Model Elasticity for Hardware Heterogeneity in Federated Learning Systems

Allen-Jasmin Farcas, Xiaohan Chen, Zhangyang Wang, Radu Marculescu
{allen.farcas,xiaohan.chen,atlaswang,radum}@utexas.com
The University of Texas at Austin
Austin, TX, USA

ABSTRACT

Most Federated Learning (FL) algorithms proposed to date obtain the global model by aggregating multiple local models that typically share the *same* architecture, thus overlooking the impact on the *hardware heterogeneity* of edge devices. To address this issue, we propose a model-architecture co-design framework for FL optimization based on the new concept of *model elasticity*. More precisely, we enable local devices to train different models belonging to the same architecture family, selected to match the resource budgets (e.g., latency, memory, power) of various edge devices. Our results on EMNIST and CIFAR-10 for both IID and non-IID cases show up to 2.44× less data transferred per communication round and up to 100× reduction in the number of communication rounds, while providing the same or better accuracy compared to existing approaches.

CCS CONCEPTS

• **Computing methodologies** → **Machine learning**; **Supervised learning**; • **Security and privacy**;

KEYWORDS

Federated Learning, Edge Devices, Hardware Heterogeneity, Data Privacy, Model-Architecture Co-Design, Communication Cost, Model Heterogeneity, Internet-of-Things

ACM Reference Format:
Allen-Jasmin Farcas, Xiaohan Chen, Zhangyang Wang, Radu Marculescu. 2022. Model Elasticity for Hardware Heterogeneity in Federated Learning Systems. In *1st ACM Workshop on Data Privacy and Federated Learning Technologies for Mobile Edge Network (FedEdge'22), October 17, 2022, Sydney, NSW, Australia.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3556557.3557954

FedEdge'22, October 17, 2022, Sydney, NSW, Australia

ACM ISBN 978-1-4503-9521-2/22/10…$15.00
https://doi.org/10.1145/3556557.3557954

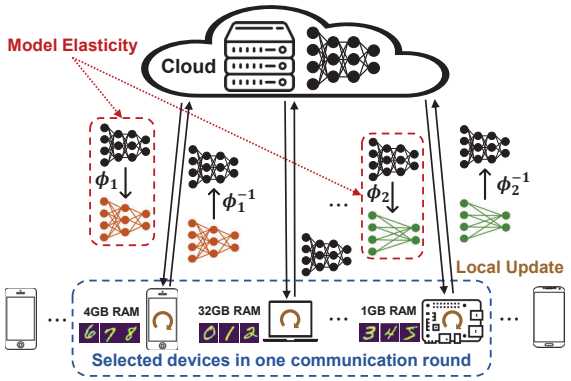


Figure 1: We propose *model elasticity* to allow the cloud to distribute elastic models that satisfy the heterogeneous hardware constraints of edge devices.

1 INTRODUCTION

Federated Learning (FL) is the de facto paradigm for Machine Learning (ML) on edge devices, where each device uses its own local data for training to protect data privacy [8, 16]. FL usually uses the *same* model architecture on the resource-limited edge devices for training. Unfortunately, current ML models are too complex for training or inference on heterogeneous edge devices. Therefore, we propose a new FL approach that can flexibly handle hardware heterogeneity while maximizing the model performance and communication-efficiency.

LotteryFL [13] shows that sparsity alone can reduce the communication cost, but it *cannot* reduce memory and computational costs of on-device inference and training, unless the device has built-in specialized hardware for sparse matrix multiplication and addition. Few edge devices in real FL systems have such capabilities. To address these challenges, we propose *model elasticity* as a solution to hardware heterogeneity (Figure 1); this relies on *elastic transformations* that adapt the complex models from the cloud to smaller models that satisfy specific hardware constraints when distributed on edge devices. We integrate the model elasticity with pruning [6] to further reduce the model size, memory, inference time, and the amount of data communicated at every round between the cloud and the edge devices. Sparsity can thus provide another form of *elasticity* (in addition to

model architecture), as we can select different sparsity ratios to satisfy the hardware constraints on every device.

