ScaleFL: Resource-Adaptive Federated Learning with Heterogeneous Clients

Fatih Ilhan *(Georgia Tech)*
Gong Su *(IBM T.J. Watson Research)*
Ling Liu *(Georgia Tech)*

THU-PM-376

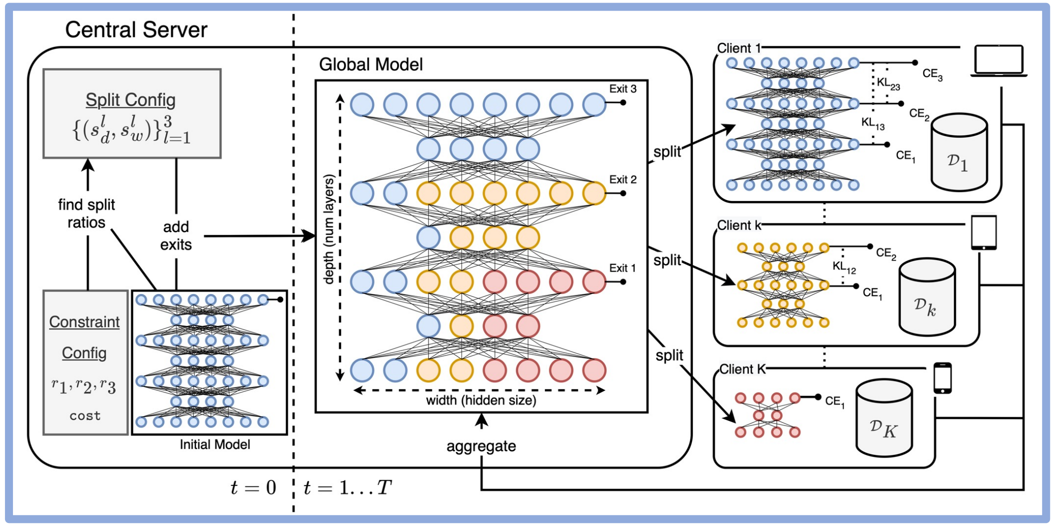
JUNE 18-22, 2023
CVPR
VANCOUVER, CANADA

Georgia Tech

ScaleFL
Summary

ScaleFL is a novel FL framework to handle system heterogeneity by using early exits, which enables

- two-dimensional model downscaling through
  - determining number/location of exits based on client resource statistics
  - computing uniform downscaling ratios based on level constraints
- optimization with self-distillation



On five different image/text classification datasets compared to existing approaches,

- improved global model performance up to 3%
- in local models at lower complexity levels, we obtain 2.5x inference speed and 5x model size reduction with less than 2% performance decrease

Georgia Tech

Problem
System Heterogeneity and Risks

In most studies, all clients are assumed to have similar computational capabilities and be able to finetune/train the high-cost global model

In case of clients with different capabilities, we may have to

- **Omit resource-constraint clients** and hence failing to use their data and bias
- **Switch to a smaller model** to incorporate more clients, hence lower performance

