

Optimizing sparse ternary compression with thresholds for communication-efficient federated learning

Nithyaniranjana Murthy Chittaiiah, Manjula Sunkadakatte Haladappa

Department of Computer Science and Engineering, University Visvesvaraya College of Engineering, Bangalore University, Bengaluru, India

Article Info

Article history:

Received Oct 26, 2024

Revised Oct 24, 2025

Accepted Nov 8, 2025

Keywords:

Communication efficiency
Distributed machine learning
Federated learning
Sparse ternary compression
STC threshold

ABSTRACT

Federated learning (FL) enables decentralized model training while preserving client data privacy, yet suffers from significant communication overhead due to frequent parameter exchanges. This study investigates how varying sparse ternary compression (STC) thresholds impact communication efficiency and model accuracy across the CIFAR-10 and MedMNIST datasets. Experiments tested thresholds ranging from 1.0 to 1.9 and batch sizes of 10, 15, and 20. Results demonstrated that selecting thresholds between 1.2 and 1.5 reduced total communication costs by approximately 10–15%, while maintaining acceptable accuracy levels. These findings suggest that careful threshold tuning can achieve substantial communication savings with minimal compromise in model performance, offering practical guidance for improving the efficiency and scalability of FL systems.

This is an open access article under the [CC BY-SA](#) license.



Corresponding Author:

Nithyaniranjana Murthy Chittaiiah

Department of Computer Science and Engineering, University of Visvesvaraya College of Engineering

Bangalore University

Bengaluru, Karnataka, India

Email: nithya.semantic@gmail.com

1. INTRODUCTION

Federated learning (FL) represents a decentralized approach to machine learning, where several clients work together to train a shared model without directly sharing local datasets. Unlike traditional approaches that rely on centralized data aggregation for model development, FL ensures that data remains on each client device, thus enhancing privacy. This decentralized setup is particularly advantageous in domains like healthcare, finance, and mobile systems, where data sensitivity is a primary concern [1]. Despite the potential advantages, FL faces significant challenges, particularly in terms of communication efficiency. In a typical FL setup, model updates are iteratively sent by clients to a central server, which aggregates these updates and returns a global model to the clients. Modern machine learning models often consist of millions of parameters, and communicating these updates can lead to substantial network bandwidth consumption, especially in resource-constrained environments. As a result, the frequency and volume of these communications can become a bottleneck, affecting both the scalability and efficiency of FL systems [2].

FL integrates data from a variety of devices, such as sensors, mobile phones, and IoT systems, each possessing distinct data characteristics and serving applications across domains like healthcare, finance, and online education. However, FL encounters challenges including privacy risks, implementation difficulties, hardware constraints, communication costs, and device unavailability [3]-[5]. Many applications across various domains involve sensitive personal data, making it crucial for all stakeholders, including companies, agencies,

and researchers, to take collective responsibility in safeguarding this information from misuse and abuse [6]-[8].

Various model compression strategies have emerged to mitigate communication bottlenecks. One such technique is sparse ternary compression (STC), which reduces the size of the model updates by compressing them into three discrete states: -1, 0, and 1. By representing the model updates in this ternary form, the number of bits required for transmission is significantly reduced, leading to lower communication costs. However, the effectiveness of STC is closely tied to the selection of the compression threshold, which influences the extent of sparsity in the updates. Selecting an appropriate threshold helps balance communication savings with model performance; however, a poorly chosen value may cause significant information degradation or fail to compress sufficiently [9].

FL can be applied in various domains, including smart retail, energy management, smart cities, finance, vehicle-to-vehicle communication, and healthcare [10], [11]. Applications such as patient monitoring, moisture detection in crops, and predictive text in keyboards use FL to obtain data-driven insights. With the involvement of sensitive personal data, ensuring privacy is paramount. For instance, the General Data Protection Regulation (GDPR) is one of several regulatory frameworks adopted in regions including the EU and UK to safeguard personal data [12], [13].

The paper investigates the impact of varying STC thresholds, specifically focusing on values of 1.0, 1.2, 1.5, 1.7, and 1.5, on communication efficiency and model accuracy in FL. Systematic experimentation is conducted with different threshold values, followed by an analysis of the effects on a well-established machine learning task. Insights are offered on how STC thresholds affect the trade-off between communication cost and model performance. Our study aims to offer guidance on choosing optimal STC thresholds to enhance efficiency, scalability, and the practical use of FL in real-world scenarios.

2. RELATED WORK

McMahan *et al.* [14] introduced the concept of FL through the development of the Federated Averaging (FedAvg) algorithm, which enables aggregation of locally computed gradients from multiple decentralized devices. Following this work, FL began receiving increased attention due to its ability to perform machine learning directly on local client data without requiring central data transfer. This foundational study demonstrated the feasibility of training models across distributed systems while preserving user privacy, paving the way for wider adoption in both research and industry.

Sattler *et al.* [15] introduced STC to reduce communication costs in FL by compressing model updates. STC builds on top-k gradient sparsification and adds ternarization with optimal Golomb encoding, enabling more efficient downstream communication. Quantized SGD (QSGD), proposed by Alistarh *et al.* [16], is a technique that minimizes communication overhead in distributed machine learning by quantizing gradients. QSGD enhances training efficiency and scalability by reducing the data exchanged between workers during gradient descent while maintaining convergence. Bernstein *et al.* [17] presented signSGD, a compression technique that provides theoretical convergence guarantees on independent and identically distributed (IID) data. The technique reduces the bit size for each gradient update by a factor of 32 by converting each update into a binary sign. Additionally, signSGD performs download compression by aggregating binary updates from all clients through a majority voting mechanism.