The contributions of this work are as follows:

• **A new notion of model elasticity**. We propose *model elasticity* to better fit the hardware constraints of heterogeneous devices by transforming the models into larger or smaller versions of themselves via *elastic transformations*.

• **Integration of model elasticity and sparsity**. We integrate sparsity into model elasticity by adopting Iterative Magnitude Pruning (IMP) to reduce both communication and on-device training costs. This additional form of elasticity enabes a selection of sparsity ratios produced by IMP.

• **A new optimization framework**. We propose a constrained optimization framework called **eLF** (**e**lastic **L**earning for **F**ederated systems) that considers heterogeneous resource budgets, model elasticity, and model sparsity for various edge devices. To the best of our knowledge, we are the first to unify heterogeneous hardware, model elasticity and model sparsity to improve communication efficiency for FL on heterogeneous edge devices.

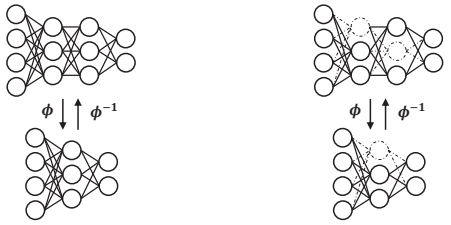
• **Results validation**. We validate the proposed framework using Conv and ResNet model architectures on EMNIST [2] and CIFAR-10 [11] datasets for both IID and non-IID cases. The results show up to 2.44× less data transferred per communication round and up to 100× reduction in the number of communication rounds, while providing the same or better accuracy compared to the baselines.

2 RELATED WORK

Numerous FL approaches [22, 24] focus on communication efficiency. For instance, FedKD [24] is proposed as a communication-efficient FL method, yet it trains on-device not one, but two models (i.e., a teacher and a student) and only shares the smaller (i.e., student) model with the cloud, thus creating a computational overhead for the edge devices.

Sparse neural networks (NNs) have been intensively investigated in the form of pruning for efficient and low-latency on-device inference [4, 6]. Recently, the "Lottery Ticket Hypothesis" (LTH) [3] finds sparse subnetworks in randomly initialized NNs (termed as *winning tickets*) that can be trained in isolation from scratch to match the performance of well-trained dense networks.

Several recent works discuss the combination of pruning with FL [9, 10, 12–14, 23]. The authors in [23] exploit the downlink broadcast to enhance communication efficiency of LotteryFL, yet overlook the hardware heterogeneity of devices. The authors of [12] extend the LotteryFL idea to learn device-personalized and structured sparse masks, while aggregating only their overlapping nonzero elements. However, these methods start from dense models and run on-device pruning based on local validation performance metrics, thus increasing the memory and computational overhead on the edge devices. PruneFL [10] uses unstructured pruning to add and remove weights every communication round eventually obtaining a lottery ticket, but not necessarily the smallest.



(a) Model elasticity **(b) Model elasticity with sparsity**

Figure 2: (a) We elastically transform the complex global model to fit the hardware constraints of heterogeneous edge devices. (b) Sparsity integrated with the elastic transformation (dotted nodes/links).

Heterogeneity in FL has been addressed mostly from a data heterogeneity point of view [16, 18, 25]. Very few FL papers dive deep into the effects of heterogeneous hardware and the corresponding needs for design elasticity [8]. FedMD [15] proposes a framework that enables each device to design its own model locally, while enabling global aggregation with transfer learning and knowledge distillation.

Works like [5, 19] take different perspectives over the term "elasticity". In [19], the authors propose an elastic term in the local objective function to improve the optimization with non-IID data and low participation rate of devices. The "elasticity" mentioned in [5] is related to the weights assigned to each layer from the local models that adjust the layer-wise contributions to the global model.

In contrast, we propose a new dimension of "elasticity" that enables elastic models to fit the hardware constraints of each device. Our proposed model elasticity is orthogonal to other existing elasticity ideas and hence they can work synergistically [5, 19, 21]. We also move the extra computation for model selection from the resource constrained edge devices to the cloud, thus differentiating our approach from previous work such as [9, 13, 15].

