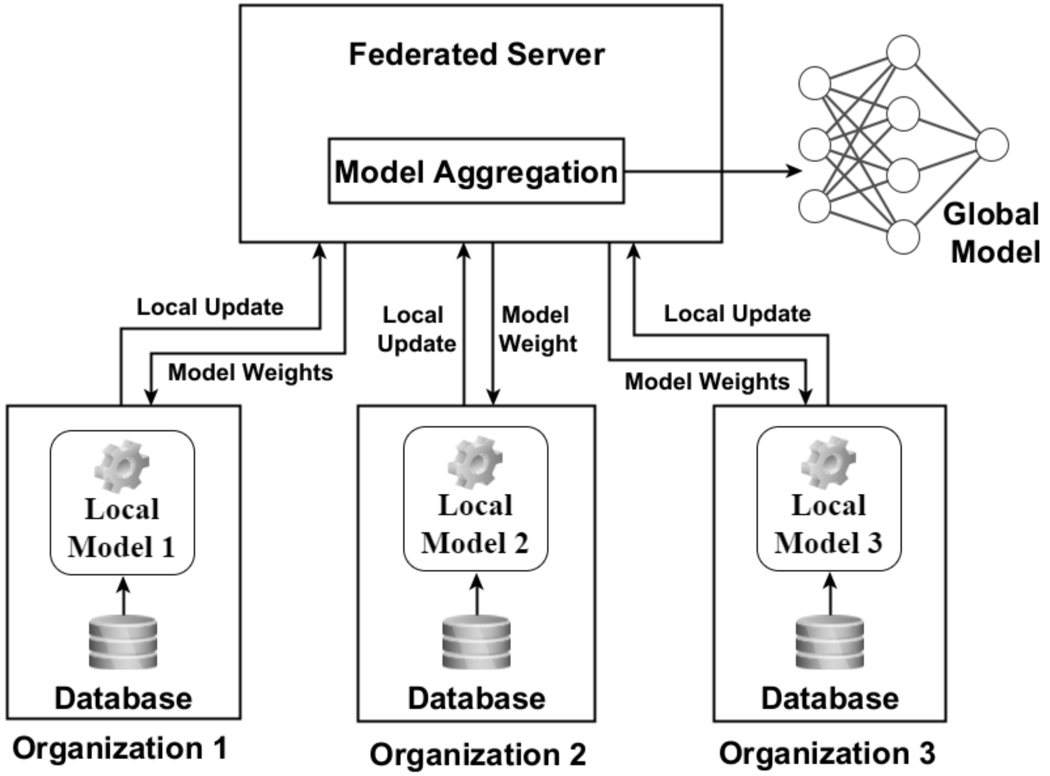


Fig 1.2 Federated Learning with example of 3 organization participating in FL training.

New Challenges

Addressed in paper:

- Clients typically lack Independent and Identically Distributed (IID) training data
- Use of inappropriate metric to evaluate performance
- Variability in performance by using global model on local data (non-IID)

Other Problems :

- Model aggregation and communication overhead in aggregation

Related Work

- Facilitating pairwise collaborations between clients with similar data.
  (Huang, Yutao, et al. "Personalized Cross-Silo Federated Learning on Non-IID Data." AAAI. 2021.)

GSU

Proposed Approach

- We convert the biological sequences to a fixed-length numerical representation, afterward, we compute different kernel matrices and show their effect along with several initialization methods on the quality of t-SNE.
- We show that the Kernel selection can play an important role and should be considered carefully rather than using the typical Gaussian kernel for t-SNE computation.
- We show that random initialization is inefficient for t-SNE computation. Alternatively, the Ensemble approach is a better choice to start with as an initial solution than the Random, Principal Component Analysis (PCA), and Independent Component Analysis (ICA) approaches.
- We evaluate the performance of the t-SNE using subjective (t-SNE plots) and objective (AU CRN X) criteria, and report results for different kernel computation methods along with different initialization approaches.
- We show that our proposed setting that includes the use of Laplacian kernel, along with ensemble initialization, outperforms the typical Gaussian and Isolation kernel-based methods in terms of AUC_RNX .

Workflow Steps

Workflow can be described in six steps –

1. Clients are sent a Work Request to take part in the FL round
2. Clients respond - based on their physical condition and regional preference
3. The global model and any optimization parameters are downloaded by clients.
   Apply tailored patches. (Batch Normalization layers as patches, are local)
4. After training, clients keep their distinctive patches for the next round.
   Authors used Adam Optimizer here instead of SGD for local clients.
5. The server waits until clients uploaded their model without patches and optimizer parameters or it waits till threshold time.
6. All models are averaged by the server, which then are stored as aggregate and a new round starts.

