Recent Trends in AI & Federated Learning

Dr. Gagan Raj Gupta

Associate Professor

Indian Institute Of Technology, Bhilai

Department of Computer Science & Engineering

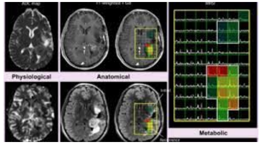
December 12, 2024

Presentation Outline

- Background
- Federated Learning
- Split Learning
- Literature Review
- Research Gap
- ESL System Architecture
- ESL Algorithm
- Experimental Goals
- Experimental Setup
- Results and Use Case in Medical Imaging
- Conclusion
- Future Work

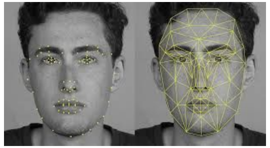
Background

- Deep Learning models thrive on large datasets.
- Lack of samples at single location.
- Data privacy: Sensitive data in healthcare, finance, and other industries.
- Effective utilization of distributed computation resources.



I. Medical Imaging

II. Object Detection



III. Image Recognition

This motivates to develop communication efficient, privacy preserving collaborative learning learning on resource constrained devices. Federated and Split Learning are two representative emerging collaborative learning methods

[I] https://bids.berkeley.edu/sites/default/files/styles/400x225/public/projects/vareth_-_nmr_cover_new_project_page_banner.png?itok=FGmASmU2
[II] https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcSNcllVUlYAa5IQN1wiSbJbdcowkH84NFAcPA
[III] https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcQwJns0JqaFaxnXG_zbRtFFVzOfco1uIxypgQ

Federated Learning-Setup

There are two types of entities in the FL system-

- ➢ The **data owners** (FL clients) and
- ➢ The **model owner** (FL server)

Let N = 1,2,...,K denote a set of K clients each having a dataset D_k (k ϵ K), then the entire dataset is $D = U^K D_k$.

At the beginning of a learning process, the FL server first initializes a model training task.

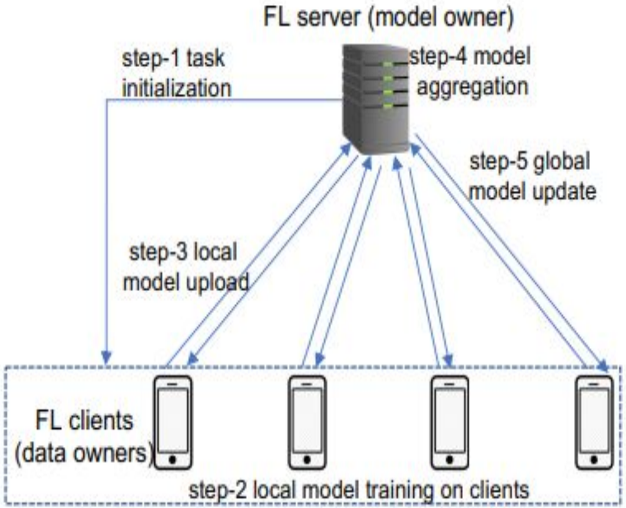


Figure:The framework architecture of Federated Learning.

Image Reference:]https://arxiv.org/abs/2207.09611

Federated Learning-Basic Working

- Each selected client k uses its own dataset D_k to train a **local model** and uploads the trained model parameters to the server.
- At the end of each training iteration, the trained models from all clients are aggregated by a server into a **global model**.
- The above process of local training–global aggregation is repeated for multiple rounds until a certain level of accuracy is achieved for the global model.

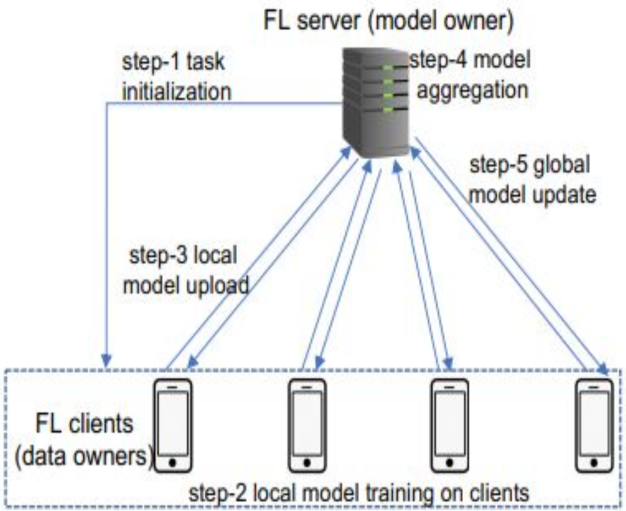


Figure:The framework architecture of Federated Learning.

Image Reference:]https://arxiv.org/abs/2207.09611

Federated Learning-Benefits and Challenges

Benefits:

- Data Privacy
- Scalable Architecture

Limitation:

- Requires heavy resource consumption
- Disparities in data distribution
- System Heterogeneity
- Communication Overhead
- System Dynamism
- Privacy and Security

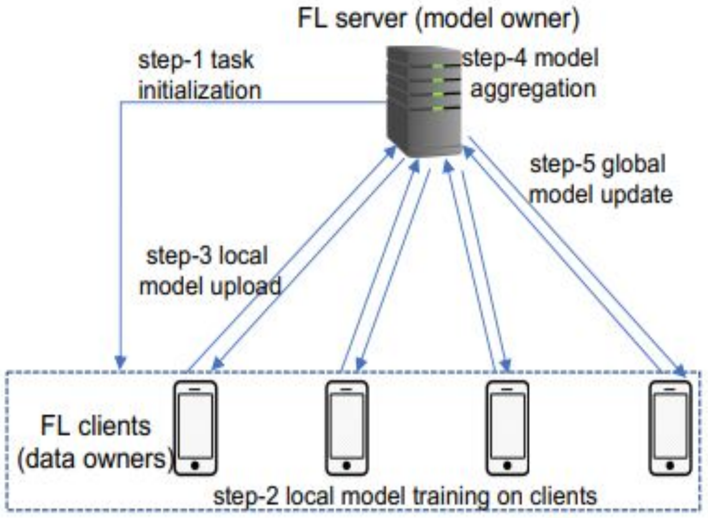


Figure:The framework architecture of Federated Learning.

Image Reference:]https://arxiv.org/abs/2207.09611

Enabling FL in Resource Constrained Environment

	FL Algorithms	Model Aggregation	Client Selection	Communication Control	Privacy/Security Protection
Data Heterogeneity	+	+	+		
System Heterogeneity	+	+	+		
Communication Overheads				+	
Constrained Resources			+	+	
System Scalability			+	+	
System Dynamism		+	+		
Privacy & Security					+

The main challenges to FL in resource constrained environment and representative technical strategies for addressing them.

Constrained Resources motivated the development of Split Learning Approaches

Image Reference:]https://arxiv.org/abs/2207.09611

Split Learning (SL)-Setup

- In the framework, an ML model is split into two portions-
    - the client-side model W_C
    - the server-side model W_S

- Similar to FL, all the raw training data are stored on the client without being transmitted to the server.

- The training of the full model is performed by executing a sequence of forward propagation and backpropagation between the client and server.

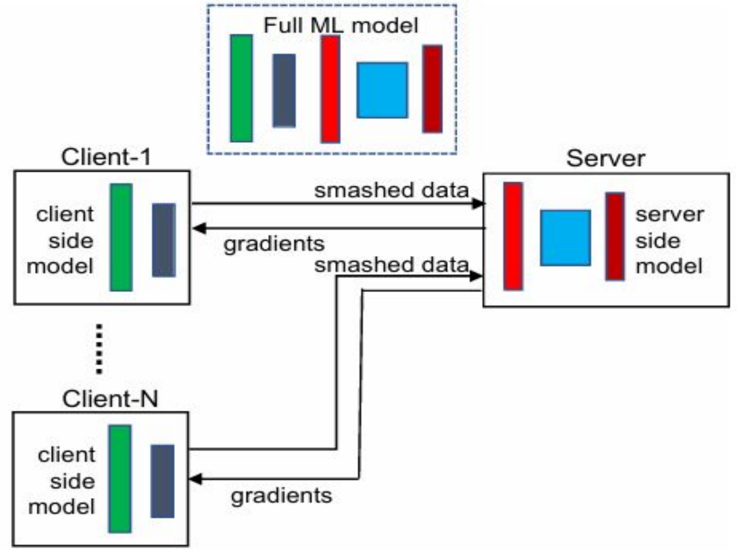


Figure: Framework architecture for multi-client split learning.

Image Reference: https://arxiv.org/abs/2207.09611

Split Learning (SL)-Working

- The client uses the training data to feed the model W_C and performs forward propagation until the cut layer.
- Then the cut layer's activations are transmitted to the server.
- The server uses the smashed data received from the client as the inputs to its model W_S and completes forward propagation on the remainder of the full model.

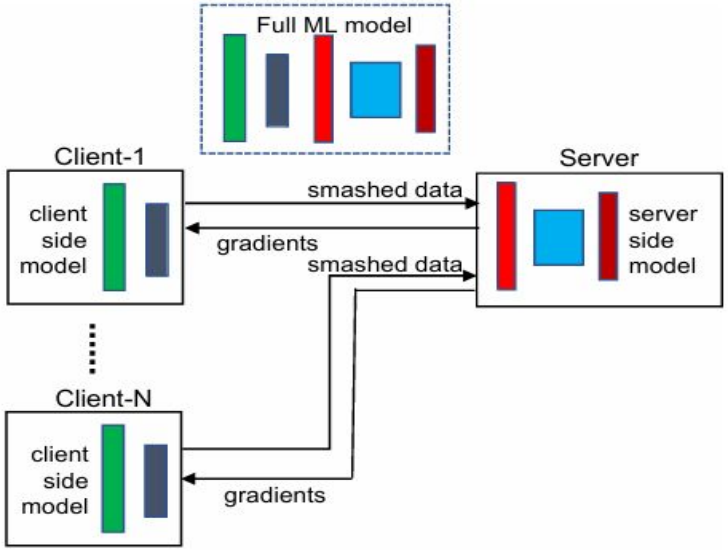


Figure: Framework architecture for multi-client split learning.

Image Reference: https://arxiv.org/abs/2207.09611

Split Learning (SL)-Working

- After calculating the loss function, the server starts the backpropagation process in which it computes gradients and updates the weights of each layer of W_S until reaching the cut layer.
- Then the server transmits the gradients of smashed data back to the client.
- Upon receiving the gradients from the server, the client executes its backpropagation on W_C to complete a single pass of backpropagation of the full model.
- In SL, the forward propagation and backpropagation between the client and server continues until a convergence point is reached for the full model.

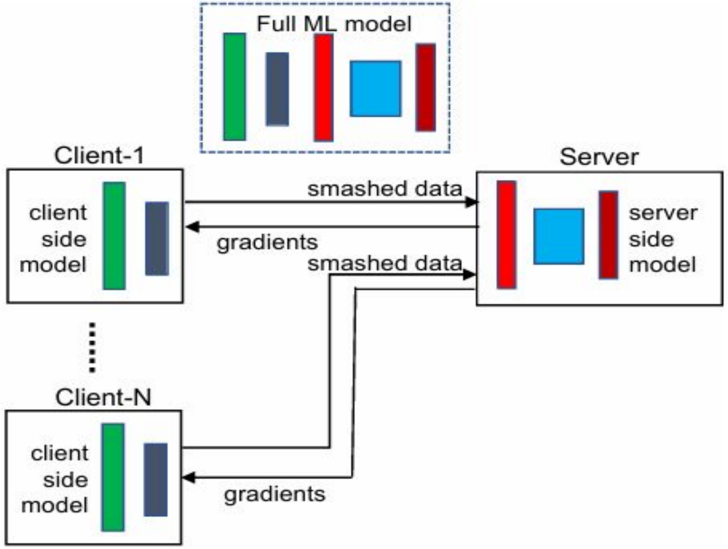


Figure: Framework architecture for multi-client split learning.

Image Reference: https://arxiv.org/abs/2207.09611

Split Learning (SL)-Configuration

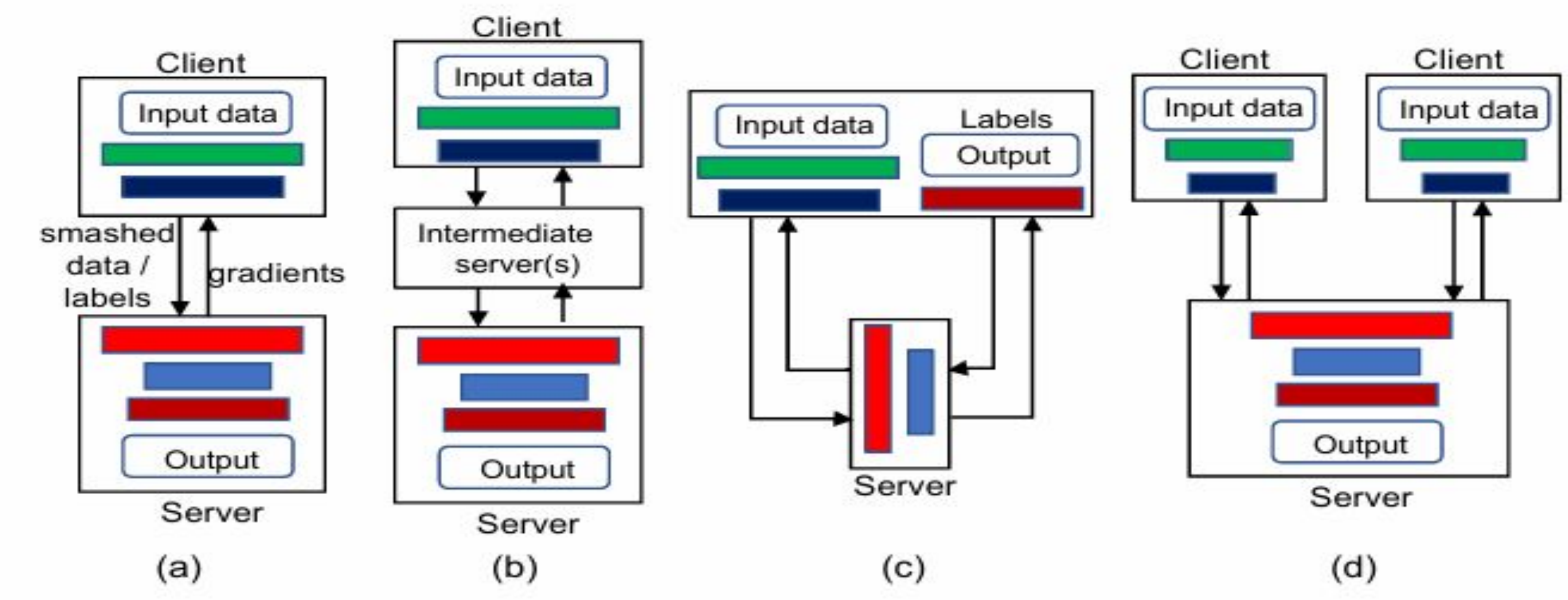


Figure: Configurations for split learning frameworks: (a) basic configuration, (b) extended configuration, (c) U-shape configuration, (d) vertical configuration

Image Reference: https://arxiv.org/abs/2207.09611

Split Learning (SL)

Benefits:

- Data Privacy
- Scalable Architecture and Edge Computing Benefits
- Reduces Computation overhead

Limitation:

- Disparities in data distribution
- Introduces additional communication overheads.