3 PROPOSED METHODOLOGY

Model Elasticity for Hardware Heterogeneity Currently, most FL methods force all devices to learn the *same* model architecture and thus cannot directly learn using different model architectures collaboratively. Therefore, we propose the concept of *model elasticity* to enable heterogeneous edge devices learn using different model architectures.

Direct and inverse elastic transformation In FL, the goal is to learn a complex global model ω, which is usually stored in the cloud (see Figure 1). To distribute ω to resource-constrained edge devices that cannot store ω as is, we shrink ω into a smaller version of itself by selectively

Algorithm 1 **e**lastic **L**earning for **F**ederated systems (**eLF**)

K devices; local epochs E; communication round R
S devices selected in each communication round
Pre-processing in the cloud
Initialize per-device resource budget $B_i^0, i \in [K]$
Initialize global model ω^0
$\Phi \leftarrow \{\phi_1(\omega^0), \ldots, \phi_P(\omega^0)\}$ ▷ Initialize model set
$M_p = \{m_p^l\}_{l=1}^L \leftarrow \text{IMP}(\phi_p(\omega^0), D_{\text{proxy}}, L), p \in [P]$
$\rho \leftarrow \rho(B_i^0) \mapsto (\phi_i^*, m_i^*)$ ▷ Set up look-up table
Elastic federated learning
for each round $r = 1, \ldots, R$; **do**
  $\mathcal{U} \leftarrow \{i_1, \ldots, i_S\}$ ▷ Sample available devices
  for each selected device $u \in \mathcal{U}$ **do in parallel**
    $(\phi_u^*, m_u^*) \leftarrow \rho(B_u^{r-1})$ ▷ Refer to look-up table
    $\omega_u^{r-1} \leftarrow \phi_u^*(\omega^{r-1} \odot m_u^*)$ ▷ Elastic transformation
    $\omega_u^r, B_u^r \leftarrow \texttt{LocalUpdate}\left(\omega_u^{r-1}, D_u, E\right)$
  end for
  $\omega^r \leftarrow \texttt{Aggregate}(\{\phi_u^{*-1}(\omega_u^r)\}_{u \in \mathcal{U}})$
end for △Inverse elastic transformation

removing some layers. We call this operation *(direct) elastic transformation*, denoted as ϕ with $\phi(\omega)$ being the transformed (smaller) model. Before FL training starts, we define a set of P elastic transformations $\Phi = \{\phi_p\}_{p=1}^P$ that transform ω into smaller models with different depths to satisfy the hardware constraints of various resources. For *each device* involved in one communication round, we select one $\phi \in \Phi$ and only transfer $\phi(\omega)$ to the target device.

The use of elastic transformations results in different model architectures for each edge device. To enable model aggregation in the cloud, we use inverse elastic transformations ϕ^{-1} to transform the local models back into the architecture of the global model. By adding extra layers, ϕ^{-1} stretches a model to its deeper version. The weight values in the newly added layers can be set in different ways, such as replicating previous layers, filling with random distributions or constants (e.g., zeros). We empirically find that filling with zeros provides stability to the model training process and thus, better results compared to the other methods. The direct and inverse elastic transformations are illustrated in Figure 2a.

Integrating Elasticity and Sparsity As illustrated in Figure 2b, applying the proposed elastic transformations to a sparse network will result in a sparse subnetwork. We adopt IMP to identify highly trainable sparse subnetworks *starting from the random initialization* of their dense versions [3]. This enables us to prune the global model in the cloud, *before* deploying the models to the edge devices. Thus, we save considerable amounts of computation and communication from the very beginning.