Architecture

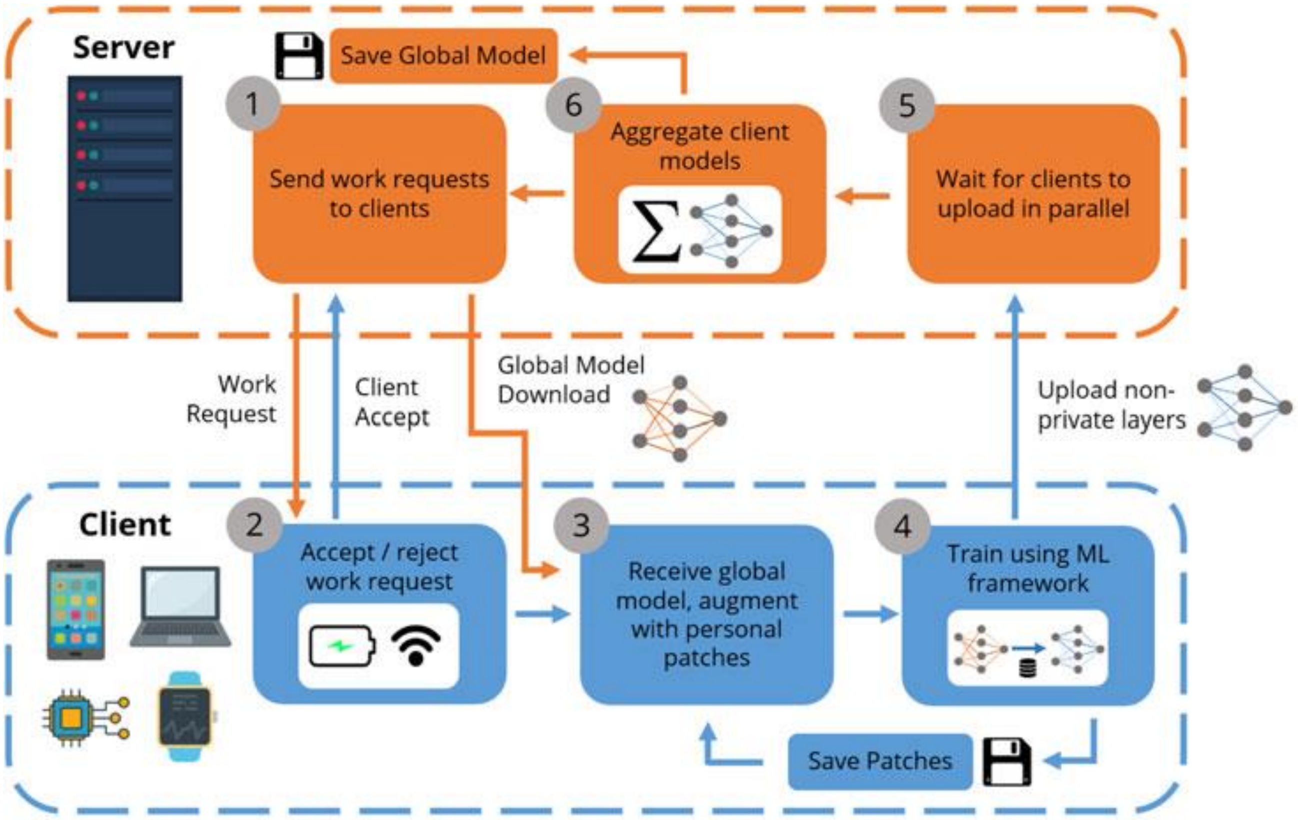


Figure 1.3: The workflow of MTFL algorithm [Mills et al.]

Personalised FL Model

What we have achieved so far from MEC to FL?

- Offloaded majority of computation at client

How it provides personalised FL Model

- By adding the private Batch Normalisation (BN) layers called as patch layers
- Preserving data privacy more strongly (not only user data is not uploaded, but key parts of their local models are not uploaded as well)
- Whereas in FedAvg and other methods the whole model is shared

Evaluation Metric –User Model Accuracy (UA)

What's wrong in existing metric ?

- Most of research work uses centralized Independent and Identically Distributed (IID) test-set to assess performance [Li et al. (2020)]
- All the user cares about is local performance

What is User Model Accuracy ?

- It measures the performance using a local test dataset
- Motivated from, adding modest per-task ”patch” layers to DNNs enhanced their performance in Multi Task Learning(MTL) scenarios [Mudrakarta et al. (2018)].

Objective Function

Aim is to minimize objective function (ultimately reduce the loss)

- In FedAvg [McMahan, Brendan, et al.] the objective function is

$$F_{\mathrm{FL}} = \sum_{k=1}^{K} \frac{n_k}{n} \ell_k(\Omega)$$

where K is the total number of clients,
n_k is the number of samples on client k,
n is the total number of samples across all clients,
ℓ_k is the loss function on client k,
and Ω is the set of global model parameters.

- When we add private BN layers patches the function of MTFL changes to

$$F_{\mathrm{MTFL}} = \sum_{k=1}^{K} \frac{n_k}{n} \ell_k(\mathcal{M}_k)$$

$$\mathcal{M}_k = \left(\Omega_1 \cdots \Omega_{i_1}, P_{k_1}, \Omega_{i_1+1} \cdots \Omega_{i_m}, P_{k_m}, \Omega_{i_m+1} \cdots \Omega_j\right),$$

where $\mathcal{M}_k$ is the patched model on client k [Mills, J., Hu, J., et al.]

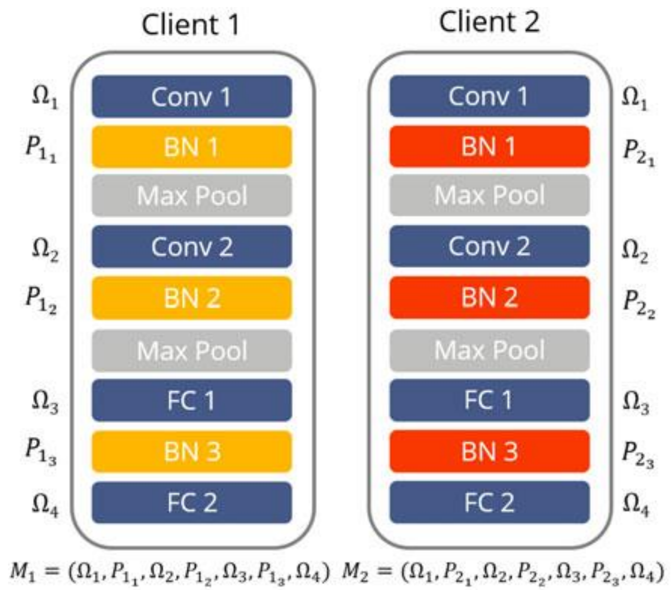


Fig. 1.3 Example composition of a DNN model used in MTFL [Mills et al.]