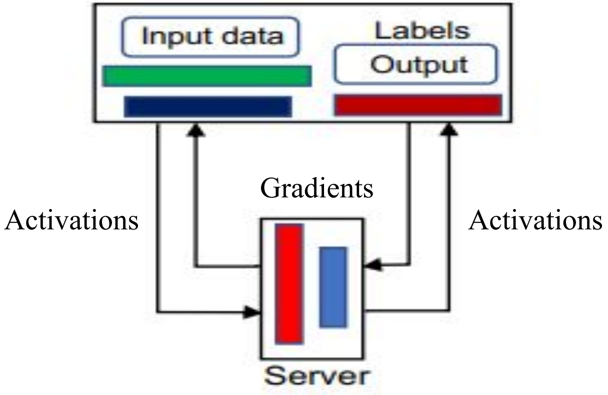


Figure: Split Learning without Label sharing

Therefore, communication efficiency is an important aspect of SL performance.

Image Reference: https://arxiv.org/abs/2207.09611

Literature Review:

Traditional approach to reduce communication overhead:

- Accelerate Convergence Strategies:
    - Momentum-based Optimization[1]
    - Asynchronous Training: [8][9]
- Communication Compression Techniques:
    - Sparsification, Quantization, and Pruning[10][11][12],
    - Model Compression[13][14][15]
    - Autoencoder-based Compression[16][17][18]

Traditional Approach tackle data heterogeneity:

- Enhance FL Algorithm[1][2][3]
- Adaptive Learning Rates[4]
- Personalised Learning[5][6][7]

Research Gap:

Challenges with Traditional Approach:

- **Trade-offs:**
  - Traditional strategies prioritize compression or fast convergence techniques to reduced communication. It affects model accuracy.
  - For example “Communication for Split Learning by Randomized Top-k Sparsification, 2023[11]” resulted in an 88% reduction in communication but caused a 3% decrease in accuracy.

- **Unaddressed Concern:**
  - Computation overhead remains a critical issue in existing FL/SL approaches.
  - The experimental study lacked diversity in complex tasks and realistic datasets.

Problem Statement

In our study, we seek to achieve optimal performance, especially in non-IID settings, with a focus on rapid convergence, minimized computation and communication overhead using Split Learning Setting.

Efficient Split Learning: Main Idea

1. **Deep Learning model as a function**	$f(x) = y$
2. **Transfer Learning**	$f(x) = f_B f_F(x)$
3. **U Shape Split**	$f(x) = f_{CB} f_{SB} f_{SF} f_{CF}(x)$
4. **Key Value Store**	$key(x) = unique\_key(client\_id,data\_id)$ $value_{server}(x) = f_{SF} f_{CF}(key(x))$ $f(x) = f_{CB} f_{SB}(value_{server}(key(x)))$
5. **Model Customization**	$f(x) = f_{Custome\_Block} f_B(value(key(x)))$
6. **Model aggregation**	$f_{Gen}(x) = \Sigma_{\forall\ clients}\ f_{Custome\_Block} f_B(value_{server}(key(x)))$
7. **Personalised Training**	$value_{client\_i}(x) = f_B(value_{server}(x_{client\_i}))$ $f_{Personal\_i}(x)=f_{Personal\_Custome\_Block}(value_{client\_i}(x_{client\_i}))$

ESL Framework:

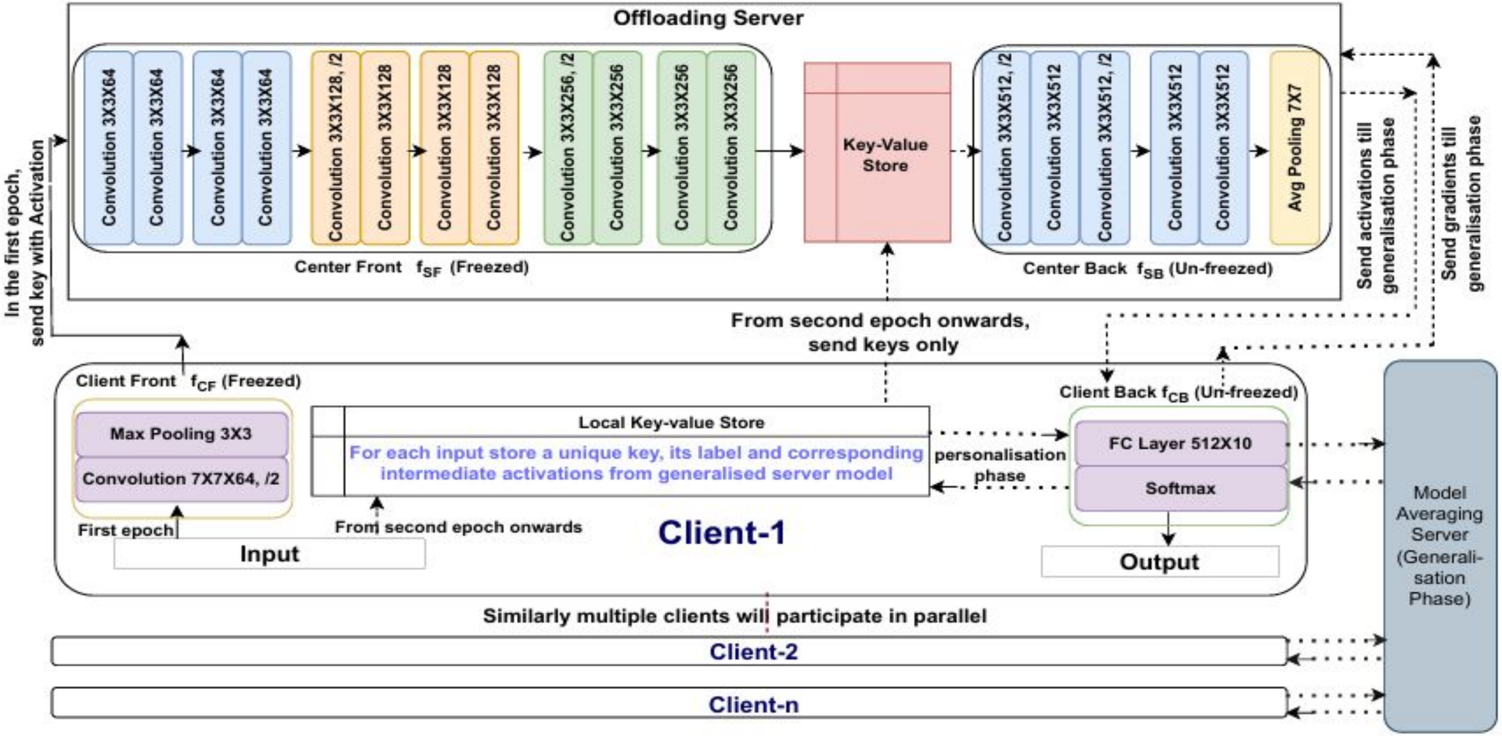


Figure: ESL System Architecture using ResNet18
As an example for splitting the model into the client front, center front, center back, and client back models for training with key-value store.

ESL Generalisation Personalisation Model:

Generalization Phase: Outputs common model working on all data distributions

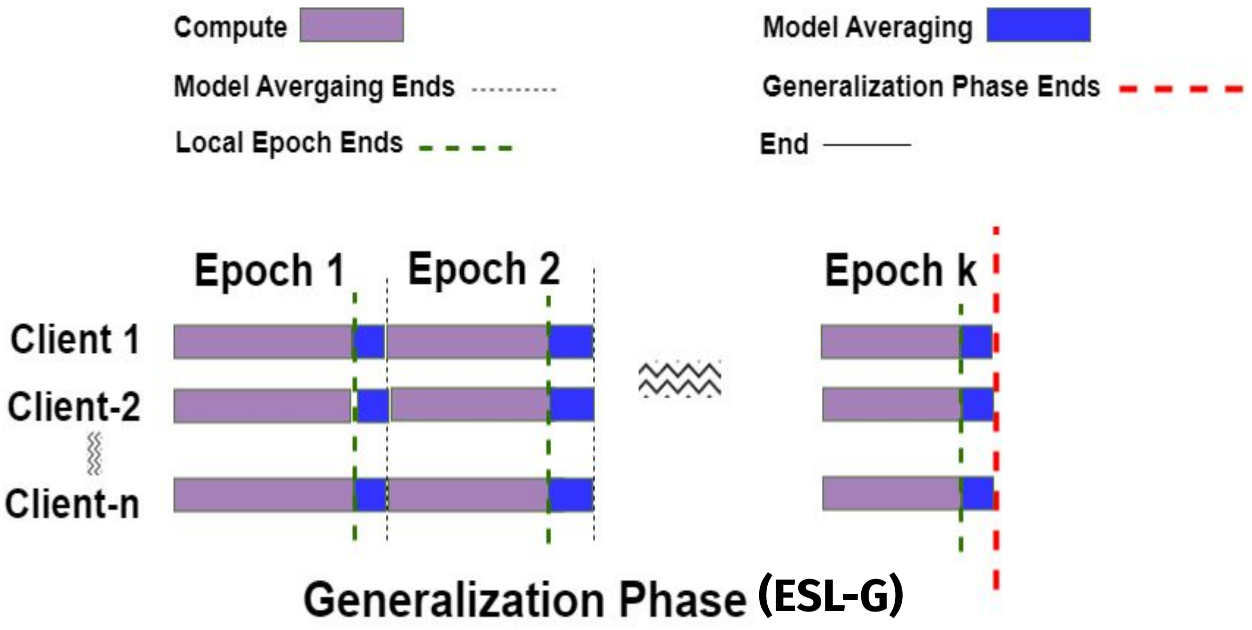


Figure: Training with multiple clients in ESL

ESL Generalisation Personalisation Model:

Generalization Phase: Outputs common model working on all data distributions.
Personalization Phase: Outputs models working best on client's own local data distribution

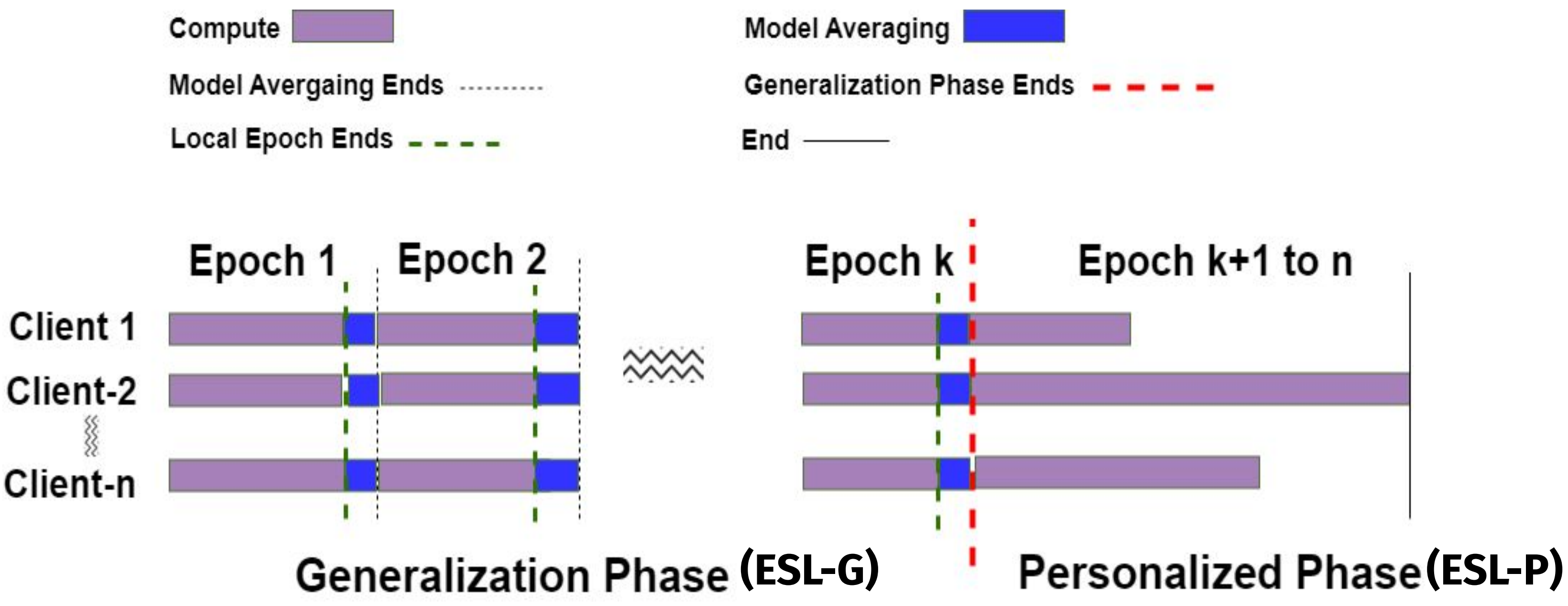


Figure: Training with multiple clients in ESL

Summary:Efficient Split Learning(ESL)

Challenges with Decentralised Training	Key-Value Store	Base Model Customization	Generalization-Personalization Model	Transfer Learning	U-shaped Configuration	Client Model Averaging Server
Statistical Heterogeneity		✓	✓			
Communication Overheads	✓		✓	✓		
Constrained Resources	✓	✓	✓	✓		
Convergence			✓	✓		
Privacy & Security					✓	✓

The main challenges to SL in IoT and representative contribution and technical strategies for addressing them.

Experiment Goal:

We have designed the experiments to answer the following questions:

1. How does varying the number of layers in the client's backside model affect SL performance and the computational workload for resource-constrained clients.

2. How does customizing the model at the client back side affect SL performance and workload distribution between clients and servers?

3. What is the overall performance of ESL compared to other existing algorithms for image classification and 3D image segmentation?

4. Is the key-value store approach effective in ESL, and how much does it mitigate overhead in terms of communication and computation compared to other methods?

5. Will the ESL framework demonstrate effective performance when applied to realistic medical datasets in a federated setting?
    - **Skin Lesion Diagnosis using ESL Framework**
    - **Diabetic Retinopathy Detection using ESL Framework**
    - **Brain 3D Image Segmentation using ESL framework**
    - **Kidney and Tumor Segmentation using ESL framework**

Experimental Setup:

Dataset Used:

Dataset	Number of Classes	Metric Used	Task	Base Model	Pretrained On Dataset
CIFAR-10[19]	10	Accuracy	Image Classification	ResNet-18[26]	ImageNet[30]
FMNIST[20]	10				
DR[21][22]	3	Balanced Accuracy[27]	Medical-Image Classification	ResNet-18	ImageNet
ISIC-2019[23]	8				
KITS-19[24]	3	DICE-Score[28]	3D-Image Segmentation	nnUNet[31]	MSD Pancreas[33]
IXI-Tiny[25]	2			3D UNet[32]	MSD Spleen[34]

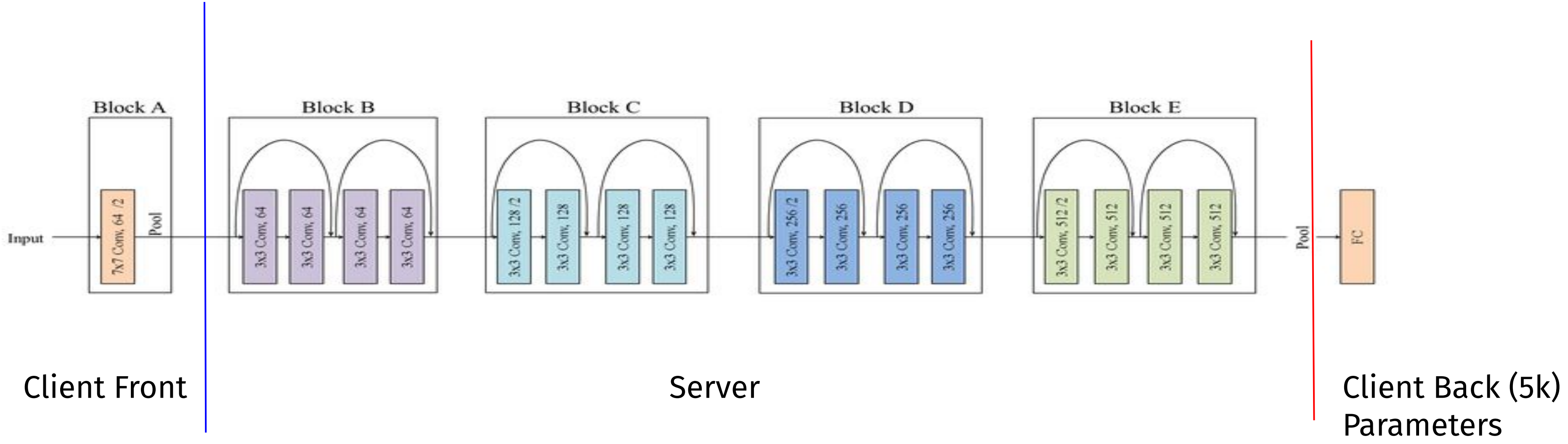
Framework Used: We have created our framework using PyTorch.