IMP is an iterative scheme that generates a series of sparse masks $\{m^l\}_{l=0}^L$ with increasing sparsity by incrementally pruning a ratio of remaining parameters in each iteration. IMP is performed in the cloud on a proxy dataset D_{proxy}. We made the dependency on a proxy dataset *insignificant* by using only 5% of each dataset with IID data for IMP. After IMP finishes, we select one of the generated subnetworks $\omega \odot m^l$ as the global model, $\odot$ being the operation that applies the sparse mask m^l to the model weights ω.

eLF: A Constrained Optimization Framework We propose a new unified optimization framework that optimizes both model elasticity and sparsity under heterogeneous hardware constraints. Let K denote the total number of devices in a FL system, each of them having a budget $B_i, i \in \{1, \ldots, K\}$ which describes the hardware constraints, e.g., latency, memory, bandwidth, power (or some combinations thereof).

Following [20], we consider a non-convex optimization problem with a finite-sum objective over all devices:

$$\underset{\omega}{\text{minimize}}\ f(\omega) = \sum_{i=1}^K p_i F_i(\omega, D_i), \tag{1}$$

where ω denotes the parameters in the global model; (1) is evaluated on the local datasets D_i to calculate the local objective function values F_i, i.e., the cross-entropy loss function. Here, i is the index of each device and the re-weighting coefficients in the finite sum are conventionally taken by $p_i = |D_i| / \sum_j |D_j|$, with $|D_i|$ being the size of D_i.

Formally, our proposed framework extends the standard FL formulation (1) with resource constraints for heterogeneous edge devices and formulates the extended problem as a constrained optimization over the global model:

$$\underset{\omega, \{\phi_i, m_i\}_{i=1}^K}{\text{minimize}}\ \sum_{i=1}^K p_i F_i(\underbrace{\phi_i(\omega \odot m_i)}_{\text{Integration of model elasticity and sparsity}}, D_i) \tag{2}$$

$$\text{subject to}\ \underbrace{C(\phi_i(\omega \odot m_i)) \le B_i,\ \text{for}\ i = 1, \ldots, K}_{\text{Heterogeneous hardware constraints}}.$$

Here, we optimize the global parameters ω, the direct elastic transformation ϕ_i and the sparse mask m_i to account for hardware heterogeneity when distributing the global model to local devices. Moreover, ϕ_i and m_i are strictly capped by the heterogeneous hardware budgets B_i, therefore both ϕ_i and m_i need to satisfy hard constraints for the hardware cost $C(\phi_i(\omega \odot m_i))$. For simplicity, we consider the $C(\cdot)$ function as the hardware performance measured in FLOPs, but in real FL systems, the device budget can be much more complex (e.g., include available memory or communication speed). The algorithm to solve the constrained optimization problem (2) is outlined in Algorithm 1.

How to optimize ϕ_i and m_i Given the non-convex nature of (1) and the complexity of integer programming over binary sparse masks, the joint optimization of ϕ_i and m_i with ω is challenging. Both ϕ_i and m_i are discrete by design: *(i)* ϕ_i is defined using network rolling/unrolling transformation

Model	EMNIST IID			EMNIST Non-IID			Model	CIFAR10 IID			CIFAR10 Non-IID		
	1	5	10	1	5	10		1	5	10	1	5	10
Conv3	77.62%	83.52%	84.80%	76.64%	81.97%	82.94%	ResNet14	66.53%	82.47%	84.38%	58.08%	74.28%	75.57%
Conv5	78.60%	84.05%	85.33%	77.59%	82.42%	83.29%	ResNet20	70.76%	84.39%	86.39%	63.45%	71.41%	71.46%
Conv7	79.84%	84.58%	85.60%	78.63%	82.86%	83.51%	ResNet32	71.79%	85.5%	86.56%	64.59%	74.61%	77.46%
eLF	**79.74%**	**83.47%**	**84.38%**	**78.53%**	**81.82%**	**82.48%**	**eLF**	**73.56%**	**83.77%**	**86.11%**	**69.92%**	**78.05%**	**77.3%**

Table 1: Test accuracy for 1, 5 and 10 local training epochs for EMNIST and CIFAR10 datasets. Compared to the IID case, in the non-IID case for both datasets eLF shows a smaller accuracy decrease compared to the baselines.