GSU

Optimization within MTFL (Faster Convergence)

MTFL runs rounds of communication until a given termination criteria (certain UA accuracy)

During these rounds - update is happening at 2 places

- The local update which uses the patch layers with federated layers to train the "local model".
- The non-private layer's parameters are aggregated to update the "Global model" at the server.

Proposed approach suggests the use of distributed optimization strategy:

The Adam moments of the local updates are also pushed to the server and averaged out to contribute to updating/creating the Global Model at the end of a respective iteration.

Optimization Strategy	Local Update	Global Update	Global Optim Update
FedAvg	SGD	Average	-
FedAdam	SGD	Adam	-
FedAvg-Adam	Adam	Average	Average

Table 1.1 Optimization and updates used by the FedAvg [McMahan, Brendan, et al.], FedAdam [Leroy, David, et al.], [Reddi, Sashank, et al.] and FedAvg-Adam FL Training Strategies

Experimental Setup

Dataset

- MNIST [LeCun, Yann, et al.] : Handwritten digits in greyscale images with a 28×28 pixels from 10 labels. This dataset was used with a “2NN” network.
- CIFAR10 [Krizhevsky et al.] : RGB images of size 32×32 pixel, objects from 10 classes and “CNN’ network is used.

Evaluation Metrics

- The target average UA is set to 97% for MNIST and 65% for the CIFAR10 dataset
- To simulate a low-powered clients, a testbed configuration was made using 10 local clients(RPi) linked by WiFi with the server [Mills et al. (2021)].

Evaluation

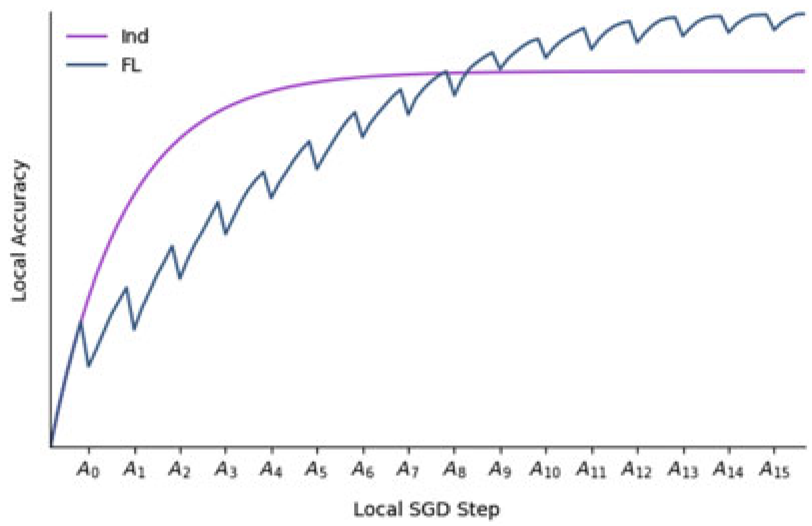


Figure 1.3.Accuracy curve for FL and Independent training [Mills et al. (2021)]

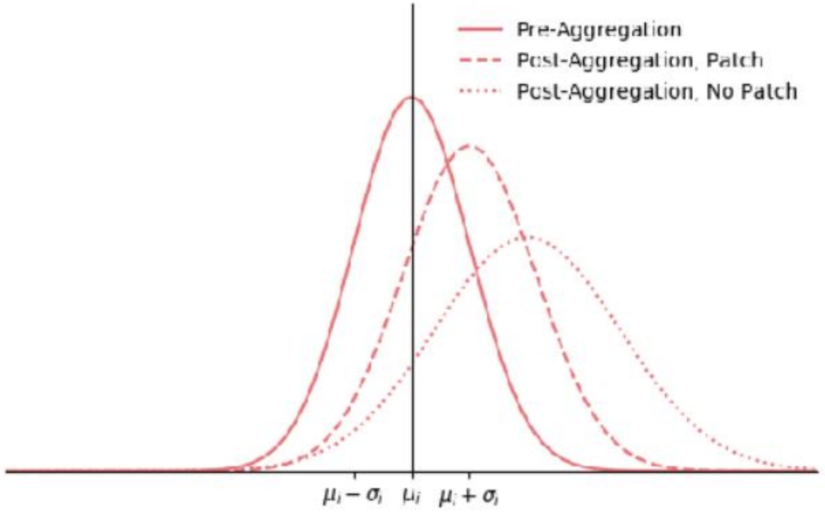


Figure 1.4: Effect of patch Layers on activation [Mills et al. (2021)]

Results

MTFL can cut the number of cycles needed to reach targeted UA by 5×, with an additional 3× improvement was seen with FedAvg-Adam