Experiments Conducted on: Workstation equipped with an Intel(R) Xeon(R) Gold 5218 CPU operating at 2.30GHz, coupled with the NVIDIA RTX A6000 GPU.

Reproducibility: Each experiment is repeated 5 times and average value is reported.

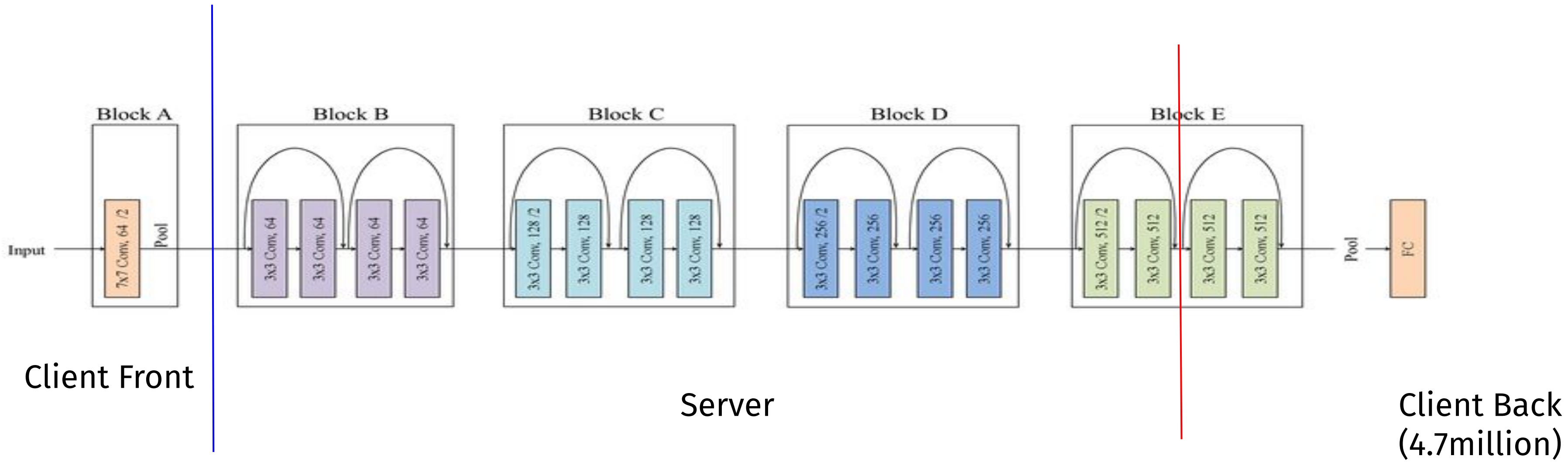
1. Effect of number of layers in client-back side.

Setup: Splits-1



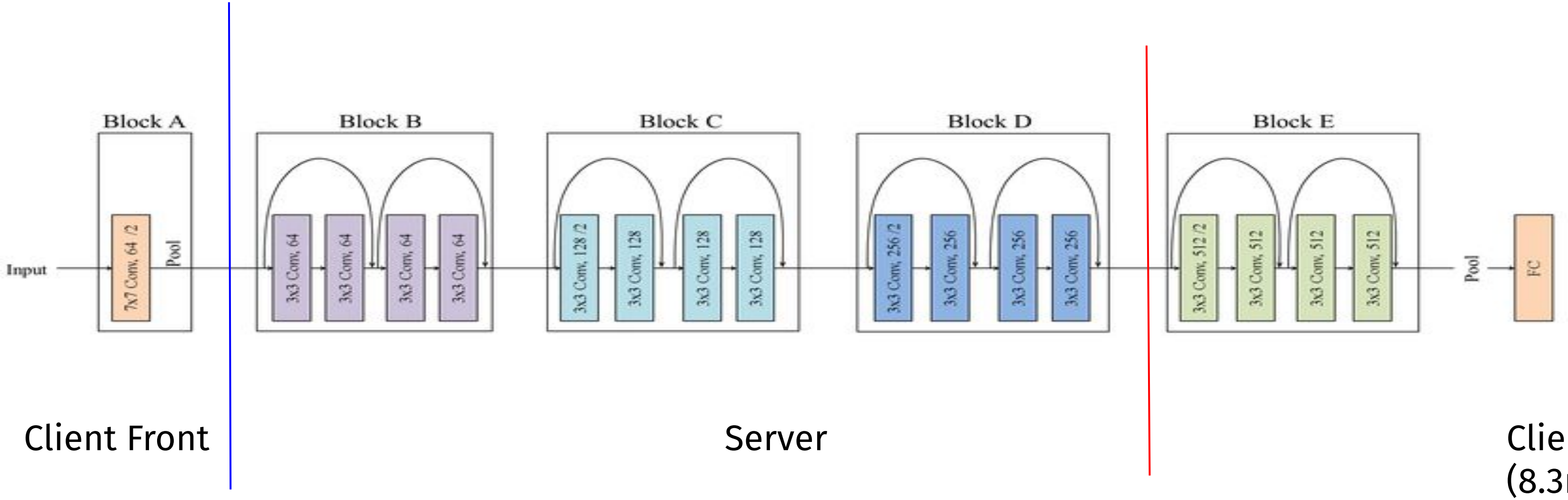
Reference:https://www.researchgate.net/figure/ResNet-with-18-layers-depicted-in-five-blocks_fig4_333787753

Setup: Splits-2



Reference:https://www.researchgate.net/figure/ResNet-with-18-layers-depicted-in-five-blocks_fig4_333787753

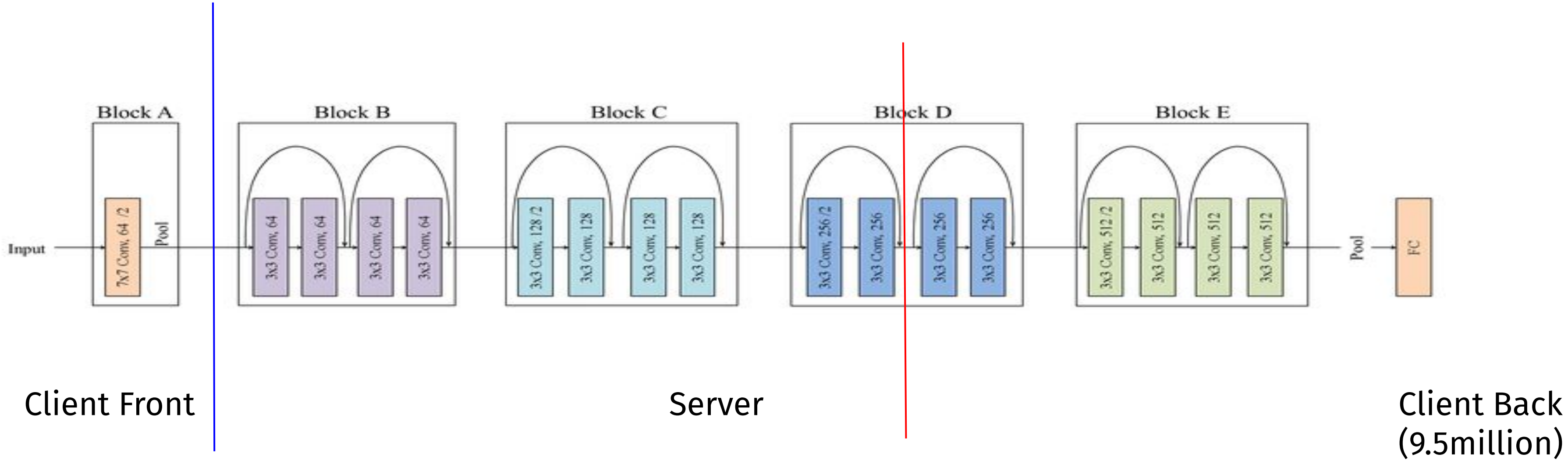
Setup: Splits-3



Client Front | Server | Client Back (8.3million)

Reference:https://www.researchgate.net/figure/ResNet-with-18-layers-depicted-in-five-blocks_fig4_333787753

Setup: Splits-4



Reference:https://www.researchgate.net/figure/ResNet-with-18-layers-depicted-in-five-blocks_fig4_333787753

IID Setting: Different Model Split Results

Model Split	Number of Parameters at Client Back (in million)	E-1 Test Accuracy	E-1 Epoch	E-2 Test Accuracy	E-2 Epoch
small → RN-S1	0.005	88.19	33	74.92	16
RN-S2	4.7	88.31	34	74.94	17
RN-S3	8.3	88.28	36	75.15	14
Large → RN-S4	9.5	88.23	33	75.35	14

Performance metrics (Test Accuracy and Epoch) for different splits in an IID setting: E-1 (500 datapoints) and E-2 (50 datapoints) on CIFAR-10 dataset with 10 clients.

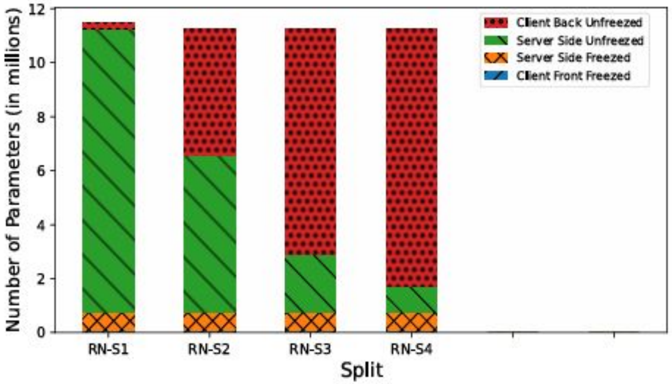


Figure:Statistics of various splits: Resnet-18. The client front freeze is at the bottom of each bar, with 5k parameters and therefore not visible

Observation:

- When we are dealing with iid data across clients, how we split the model and number of trainable layers at client side does not matter.

Conclusion:

- We can offload maximum on server side.

Non-IID: Different Model Splits Results

Model Split	Number of Parameters at Client Back (in million)	Generalised Model Test Acc.	Std	Epoch	Personalised Model Test Acc.	Std	Epoch
RN-S1	0.005	82.3	4.21	29	86.65	2.43	25
RN-S2	4.7	82.2	3.15	25	87.51	2.48	23
RN-S3	8.3	82.1	3.12	26	88.55	2.37	21
RN-S4	9.5	82.4	3.11	27	89.33	1.52	18

small → RN-S1

Large → RN-S4

Performance metrics (Average Test Accuracy, Standard Deviation, and Epoch) for different splits in a Non-IID setting for 500 data-points on CIFAR-10 dataset with 10 clients

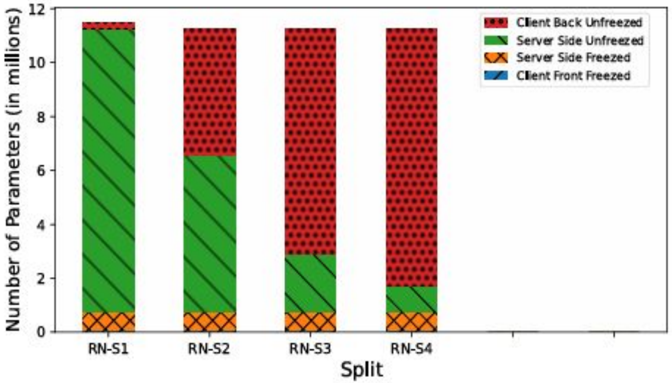


Figure:Statistics of various splits: Resnet-18. The client front freeze is at the bottom of each bar, with 5k parameters and therefore not visible

Observations:

- When we are dealing with non- iid data across clients, how we split the model and number of trainable layers at client side matters.

Conclusion:

- We have to be careful when selecting the split.

Non-IID: Customization of Client Back Model

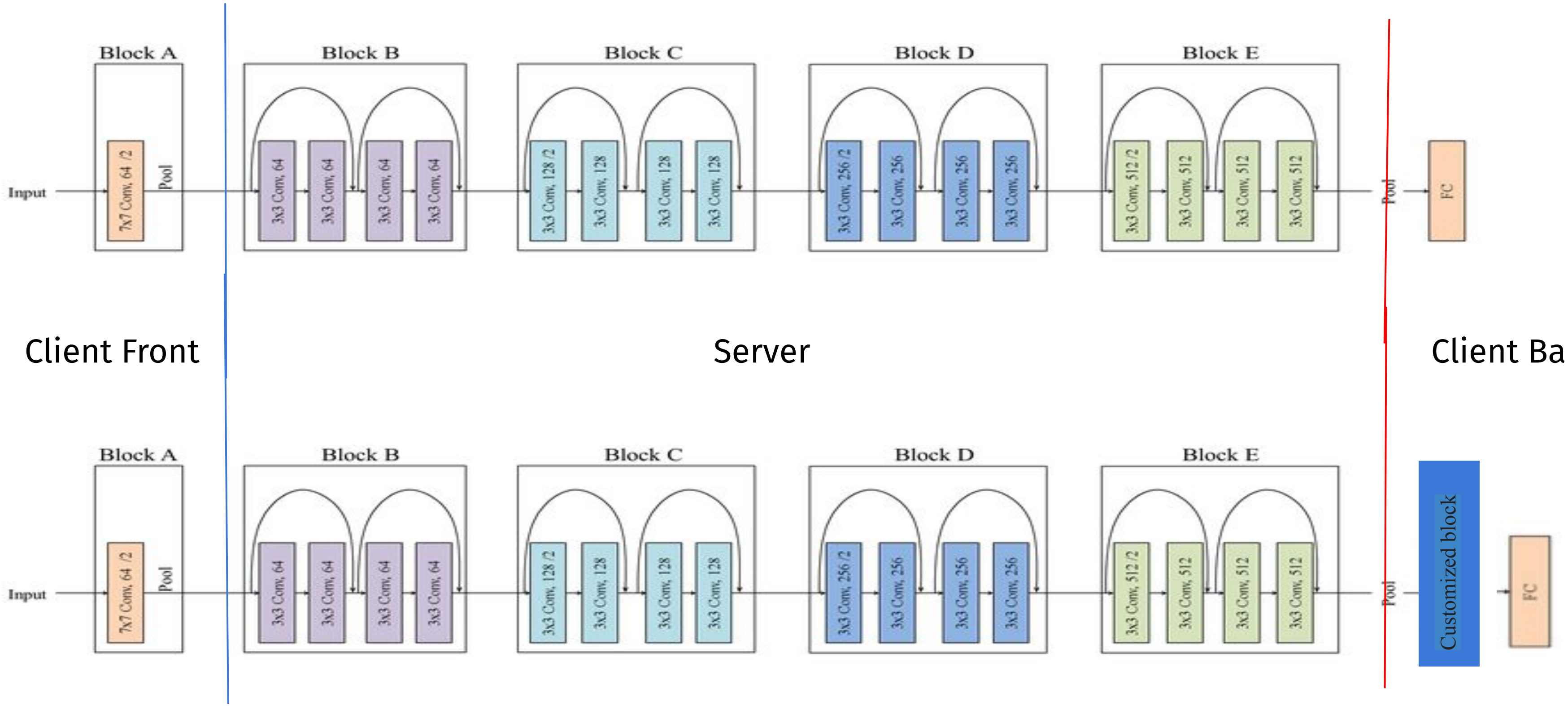
- There is a trade-off between resource requirements and performance in split learning when deciding how to allocate the model's parameters between the client and server.
- Increasing the parameters on the client's side allows for more personalized and fine-grained learning, leading to improved performance and accuracy.