Rothchild *et al.* [18] introduced FetchSGD to address communication bottlenecks and convergence issues in FL. In research by Wang *et al.* [19], communication cost optimization was identified as a crucial factor, with three primary techniques suggested for reduction: i) allowing local updates to reduce the frequency of communication, ii) compressing messages to lower the volume of data transmitted, and iii) reducing communication traffic at the server by limiting the number of participating clients per round. Research conducted by Yang *et al.* [20] focused on wireless communication using the principle of Over-the-Air computation. A model for FL was designed by exploiting signal superposition of wireless devices, combined with beamforming design and joint device selection, to enhance statistical learning performance. Chen *et al.* [21] identified challenges in wireless communication and proposed a model that jointly selects users and allocates resources, effectively reducing packet loss and improving FL model performance. Emphasis was also placed on reducing energy consumption during local training and transmission, which is a key consideration for improving FL efficiency.

Cheng *et al.* [22] examined two primary factors for reducing communication costs: i) increasing computational power on local devices to allow for more local updates before global aggregation,

and ii) selecting more participants for each round by enhancing parallelism. Simulation results indicated that increasing parallelism significantly reduced communication costs. However, its effectiveness became apparent only after reaching a specific threshold, compared to simply increasing the computational power on participant devices. Liu *et al.* [7] extended the concept to vertical federated learning (VFL), which involves the same set of users or participants with different feature sets. For example, in the same city, a local bank and a local retail company may share the same customer base but maintain distinct feature sets, facilitating collaborative model building. A novel algorithm called Federated Stochastic Block Coordinate Descent (FedBCD) was introduced, enabling several local updates before communicating with the global server for aggregation. The algorithm also ensured local convergence with fewer communication rounds.

FedZip, a compression framework for FL, was introduced by Malekijoo *et al.* [23] to address communication overhead, energy consumption, and performance degradation. Sparsification was achieved using Top-z pruning, while K-means clustering, quantization of model weights, and Huffman encoding were employed for model compression. When applied to client-to-server communication, FedZip improved communication efficiency in FL systems with minimal effects on accuracy and convergence. Haddadpour *et al.* [24] focused on reducing uplink communication costs by compressing messages sent from client devices to the central server. To prevent information loss during compression, the central server generates a convex combination of the previous global model and the aggregated updates from local models.

3. METHODOLOGY

3.1. Research design

The study aimed to evaluate the impact of varying STC thresholds on communication efficiency and model accuracy in a FL context. Experiments tested STC threshold values of 1.0, 1.2, 1.5, 1.7, and 1.9 across batch sizes of 10, 15, and 20. Two diverse image classification datasets—CIFAR-10 and MedMNIST—were selected to represent both natural images and medical imaging tasks, supporting broader applicability. Data was partitioned in a non-IID (Non-Independent and Identically Distributed) manner to mimic realistic client heterogeneity typical in federated settings. This design enabled a systematic analysis of how different threshold settings and batch sizes influence communication cost and model performance across varying data characteristics.

3.2. Experimental procedure

The experimental procedure involved several key steps designed to systematically evaluate the impact of STC thresholds on FL performance. Data partitioning: The CIFAR-10 and MedMNIST datasets were used to provide diverse evaluation scenarios. Data was partitioned among multiple simulated clients (devices) to replicate a FL system with non-IID (non-independent and identically distributed) distributions (see Table 1 for environment configuration). This setup aimed to closely mimic real-world heterogeneity, where data is unevenly distributed across participants.

Table 1. Federated learning configuration details

Configuration attribute	Client count	Selection fraction	Labels per client	Mini-batch size	Dataset balance factor
Assigned value	10	0.1	2	10, 15, 20	1.0

Model training: a convolutional neural network (CNN) architecture based on VGG11* was employed as the client-side model. Hyperparameters and model configurations used for CIFAR-10 and MedMNIST experiments are summarized in Table 2. Each client trained the model locally on its partitioned data for a fixed number of iterations. After local training, model parameters were compressed using the STC technique before transmission to the central server, aiming to reduce communication costs.

Table 2. Models and hyperparameters used in experiments

Parameter	CIFAR-10 value	MedMNIST value
Model architecture	VGG11*	VGG11*
Learning rate	0.016	0.010
Optimizer	SGD	SGD
Loss function	Cross-entropy	Cross-entropy
Iterations	20,000	20,000

Note: VGG11* denotes a streamlined adaptation of the VGG11 model [25], where dropout and batch normalization layers have been excluded, and both the convolutional filter count and the dimensions of the fully connected layers have been scaled down to half.

STC: the STC technique reduces model updates to three states: -1, 0, and 1, effectively lowering the number of bits needed for communication. Compression is controlled by a threshold parameter τ , which governs the sparsity of the updates. The STC algorithm operates as in (1).

$$\text{STC}(g_i) = \begin{cases} 1 & \text{if } g_i \geq \tau \\ -1 & \text{if } g_i \leq -\tau \\ 0 & \text{if } -\tau < g_i < \tau \end{cases} \quad (1)$$

Where g_i represents an individual model gradient component. Elements with absolute values exceeding τ are retained with their sign, while others are zeroed out. Experiments systematically varied τ across values 1.0, 1.2, 1.5, 1.7, and 1.9 to study its influence on both communication cost and accuracy. This range enabled a detailed analysis of how increased sparsification impacts the trade-off between compression efficiency and model performance. The ternarization approach also enables encoding gradients with fewer bits, reducing communication overhead significantly.