MNIST - 2NN																
	FL				**MTFL**											
	Private values = None				$\mu, \sigma, \gamma, \beta$				μ, σ				γ, β			
Optimisation	W = 200		400		200		400		200		400		200		400	
Strategy	C = 0.5	1.0	0.5	1.0	0.5	1.0	0.5	1.0	0.5	1.0	0.5	1.0	0.5	1.0	0.5	1.0
FedAvg	99	102	107	110	85	58	101	68	X	X	X	X	29	21	34	26
FedAdam	85	69	88	65	56	37	75	77	109	90	194	262	31	25	31	27
FedAvg-Adam	44	49	40	50	17	41	19	32	131	151	170	198	**9**	**9**	**10**	**9**
CIFAR10 - CNN																
FedAvg	139	138	171	164	49	33	55	36	231	280	258	266	37	24	45	30
FedAdam	105	90	83	80	21	14	22	16	67	45	48	38	24	14	25	16
FedAvg-Adam	57	43	36	31	11	9	14	**8**	82	79	62	63	**10**	**7**	**11**	**8**

Table 1.2 Communication Rounds Required to Reach Target Average User Accuracies for Different Tasks Using FL and MTFL (With Private Statistics μ, σ and/or Trained Parameters γ, β), for Different Numbers of Total Clients W, Client Participation Rates C, and Optimisation Strategies [Mills et al.] .

Note –X denotes the target UA not reached within 500 rounds.

Robustness Check

- 20 percent of the clients at random had 0-mean Gaussian noise added to their training data
- Gaussian noise with standard deviation 3 for MNIST and 0.2 for CIFAR10

MNIST - 2NN																
	FL				**MTFL**											
	Private values = None				$\mu, \sigma, \gamma, \beta$				μ, σ				γ, β			
Optimisation	W = 200		400		200		400		200		400		200		400	
Strategy	C = 0.5	1.0	0.5	1.0	0.5	1.0	0.5	1.0	0.5	1.0	0.5	1.0	0.5	1.0	0.5	1.0
FedAvg	276	X	290	X	115	76	144	144	85	58	102	68	50	36	65	48
FedAdam	X	X	X	X	76	47	110	89	56	37	75	77	43	33	46	53
FedAvg-Adam	133	260	191	X	20	16	24	27	17	41	19	32	**12**	**8**	**15**	**40**
CIFAR10 - CNN																
FedAvg	148	208	202	250	47	32	52	35	239	186	260	88	36	24	43	28
FedAdam	159	91	92	93	21	14	21	15	74	49	51	42	34	16	21	14
FedAvg-Adam	193	X	X	X	14	10	16	**9**	103	111	67	74	**12**	**8**	**13**	**9**

Table 1.3 Communication Rounds Required when 20 percent of Clients Have Noisy Training Data, for Different Numbers of Total Clients W, Client Participation Rates C, and Optimisation Strategies [Mills et al.] .

Note –X denotes the target UA not reached within 500 rounds.

Testbed Results

For 10 rounds the average time, % of time spent per round in downloading models, local training, uploading models and work performed on the server was compared (no model testing on server).

MTFL(FedAvg-Adam) took the longest per round due to the increased number of weights

However, the far fewer rounds required to reach a target average UA will outweighed the increase in communication time.

Learning Scheme	Round Time (s)	Percentage of Round Time (%)			
		Down	*Client*	*Up*	*Server*
MNIST - 2NN					
FL(FedAvg)	30	5	88	6	1
MTFL(FedAvg-Adam)	38	11	76	12	1
Independent	29	0	100	0	0
CIFAR10 - CNN					
FL(FedAvg)	108	5	86	5	4
MTFL(FedAvg-Adam)	136	11	74	12	3
Independent	100	0	100	0	0

Table 1.4 Average Time Per Round of Different Learning Schemes on the MNIST and CIFAR10 Datasets, and Percentage of Time Spent Downloading the Model (Down), Training the Model (Client), Uploading the Model (Up), and Model Aggregation/Distribution on the Server (Server) Took [Mills et al.] .

Conclusion

- MTFL provides personalised FL model with more data privacy and security
- Proposed approach can substantially reduce the number of rounds to reach target UA thus provides faster convergence
- User model Accuracy (UA) brings attention to appropriate direction for evaluating models

Shortcomings And Future Work

Shortcomings

- Heterogeneous hardware of clients is an important component not considered in this work
- Impact of Adam optimizer on Global model update is not evaluated

Future Work

- Different aggregation techniques such as decentralized aggregation or asynchronous aggregation [Jiang, Haotian, et al.]
- Peer to peer collaboration of private parameters (issue of integration with in subdivisions of organization)
- Optimizing the loss function to measure the fairness on keeping different smaller test data at client
- Use of different types of data rather than all image data(since uniformity in data storage is hard to achieve)

Dynamic-Fusion-Based Federated Learning for COVID-19 Detection

Content

- Motivation
- Problem Formulation
- Idea
- Related Work
- Proposed Approach
- Evaluation
- Conclusion
- Future Work

Motivation

- Deep learning - state-of-the-art techniques in computer vision
- Use of imaging techniques such as Computed Tomography (CT) and Positron Emission Tomography (PET) scans, MRIs, and, X-rays, by radiologists in diagnose of medical disorders.
- Computer-assisted detection and diagnosis could help in improving quality of medical services [Nguyen et al. (2022)]
- Privacy and data security concerns (HIPAA [CDC])
- Healthcare organizations are known to prefer models validated on their data [Buch et al. (2021)]

Problem Formulation

Federated Learning seems a good fit to address the privacy concern.

However when there is significant data heterogeneity [Zhang et al.(2021)] among clients, the default setup of FL works poorly and requires massive communication costs.