Question:

- Can we customize the model and get the same performance with maximum offload on the server side ?

2. Effect of client back model customization.

Setup: Custom block



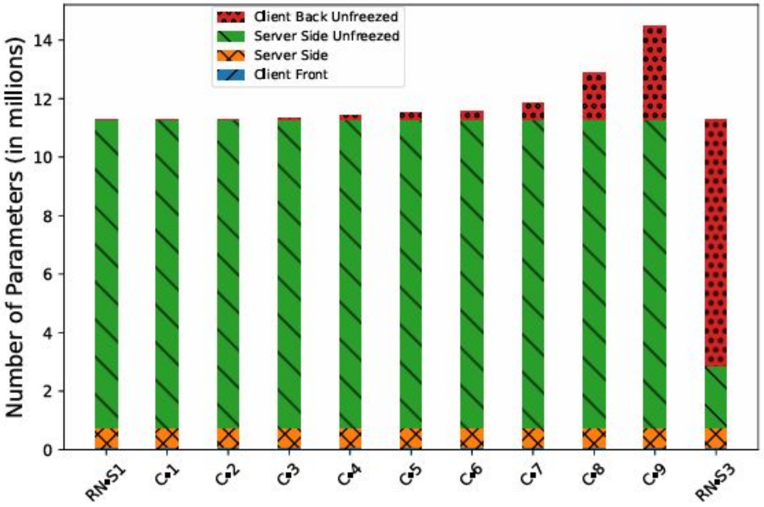
Reference:https://www.researchgate.net/figure/ResNet-with-18-layers-depicted-in-five-blocks_fig4_333787753

Result: Custom block

small → S-1; Large → S-3

Split	Client Back Side Model	No. of parameters (in million)	CIFAR-10 ESL-G	CIFAR-10 ESL-P
S-1	**Linear Layer(512x10)**	**0.005**	**82.39**	**86.65**
C-1	Linear Layer(512x64) Linear Layer(64x10)	0.033	81.65	87.75
C-2	Depthwise-Seperable(64x5x5) Linear Layer(64x10)	0.039	82.9	86.4
C-3	Depthwise-Seperable(128x5x5) Linear Layer(128x64) Linear Layer(64x10)	0.08	83.1	86
C-4	Depthwise-Seperable(256x5x5) Linear Layer(256x64) Linear Layer(64x10)	0.15	83.1	86.9
C-5	Depthwise-Seperable(512x5x5) Linear Layer(512x10)	0.28	<u>84.02</u>	88.36
C-6	Convolution Layer(64x5x5) Linear Layer(64x10)	0.29	82.6	86.8
C-7	Convolution(128x5x5) Linear Layer(128x64) Linear Layer(64x10)	0.59	81.8	87.3
C-8	Convolution(64x3x3) Linear Layer(64x10)	1.6	84.00	86.3
C-9	Convolution(128x3x3) Linear Layer(128x64) Linear Layer(64x10)	3.22	83.62	86.30
S-3	**Last Convolution block and Linear Layer(512x10) of Resnet-18 model**	**8.39**	**82.34**	**<u>88.52</u>**

Comparison of performance metric for different configurations of Client Back Models across various splits on CIFAR-10 dataset with 10 clients. The best result is underlined.



Statistics of various Customized splits: Resnet-18
The client front freeze is at the bottom of each bar, with 5k parameters and therefore not visible.

We've significantly improved ESL-G and achieved ESL-P at a comparable level, all while reducing the number of parameters in the client's backend model by **300X** times..

Effect of client back model customization:

Split	Client Back Side	Number of parameters	CIFAR-10 ESL-G	CIFAR-10 ESL-P	FMNIST ESL-G	FMNIST ESL-P
small → **RN-S1**	**LL(512x10)**	**0.005**	**82.39**	**86.65**	**82.46**	**91.48**
C-1	2LL(512x64X10)	0.033	81.65	87.75	82.02	89.72
C-5	**DS(512x5x5)+LL(512x10)**	0.28	84.02	88.36	84.47	90.53
Large → **RN-S3**	**Last CB+LL(512x10)**	**8.39**	**82.34**	**88.52**	**82.55**	**88.97**

Comparison of performance metric for different configurations of Client Back Models across various splits on various datasets. The best result is underlined.

Observation:

- By carefully selecting and designing the custom block, it is possible to achieve comparable or even improved performance compared to a larger, non-customized model. This allows for a higher offload of parameters to the server side while maintaining good accuracy and performance.

Conclusion: Customization allows for a higher offload of parameters to the server side while maintaining good accuracy and performance.

3. Overall Performance comparison with different algorithms.

Algorithms for comparative study:

- **ESL-G and ESL-P** : Efficient Split Learning Generalised and Personalised Model
- ESL-G° and ESL-P° : Personalized & Fair Split Learning Generalised and Personalised Model
- FedAvg : Federated Learning [1]
- Fedavg_TL : Federated Learning with Transfer Learning
- SL : Vanilla Split learning
- SFLv2 : SplitFedv2 (Split Learning +federated Learning)
- FedProx_TL : Improved FedAvg [2]
- Scaffold_TL : Improved FedAvg [3]
- UN-G and UN-P: ResNet-18 pretrained weights with all the layers unfrozen for generalization and personalization in training.
- NF-G and NF-P: ResNet-18 pretrained weights with adjustments made to the last layer based on the dataset requirements.

Overall Performance Comparison-CIFAR10

Motivation: Standard Dataset

Set Up:

- Total 10 clients, Non-iid data distribution.
- Train(500 samples) and Test(1000 samples) data distribution is same for each client.
- Benefits of personalization and model customization is quantified.

Observation and Conclusion:

- ESL's performance aligns closely with Pooled emphasizes the effectiveness of our method in handling decentralized or Non-IID scenarios.
- ESL demonstrating **30%** and **35%** better performance over FedAvg_TL and SLv2, respectively.

Method	CIFAR-10	FMNIST	ISIC	DR
ESL-G	84.02	84.47	61.13	53.46
ESL-P	88.36	91.53	71.40	61.30
Pooled_TL	**89.12**	**91.88**	**75.50**	**65.4**
ESL-G°	**82.39**	**82.46**	**61.90**	**50.87**
ESL-P°	**86.65**	**91.48**	**68.97**	**56.50**
NF-G	**70.25**	**68.57**	**49.21**	**41.74**
NF-P	**78.97**	**83.65**	**59.82**	**46.51**
UN-G	**85.15**	**85.57**	**55.30**	**46.56**
UN-P	**87.11**	**91.95**	**60.11**	**49.32**
FedAvg	61.01	72.42	48.95	40.21
SL	65.42	76.22	51.95	41.45
SLV2	65.23	76.80	49.78	41.25
FedAvg_TL	67.60	78.71	54.11	42.05
FedProx_TL	67.06	72.51	55.22	42.06
Scaffold_TL	72.01	77.33	54.82	46.42

Performance comparison across different methods on various datasets with Non-IID setting.

Convergence Curve

Observation and Conclusion:

- ESL have faster and smoother convergence than competing techniques.
- When personalization begins, all loss curve slopes increase significantly, indicating accelerated convergence.
- Unfreezing all layers (UN-G and UN-P) has result in a slight accuracy increase for CIFAR-10.
- NF-G and NF-P exhibit lower accuracy compared to ESL-G and ESL-P, indicating the limitations of solely relying on pre-training without fine-tuning.

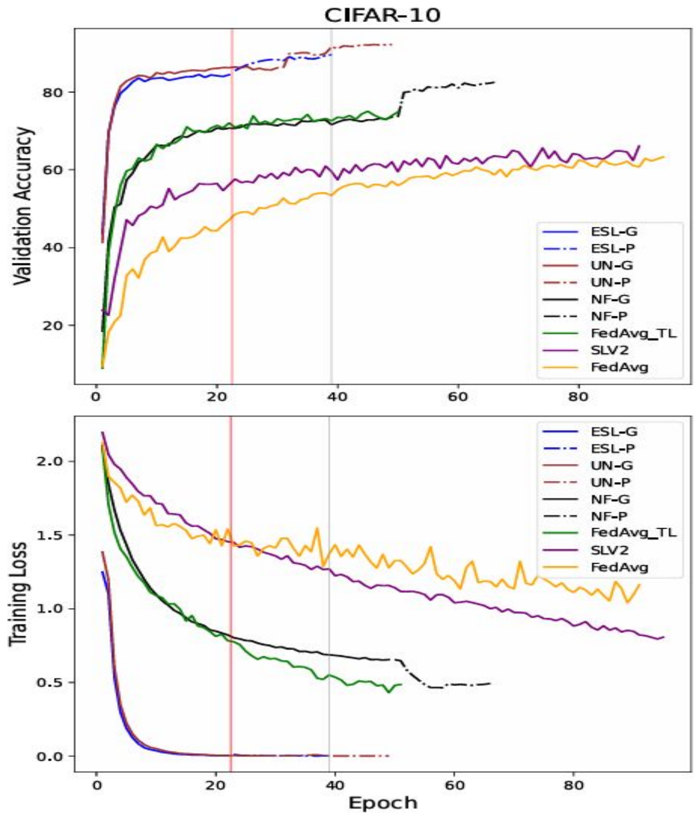


Fig: The convergence curves for training loss, and validation performance for CIFAR-10 using various methodologies.

Overall Performance Comparison-FMNIST

Motivation: Standard Dataset

Set Up:

- Total 10 clients, Non-iid data distribution.
- Train(750 samples) and Test(1000 samples) data distribution is same for each client.
- Benefits of personalization and model customization is quantified.

Observation and Conclusion:

- ESL's performance aligns closely with Pooled emphasizes the effectiveness of our method in handling decentralized or Non-IID scenarios.
- ESL demonstrating **16%** and **19%** better performance over FedAvg_TL and SLv2, respectively.

Method	CIFAR-10	FMNIST	ISIC	DR
ESL-G	84.02	84.47	61.13	53.46
ESL-P	88.36	91.53	71.40	61.30
Pooled_TL	**89.12**	**91.88**	**75.50**	**65.4**
ESL-G°	**82.39**	**82.46**	**61.90**	**50.87**
ESL-P°	**86.65**	**91.48**	**68.97**	**56.50**
NF-G	**70.25**	**68.57**	**49.21**	**41.74**
NF-P	**78.97**	**83.65**	**59.82**	**46.51**
UN-G	**85.15**	**85.57**	**55.30**	**46.56**
UN-P	**87.11**	**91.95**	**60.11**	**49.32**
FedAvg	61.01	72.42	48.95	40.21
SL	65.42	76.22	51.95	41.45
SLV2	65.23	76.80	49.78	41.25
FedAvg_TL	67.60	78.71	54.11	42.05
FedProx_TL	67.06	72.51	55.22	42.06
Scaffold_TL	72.01	77.33	54.82	46.42

Performance comparison across different methods on various datasets with Non-IID setting.

4. Evaluation of key-value store: Communication Reduction

Communication Reduction:

Method	Epochs till convergence	Client Front to Server (Activation)	Key in KV Store	Server to/from Client Back (Activation+gradient)	Model Weights to Server	Total data-transfer till convergence	Reduction compared to other method
ESL-G	22	401, 408, 000	22, 000	45, 056, 000	902, 880	447, 388, 880	–
ESL-P	20	0	0	0	0	0	–
Total ESL	42	401, 408, 000	22, 000	45, 056, 000	902, 880	447, 388, 880	–
ESL-G⁻	22	8, 830, 976, 000	0	45, 056, 000	902, 880	8, 876, 934, 880	–
ESL-P⁻	20	8, 028, 160, 000	0	20, 480, 000	0	8, 048, 640, 000	–
Total ESL⁻	42	16, 859, 136, 000	0	65, 536, 000	902, 880	16, 925, 574, 880	**37.8X**
UN-G	30	24084480000	0	61440000	2288640	24148208640	–
UN-P	18	14450688000	0	36864000	0	14487552000	–
Total UN	48	38535168000	0	98304000	2288640	38635760640	**86.35X**
FL	94	0	0	0	8, 408, 594, 784	8, 408, 594, 784	**18.8X**
FL_TL	50	0	0	0	3, 359, 543, 200	3, 359, 543, 200	**7.5X**
SL	92	72, 253, 440, 000	0	188, 416, 000	3, 775, 680	72, 437, 760, 000	**161.9X**

Communication overhead (in Bytes) of various methods on CIFAR-10 dataset with Non-IID setting per client.
ESL− is without Key-Value Store version of ESL

Observations: Upto 7.5X and 161.9X reduction in communication overhead per Client as compared to FL_TL and SL, during training.

Conclusion: The key-value store approach is even better with personalisation.

Comparison: ESL Vs Sparsification Technique

Paper: “Reducing Communication for Split Learning by Randomized Top-k Sparsification[11]”.

Dataset Used: CIFAR-100[36]

Model: ResNet-18

Observation:

- A 13% better accuracy with reduced communication overhead during the training phase.

Method	Accuracy (%)	Compressed Size
TOP-K Sparsification	66.01	12.5
ESL-P	**75.1**	**1.21**

Comparison of Accuracy and Compressed Size of ESL with [11]. We assume the original communication size(non-compression case) to be 100.

5. Evaluation of key-value store: Computation Reduction

Computation Reduction:

Method	Total Epoch till convergence	Client Side Front Layer	Server Side Central Layer	Client Side Back Layer	Total GLOPS till convergence	Reduction compared to other method
ESL-G	22	0.242	0	0.000676	0.243126	–
ESL-P	20	0	0	0.000614	0.000614	–
Total ESL	42	0.242	0	0.001290	0.243740	–
ESL-G^{-}	22	5.333	0	0.000676	5.334585	–
ESL-P^{-}	20	4.849	0	0.000614	4.849623	–
Total ESL^{-}	42	10.18	0	0.001290	10.18421	**41.78X**
Total UN	48	21.82	0	0.001474	34.91433	**143.24X**
FL	94	68.37	960.1	0.002887	1028.469	**4219.5X**
FL_TL	50	12.12	334.9	0.001536	347.0689	**1423.9X**
SL	92	66.92	0	0.002826	66.91914	**274.4X**

Floating-point operations at the client in various methods on CIFAR-10 dataset with Non-IID setting per client. Without the Key-Value Store version of ESL is shown as ESL−.

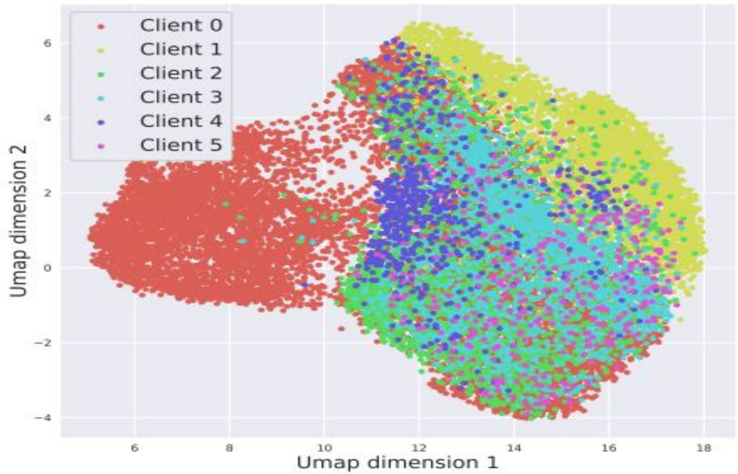
Observations: Upto 4216X reduction in GFLOPs/Work done per Client, during training.
Conclusion: The key-value store approach is even better with personalisation.