Model aggregation and update: the central server aggregated compressed updates from all participating clients using weighted averaging, as in (2).

$$w^{(t+1)} = \sum_{k=1}^K \frac{n_k}{n} w_k^{(t)} \quad (2)$$

Here, $w^{(t+1)}$ denotes the global model at round $t + 1$, $w_k^{(t)}$ are the parameters from client k , n_k is the number of samples on client k , and n is the total number of samples across all clients. The aggregated global model was then redistributed to all clients for the next round of local training.

Performance evaluation: performance was assessed using two primary metrics: model accuracy and communication cost, measured in bits transmitted. Accuracy was evaluated on held-out validation data, while communication cost accounted for the total number of bits sent during each communication round. Experiments were repeated over multiple rounds to analyze convergence behavior and to compare the effects of different thresholds and batch sizes across both datasets.

3.3. Testing and data acquisition

The testing phase involved running the FL training process with each STC threshold value. Results, including model accuracy and communication cost, were recorded at each iteration. The data acquisition process was automated using scripts that logged the necessary metrics, ensuring consistent and reliable data collection. The recorded data was then analyzed to identify trends and draw conclusions about the effectiveness of different STC thresholds in FL systems. The research followed a rigorous experimental protocol, ensuring that the results are scientifically valid and reproducible. The methods used for testing and data acquisition were based on established practices in FL research providing a solid foundation for the conclusions drawn in the study.

4. RESULTS

The experimental findings evaluating the impact of STC thresholds on communication efficiency and model performance in FL are presented in this section. Results were systematically analyzed for different thresholds (1.0, 1.2, 1.5, 1.7, and 1.9) and batch sizes (10, 15, and 20), using two datasets: CIFAR-10 and MedMNIST. Key aspects examined include total communication cost, model accuracy across training iterations, and maximum accuracy achieved across batch sizes. These analyses reveal the balance between reducing communication overhead and maintaining model effectiveness, offering practical guidance for selecting appropriate STC thresholds in diverse FL scenarios. Threshold 1.9 is excluded from all plots due to consistently low accuracy. Its inclusion would distort axis scaling and obscure meaningful comparisons across other thresholds.

4.1. Total bits sent up vs. batch size for different sparse ternary compression thresholds (in MB)

The first aspect of the analysis focused on evaluating communication efficiency in the FL system by measuring the total bits sent during the training process. Figures 1 and 2 illustrate how total bits sent varied with different batch sizes and STC thresholds across two datasets: CIFAR-10 and MedMNIST. For the CIFAR-10 dataset (Figure 1), the results indicated a general trend where increasing the STC threshold resulted in lower total bits sent across batch sizes. Higher thresholds (such as 1.5) effectively reduced the volume of communication, particularly at larger batch sizes. This behavior suggests that aggressive compression successfully reduces the data that must be transmitted between clients and the server, thereby lowering communication overhead.

The MedMNIST dataset results (Figure 2) generally reinforced these findings. However, while higher thresholds still reduced communication costs, the extent of this reduction was less uniform across batch sizes, likely attributable to the dataset's inherent complexity and distribution characteristics. Overall, the findings across both datasets emphasize the value of tuning STC thresholds to optimize communication efficiency in FL setups. Careful selection of the threshold parameter enables significant reductions in transmission volume, but it remains essential to balance this gain with potential impacts on model accuracy, as discussed in subsequent sections. Results for threshold 1.9 are excluded from the table and figures due to consistently poor accuracy, illustrating that excessive compression degrades model performance beyond acceptable limits. The numerical values corresponding to Figures 1 and 2 are summarized in Table 3, detailing total bits sent for each STC threshold and batch size combination.

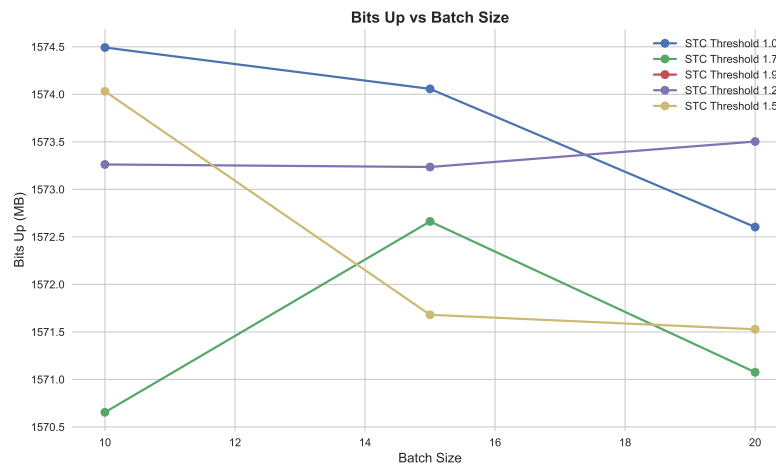


Figure 1. Total bits sent up vs. batch size for different STC thresholds on CIFAR-10