Model	EMNIST		Model	CIFAR10	
	Data Transferred	**eLF Improv.**		Data Transferred	**eLF Improv.**
Conv3	4.55 MB	1.16×	ResNet14	1.34 MB	0.93×
Conv5	5.09 MB	1.30×	ResNet20	2.07 MB	1.44×
Conv7	5.64 MB	1.44×	ResNet32	3.52 MB	2.44×
eLF	3.92 MB	-	**eLF**	1.44 MB	-

Table 2: Average data transferred per communication round. eLF results in significant communication reduction for both Conv and ResNet.

over the unit of building blocks (i.e., add or reduce only a fixed number of blocks each time); *(ii)* m_i is solved by IMP to ensure the "lottery ticket" property. Therefore, we implement a look-up table as an indicator of the performance that can be achieved for a given configuration of elastic transformation and sparse mask.

We use the lookup table to implement a fast mapping function $\rho(\cdot)$ from the resource budget B_i of a given device to the (approximately) optimal configuration (ϕ_i^*, m_i^*) for that device that accommodates the budget constraints. As described in Line 4, Algorithm 1, we apply IMP with L iterations to each elastic model $\phi_p(\omega^0)$ with the initial global model ω^0, to generate a series of winning tickets with increasing sparsity ratios $M_p = \text{IMP}(\phi_p(\omega^0), D_{\text{proxy}}, L) = \{m_p^1, \ldots, m_p^L\}$. This forms a pool of $P \times L$ winning tickets $\{\phi_p(\omega^0 \odot m_p^l)\}_{p,l}$. For each entry in the pool, we also record its accuracy $\theta_{p,l}$ on the proxy dataset D_{proxy}. For every communication round r, when we distribute an elastic model to the i-th device with a resource budget B_i^r during eLF, we use the look-up table to select the elastic transformation ϕ_i^* and sparse mask m_i^* with the highest accuracy $\theta_{p,l}$ on the proxy dataset among all feasible configurations $(\phi_{p^*}, m_{p^*}^{l^*})$. Formally, the fast mapping function $\rho(B_i^r) = (\phi_i^*, m_i^*) \overset{\text{def}}{=} (\phi_{p^*}, m_{p^*}^{l^*})$ where:

$$p^*, l^* = \underset{p \in [P], l \in [L]}{\text{argmax}} \ \theta_{p,l}\,, \ \textbf{s.t.}\ C\left(\phi_p(\omega^0 \odot m_p^l)\right) \leq B_i^r \quad (3)$$

Our approach uses a "warm-up" initialization for ϕ_i and m_i and allows them, during the FL training process, to *adapt to* the budget changes of resource constrained edge devices (see Line 11, Algorithm 1). We follow the standard `LocalUpdate` and `Aggregation` (FedAvg) functions in Algorithm 1 Line 11 and Line 13, respectively. eLF is compatible with more advanced techniques for global aggregation and local training.

4 EMPIRICAL ANALYSIS

Experimental Setup We validate eLF on EMNIST (with 62 classes, having lower and upper case letters and digit classes) and CIFAR10, under both IID and non-IID settings. For the IID case, samples of each dataset are distributed evenly class-wise across all devices. For the non-IID case, we use 5 classes per device on CIFAR10 and 20 classes per device on EMNIST. We use the non-IID sampling function from [13] that creates data distributions using class and quantity skewness. By default, we split data between 100 devices. In the experiments,we use as baselines the Conv and ResNet model architectures for EMNIST and CIFAR10, respectively.

For the baselines, we randomly sample 10 devices every communication round using the same model for all devices. For eLF we sample a total of 9 devices: 3 devices with ResNet14/Conv3, 3 devices with ResNet20/Conv5 and 3 devices with ResNet32/Conv7. Conv models are selected for EMNIST and ResNet models are selected for CIFAR10, all models using a sparsity of 50%. Each eLF scenario uses *only* 5% of the datasets as the proxy dataset to perform IMP. For eLF aggregation we use the *largest model* (i.e., ResNet32/Conv7) as the global model. For the baselines, the same model is used by every device and as a global model as well. By default, we use FedAvg [20] optimization for FL.