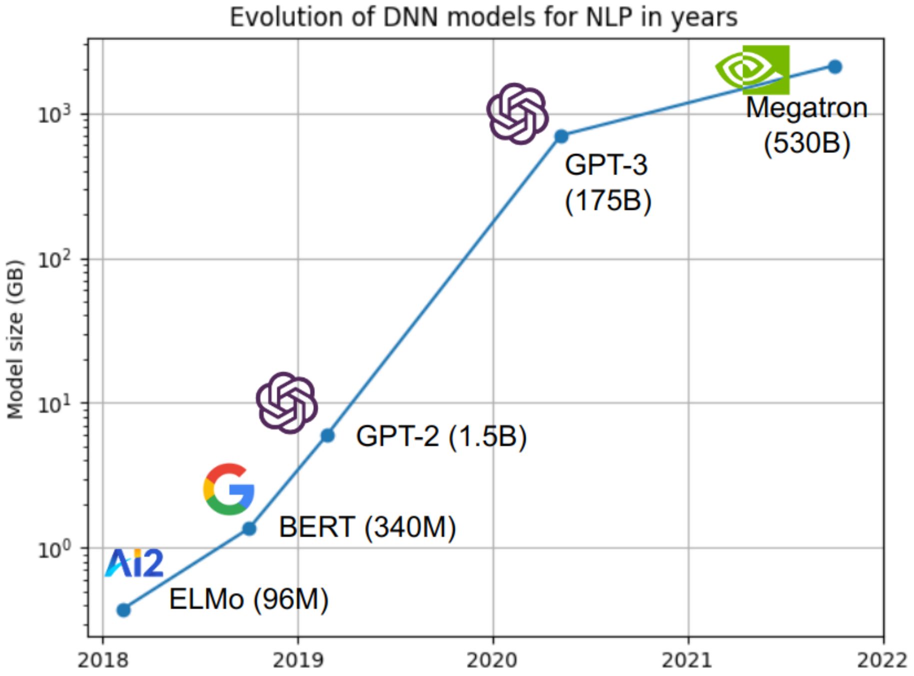


Figure 2: Evolution of DNN models for NLP tasks. Model size increases each year to increase modeling capabilities with deeper and wider model architectures.

Georgia Tech

Methodology
2-D Downscaling Motivation

The dimensions of a deep learning model:

- **width** (# hidden dimensions) enables capturing more **low-level, basic** patterns
- **depth** (# layers) enables capturing **high-level, complex** patterns

Uniformly scaling the dimensions in a model is crucial for efficient model design [3]:

- wide but shallow networks struggle to learn complex patterns
- deep but narrow networks has low capacity for basic patterns

Our approach uniformly downscales the global model into smaller subnetworks using a two-dimensional split approach, which enables efficiently balancing access to basic/complex features.

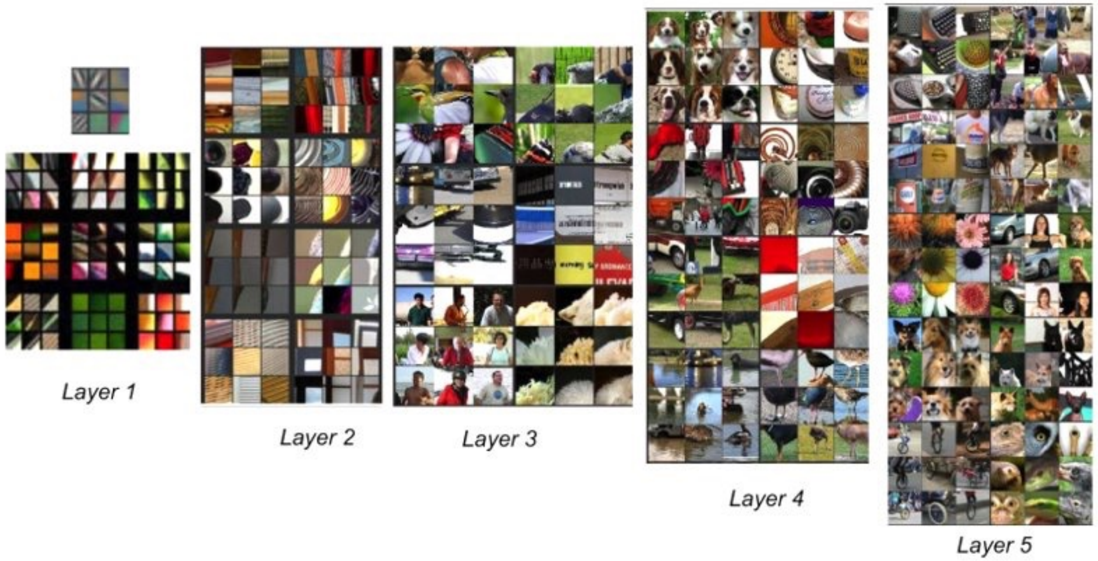


Figure 4: Activation maps of a CNN, early layers learn basic features (lines, colors, words etc.) and deeper layers specialize in complex features (objects, sentences etc.) [4]