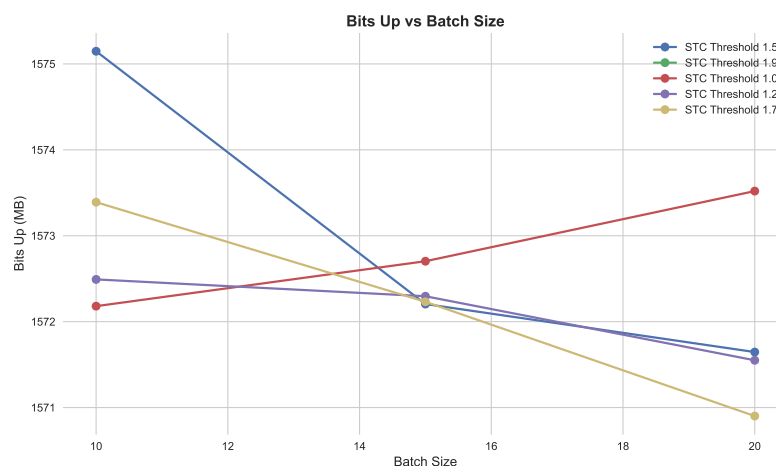


Figure 2. Total bits sent up vs. batch size for different STC thresholds on MedMNIST

Table 3. Total bits sent up (MB) vs. batch size for different STC thresholds on CIFAR-10 and MedMNIST

Batch size	CIFAR-10				MedMNIST			
	1.0	1.2	1.5	1.7	1.0	1.2	1.5	1.7
10	1574.49	1573.26	1574.03	1570.65	1572.18	1572.49	1575.15	1573.39
15	1574.06	1573.24	1571.68	1572.66	1572.70	1572.30	1572.20	1572.23
20	1572.60	1573.50	1571.53	1571.08	1573.52	1571.55	1571.65	1570.90

Note: Threshold 1.9 is excluded due to consistently poor accuracy

4.2. Accuracy vs. iterations for different sparse ternary compression thresholds

The second aspect of the analysis focused on evaluating model accuracy as training progressed over iterations, highlighting convergence behavior under different STC thresholds. Figures 3 and 4 show the accuracy curves for the CIFAR-10 and MedMNIST datasets, respectively, across thresholds 1.0, 1.2, 1.5, 1.7, and 1.9. For the CIFAR-10 results (Figure 3), accuracy improved steadily over 20,000 iterations for thresholds 1.0, 1.2, and 1.5. Threshold 1.0 consistently delivered the highest accuracy at convergence, demonstrating more stable learning with less accuracy drop. Threshold 1.5 showed competitive early convergence but exhibited slightly more variability in late training stages. Thresholds 1.7 and 1.9 led to noticeably degraded accuracy, with threshold 1.9 failing to improve significantly during training. This pattern indicates that overly aggressive compression (high thresholds) can harm learning by discarding too much gradient information.

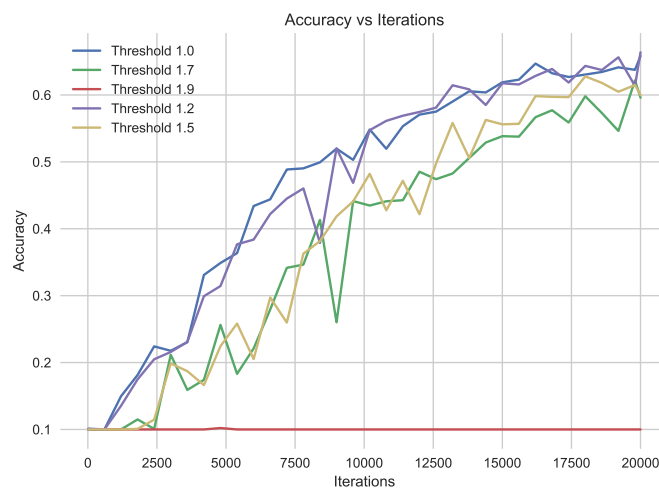


Figure 3. CIFAR-10: accuracy vs. iterations for different STC thresholds

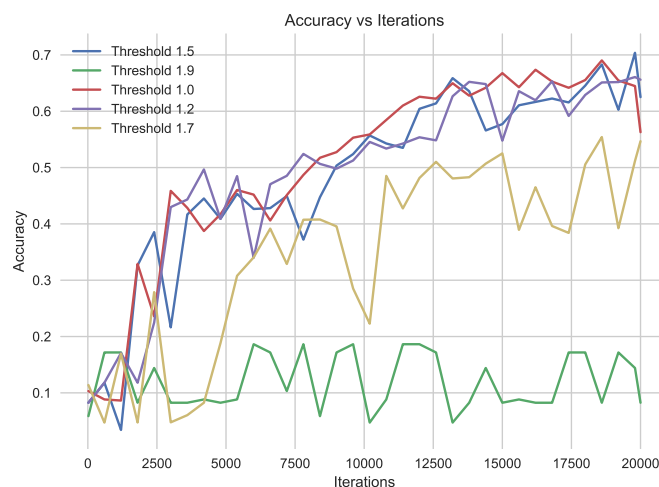


Figure 4. MedMNIST: accuracy vs. iterations for different STC thresholds

For MedMNIST (Figure 4), similar trends were observed. Thresholds 1.0 and 1.5 achieved higher and smoother accuracy gains across training rounds, with threshold 1.0 generally converging at slightly higher accuracy. Threshold 1.2 also delivered competitive performance, reinforcing that moderately low thresholds preserve critical learning signals. Conversely, thresholds 1.7 and 1.9 consistently lagged in accuracy across iterations, indicating diminished learning capacity due to excessive sparsification.

These results confirm that selecting lower STC thresholds (e.g., 1.0, 1.2) supports better convergence and model accuracy while still offering meaningful compression benefits. The trade-off between communication reduction and model performance is evident: while higher thresholds reduce data transfer costs, they risk compromising accuracy. Careful selection of STC thresholds is thus essential for balancing efficiency and learning quality in FL settings.