6. Skin Lesion Diagnosis

Fed-ISIC2019 Dataset

Motivation:

- ➢ Natural Splits(HealthCare-Skin Disease)
- ➢ High-quality dermoscopy images of skin lesions
- ➢ Limited Clients(Cross-Silo Setting)
- ➢ Real-world problems:
  - heterogeneity between clients
  - high label imbalance
  - Multiclass classification



UMAP of deep network features of the raw images, colored by clients.

Main Challenge:

Model training bias or poor performance on certain clients.

Number	Client	Dataset size	Train	Test
0	Hospital Clínic de Barcelona	12413	9930	2483
1	ViDIR Group, Medical University of Vienna (MoleMax HD)	3954	3163	791
2	ViDIR Group, Medical University of Vienna (DermLite FOTO)	3363	2691	672
3	The skin cancer practice of Cliff Rosendahl	2259	1807	452
4	Memorial Sloan Kettering Cancer Center	819	655	164
5	ViDIR Group, Medical University of Vienna (Heine Dermaphot)	439	351	88

Information for the different clients in Fed-ISIC2019

Image and Table Reference: .https://arxiv.org/pdf/2210.04620.pdf

Fed-ISIC2019 Dataset

1.**Metric Used**: Balanced Accuracy[27]

(The average of the recalls calculated for each class)

It accounts for both the positive and negative outcome classes and doesn’t mislead with imbalanced data.

2. **Loss Function Used**: Weighted Focal Loss[35]

Focal loss applies a modulating term to the cross entropy loss in order to focus learning on hard misclassified examples.

$$FL(p_t) = -\alpha_t(1-p_t)^\gamma \log(p_t)$$

Pt : probability output by our model for the ground-truth class

αt : is the weight of the ground-truth class

γ : is a hyperparameter

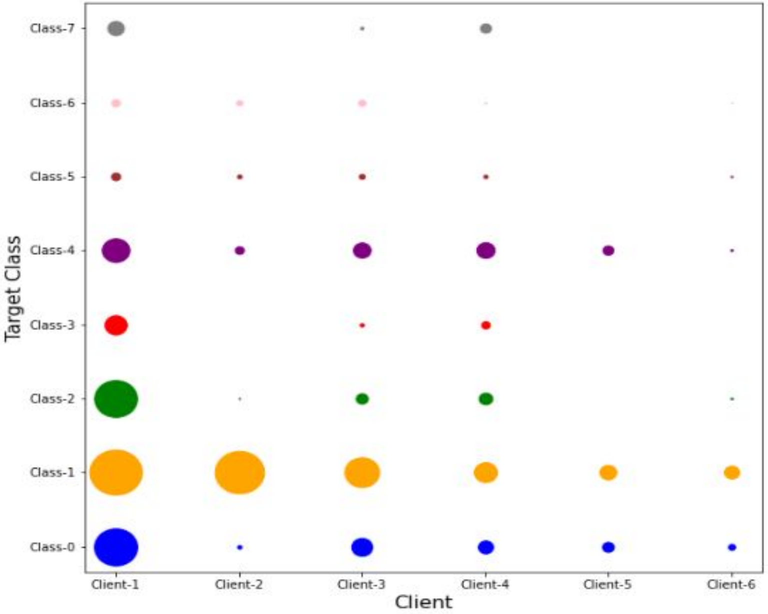


Figure: Data Distribution among clients ,where larger circle indicates more number of training samples for that particular class and client

Fed-ISIC2019 Dataset: ESL Vs Flamby[23]

Observation and Conclusion:

- ESL-P demonstrating **5.7%** better performance over FedAvg.
- ESL-P demonstrating **22%** better performance over FedAvg.
- It shows lower variability(standard deviation).

Method	Balanced Accuracy(%)	Standard Deviation
FedAvg	58.06	13.19
FedProx	55.02	13.21
Scaffold	61.9	13.90
ESL-G	61.1	13.18
ESL-P	**71.4**	**11.62**

Performance comparison across different methods on Fed-ISIC2019 dataset.

Overall Performance Comparison-ISIC2019

Observation and Conclusion:

- ESL's performance aligns closely with Pooled emphasizes the effectiveness of our method in handling decentralized or Non-IID scenarios.

- ESL demonstrating **32%** and **45%** better performance over FedAvg_TL and SLv2, respectively.

Method	CIFAR-10	FMNIST	ISIC	DR
ESL-G	84.02	84.47	61.13	53.46
ESL-P	88.36	91.53	71.40	61.30
Pooled_TL	**89.12**	**91.88**	**75.50**	**65.4**
ESL-G°	**82.39**	**82.46**	**61.90**	**50.87**
ESL-P°	**86.65**	**91.48**	**68.97**	**56.50**
NF-G	**70.25**	**68.57**	**49.21**	**41.74**
NF-P	**78.97**	**83.65**	**59.82**	**46.51**
UN-G	**85.15**	**85.57**	**55.30**	**46.56**
UN-P	**87.11**	**91.95**	**60.11**	**49.32**
FedAvg	61.01	72.42	48.95	40.21
SL	65.42	76.22	51.95	41.45
SLV2	65.23	76.80	49.78	41.25
FedAvg_TL	67.60	78.71	54.11	42.05
FedProx_TL	67.06	72.51	55.22	42.06
Scaffold_TL	72.01	77.33	54.82	46.42

Performance comparison across different methods on various datasets with Non-IID setting.

Convergence Curve

Observation and Conclusion:

- ESL have faster and smoother convergence than competing techniques.
- When personalization begins, all loss curve slopes increase significantly, indicating accelerated convergence.
- Unfreezing all layers (UN-G and UN-P) approach tends to lead to overfitting when applied to datasets from different domains, as seen with ISIC-2019.
- NF-G and NF-P exhibit lower accuracy compared to ESL-G and ESL-P, indicating the limitations of solely relying on pre-training without fine-tuning.

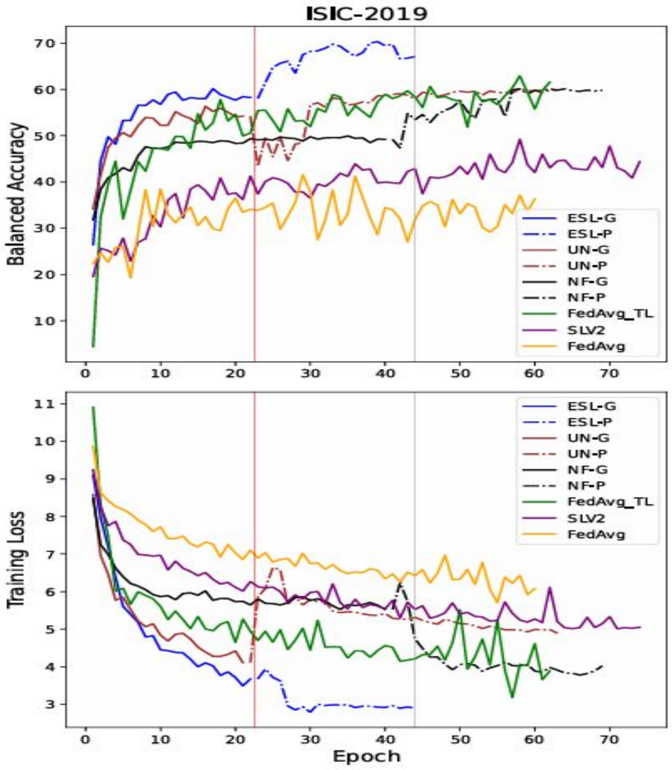


Fig: The convergence curves for training loss, and validation performance for ISIC-2019 using various methodologies.

7. Diabetic Retinopathy Detection

Overall Performance Comparison-Diabetic Retinopathy

Motivation: Distinct geographical locations and healthcare setting

Set Up:

- High-Resolution Retina Images
- Two datasets are used: EyePACS and APTOS.
- Total 10 clients, first 5 clients are provided the APTOS dataset and the next 5 clients are provided the EyePACS dataset.

Observation and Conclusion:

- ESL's performance aligns closely with Pooled emphasizes the effectiveness of our method in handling decentralized or Non-IID scenarios.
- ESL demonstrating **46%** and **49%** better performance over FedAvg_TL and SLv2, respectively.

Method	CIFAR-10	FMNIST	ISIC	DR
ESL-G	84.02	84.47	61.13	53.46
ESL-P	88.36	91.53	71.40	61.30
Pooled_TL	**89.12**	**91.88**	**75.50**	**65.4**
ESL-G°	**82.39**	**82.46**	**61.90**	**50.87**
ESL-P°	**86.65**	**91.48**	**68.97**	**56.50**
NF-G	**70.25**	**68.57**	**49.21**	**41.74**
NF-P	**78.97**	**83.65**	**59.82**	**46.51**
UN-G	**85.15**	**85.57**	**55.30**	**46.56**
UN-P	**87.11**	**91.95**	**60.11**	**49.32**
FedAvg	61.01	72.42	48.95	40.21
SL	65.42	76.22	51.95	41.45
SLV2	65.23	76.80	49.78	41.25
FedAvg_TL	67.60	78.71	54.11	42.05
FedProx_TL	67.06	72.51	55.22	42.06
Scaffold_TL	72.01	77.33	54.82	46.42

Performance comparison across different methods on various datasets with Non-IID setting.

Evaluation of key-value store

Observations:

- From 1.16X to maximum 109X times reduction in communication overhead per Client.
- From 41X to maximum 4216X times reduction in GFLOPs/Work done per Client.

Method	CIFAR-10	FMNIST	ISIC	DR
FL	18.8	8.21	1.94	2.94
FL_TL	7.51	5.21	1.16	1.80
SL	161.9	93.7	109.3	109.7
ESL^{-}	37.8	40.1	40.4	42.5

Reduction in Communication Overhead of ESL concerning different techniques across all datasets.
Without the Key-Value Store version of ESL is shown as ESL−

Conclusion:

- The key-value store approach in ESL provides a practical solution to reduce communication overhead and optimize computations.

Method	CIFAR-10	FMNIST	ISIC	DR
FL	4219.5	2915.9	3499.6	3364.1
FL_TL	1423.9	1565.3	1765.0	1735.8
SL	274.55	164.03	181.97	184.88
ESL^{-}	41.783	46.727	44.751	47.715

Reduction in GFLOPs of ESL concerning different techniques across all datasets.
Without the Key-Value Store version of ESL is shown as ESL−

8. Brain Image Segmentation

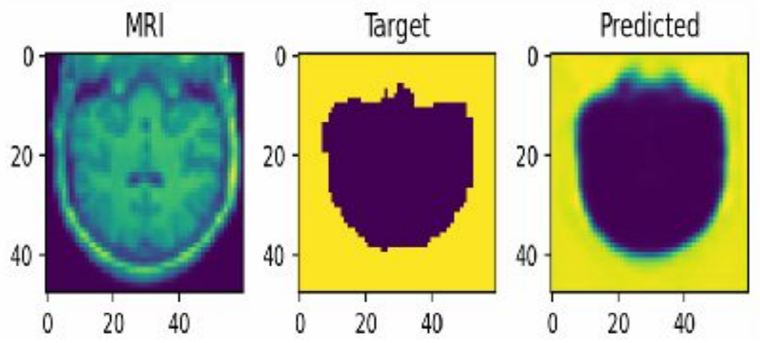
Fed-IXI-Tiny Dataset

Motivation:

- T1-weighted brain MR images along with a set of corresponding brain image segmentations labels, taking the form of binary image masks.
- Natural Splits from three different London hospitals.
- Limited Clients(Cross-Silo Setting)

Hospital Name	Sex	Dataset size	Age
Guys	Female	184	53.23 ± 15.25
	Male	144	51.02 ± 17.26
HH	Female	93	50.28 ± 16.93
	Male	85	44.43 ± 15.67
IOP	Female	44	43.90 ± 18.43
	Male	24	39.57 ± 12.46

Demographics information for Fed-IXI.



Brain MRI images, Target Mask and Predicted Mask

Task: The task is to segment the brain on the volume.

Table Reference: .https://arxiv.org/pdf/2210.04620.pdf

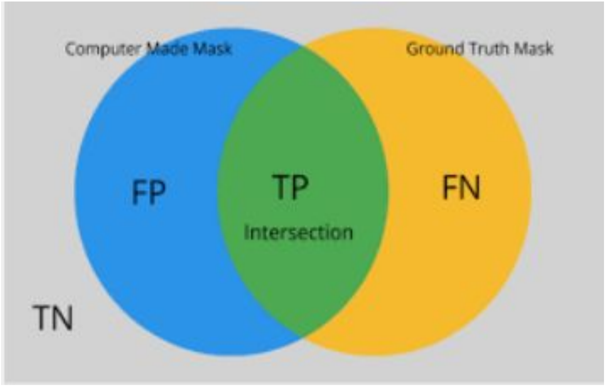
Fed-IXI-Tiny Dataset

1. **Metric Used**: Dice Score[37]
   It measures the overlap between the segmented region and ground truth, providing a comprehensive measure of segmentation accuracy.

2. **Loss Function Used**: Dice Loss[38]
   The model was directly trained for the DICE loss

$$\ell_{DICE} = 1 - S_{DICE} = 1 - \frac{2\mathrm{TP}}{2\mathrm{TP} + \mathrm{FP} + \mathrm{FN} + \epsilon},$$

TP : True positive rate
FP : False positive rate,
FN : False negative rate
ϵ : Ensures numerical stability



Dice Score Visualization

Image Reference:https://fedbiomed.gitlabpages.inria.fr/latest/tutorials/medical/medical-image-segmentation-unet-library/

Main Challenge - Where to split?

- Symmetric architecture with two main parts: the contracting path and the expansion path.
- Splitting at the last layers significantly communication because of larger size activations.
- Splitting in the middle layers significantly boosts client-side computations.
- Deciding where to split such a model is quite challenging.