Methodology
System Architecture

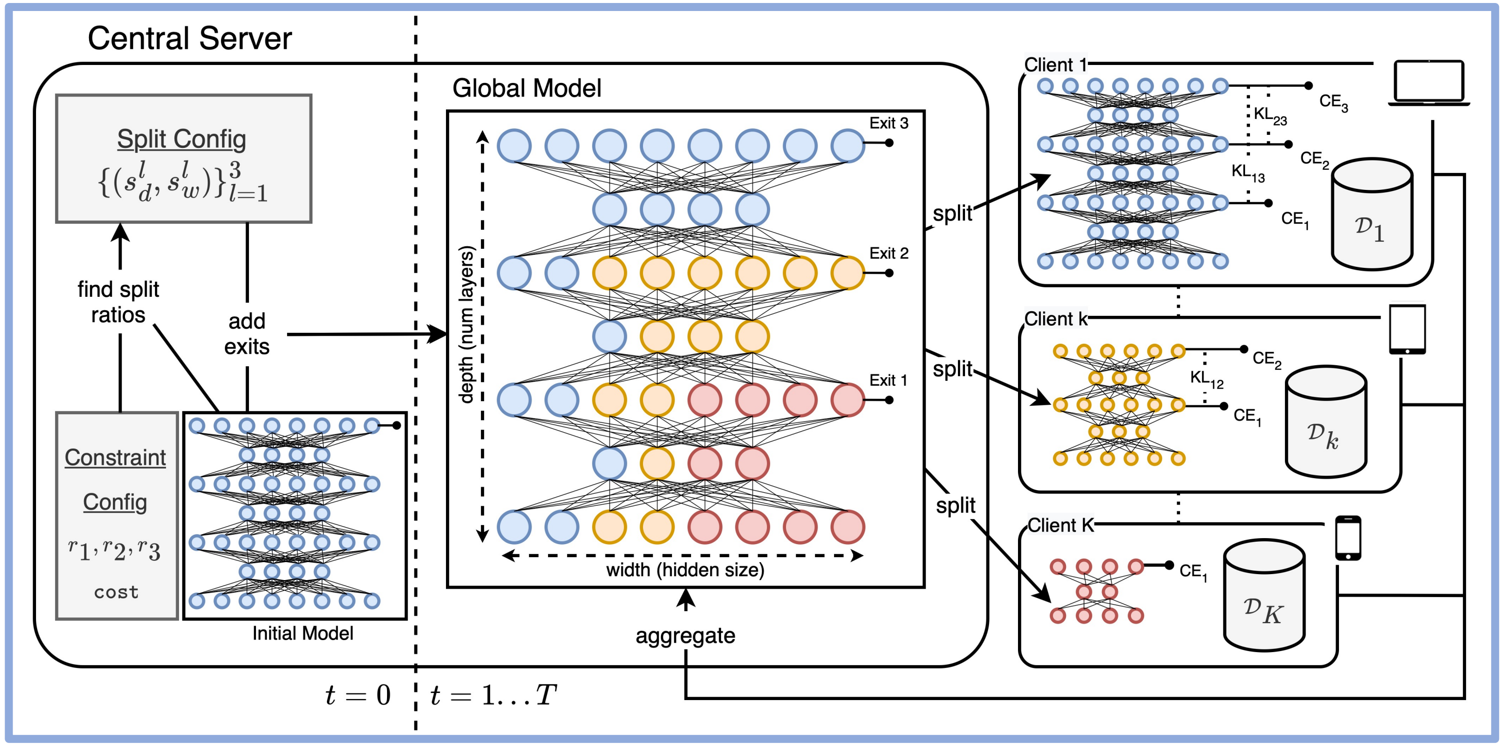


Figure 5: System architecture of ScaleFL

Methodology
Resource-aware Early Exit Injection and Downscaling

We perform cluster analysis over the set of client resources to determine the number of complexity levels (L) and target cost reduction ratios (r_l) at each level l.

Training constraint/cost can be defined based on the application scenario:

- model parameters (model size)
- RAM usage
- number of floating-point operations (#FLOPs)
- latency
- power consumption

Georgia Tech

Methodology
Resource-aware Early Exit Injection and Downscaling

Given a model and constraint definition, we find the most uniform scaling factor that satisfies the target cost reduction ratio through a grid search for each complexity level

$$(s_w^{(l)}, s_d^{(l)}) = \underset{s'_w \in (0,1], s'_d \in (0,1]}{\arg\min} |s'_w - s'_d| \quad \text{such that}$$
$$\left|\frac{\texttt{cost}(\texttt{split}(M; s'_w, s'_d))}{r_l \texttt{cost}(M)} - 1\right| \leq \epsilon_l.$$

- Early exits are injected to Ns_d th layers.
- Each client k is assigned to a complexity level (l_k) based on the available resources such that the cost of the subnetwork (cost(M_l_k)) will not exceed the budget of the client (B_k).