4.3. Max accuracy vs. batch size for different sparse ternary compression thresholds

The final aspect of the analysis examined the maximum accuracy achieved at different batch sizes for each STC threshold. Figures 5 and 6 display these results for the CIFAR-10 and MedMNIST datasets, respectively, showing how threshold selection and batch size interact to influence model performance. For the CIFAR-10 dataset (Figure 5), thresholds 1.0 and 1.2 consistently achieved higher maximum accuracy across all batch sizes compared to higher thresholds such as 1.5, 1.7, and 1.9. The results showed that as batch size increased from 10 to 20, accuracy improved most noticeably for lower thresholds. For instance, threshold 1.0 peaked near 71% at batch size 20, highlighting its stability and robustness even as communication cost increased. By contrast, thresholds 1.7 and 1.9 showed lower and flatter trends, indicating degraded performance likely due to excessive sparsification of gradient updates. This behavior suggests that aggressive compression harms learning signal retention, particularly in larger batch configurations.

For the MedMNIST dataset (Figure 6), a different pattern emerged. Threshold 1.5 generally achieved competitive or even superior accuracy at batch size 15, peaking above 70%. Threshold 1.0 maintained strong performance but did not always outperform 1.5, especially at mid-sized batches. Meanwhile, thresholds 1.7 and 1.9 again showed weaker accuracy across batch sizes, reinforcing that overly aggressive sparsification reduces learning capacity. The variability across batch sizes and thresholds in MedMNIST emphasizes that optimal threshold selection may be dataset-specific and should consider data complexity and distribution.

Overall, these results highlight the importance of selecting STC thresholds carefully to balance communication efficiency with model accuracy. Lower thresholds (e.g., 1.0, 1.2) generally favor accuracy at the cost of higher communication volume, while higher thresholds reduce communication but risk learning degradation, especially at larger batch sizes. Tailoring thresholds to dataset properties and batch size can thus enhance FL performance. The detailed numerical values corresponding to Figures 5 and 6 are provided in Table 4, summarizing the maximum accuracy achieved for each threshold and batch size combination.

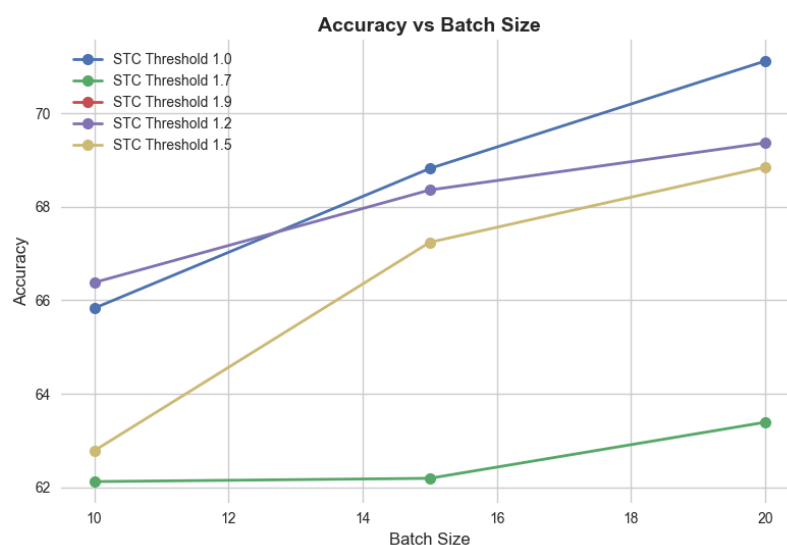


Figure 5. CIFAR-10: max accuracy vs. batch size for different STC thresholds

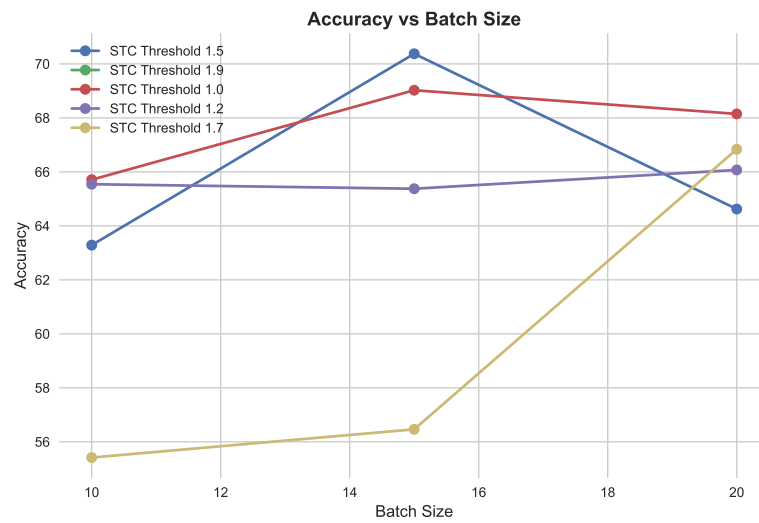


Figure 6. MedMNIST: max accuracy vs. batch size for different STC thresholds

Table 4. Max accuracy (%) vs. batch size for different STC thresholds on CIFAR-10 and MedMNIST

Batch size	CIFAR-10				MedMNIST			
	1.0	1.2	1.5	1.7	1.0	1.2	1.5	1.7
10	65.83	66.38	62.78	62.12	65.71	65.54	63.29	55.42
15	68.82	68.36	67.24	62.19	69.03	65.38	70.38	56.46
20	71.12	69.37	68.85	63.39	68.15	66.07	64.62	66.84

4.4. Summary heatmaps of accuracy and communication cost

To complement the detailed analyses presented in the previous subsections, Figures 7 and 8 provide heatmap visualizations summarizing the combined effects of STC threshold and batch size on both maximum accuracy and total communication cost. For the CIFAR-10 dataset (Figure 7), Figure 7(a) shows that lower STC thresholds (1.0, 1.2) combined with larger batch sizes (20) achieve the highest maximum accuracy (exceeding 71%), while Figure 7(b) reveals a clear reduction in communication cost as the threshold increases, confirming the trade-off between model performance and transmission efficiency.