To better emulate resource-constrained edge devices, we use convolutional (Conv) models with 500-750K parameters and ResNet models with 175-500K parameters. The Conv models have an initial convolution layer followed by two groups of convolutional layers that increase in depth as we go from Conv3 to Conv5 and Conv7 models. Finally, the Conv models have two fully connected layers for classification. We use the ResNet model architecture from [7]. All experiments are done via simulation on GPU servers.

Experiments on Conv and ResNet models In Table 1, we observe that the drop in accuracy between IID and non-IID cases for eLF is lower compared to the baselines for CIFAR10. As we can see in Table 2, eLF obtains up to 1.44× data communication reduction for Conv models and up to 2.44× data communication reduction for ResNet models, while having

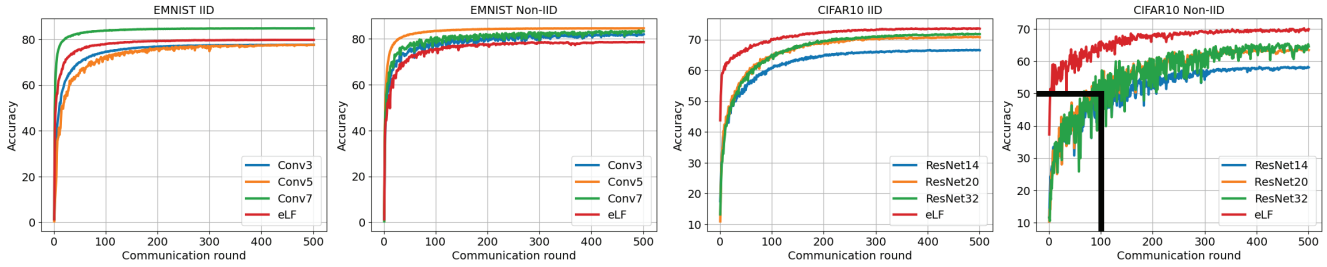


Figure 3: Results for one local epoch on EMNIST using Conv models (a-b) and on CIFAR10 using ResNet models (c-d) for IID and non-IID cases. In (d), eLF with ResNet models obtains 50% accuracy on CIFAR10 using non-IID data in just one communication round, while the baselines take at least 100 rounds to reach the same accuracy.

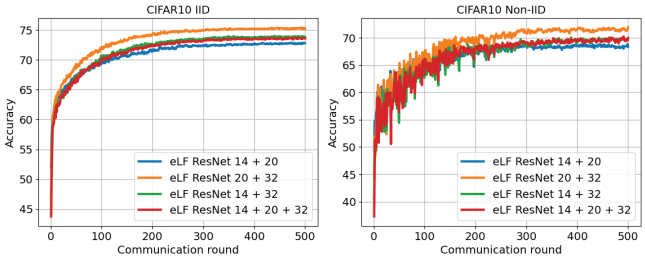


Figure 4: Results on CIFAR10 for one local epoch using ResNet models showing eLF with different elastic model combinations. We observe clear improvements when eLF uses a larger global model (i.e., ResNet 32).

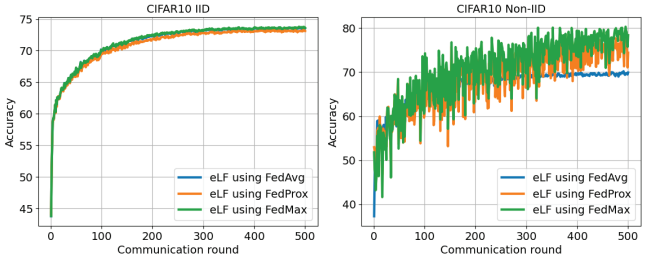


Figure 5: Results on CIFAR10 for one local epoch using eLF with ResNet models combined with FedMax [1] and FedProx [17]; this shows that eLF can easily work with other FL optimizations.

similar or even better accuracy performance compared to the baselines (Table 1). On the CIFAR10 dataset, our method achieves the highest accuracy using one local epoch.