ResNet110	Split Ratios		Cost		
Level	s_d	s_w	#PARAMS	#FLOPS	r_l
4	1.00	1.00	1.73 M	253.1 M	1.000
3	0.88	0.75	0.86 M	138.5 M	0.500
2	0.77	0.70	0.46 M	99.7 M	0.250
1	0.66	0.70	0.21 M	83.4 M	0.125

Table 1: Split ratios and resulting local model statistics (#PARAMS, #FLOPS) for ResNet110 at each level.

Methodology

Split and Aggregate

Splitting along depth

- Early exit classifiers were injected to Ns_d th layer
- Layers after the corresponding early exit are removed

Splitting along width

- Weight matrices at hidden layers are split with ratio s_w
- The index function returns a Boolean matrix to access the upper-left submatrix (first Ds_w elements along each dimension with size D)
- Can be thought as block-wise dropout

Overlapping parts of subnetworks are scaled by the number of contributing local clients during aggregation.

Algorithm 2: `split`

Inputs: Model M with weights θ and N layers
Parameters: Split ratio pair (s_d, s_w)
Outputs: Split model M' with weights θ'

$M' \leftarrow M, \theta' \leftarrow \theta$
Remove all layers in M' after $\lfloor s_d N \rfloor$-th layer
for $\mathbf{W} \in \theta'$ **do**
  $\mathbf{Z} \leftarrow \texttt{index}(\texttt{size}(\mathbf{W}, s_w))$
  $\mathbf{W} \leftarrow \mathbf{W}[\mathbf{Z}]$
end for
return M' with θ'

Algorithm 3: `aggregate`

Inputs: Global model weights θ, set of local model weights $\{\theta^{(k)}\}_{k \in S}$, split ratio pairs $\{(s_d^{(l)}, s_w^{(l)})\}_{l=1}^{L}$
Outputs: Aggregated model weights θ'

$\theta' \leftarrow \theta$
for $\mathbf{W}$ in θ' **do**
  $\widetilde{\mathbf{W}} \leftarrow \texttt{zeros\_like}(\mathbf{W})$
  $\mathbf{C} \leftarrow \texttt{zeros\_like}(\mathbf{W})$
  for client $k \in \mathcal{S}$ **do**
    if $\texttt{key}(\mathbf{W}) \in \texttt{key}(\theta^{(k)})$ **then**
      $\mathbf{Z} \leftarrow \texttt{index}(\texttt{size}(\mathbf{W}), s_w^{(l_k)})$
      $\widetilde{\mathbf{W}}[\mathbf{Z}] \leftarrow \widetilde{\mathbf{W}}[\mathbf{Z}] + \mathbf{W}_k$
      $\mathbf{C}[\mathbf{Z}] \leftarrow \mathbf{C}[\mathbf{Z}] + 1$
    end if
  end for
  $\overline{\mathbf{C}} = \mathbf{C} > 0$
  $\mathbf{W}[\overline{\mathbf{C}}] \leftarrow \widetilde{\mathbf{W}}[\overline{\mathbf{C}}]$
  $\mathbf{W}[\overline{\mathbf{C}}] \leftarrow \mathbf{W}[\overline{\mathbf{C}}] / \mathbf{C}[\overline{\mathbf{C}}]$
end for
return θ'

Methodology

Optimization with Self-Distillation

Knowledge distillation is the method to transfer knowledge from a large (teacher) model to a smaller (student) model [5]

- Iterative training of a student network using the teacher network predictions as soft-labels (over an additional distillation dataset)

Early exits enable performing self-distillation through utilizing the final prediction as soft-label for earlier exit predictions.
KL divergence among predictions is also minimized in the optimization objective during local training iterations:

$$\mathcal{L} = \frac{1}{j(j+1)} \sum_{i=1}^{j} i(\alpha \mathcal{L}_{KL}(\hat{y}_{i,j}, \hat{y}_{j,j}; \tau) + (1-\alpha)\mathcal{L}_{CE}(\hat{y}_{i,j}, y))$$