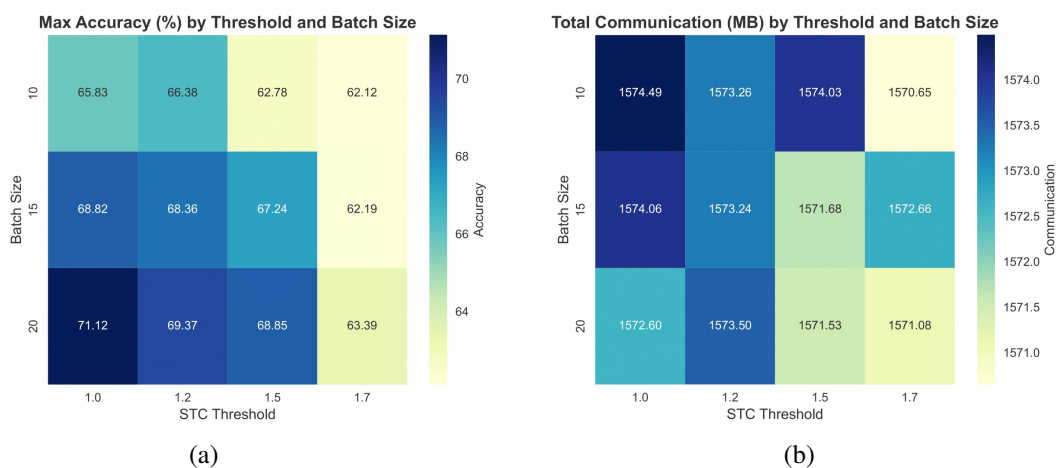


Figure 7. CIFAR-10 Heatmaps: (a) max accuracy (%) by STC threshold and batch size and (b) total communication (MB) by STC threshold and batch size

For MedMNIST (Figure 8), a similar pattern is observed, though with notable variations across

thresholds and batch sizes as presented in Figures 8(a) and 8(b). The accuracy heatmap highlights that threshold 1.5 at batch size 15 yields strong accuracy, while communication cost remains consistently sensitive to threshold adjustments. These visual summaries underscore the importance of careful threshold tuning tailored to both the data distribution and the desired balance between accuracy and communication savings in FL deployments.

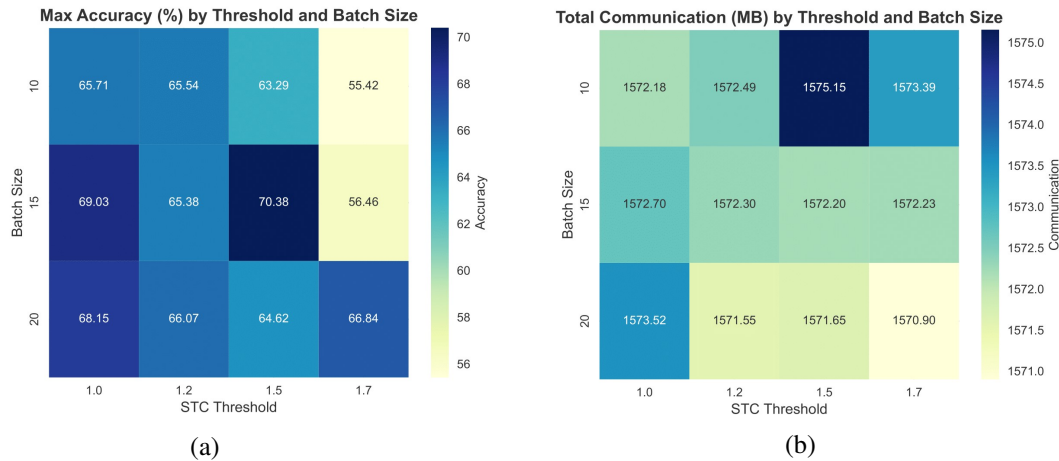


Figure 8. MedMNIST Heatmaps: (a) max accuracy (%) by STC threshold and batch size and (b) total communication (MB) by STC threshold and batch size

These findings provide a comprehensive understanding of the role of STC threshold selection in FL across different datasets and batch sizes. The results demonstrate a clear trade-off between communication efficiency and model accuracy that varies with both dataset characteristics and training configuration. Higher STC thresholds generally reduced total communication costs in both CIFAR-10 and MedMNIST experiments, confirming their effectiveness for minimizing data transmission overhead. However, this benefit came at the cost of accuracy, particularly evident at larger batch sizes and for thresholds above 1.5, where excessive sparsification degraded learning performance. In the accuracy vs. iterations analysis, lower thresholds (1.0, 1.2) consistently supported smoother and higher convergence, while higher thresholds failed to maintain learning stability across training rounds.