Communication-efficiency We reduce the total number of communication rounds required to reach a certain accuracy threshold and improve the average communication efficiency per communication round as shown in Table 2. The results in Table 1 and Table 2 show that eLF obtains a good balance between accuracy performance and communication costs (i.e., amount of data communicated and the number of communication rounds needed to achieve a certain accuracy). In Figure 3 we also observe how fast eLF can learn in a FL setting compared to the baselines. For instance, on CIFAR10 using non-IID data, eLF obtains 50% accuracy in just one communication round, while the baselines reach the same accuracy after 100 more communication rounds (Figure 3d). Moreover, eLF trains faster at the device level since models are optimized for specific hardware constraints.

Ablation on choices of elastic models In Figure 4 we show how the choice of different elastic model combinations in the FL system impacts the overall performance. For all combinations of elastic models, the largest model used is also the global model. Therefore, when eLF uses ResNet14 and ResNet20 models with ResNet20 as the global model, the overall performance is also lower. Every communication round, eLF 14+20 selects 5 devices with ResNet14 and 5 devices with ResNet20; eLF 14+20+32 selects 3 devices with ResNet14, 3 devices with ResNet20 and 3 devices with ResNet32. eLF does not dramatically decrease the performance when using more elastic models as shown in Figure 4.

eLF compatibility with existing FL methods Finally, we conduct experiments using the same three ResNet elastic models from previous experiments for FL optimization based on FedProx [17] and FedMax [1] (see Figure 5). For FedProx we use $\mu = 200$ and for FedMax $\beta = 10$. These results show that eLF is orthogonal to state-of-the-art FL optimization methods and can synergistically work with them.

5 CONCLUSION

In this paper, we have introduced model elasticity, a new approach that addresses hardware heterogeneity in FL systems. We have also implemented an optimization framework that considers model elasticity, sparsity, and heterogeneous hardware to create a communication-efficient, hardware-aware FL system. Results show up to 2.44× improvement in communication efficiency and up to 100× reduction in communication rounds, with similar or better accuracy compared to the baselines.

6 ACKNOWLEDGMENTS

This research was supported in part by NSF Grant CCF-2107085 and in part by Cisco Corp.

REFERENCES

[1] Wei Chen, Kartikeya Bhardwaj, and Radu Marculescu. 2020. Fedmax: mitigating activation divergence for accurate and communication-efficient federated learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 348–363.

[2] Gregory Cohen, Saeed Afshar, Jonathan Tapson, and Andre Van Schaik. 2017. EMNIST: Extending MNIST to handwritten letters. In *2017 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2921–2926.

[3] Jonathan Frankle and Michael Carbin. 2018. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635* (2018).

[4] Trevor Gale, Erich Elsen, and Sara Hooker. 2019. The state of sparsity in deep neural networks. *arXiv preprint arXiv:1902.09574* (2019).

[5] Yujia Gao, Liang Liu, Xiaolong Zheng, Chi Zhang, and Huadong Ma. 2021. Federated sensing: Edge-cloud elastic collaborative learning for intelligent sensing. *IEEE Internet Things J.* 8, 14 (July 2021), 11100–11111.

[6] Song Han, Huizi Mao, and William J Dally. 2015. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149* (2015).

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[8] Ahmed Imteaj, Urmish Thakker, Shiqiang Wang, Jian Li, and M Hadi Amini. 2020. Federated Learning for Resource-Constrained IoT Devices: Panoramas and State-of-the-art. (Feb. 2020). arXiv:2002.10610 [cs.LG]

[9] Yuang Jiang, Shiqiang Wang, Victor Valls, Bong Jun Ko, Wei-Han Lee, Kin K Leung, and Leandros Tassiulas. 2019. Model pruning enables efficient federated learning on edge devices. *arXiv preprint arXiv:1909.12326* (2019).