KL-divergence between the ith and the last exit of level-j model

Cross-entropy loss at the ith exit of level-j model

$$\mathcal{L}_{KL}(\hat{\boldsymbol{y}}_s, \hat{\boldsymbol{y}}_t; \tau) = \text{sum}(\sigma(\hat{\boldsymbol{y}}_t/\tau) \log \frac{\sigma(\hat{\boldsymbol{y}}_t/\tau)}{\sigma(\hat{\boldsymbol{y}}_s/\tau)})\tau^2$$

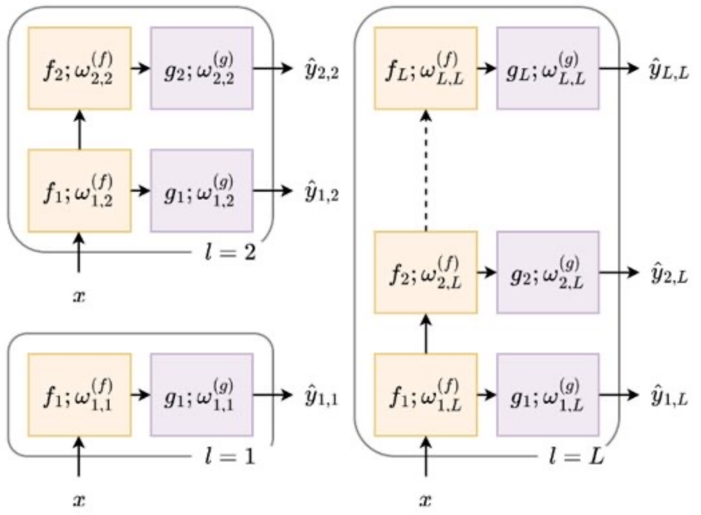


Figure 6: Subnetwork structure. For the level j local model M_j, f_i is the ith core subnetwork with weights $w_{i,j}^{(f)}$. Likewise, g_i is the ith exit classifier subnetwork with weights $w_{i,j}^{(g)}$. $\hat{y}_{i,j}$ is the output at the ith exit of the model at level j.

Since distillation is performed within the network, we don't need any additional distillation dataset or perform distillation operations at the central server. Therefore, there is **no additional communication/computation cost due to self-distillation** neither in clients or central server.

Related Work

[1] FedAvg
[6] HeteroFL
[7] FedDF
[8] FedProx
[9] SplitFed
[10] FedMD
[11] FL with Compression

Existing Approaches differ in the following eight aspects:

Applicability for
- Computation Constraints
- Storage Constraints
- Communication Constraints

Requirement of
- Additional Training on Shared Data
- Additional Training on Server / Clients
- Sharing intermediate layer output

Other Properties:
- Distillation for client-server integration
- Capable of Adaptive Inference

Related Work
Qualitative Comparisons

	Applicable Constraint Type			NO requirement of				
Method	Computation constraints	Storage constraints	Communication constraints	Additional training on shared data	Additional training on server/clients	Sharing intermediate layer output	Distillation	Adaptive inference
FedAVG [1]	✗	✗	✗	✓	✗	✓	✗	✗
HeteroFL [6]	✓	✓	✓	✓	✓	✓	✗	✗
FedDF [7]	✓	✓	✓	✗	✗	✓	✓	✗
FedProx [8]	✓	✗	✗	✓	✓	✓	✗	✗
SplitFed [9]	✓	✓	✗	✓	✗	✗	✗	✗
FedMD [10]	✓	✓	✓	✗	✗	✓	✓	✗
Compression [11]	✗	✗	✓	✓	✓	✓	✗	✗
ScaleFL	✓	✓	✓	✓	✓	✓	✓	✓

Georgia Tech

Experiments
Datasets and Setup Details

Dataset	Train size	Test size	Resolution	# Classes
CIFAR-10	50K	10K	32	10
CIFAR-100	50K	10K	32	100
ImageNet	1.2M	150K	224	1000
SST-2	67K	872	-	2
AgNews	120K	7.6K	-	4

System Topology

- 100 clients with 10% availability at each round
- Four complexity levels with target cost reduction ratios of 12.5%, 25%, 50%, 100% in terms of #PARAMs
- Client level distribution is uniform (25% each level)

Data Heterogeneity

- Dirichlet distribution with varying concentration parameters to control nonIID data simulation (label distribution skew)