Problem:

High communication cost and decision to make for local model participation in building FL global model

Related Work

- Using federated Learning for Covid19 detection in medical image data [Kumar, Rajesh, et al.], [Yan, Bingjie, et al.].
  (Kumar, Rajesh, et al. "Blockchain-federated-learning and deep learning models for covid-19 detection using ct imaging." *IEEE Sensors Journal* 21.14 (2021): 16301-16314.)
  (Yan, Bingjie, et al. "Experiments of federated learning for COVID-19 chest X-ray images." *International Conference on Artificial Intelligence and Security*. Springer, Cham, 2021.)

  Weakness: Focus on data aggregation and accuracy
  Does not mention about efficiency in training; rather focus on finding better deep learning architecture

- Model compression methods are utilized to reduce the communication cost by keeping check on updated parameters [Luping, W., et al.].
  (Luping, W., et al. "CMFL: Mitigating communication overhead for federated learning." 2019 IEEE 39th international conference on distributed computing systems (ICDCS). IEEE, 2019.)
  Weakness: Additional computation load was on low powered client in order to evaluate if global parameters make any relevancy
  Communication-Mitigated Federated Learning (CMFL) improves the communication efficiency by only 5.7x.

- Additionally, communication techniques are introduced to improve communication efficiency, e.g., over-the-air computation technique [Yang, Kai, et al.] and multichannel random access communication mechanism [Choi, Jinho, et al.].
  (Yang, Kai, et al. "Federated learning via over-the-air computation." IEEE Transactions on Wireless Communications 19.3 (2020): 2022-2035.)
  (Choi, Jinho, and Shiva Raj Pokhrel. "Federated learning with multichannel ALOHA." IEEE Wireless Communications Letters 9.4 (2019): 499-502.)
  Weakness: These methods are focused on the superposition property of a wireless multiple-access channel to improve the communication efficiency and reduce the required bandwidth. Local models have less control in decision making.

Proposed Approach

A dynamic Fusion based solution which takes two things into account

- Performance of local model in previous and ongoing round of training
- Time taken by local model to train

A **dynamic fusion federated learning [Zhang et al. (2021)]** involves two important decision making

- Decide client participation – each **client** makes a decision for participating in a model update for the round.
- Client selection - the **server** chooses a client based on *waiting time*

Communication Workflow

<u>Client</u>

Model Trainer - downloads training job; perform training with local data, training time is sent back to the coordinating server

Model Assessor –

- Receives aggregation request and act based on performance compares the previous model's performance with the current one.
- If performing better, it sends the request for model upload; if not, it declines to take part in this round's global model update
- Models that cannot finish the training epochs in the allotted time are discarded

<u>Server</u> (coordinator)

Job Creator - configures the initial model and the initial waiting time, create training jobs

Aggregate scheduler - modify the waiting time in accordance to the previous training times and sends the aggregation request to the client

Model Aggregator - updates the global model and measures its performance

Architecture

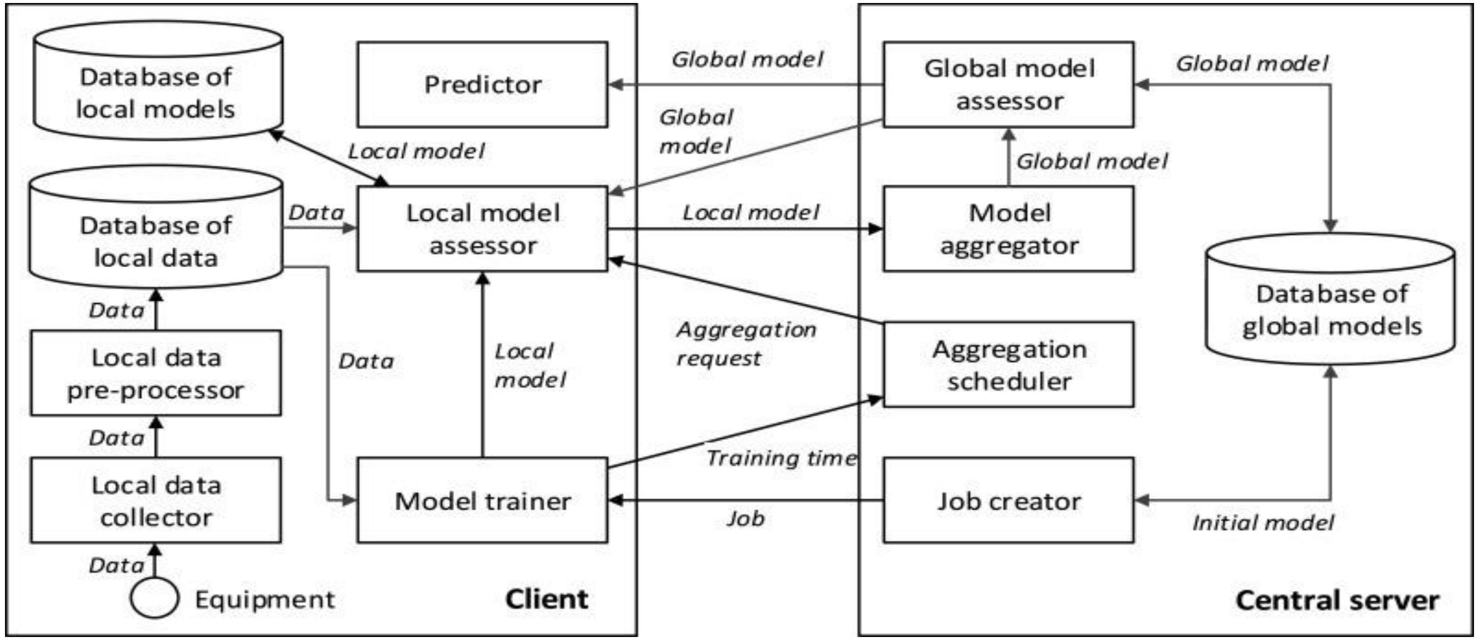


Figure 2.1 Architecture dynamic fusion federated learning (DF_FL) [Zhang, Weishan, et al]