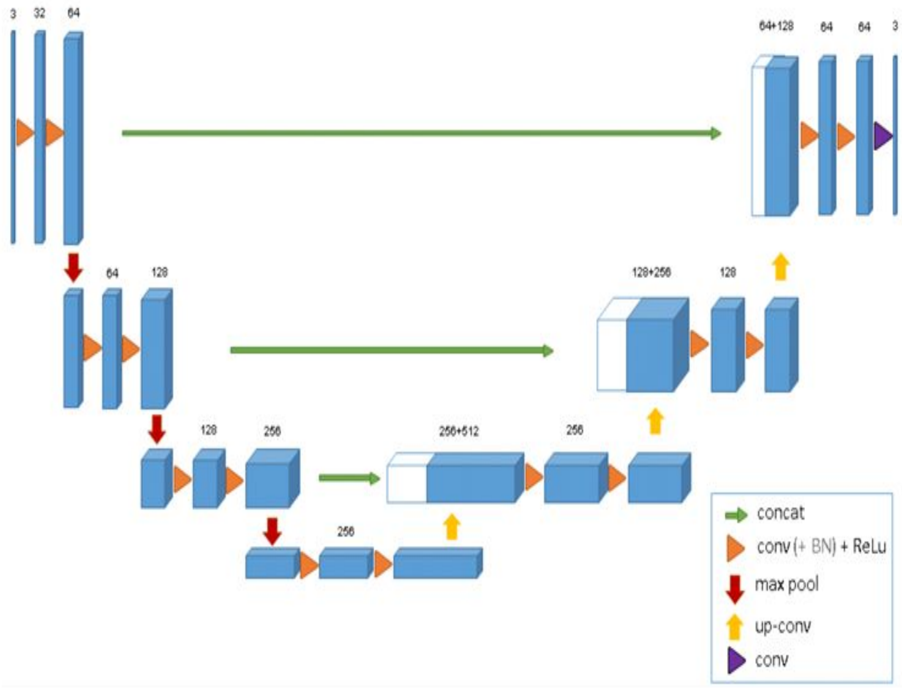


Fig: The 3D U-Net architecture. Blue boxes represent feature maps. The number of channels is denoted above each feature map[32]

Setup Split-1

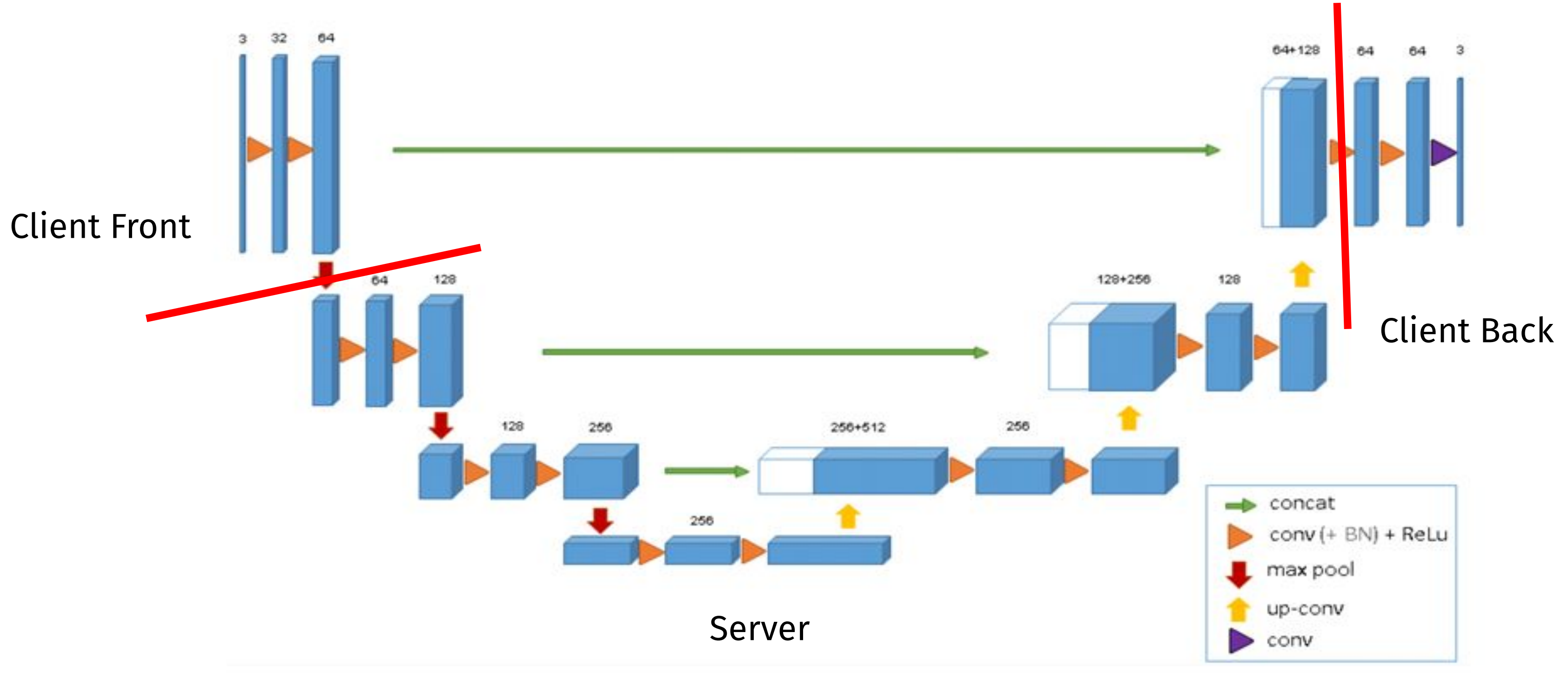


Fig: The 3D U-Net architecture. Blue boxes represent feature maps.
The number of channels is denoted above each feature map [32].

Setup Split-2

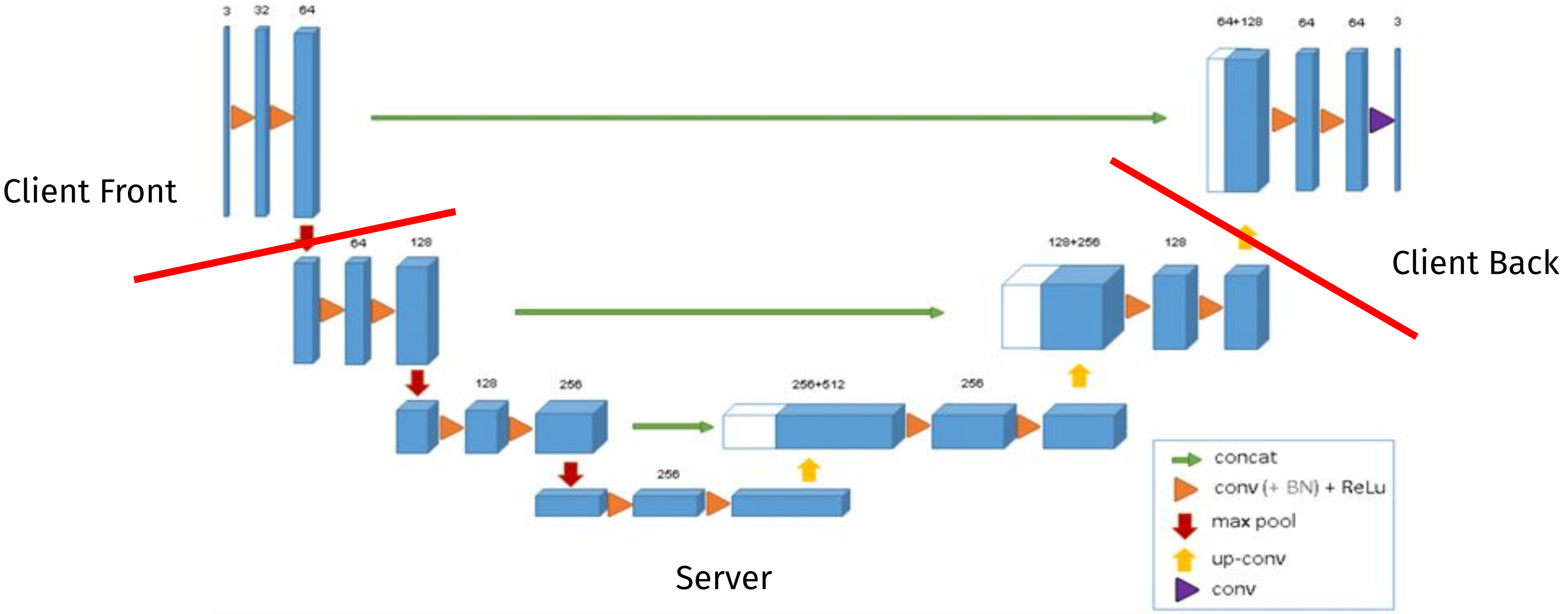


Fig: The 3D U-Net architecture. Blue boxes represent feature maps.
The number of channels is denoted above each feature map[32]

Setup Split-3

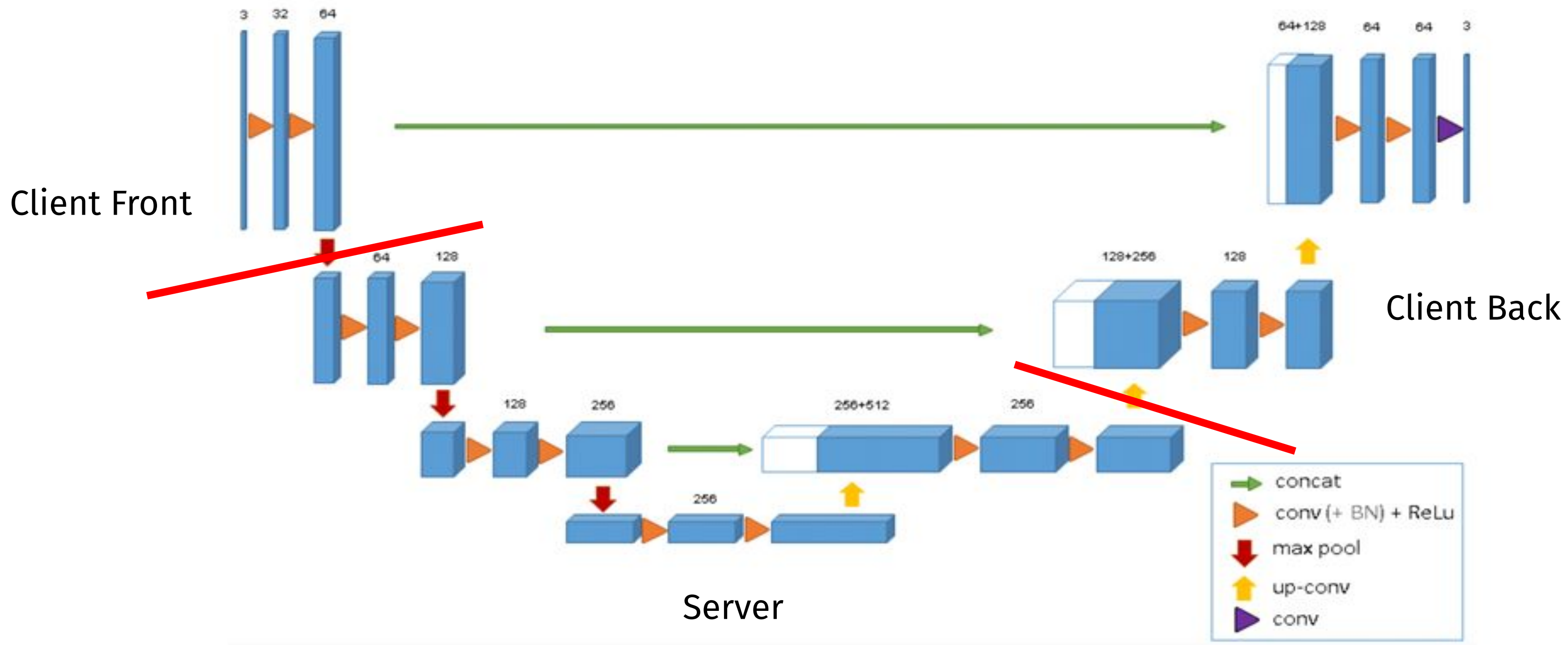


Fig: The 3D U-Net architecture. Blue boxes represent feature maps.
The number of channels is denoted above each feature map [32]

Setup Split-4

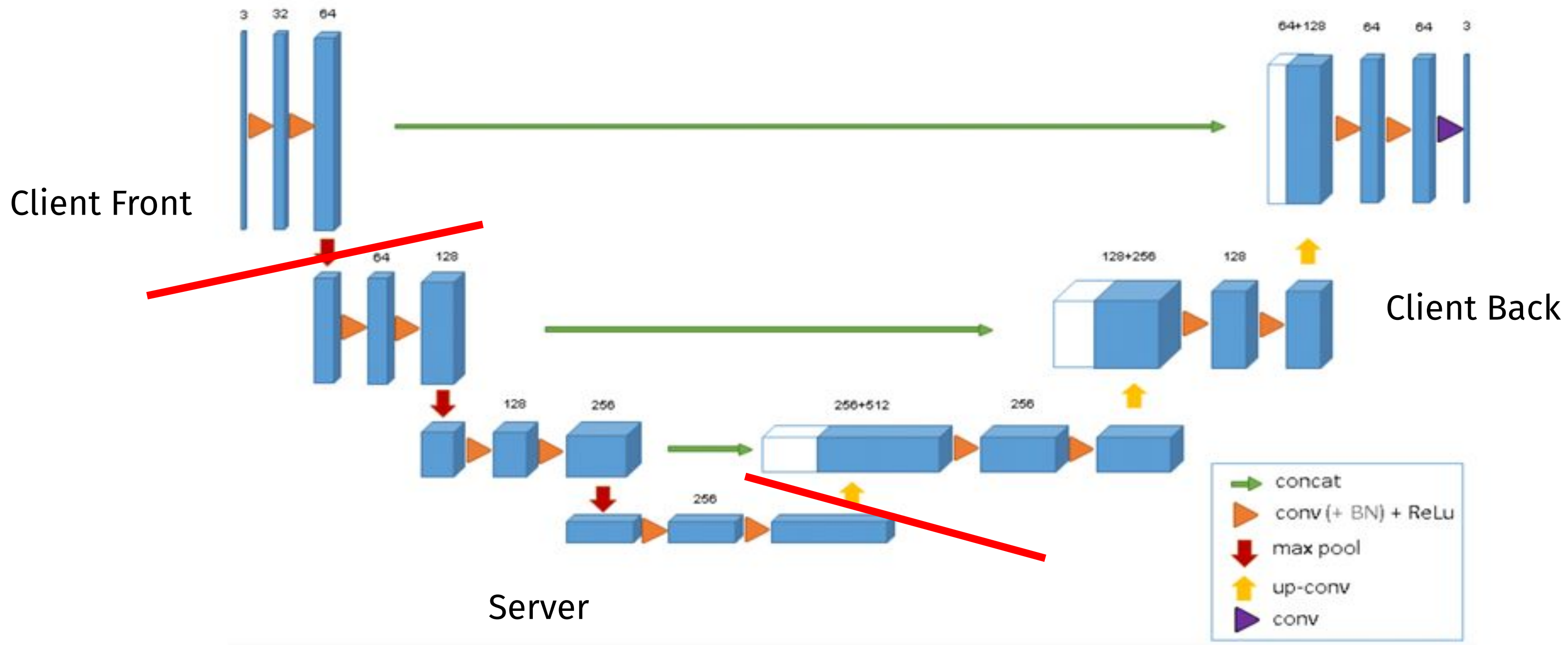


Fig: The 3D U-Net architecture. Blue boxes represent feature maps.
The number of channels is denoted above each feature map [32].

Performance Comparison for Different Splits

Model Split	Total data transfer till convergence	Total GFLOPs till convergence	Test Dice Dice score ± std dev	Epochs
UN-S1	7,941,624,851	**4.60**	0.782±0.020	10
UN-S2	7,943,055,731	16.24	0.792±0.001	10
UN-S3	2,547,619,481	22.02	0.877±0.010	10
UN-S4	2,578,357,761	25.34	**0.916**±0.001	10
UN-S5	**926,034,663**	22.02	0.872±0.001	10
UN-S5*	926,034,663	25.02	0.876±0.001	15

Performance metrics for different splits on IXI-Tiny dataset for 3D UNet model. Only global generalized model results except for UN-S5* where personalization after generalization has been performed

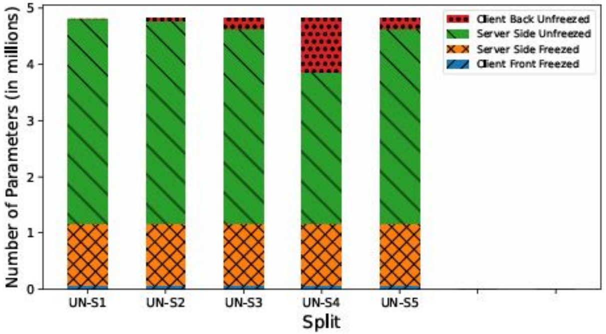


Figure: Statistics of various splits: 3D U-Net. The client front freeze is at the bottom of each bar, with 5k parameters and therefore not visible

Observations: Assigning more layers to the client's backside improved the Dice Score, but it is computationally demanding for the client. Also Personalization does not improve performance for 3D segmentation.

Conclusion: The optimal/best split depends on the resource constraints.