Georgia Tech

Experiments
Model and Implementation Details

Baselines:

- *FedAVG:* level-1 subnetwork is trained using federated averaging algorithm
- *Decoupled:* one model for each complexity level is trained in a decoupled way

Existing Methods:

- *HeteroFL:* employs vertical model splitting along width [6]
- *FedDF:* uses ensemble distillation on central server over an additional dataset after each round [7]

Models:

- *ResNet110* on CIFAR10/100 experiments
- *MsdNet24* on CIFAR10/100 experiments
- *EfficientNetB4* on ImageNet experiments
- *BERT* on SST2 and AgNews experiments (pr etrained model is finetuned in federated setting)

Experiments
Results – Image Classification

Resnet110	CIFAR-10				CIFAR-100			
	$\alpha = 100$		$\alpha = 1$		$\alpha = 100$		$\alpha = 1$	
	local	global	local	global	local	global	local	global
FedAVG	81.46	81.46	77.72	77.72	44.26	44.26	42.75	42.75
Decoupled	77.16	77.16	74.83	74.83	36.60	36.60	35.78	35.78
HeteroFL	82.93	84.35	77.60	79.91	44.66	47.12	42.97	42.95
FedDF	83.35	84.44	77.08	78.57	43.50	46.99	42.29	44.50
ScaleFL (Ours)	**84.49**	**85.53**	**79.61**	**80.83**	**46.63**	**49.94**	**43.52**	**44.95**

MSDNet24	CIFAR-10				CIFAR-100			
	$\alpha = 100$		$\alpha = 1$		$\alpha = 100$		$\alpha = 1$	
	local	global	local	global	local	global	local	global
FedAVG	82.69	82.69	75.28	75.28	46.44	46.44	41.44	41.44
HeteroFL	81.54	83.02	75.77	76.74	44.65	47.77	42.32	43.00
ScaleFL (Ours)	**84.61**	**84.77**	**77.81**	**78.69**	**49.19**	**50.25**	**46.25**	**46.12**

EfficientNetB4	ImageNet	
	$\alpha = 100$	
	local	global
FedAVG	45.00	45.00
Decoupled	-	-
HeteroFL	43.68	46.61
FedDF	-	-
ScaleFL (Ours)	**46.63**	**48.95**

Experiments
Results – Text Classification

BERT	SST-2				AG News			
	$\alpha = 100$		$\alpha = 1$		$\alpha = 100$		$\alpha = 1$	
	local	global	local	global	local	global	local	global
FedAVG	79.94	79.94	70.01	70.01	85.14	85.14	81.10	81.10
HeteroFL	76.02	88.83	76.21	82.86	89.92	91.51	88.85	90.85
FedDF	77.67	**88.95**	77.41	82.95	89.79	90.93	88.93	91.05
ScaleFL (Ours)	**83.72**	88.58	**79.65**	**83.79**	**90.53**	**92.13**	**89.72**	**91.20**

Improvements are consistently more significant for local model performances with a range of 1-6% accuracy increase.

Georgia Tech

Experiments

Results – Local Performance Analysis (CIFAR10/100)

Performance improvements are greater at lower complexity levels, which shows the efficiency of submodels created with two-dimensional model downscaling.