Dataset and Evaluation Metric

Dataset

Type	Number	Data Size (in MB)	COVID-19 Positive	COVID-19 Negative	Viral Pneumonia
CT	746	92.6	349	397	0
X-Ray	2960	1182.2	274	1341	1345

Table 2.1 Category of Medical Diagnostic Image Data Sets for COVID-19 Detection [Zhang, Weishan, et al]

Evaluation Metrics

- The experiment was done using three models, GhostNet, ResNet50, and ResNet101.
- The metric used is **accuracy** to evaluate the performance of three models for 6 different configuration.
- Communication efficiency is measured by counting the number of uploads and time taken to upload models.

Experimental Setup

- Six different configuration (combination) of CT and X-ray images are used
- 2800 images were used for training and 526 for testing, 3 Clients are used for FL.
- In total we get $6 \times 3 = 18$ experiments. Random crop is used to process test set and its performance for each model is also evaluated

Local Client 1 (CT/X-ray)	**Data Size (in MB)**	**Local Client 2** (CT/X-ray)	**Data Size (in MB)**	**Local Client 3** (CT/X-ray)	**Data Size (in MB)**	**Ratio**	**Total Data Size (in MB)**	**Amount**
600/0	76.8	0/900	76.8	600/0	545.7	600/2200	1013.8	2800
300/300	168.5	0/900	392.5	300/300	546.6	300/2500	1107.6	
200/400	196.8	0/900	389.1	0/1300	534.5	200/2600	1120.4	
150/450	209.9	0/900	381.6	0/1300	544	150/2650	1135.5	
200/400	197.4	200/700	318.9	0/1300	557.5	400/2400	1073.8	
200/400	198.6	200/700	317.2	200/1100	497	600/2200	1012.8	

Table 2.2 Client wise data configuration [Zhang, Weishan, et al]

Evaluation

- Accuracy plots for the Resnet101 model
- Number of rounds needed is significantly less for DF_FL
- Random crop is used to process test set and its performance is also evaluated.
- The findings show that in 14 out of 18 experiments, the suggested DF_FL achieves more accuracy than the D_FL.
- Other 4 also differ marginally(0.57%, 1.331%, 0.951%, and 1.141%).
- Group 4 results show that the proposed method can ensure fault tolerance and is resilient (data is meddled to test)