9. Kidney Tumor Segmentation

Fed-KITS-19 Dataset

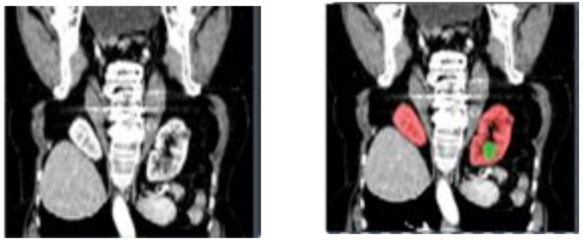
Motivation:

- ➢ The KiTS19 stems from the Kidney Tumor Segmentation Challenge 2019
- ➢ Natural Splits from six different London hospitals, contains CT scans of patients
- ➢ Limited Clients(Cross-Silo Setting)

Local ID Number	Dataset size	Train	Test
0	12	9	3
1	14	11	3
2	12	9	3
3	12	9	3
4	16	12	4
5	30	24	6

Information for the selected clients in Fed-KiTS19

Task: The task consists of both kidney and tumor segmentation, labeled 1 and 2, respectively.



An example of a coronal section of one of the training cases with its ground truth segmentation overlaid (kidney in red, tumor in blue).

Image Reference: .https://arxiv.org/pdf/1912.01054
Table Reference: https://arxiv.org/abs/2210.04620

Setup Split-1

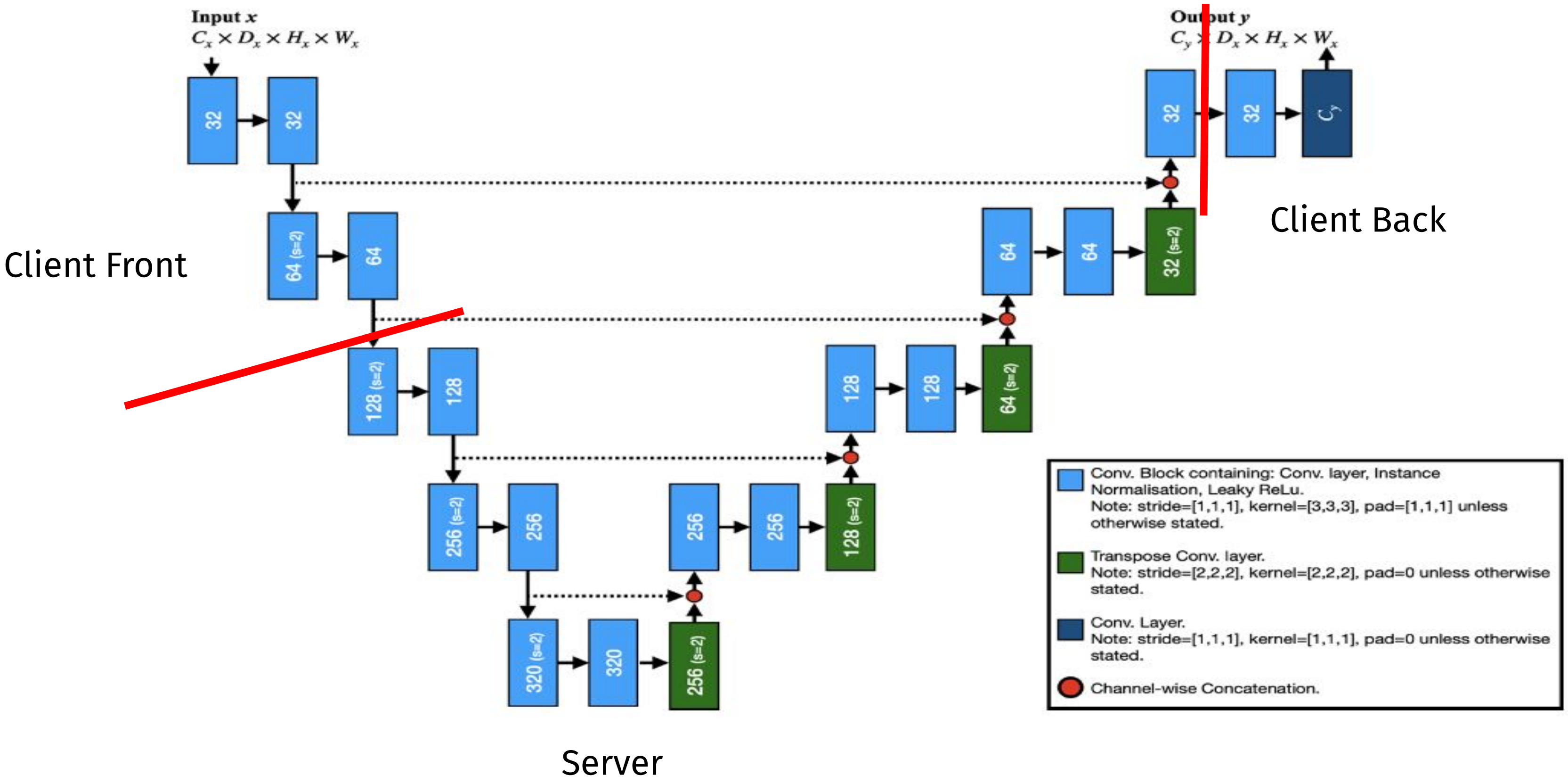


Figure: Baseline nnUNet architecture representation [36]

Setup Split-2

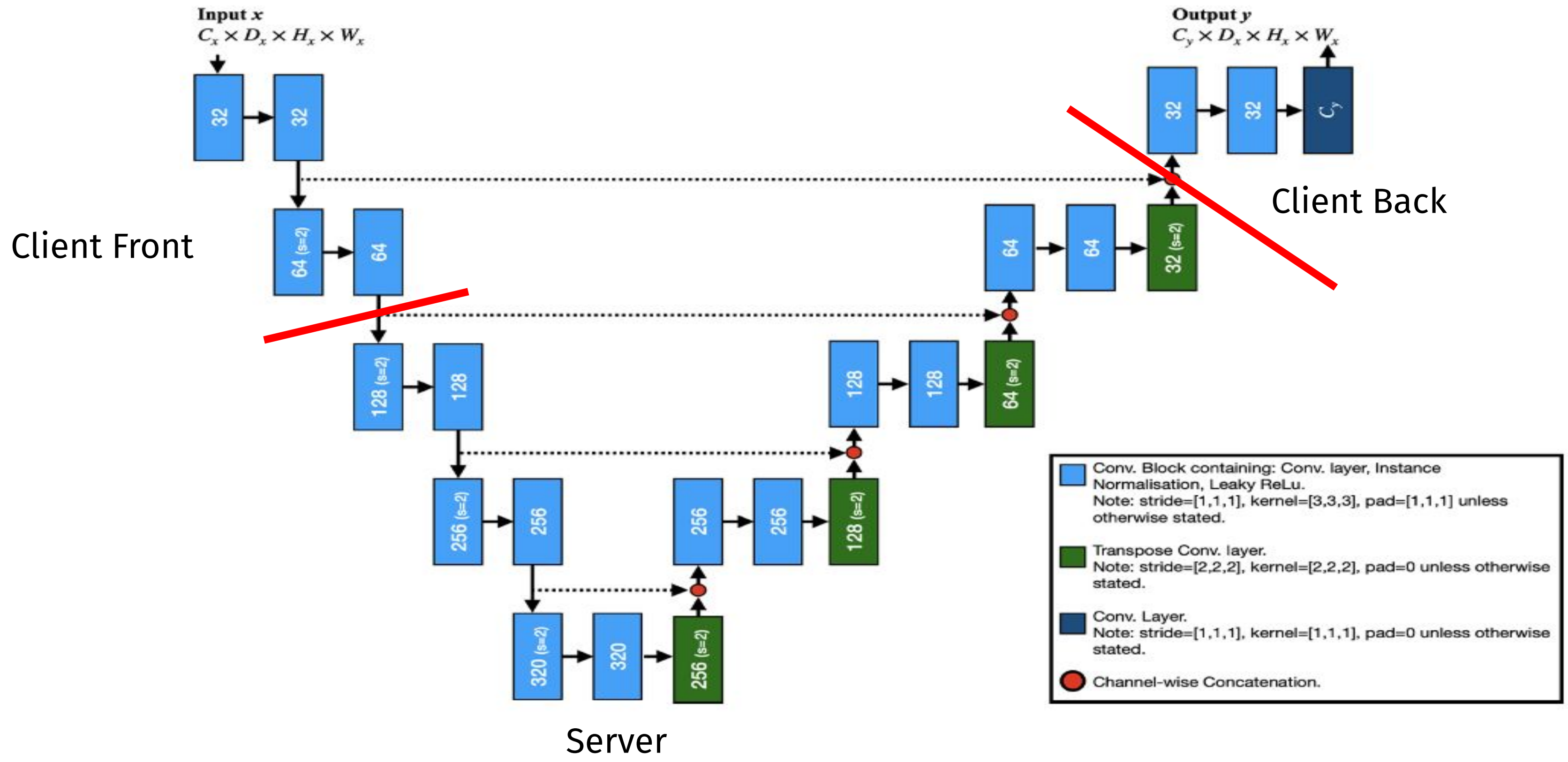


Figure: Baseline nnUNet architecture representation [36]

Setup Split-3

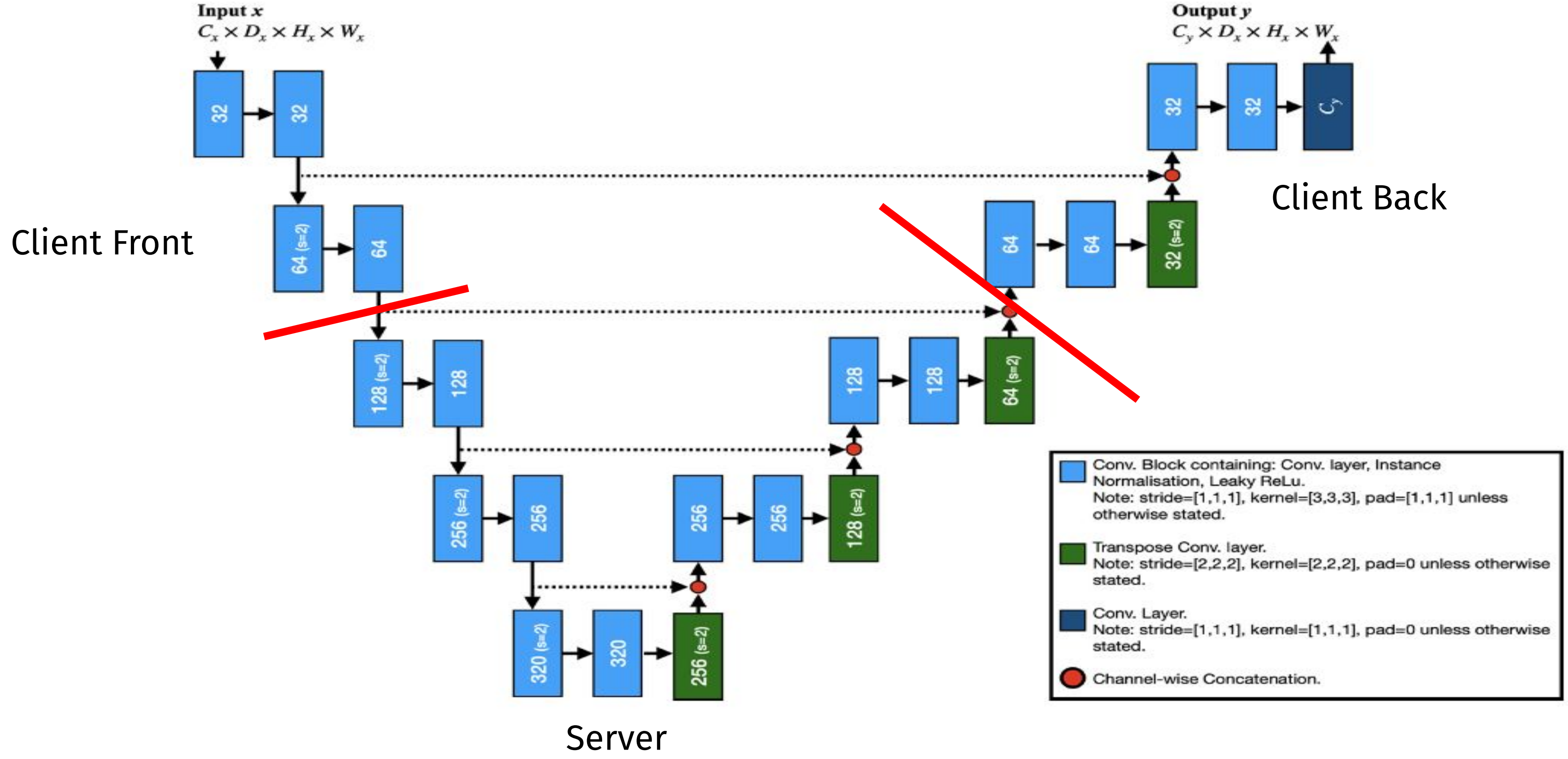


Figure: Baseline nnUNet architecture representation [36]

Setup Split-4

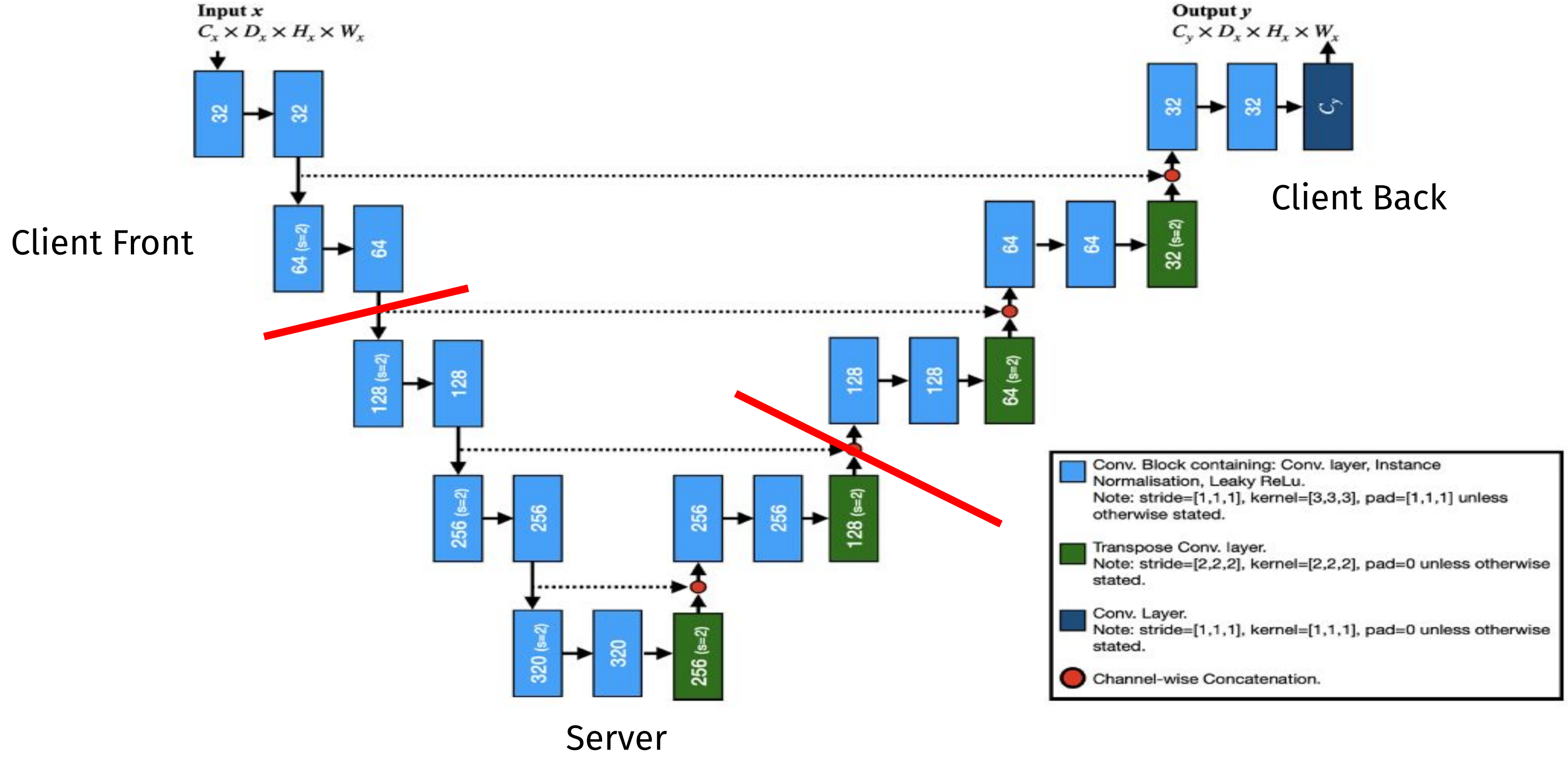


Figure: Baseline nnUNet architecture representation [36]