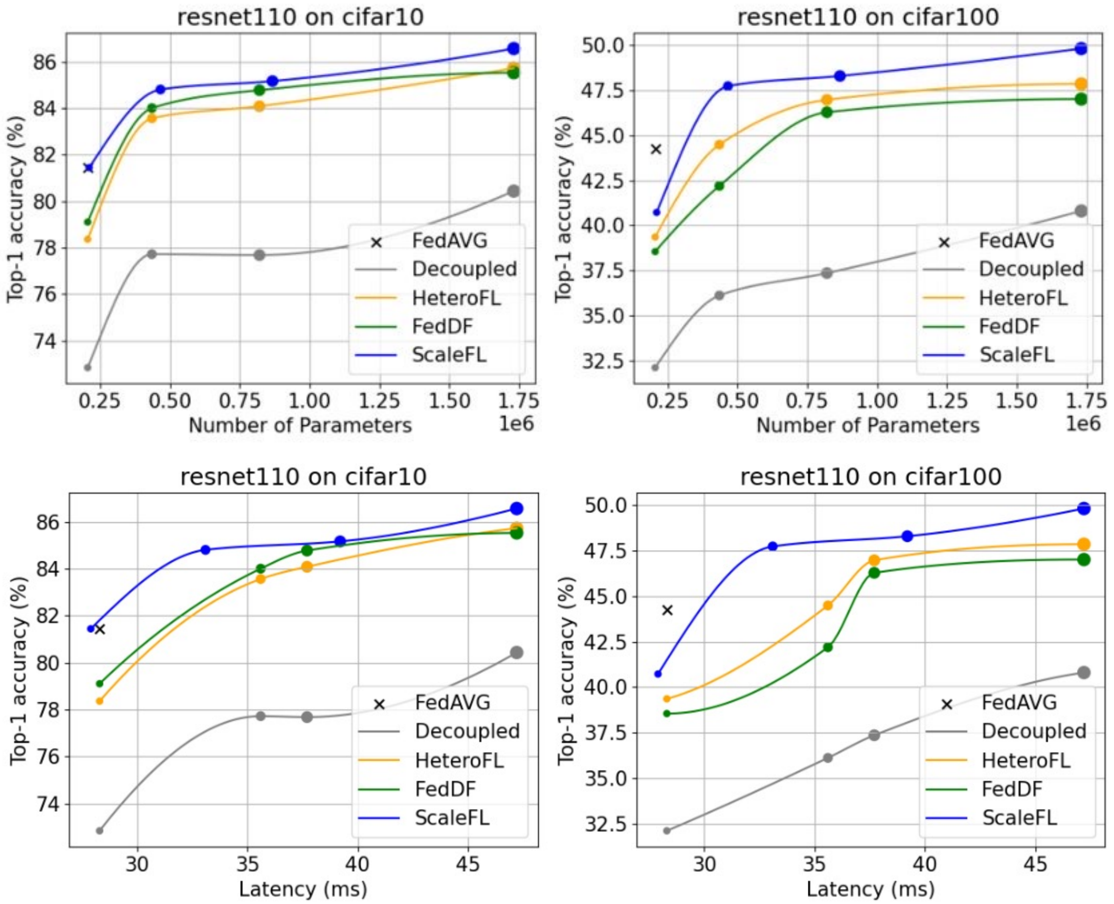


Figure 6: Local model performances (ResNet110)

Georgia Tech

Experiments

Results – Local Performance Analysis (SST-2/AgNews)

For instance, level-2 model on AgNews has 6x faster inference and 0.25x of model size compared to global model while causing 2.5% (vs. 3-4% for other ods) performance drop.