Similarly, in maximum accuracy vs. batch size evaluations, CIFAR-10 results favored lower thresholds for achieving the highest accuracy, especially at larger batch sizes. MedMNIST showed more varied behavior, with threshold 1.5 performing competitively at mid-sized batches but still revealing accuracy losses at higher thresholds like 1.7 and 1.9. Overall, these results highlight the importance of carefully selecting and tuning STC thresholds based on dataset complexity and batch size to balance communication efficiency with model fidelity in FL systems.

4.5. Comparative analysis

The study extends the foundational work of Sattler *et al.* [15] by thoroughly investigating STC thresholds beyond the baseline value ($\tau = 1.0$). Our experimental results, as detailed in Figures 1-6 and Tables 3-4, reveal distinct advantages of our proposed thresholds ($\tau = 1.2$, $\tau = 1.5$, and $\tau = 1.7$) over the base paper's $\tau = 1.0$. We consistently observe reduced communication overhead, with significant savings (e.g., up to ~ 2.5 MB), while simultaneously achieving competitive or superior accuracy. For instance, $\tau = 1.5$ frequently outperforms $\tau = 1.0$ in MedMNIST at mid-sized batches, demonstrating that careful STC tuning can yield better communication-accuracy trade-offs without significant performance compromise.

Beyond STC-specific comparisons, our findings contribute to the broader field of communication-efficient FL. While methods like QSGD [16], signSGD [17], and FedZip [23] offer various compression mechanisms, our lightweight threshold tuning strategy for STC stands out. It provides dataset-adaptive compression that avoids the more complex server-side decoding or architectural changes often associated with other techniques.

5. CONCLUSION

This study investigated the effect of varying STC thresholds specifically $\tau = 1.2, 1.5, 1.7$, and 1.9 —on communication efficiency and model accuracy in FL using CIFAR-10 and MedMNIST datasets. The experimental results demonstrate that careful threshold tuning plays a vital role in balancing communication cost with model performance. Threshold $\tau = 1.2$ consistently achieved the best trade-off by offering high accuracy with reduced communication, while $\tau = 1.5$ performed particularly well on MedMNIST with mid-sized batches. Though thresholds $\tau = 1.7$ and 1.9 resulted in higher compression, they showed reduced convergence quality at larger batch sizes, indicating potential over-sparsification. Overall, our findings emphasize the need to explore thresholds beyond the conventional baseline ($\tau = 1.0$), as moderate values (1.2 – 1.5) can provide meaningful improvements without compromising accuracy. Future research may extend this work by quantifying energy consumption, evaluating computational overhead, and designing adaptive thresholding mechanisms that dynamically adjust to varying data distributions and training conditions in heterogeneous FL environments.

FUNDING INFORMATION

The authors declare that no funding was received for conducting this study.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Nithyaniranjana Murthy Chittaiiah	✓	✓	✓	✓	✓	✓		✓	✓	✓			✓	
Manjula Sunkadakatte Haladappa		✓				✓				✓	✓	✓		

C : Conceptualization

M : Methodology

So : Software

Va : Validation

Fo : Formal Analysis

I : Investigation

R : Resources

D : Data Curation

O : Writing - Original Draft

E : Writing - Review & Editing

Vi : Visualization

Su : Supervision

P : Project Administration

Fu : Funding Acquisition

CONFLICT OF INTEREST STATEMENT

The authors state no conflict of interest.

DATA AVAILABILITY

The supporting data of this study are openly available in the following sources: CIFAR-10: <https://www.cs.toronto.edu/~kriz/cifar.html>, and MedMNIST: <https://medmnist.com/>.





REFERENCES

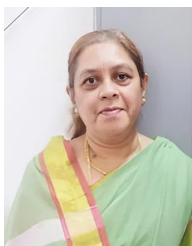
- [1] J. Chen, H. Yan, Z. Liu, M. Zhang, H. Xiong, and S. Yu, "When federated learning meets privacy-preserving computation," *ACM Computing Surveys*, vol. 56, no. 12, 2024, doi: 10.1145/3679013.
- [2] M. Liu, H. Jiang, J. Chen, A. Badokhon, X. Wei, and M. C. Huang, "A collaborative privacy-preserving deep learning system in distributed mobile environment," *Proceedings - 2016 International Conference on Computational Science and Computational Intelligence (CSCI)*, pp. 192–197, 2017, doi: 10.1109/CSCI.2016.0043.
- [3] Y. Liu, J. J. Q. Yu, J. Kang, D. Niyato, and S. Zhang, "Privacy-preserving traffic flow prediction: a federated learning approach," *IEEE Internet of Things Journal*, vol. 7, no. 8, pp. 7751–7763, 2020, doi: 10.1109/IIOT.2020.2991401.
- [4] J. Xu, B. S. Glicksberg, C. Su, P. Walker, J. Bian, and F. Wang, "Federated learning for healthcare informatics," *Journal of Healthcare Informatics Research*, vol. 5, no. 1, 2021, doi: 10.1007/s41666-020-00082-4.
- [5] K. Yang, T. Jiang, Y. Shi, and Z. Ding, "Federated learning via over-the-air computation," *IEEE Transactions on Wireless Communications*, vol. 19, no. 3, pp. 2022–2035, 2020, doi: 10.1109/TWC.2019.2961673.
- [6] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-IID data," *arXiv:1806.00582v2*, 2018.
- [7] Y. Liu et al., "A communication efficient vertical federated learning framework," *The 2nd International Workshop on Federated Learning for Data Privacy and Confidentiality (in Conjunction with NeurIPS 2019)*, Vancouver, Canada, 2019.