Setup Split-4

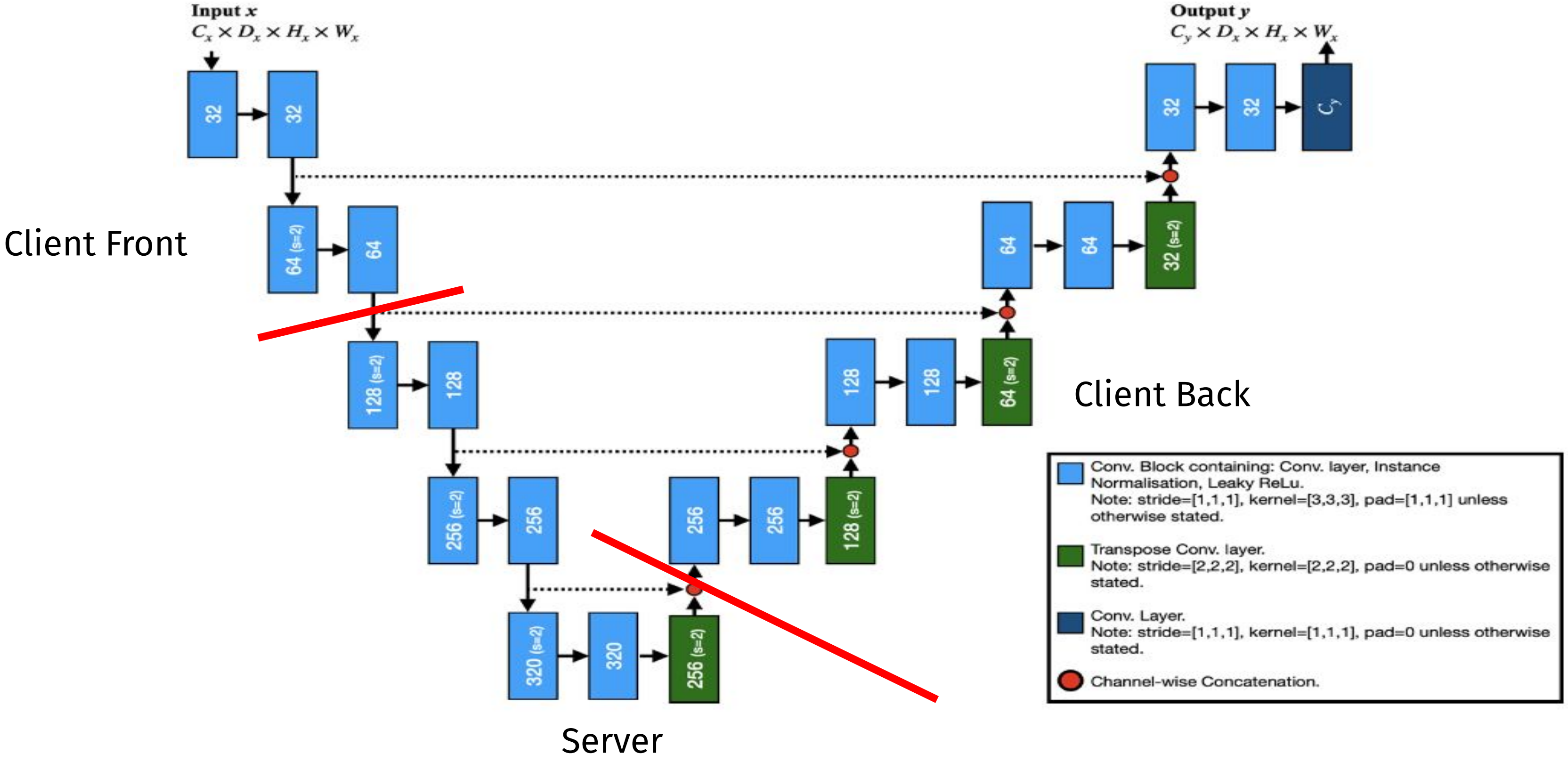


Figure: Baseline nnUNet architecture representation

Performance Comparison for Different Splits

Model Split	Total data transfer till convergence	Total GFLOPs till convergence	Test Dice Dice score ± std dev	Epochs
UN-S1	69,033,901,080	**167,010.24**	0.5655±0.04	30
UN-S2	69,047,239,440	178,456.14	0.6747±0.08	30
UN-S3	27,039,069,990	189,847.72	0.7463±0.06	30
UN-S3*	27,039,069,990	195,662.02	**0.7411**±0.08	30 + 5
UN-S4	14,479,142,910	237,309.65	0.8056±0.03	30
UN-S5	15,113,392,140	239,682.74	0.8048±0.02	30

Performance metrics for different splits on KITS-19 dataset for nnUNet model. Only global generalized model results except for UN-S3* where personalization after generalization has been performed

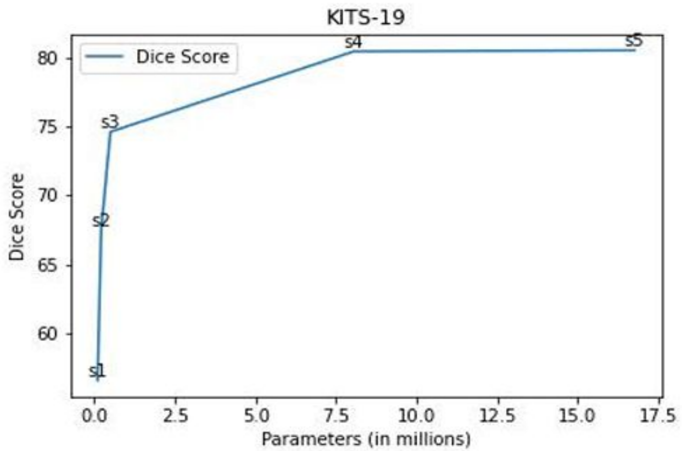


Figure: Parameters at client back side vs. dice Score for KITS-19 dataset for 3D nnU-Net model

Observations: Assigning more layers to the client's backside improved the Dice Score, but it is computationally demanding for the client. Personalization does not improve performance for 3D segmentation.

Conclusion: The optimal/best split depends on the resource constraints.

Overall Performance Comparison-3D Segmentation

Dataset	Technique	ESL	Pooled	FedAvg	FedProx	Scaffold	Communication Reduction	Computation Reduction
KITS-19	ESL(UN-S3)	**0.746**	0.626	0.514	0.561	0.553	2.38	45.43
IXI-Tiny	ESL(UN-S5)	**0.872**	0.954	0.769	0.768	0.769	0.20	2.34

Figure:Performance metrics for different splits on IXI-Tiny dataset for 3D UNet model.

Observation:

- KITS showed a **45%** improvement compared to FedAvg results.
- The IXI-Tiny dataset outperformed several algorithms, with an average improvement of **17%** over FedAvg results.

Conclusion

- Split learning is feasible and effective within federated domains.
- It is well-suited for edge computing applications, enabling on-device model training.
- Optimal performance can be achieved with customized models, reducing computational overhead.
- By careful configuration, we can reduce the communication and computation overhead of split learning without reduction model accuracy.
- Its applicability extends to various domains like NLP, where significant computational resources are typically required.

Future Work

- Support for Heterogeneous datasets and Heterogeneous Devices.
- Dynamic Model Selection and Customization.
- Support for out of order distributions.
- Secure Aggregation Techniques.
- Real-World Deployments.
- New architectures for Split Learning.

References

1. H.Brendan McMahan et. al. “Federated Learning of Deep Networks using Model Aver aging”. CoRR abs/1602.05629 (2016). arXiv: 1602.05629. URL: http://arxiv.org/ abs/1602.05629.
2. Tian Li et. al. Federated Optimization in Heterogeneous Networks. 2020. arXiv: 1812. 06127 [cs.LG]
3. Sai Praneeth Karimireddy et. al. SCAFFOLD: Stochastic Controlled Averaging for Fed erated Learning. 2021. arXiv: 1910.06378 [cs.LG]
4. M. Yashwanth et. al. Federated Learning on Heterogeneous Data via Adaptive Self Distillation. 2023. arXiv: 2305.19600 [cs.LG]
5. Sebastian Caldas, Virginia Smith, and Ameet Talwalkar. “Federated kernelized multi task learning”. Proc. SysML Conf. 2018, pp. 1–3
6. Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, “Federated learning with non-iid data,” 2018. [Online]. Available: https://arxiv.org/abs/1806.00582
7. T. Yu, E. Bagdasaryan, and V. Shmatikov, “Salvaging federated learning by local adaptation,” 2020. [Online]. Available: https: //arxiv.org/abs/2002.04758
8. YusukeKanamoriet. al. “An asynchronous federated learning focusing on updated mod els for decentralized systems with a practical framework”. 2023 IEEE 47th Annual Com puters, Software, and Applications Conference (COMPSAC). 2023, pp. 1147–1154. DOI: 10.1109/COMPSAC57700.2023.00173.
9. Xing Chen, Jingtao Li, and Chaitali Chakrabarti. “Communication and computation re duction for split learning using asynchronous training”. 2021 IEEE Workshop on Signal Processing Systems (SiPS). IEEE. 2021, pp. 76–81.
10. PengchaoHanet.al. “Adaptive Gradient Sparsification for Efficient Federated Learning: An Online Learning Approach”. CoRR abs/2001.04756 (2020). arXiv: 2001.04756. URL: https://arxiv.org/abs/2001.04756. [23] Sebastian U. Stich at. al. “Sparsified SGD with Memory”. CoRR abs/1809.07599 (2018). arXiv: 1809.07599. URL: http://arxiv.org/abs/1809.07599.
11. Fei Zheng et. al. Reducing Communication for Split Learning by Randomized Top-k Sparsification. 2023. arXiv: 2305.18469 [cs.LG]. [25]
12. DiChaiet.al.“FederatedSingularVectorDecomposition”. CoRRabs/2105.08925(2021). arXiv: 2105.08925. URL: https://arxiv.org/abs/2105.08925.
13. Suhail Mohmad Shah and Vincent K. N. Lau. “Model Compression for Communication Efficient Federated Learning”. IEEE Transactions on Neural Networks and Learning Systems 34.9 (2023), pp. 5937–5951. DOI: 10.1109/TNNLS.2021.3131614.
14. Felix Sattler et. al. “Sparse Binary Compression: Towards Distributed Deep Learning with minimal Communication”. CoRR abs/1805.08768 (2018). arXiv: 1805.08768. URL: http://arxiv.org/abs/1805.08768.

References

15. Thijs Vogels et. al. “PowerSGD: Practical Low-Rank Gradient Compression for Dis tributed Optimization”. CoRR abs/1905.13727 (2019). arXiv: 1905.13727. URL: http: / /arxiv.og/abs/1905.13727
16. Mahdi Beitollahi and Ning Lu. “FLAC: Federated Learning with Autoencoder Com pression and Convergence Guarantee”. GLOBECOM 2022- 2022 IEEE Global Com munications Conference. 2022, pp. 4589–4594. DOI: 10.1109/GLOBECOM48099.2022. 10000743
17. Novoa-Paradelaet. al. “Fast deep autoencoder for federated learning”. Pattern Recogni tion 143 (Nov. 2023), p. 109805. ISSN: 0031-3203. DOI: 10.1016/j.patcog.2023. 109805. URL: http://dx.doi.org/10.1016/j.patcog.2023.109805.
18S oeren Becker et. al. Federated Learning for Autoencoder-based Condition Monitoring in the Industrial Internet of Things. 2022. arXiv: 2211.07619 [cs.LG]
19 AlexKrizhevsky. Learning multiple layers of features from tiny images. Tech. rep. 2009
20. Xiaoetal.Fashion-MNIST:aNovelImageDatasetforBenchmarkingMachineLearning Algorithms. 2017.
21. Kaggle.APTOS2019DiabeticRetinopathydataset.2019. URL:https://www.kaggle. com/competitions/aptos2019-blindness-detection/data.
22. Kaggle. Kaggle DR dataset (EyePACS). URL: https://www.kaggle.com/datasets/ mariaherrerot/eyepacspreprocess
23. Ogier du Terrail et. al. “Flamby: Datasets and benchmarks for cross-silo federated learn ing in realistic healthcare settings”. Advances in Neural Information Processing Systems 35 (2022), pp. 5315–5334.
24. Nicholas et. al. Heller. “The state of the art in kidney and kidney tumor segmentation in contrast-enhanced CT imaging: Results of the KiTS19 Challenge”. Medical Image Analysis (2020), p. 101821.
25. xitiny dataset. https://torchio.readthedocs.io/datasets.html#torchio. datasets.ixi.IXITiny.. Accessed: 2022-05-18.
26. Resnet-18. https://www.kaggle.com/datasets/pytorch/resnet18.
27. Margherita Grandini, Enrico Bagli, and Giorgio Visani. “Metrics for Multi-Class Clas sification: an Overview”. ArXiv abs/2008.05756 (2020).
28. LeeRaymondDice.“MeasuresoftheAmountofEcologicAssociationBetweenSpecies”. Ecology 26.3 (July 1945), pp. 297–302. URL: http://www.jstor.org/pss/1932409
29. https://paperswithcode.com/dataset/cifar-100
30. FabianIsenseeet.al. “nnU-Net: aself-configuring method for deeplearning-based biomed ical image segmentation”. Nature Methods 18.2 (Feb. 2021), pp. 203–211. ISSN: 1548 7105. DOI:10.1038/s41
31. JiaDenget.al.“ImageNet:Alarge-scale hierarchical image database”. IEEE Conference on Computer Vision and Pattern Recognition. 2009, pp. 248–255. DOI: 10.1109/CVPR. 2009.5206848. [45]
32. M. Jorge et. al. Cardoso. MONAI Model Zoo. https://monai.io/model-zoo.html. Accessed: 2023-06-16.
33. Michela Antonelli et. al. “The Medical Segmentation Decathlon”. Nature Communica tions 13.1 (July 2022), p. 4128. ISSN: 2041-1723. DOI: 10.1038/s41467-022-30695 9. URL: https://doi.org/10.1038/s41467-022-30695-9.
34. Michela Antonelli et. al. “The Medical Segmentation Decathlon”. Nature Communica tions 13.1 (July 2022), p. 4128. ISSN: 2041-1723. DOI: 10.1038/s41467-022-30695 9. URL: https://doi.org/10.1038/s41467-022-30695-9.
35. Ruoxi Qin et al. “Weighted focal loss: An effective loss function to overcome unbalance problem of chest X-ray14”. IOP Conference Series: Materials Science and Engineering. Vol. 428. 1. IOP Publishing. 2018, p. 012022.
36. https://www.sciencedirect.com/science/article/pii/S0925231223009608?dgcid=rss_sd_all

Thank You