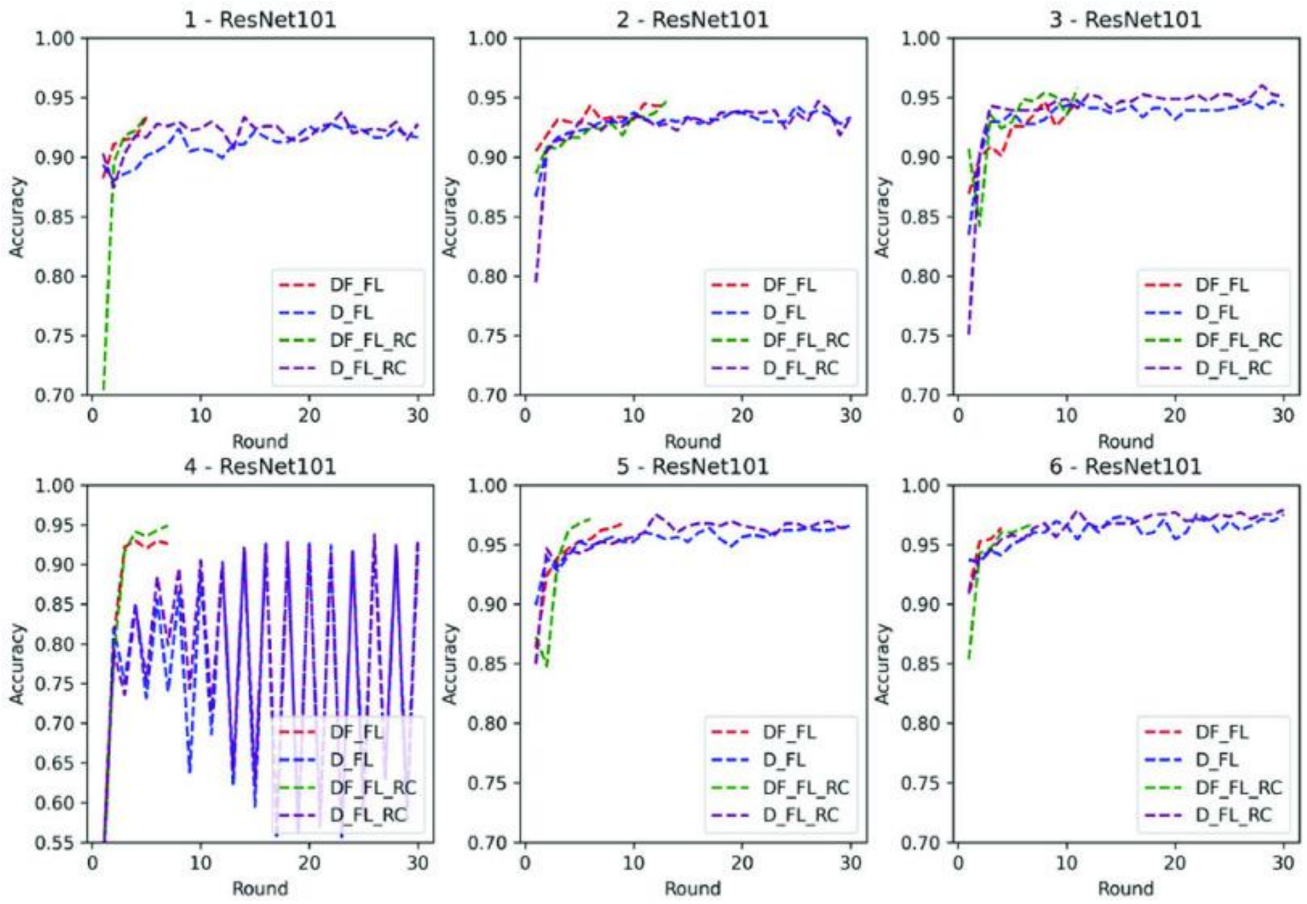


Figure 2.2 Accuracy of Resnet101 [Zhang, Weishan, et al]

Results

- DF_FL cuts the upload time and number of uploads required to reach accepted accuracy
- Average time for uploads reduced by an average of

1/3 1/10 1/16

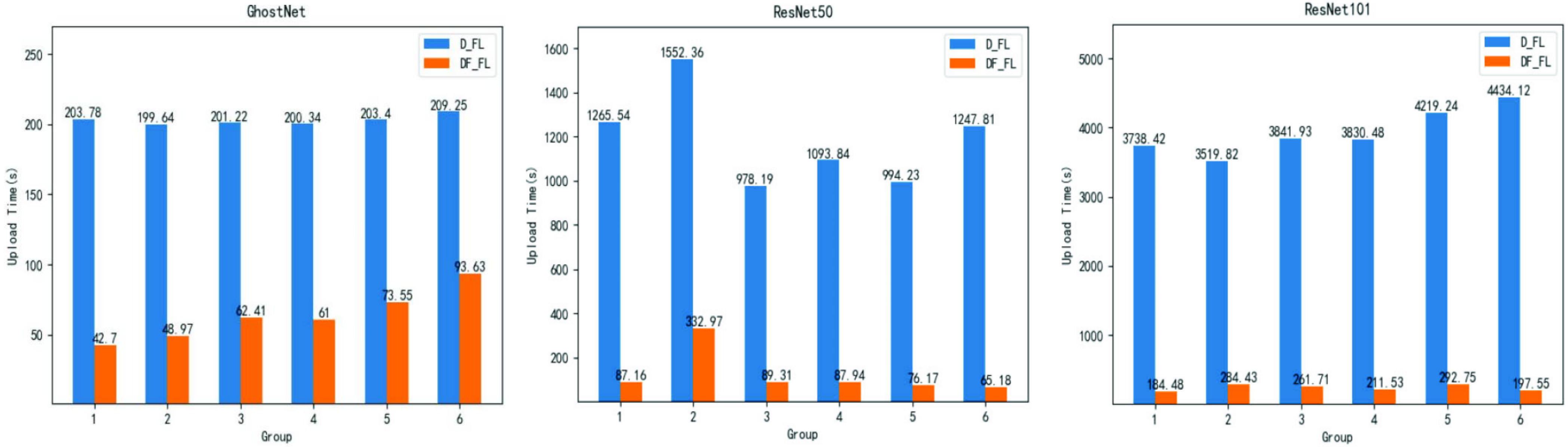


Figure 2.2 Upload time for 3 models - Ghostnet, Resnet50, Resnet101 [Zhang, Weishan, et al]

Number of Upload

- DF_FL cuts the upload time and number of uploads required to reach accepted accuracy
- Average number of uploads reduced significantly for models by
  GhostNet - **61** ResNet50 – **80** Resnet101 - **78**

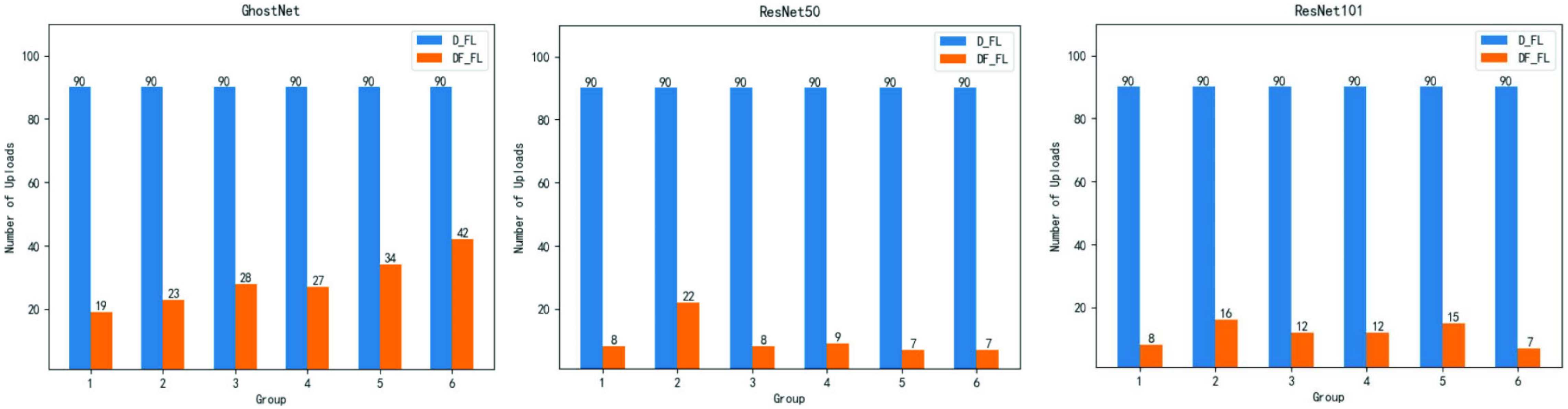


Figure 2.2 Number of upload for 3 models - Ghostnet, Resnet50, Resnet101 [Zhang, Weishan, et al]