- [8] S. J. Reddi *et al.*, “Adaptive federated optimization,” *International Conference on Learning Representations (ICLR)*, Vienna, Austria, 2021.
- [9] V. Mothukuri, P. Khare, R. M. Parizi, S. Pouriyeh, A. Dehghantanha, and G. Srivastava, “Federated-learning-based anomaly detection for IoT security attacks,” *IEEE Internet of Things Journal*, vol. 9, no. 4, pp. 2545–2554, 2022, doi: 10.1109/JIOT.2021.3077803.
- [10] Y. Zhao *et al.*, “Mobile edge computing, blockchain, and reputation-based crowdsourcing IoT federated learning: a secure, decentralized, and privacy-preserving system,” *IEEE Internet of Things Journal*, vol. 8, no. 3, pp. 1817–1829, 2019.
- [11] R. Yu and P. Li, “Toward resource-efficient federated learning in mobile edge computing,” *IEEE Network*, vol. 35, no. 1, pp. 148–155, 2021, doi: 10.1109/MNET.011.2000295.
- [12] N. Truong, K. Sun, S. Wang, F. Guitton, and Y. K. Guo, “Privacy preservation in federated learning: an insightful survey from the GDPR perspective,” *Computers and Security*, vol. 110, 2021, doi: 10.1016/j.cose.2021.102402.
- [13] J. Kang, Z. Xiong, D. Niyato, Y. Zou, Y. Zhang, and M. Guizani, “Reliable federated learning for mobile networks,” *IEEE Wireless Communications*, vol. 27, no. 2, pp. 72–80, 2020, doi: 10.1109/MWC.001.1900119.
- [14] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, “Communication-efficient learning of deep networks from decentralized data,” *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, Florida, USA, vol. 60, no. 309, 2016.
- [15] F. Sattler, S. Wiedemann, K.-R. Müller, and W. Samek, “Robust and communication-efficient federated learning from non-IID data,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 9, pp. 3400–3413, 2020, doi: 10.1109/TNNLS.2019.2944481.
- [16] D. Alistarh, D. Grubic, J. Z. Li, R. Tomioka, and M. Vojnovic, “QSGD: randomized quantization for communication-optimal stochastic gradient descent,” *arXiv:1610.02132v4*, 2017.
- [17] J. Bernstein, Y.-X. Wang, K. Azizzadenesheli, and A. Anandkumar, “signSGD: compressed optimization for non-convex problems,” *Proceedings of the 35th International Conference on Machine Learning, Stockholm, Sweden*, 2018.
- [18] D. Rothchild *et al.*, “Fetchsgd: communication-efficient federated learning with sketching,” *37th International Conference on Machine Learning (ICML)*, pp. 8223–8235, 2020.
- [19] J. Wang *et al.*, “A field guide to federated optimization,” *arXiv:2107.06917v1*, 2021.
- [20] K. Yang, T. Jiang, Y. Shi, and Z. Ding, “Federated learning via over-the-air computation,” *IEEE Transactions on Wireless Communications*, vol. 19, no. 3, pp. 2022–2035, 2020, doi: 10.1109/TWC.2019.2961673.
- [21] M. Chen, Z. Yang, W. Saad, C. Yin, H. V. Poor, and S. Cui, “A joint learning and communications framework for federated learning over wireless networks,” *IEEE Transactions on Wireless Communications*, vol. 20, no. 1, pp. 269–283, 2021, doi: 10.1109/TWC.2020.3024629.
- [22] K. Cheng *et al.*, “SecureBoost: a lossless federated learning framework,” *IEEE Intelligent Systems*, vol. 36, no. 6, pp. 87–98, 2021, doi: 10.1109/MIS.2021.3082561.
- [23] A. Malekijoo, M. J. Fadaeieslam, H. Malekijou, M. Homayounfar, F. A.- Shabdiz, and R. Rawassizadeh, “FedZip: a compression framework for communication-efficient federated learning,” *arXiv:2102.01593v1*, 2021.
- [24] F. Haddadpour, M. M. Kamani, A. Mokhtari, and M. Mahdavi, “Federated learning with compression: unified analysis and sharp guarantees,” *Proceedings of the 24th International Conference on Artificial Intelligence and Statistics (AISTATS)*, California, USA, vol. 130, pp. 2350–2358, 2021.
- [25] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *3rd International Conference on Learning Representations (ICLR)*, pp. 1–14, 2015.

BIOGRAPHIES OF AUTHORS



Nithyaniranjana Murthy Chittiah     is a research scholar at University Visvesvaraya College of Engineering, Bangalore University, Bengaluru, with a Master’s degree in Information Technology from Visvesvaraya Technological University, Karnataka. Possesses extensive industry experience, having worked as a data engineer and machine learning engineer across various domains. He can be contacted at email: nithya.semantic@gmail.com.



Dr. Manjula Sunkadakatte Haladappa     currently working as a professor in the Department of Computer Science and Engineering at the University Visvesvaraya College of Engineering, Bangalore University, Bengaluru, she has established herself as a prominent figure in the field. Holding a B.E., M.Tech., and Ph.D. in Computer Science and Engineering, she has honed her expertise in computer networks, wireless sensor networks, data mining, cloud computing, artificial intelligence, machine learning, and federated learning. With an impressive track record, she has authored 69 journals, presented 72 conference papers, holds 5 patents, and has contributed to 4 book chapters while publishing 4 books, showcasing her prolific and diverse contributions to the academic and technological community. Additionally, she is currently a respected member of the Executive Council and has served as a former member of the Academic Senate at VTU Belagavi. She can be contacted at email: shmanjula@gmail.com.