[10] Yuang Jiang, Shiqiang Wang, Victor Valls, Bong Jun Ko, Wei-Han Lee, Kin K Leung, and Leandros Tassiulas. 2022. Model pruning enables efficient federated learning on edge devices. *IEEE Transactions on Neural Networks and Learning Systems* (2022).

[11] Alex Krizhevsky, Geoffrey Hinton, et al. 2009. Learning multiple layers of features from tiny images. (2009).

[12] Ang Li, Jingwei Sun, Pengcheng Li, Yu Pu, Hai Li, and Yiran Chen. 2021. Hermes: an efficient federated learning framework for heterogeneous mobile clients. In *Proceedings of the 27th Annual International Conference on Mobile Computing and Networking*. 420–437.

[13] Ang Li, Jingwei Sun, Binghui Wang, Lin Duan, Sicheng Li, Yiran Chen, and Hai Li. 2020. Lotteryfl: Personalized and communication-efficient federated learning with lottery ticket hypothesis on non-iid datasets. *arXiv preprint arXiv:2008.03371* (2020).

[14] Ang Li, Jingwei Sun, Xiao Zeng, Mi Zhang, Hai Li, and Yiran Chen. 2021. FedMask: Joint Computation and Communication-Efficient Personalized Federated Learning via Heterogeneous Masking. In *Proceedings of the 19th ACM Conference on Embedded Networked Sensor Systems*. 42–55.

[15] Daliang Li and Junpu Wang. 2019. FedMD: Heterogenous Federated Learning via Model Distillation. (Oct. 2019). arXiv:1910.03581 [cs.LG]

[16] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. 2020. Federated Learning: Challenges, Methods, and Future Directions. *IEEE Signal Process. Mag.* 37, 3 (May 2020), 50–60.

[17] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. 2018. Federated optimization in heterogeneous networks. *arXiv preprint arXiv:1812.06127* (2018).

[18] Yi Liu, Xingliang Yuan, Zehui Xiong, Jiawen Kang, Xiaofei Wang, and Dusit Niyato. 2020. Federated learning for 6G communications: Challenges, methods, and future directions. , 105–118 pages.

[19] Zichen Ma, Yu Lu, Zihan Lu, Wenye Li, Jinfeng Yi, and Shuguang Cui. 2021. Towards Heterogeneous Clients with Elastic Federated Learning. (June 2021). arXiv:2106.09433 [cs.LG]

[20] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*. PMLR, 1273–1282.

[21] Vladimir Podolskiy. 2020. Rubberband: Enabling Elastic Federated Learning with the Temporary State Management. In *Proceedings of the 21st International Middleware Conference Doctoral Symposium* (Delft, Netherlands) *(Middleware'20 Doctoral Symposium)*. Association for Computing Machinery, New York, NY, USA, 9–14.

[22] Amirhossein Reisizadeh, Aryan Mokhtari, Hamed Hassani, Ali Jadbabaie, and Ramtin Pedarsani. 2020. FedPAQ: A Communication-Efficient Federated Learning Method with Periodic Averaging and Quantization. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics (Proceedings of Machine Learning Research, Vol. 108)*, Silvia Chiappa and Roberto Calandra (Eds.). PMLR, 2021–2031.

[23] Sejin Seo, Seung-Woo Ko, Jihong Park, Seong-Lyun Kim, and Mehdi Bennis. 2021. Communication-Efficient and Personalized Federated Lottery Ticket Learning. (April 2021). arXiv:2104.12501 [cs.LG]

[24] Chuhan Wu, Fangzhao Wu, Ruixuan Liu, Lingjuan Lyu, Yongfeng Huang, and Xing Xie. 2021. FedKD: Communication Efficient Federated Learning via Knowledge Distillation. (Aug. 2021). arXiv:2108.13323 [cs.LG]

[25] Xuefei Yin, Yanming Zhu, and Jiankun Hu. 2021. A Comprehensive Survey of Privacy-preserving Federated Learning: A Taxonomy, Review, and Future Directions. *ACM Comput. Surv.* 54, 6 (July 2021), 1–36.