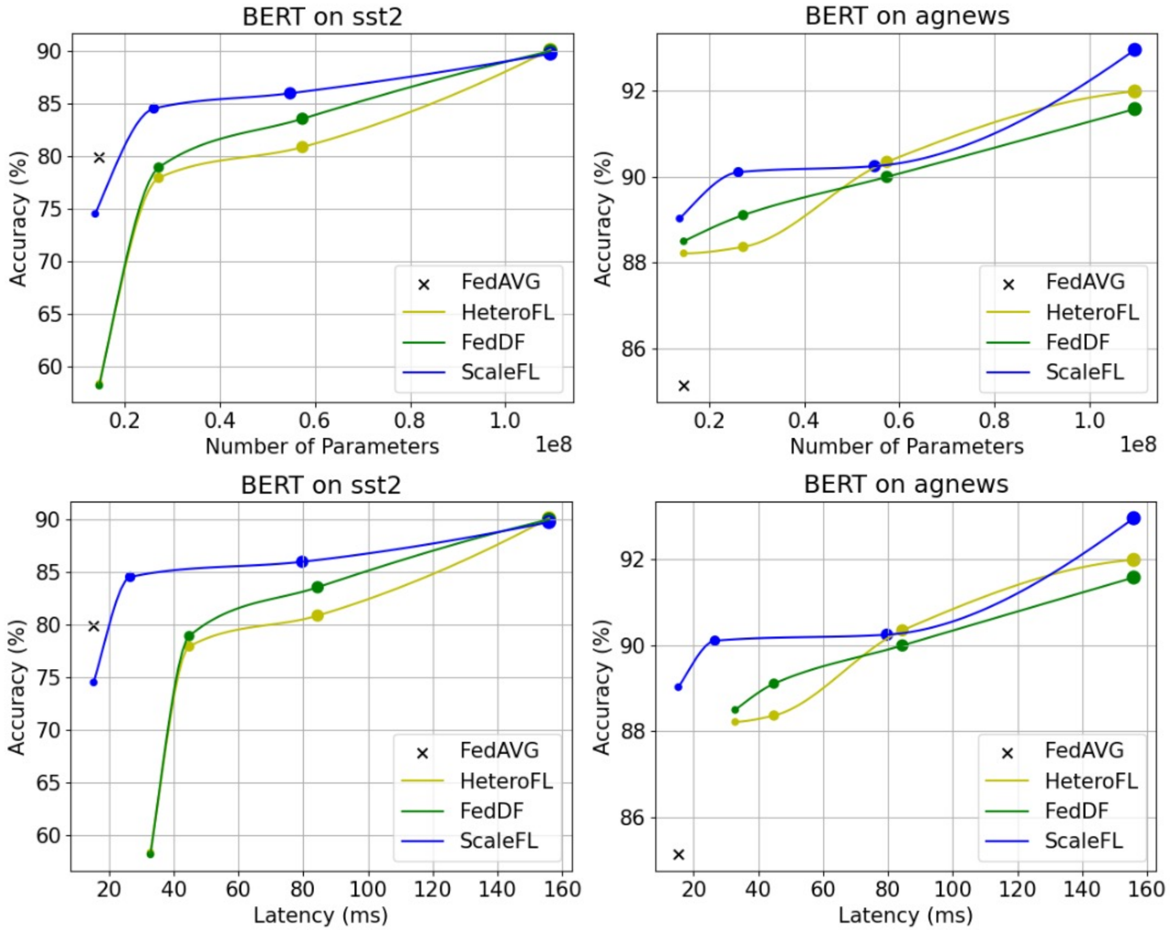


Figure 7: Local model performances (BERT)

References

[1] McMahan, H. B.; Moore, E.; Ramage, D.; Hampson, S.; and y Arcas, B. A. "Communication-Efficient Learning of Deep Networks from Decentralized Data". In AISTATS 2017.

[2] https://medium.com/accenture-the-dock/instilling-responsible-and-reliable-ai-development-with-federated-learning-d23c366c5efd.

[3] Tan, M.; and Le, Q. "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks". In PMLR 2019.

[4] Zeiler, M; Fergus, R. "Visualizing and Understanding Convolutional Networks". In ECCV 2014.

[5] Hinton, G.; Vinyals, O.; and Dean, J. "Distilling the Knowledge in a Neural Network". 2015.

[6] Diao, E.; Ding, J.; and Tarokh, V. "HeteroFL: Computation and Communication Efficient Federated Learning for Heterogeneous Clients". In ICLR 2021.

[7] Lin, T.; Kong, L.; Stich, S. U.; and Jaggi, M. "Ensemble Distillation for Robust Model Fusion in Federated Learning". In NeurIPS 2020.

[8] Li T.; Sahu A. K.; Zaheer M.; Sanjabi M.; Talwalkar A.; and Smith V. "Federated Optimization in Heterogeneous Networks". In MLSys 2020.

[9] Thapa C.; Chamikara M. A. P.; Camtepe S. and Sun L. "SplitFed: When Federated Learning Meets Split Learning". In AAAI 2022.

[10] Li D. and Wang J. "Heterogenous Federated Learning via Model Distillation". In NeurIPS Workshop 2019.

[11] Haddadpour et. al. "Federated Learning with Compression: Unified Analysis and Sharp Guarantees". In PMLR 2021.

[12] Teerapittayanon, S.; McDanel, B.; and Kung, H. T. "BranchyNet: Fast inference via early exiting from deep neural networks". In ICPR 2016.

[13] Huang, G. et. al. "Multi-Scale Dense Networks for Resource Efficient Image Classification". In ICLR 2018.

[14] Zhou et. al. "BERT Loses Patience: Fast and Robust Inference with Early Exit". In NeurIPS 2020.

[15] Dai et. al. "EPNet: Learning to Exit with Flexible Multi-Branch Network". In CIKM 2020.

[16] Chen et. al. "Learning to Stop While Learning to Predict". In ICML 2020.