Conclusion

- Dynamic Fusion based FL model reduces the communication overhead through fewer models uploads while improving accuracy, and keeping the data privacy and security

- Especially while working with complicated models with large number of parameters like Resnet50 and Resnet101.

- In terms of fault tolerance, model accuracy, resilience, and efficient communication, the assessment results show proposed approach performs better than the default configuration of FL

Shortcomings And Future Work

Shortcomings

- As we can see in Table 2.1 dataset is highly class imbalanced
- Accuracy is used as metric whereas Recall is a better measure for this case (can not afford the false-negatives).
- Authors claimed even after meddling the data 4th group, where images of negative Covid-19 are marked as positive. It is still unclear why dynamic fusion-based FL works better.
- Random crop data augmentation is used in test set but not in training dataset.

Future Work

- Good idea to combine Paper 1 into the paper 2
- Different aggregation methods could be used such as asynchronous aggregation [Xie et al.]
- Peer to peer collaboration of private parameters (issue of integration with in subdivisions of organization)
- Combining heterogeneous data set rather than all image data (combining with decision tree and neural network)

Summary and Relationship of both papers

The first paper focuses on the evolution of edge computing, a journey from traditional Edge computing to MULTI-ACCESS Edge Computing (MEC),

and to further use of federated learning to counter the challenges in the former ones.

The second paper shows use of federated learning with a dynamic fusion technique to recognize Covid-19 from medical images. Proposed approach solves the problem by means of communication efficiency

In Conclusion both papers are motivated by concerns of data privacy and security. Both of them shows that, the use of FL outweighs traditional approaches with few changes in the default FL.

References[1]

1. Li, Tian, et al. "Federated learning: Challenges, methods, and future directions." IEEE Signal Processing Magazine 37.3 (2020): 50-60.
2. Mills, J., Hu, J., and Min, G. (2021). “Multi-task federated learning for personalised deep neural networks in edge computing.” IEEE Transactions on Parallel and Distributed Systems, 33(3), 630–641.
3. McMahan, Brendan, et al. "Communication-efficient learning of deep networks from decentralized data." Artificial intelligence and statistics. PMLR, 2017.
4. Leroy, David, et al. "Federated learning for keyword spotting." ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2019.
5. Reddi, Sashank, et al. "Adaptive federated optimization." arXiv preprint arXiv:2003.00295 (2020).
6. Krizhevsky, Alex, and Geoffrey Hinton. "Learning multiple layers of features from tiny images." (2009): 7.
7. LeCun, Yann, et al. "Learning algorithms for classification: A comparison on handwritten digit recognition." Neural networks: the statistical mechanics perspective 261.276 (1995): 2.
8. Aledhari, M., Razzak, R., Parizi, R. M., and Saeed, F. (2020). “Federated learning: A survey on enabling technologies, protocols, and applications.” IEEE Access, 8, 140699–140725
9. Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., et al. (2021). “Advances and open problems in federated learning.” Foundations and Trends® in Machine Learning, 14(1–2), 1–210
10. Jiang, Haotian, et al. "Distributed deep learning optimized system over the cloud and smart phone devices." IEEE Transactions on Mobile Computing 20.1 (2019): 147-161.

References [2]

1. Zhang, Weishan, et al. "Dynamic-fusion-based federated learning for COVID-19 detection." IEEE Internet of Things Journal 8.21 (2021): 15884-15891.
2. Mills, J., Hu, J., and Min, G. (2021). “Multi-task federated learning for personalised deep neural networks in edge computing.” IEEE Transactions on Parallel and Distributed Systems, 33(3), 630–641.
3. Xie, C., Koyejo, S., and Gupta, I. (2019). “Asynchronous federated optimization.” arXiv preprint arXiv:1903.03934.
4. Shaheen, M., Farooq, M. S., Umer, T., and Kim, B.-S. (2022). “Applications of federated learning; taxonomy, challenges, and research trends.” Electronics, 11(4), 670.
5. Dayan, Ittai, et al. "Federated learning for predicting clinical outcomes in patients with COVID-19." Nature medicine 27.10 (2021): 1735-1743.
6. Buch, Varun, et al. "Development and validation of a deep learning model for prediction of severe outcomes in suspected COVID-19 Infection." *arXiv preprint arXiv:2103.11269* (2021).
7. Kumar, Rajesh, et al. "Blockchain-federated-learning and deep learning models for covid-19 detection using ct imaging." *IEEE Sensors Journal* 21.14 (2021): 16301-16314.
8. Yan, Bingjie, et al. "Experiments of federated learning for COVID-19 chest X-ray images." *International Conference on Artificial Intelligence and Security*. Springer, Cham, 2021.
9. CDC, Health insurance portability and accountability act of 1996 (HIPAA), https://www.cdc.gov/phlp/publications/topic/hipaa.html, [Online; accessed 09-September-2022] (2021).

Thank you.