Reducing Forgetting in Federated Learning with Truncated Cross-Entropy

Gwen Legate
University of Montreal, Mila
gwendolyne.legate@umontreal.ca

Lucas Caccia
McGill University, Mila
lucas.page-caccia@mail.mcgill.ca

Eugene Belilovsky
Concordia University, Mila
eugene.belilovsky@concordia.ca

Abstract

In federated learning (FL), a global model is learned by aggregating model updates computed from a set of client nodes, each having their own data. A key challenge in FL is the heterogeneity of data across clients whose data distributions differ from one another. Standard FL algorithms perform multiple gradient steps before synchronizing the model, which can lead to clients overly minimizing their local objective and diverging from other client solutions. We demonstrate that in such a setting individual client models experience “catastrophic forgetting" with respect to other client data. We propose a simple yet efficient approach that modifies the cross-entropy objective on a per-client basis such that classes outside a client’s label set are shielded from abrupt representation change. Through empirical evaluations, we demonstrate our approach can alleviate this problem, especially under the most challenging FL settings with high heterogeneity, low client participation.

1 Introduction

federated learning (FL) is a distributed machine learning paradigm in which a shared global model is learned from a decentralized set of data located at a number of independent client nodes [1, 2]. Due to communication constraints, FL algorithms typically perform many local gradient update steps before synchronizing with the global model. FL was conceptualized as a learning technique to train a shared model without sharing user sensitive data but allowing users to benefit from data stored at other nodes, such as phones and tablets of decentralized users. Under realistic settings, client data will often have non-iid distributions and for the case of supervised multi-class classification, users may frequently be missing data from an entire class or multiple classes. The data inhomogeneity across clients frequently induces client drift, a phenomenon in which clients progress too far towards optimizing their own local objective, leading to a solution that has severely "drifted" from an optimal global solution [3]. Continual learning (CL) [4] is another emerging paradigm in which a learner is presented with a sequence of tasks to be learned in succession. Similar to how FL clients may have non-iid data, different tasks in CL also typically contain data drawn from different distributions. In CL the problem of catastrophic forgetting is a focus of study since it is desirable for a model to retain its knowledge of previous tasks after learning a new one. We consider one round of FL in which C random clients are selected and sent a copy of the current global model and each client performs a number of local update steps to optimize the objective using their local data. The round ends with an update to the global model achieved by the aggregating the updates from each client. As local training proceeds for a typical round, the client model becomes increasingly biased towards that clients dataset and will experience a catastrophic forgetting with respect to the data from other clients.

Workshop on Distribution Shifts, 36th Conference on Neural Information Processing Systems (NeurIPS 2022).

In this way we can draw a connection between the catastrophic forgetting problem of CL and client drift problem of FL. We denote this problem as *local client forgetting*.

Reducing client forgetting would moderate an individual clients increase in loss with respect to other clients data, improving the loss of the client model over the combined data. Therefore we conclude that tackling local client forgetting can reduce client drift. Many methods to control catastrophic forgetting have been proposed in the CL literature [5–9] but they are largely impractical for FL. Experience Replay methods [7] require access to other clients data violating FL's data communication constraints. Many regularization methods such as EWC require communicating additional information and moreover typically require many steps to converge [10] due to additional conflicting objectives. For the supervised CL setting [11, 12] propose a modification of the standard cross entropy (CE) objective function that truncates the softmax denominator, removing terms corresponding to classes from old tasks thereby by reducing the bias on the model to avoid predicting old classes. In FL, as local client optimization proceeds, optimizing the terms in the CE corresponding to classes not present in that client's data distribution will encourage absent classes to be forgotten by the local model. Furthermore, as discussed in [13], it can cause spurious features to emerge. Inspired by these observations and the parallels between FL and CL we adapt the solution of [11, 12] to FL by correcting each clients loss function based on its class distribution. We show this approach can greatly reduce client level forgetting in the heterogeneous setting and lead to substantially improved overall global model convergence. It also yield optimization more robust to hyper-parameters and normalization layers.

2 Background and Methods

In FL [2] training data is distributed and optimization occurs over K clients with each client $k \in 1, ..., K$ having data $\mathbf{X}_k$ drawn from distribution D_k. We define $n_k = |\mathbf{X}_k|$ and $n = \sum_{k=1}^{K} n_k$ for n samples. The data $\mathbf{X}_k$ at each node may be drawn from different distributions and/or may be unbalanced with some clients possessing more training samples than others. The typical objective function for federated optimization is given by

$$\min_{\mathbf{w} \in \mathbb{R}^d} \sum_{k=1}^{K} \frac{n_k}{n} \mathcal{L}(\mathbf{w}, \mathbf{X}_k), \tag{1}$$

with $\mathcal{L}(\mathbf{w}, \mathbf{X}_k)$ measuring client k's local objective, and $\mathbf{w}$ representing the global model parameters. In this work we will restrict ourselves to the common case where $\mathcal{L}$ is the CE loss. There are many possible variations of FL algorithms. In general, they follow the same structure as FedAvg [1]: (a) **client selection:** for a set of K clients, $K * C$ are selected at each round $\{t_i\}_{i=1}^{T}$, where $0 < C \leq 1$ is a pre-determined fraction. (b) **client updates:** At the beginning of round, client models are initialized with the current weights of the server model. Each client selected for the round performs E local iterations of SGD. (c) **server update:** The weights of the individual client models are aggregated to form an update to the shared global model.

Truncated Cross Entropy Consider a neural network $f : \mathcal{R}^D \rightarrow \mathcal{R}^C$ where C is the total number of classes. The standard CE is given by equation 2

$$\mathcal{L}_{CE}(\mathbf{X}_k, \mathbf{Y}_k, \mathbf{w}) = -\sum_{\mathbf{x} \in \mathbf{X}} \log \frac{\exp(f_{\mathbf{w}}(\mathbf{x})_{y(\mathbf{x})})}{\sum_{c \in \mathcal{C}} \exp(f_{\mathbf{w}}(\mathbf{x})_c)}, \tag{2}$$

Here $y(\mathbf{x})$ is the label of $\mathbf{x}$ and $\mathcal{C}$ is the set of all classes available to the clients. We now consider the truncated cross entropy (TCE) characterized by equation 3 where the denominator is a function of the labels for the client data $\mathbf{Y}_k$.

$$\mathcal{L}_{TCE}(\mathbf{X}_k, \mathbf{Y}_k, \mathbf{w}) = -\sum_{\mathbf{x} \in \mathbf{X}} \log \frac{\exp(f_{\mathbf{w}}(\mathbf{x})_{y(\mathbf{x})})}{\sum_{c \in \mathcal{C}(\mathbf{Y}_k)} \exp(f_{\mathbf{w}}(\mathbf{x})_c)}, \tag{3}$$

In our work we consider $\mathcal{C}(\mathbf{Y}_k)$ to be the set unique labels for the client. Aggressively optimizing $\mathcal{L}_{CE}(\mathbf{X}_k, \mathbf{Y}_k)$ through multiple gradient steps during a client round can lead to a drastic increase in $\mathbb{E}_{x,y \sim D_{j \neq k}}[l_{CE}(\mathbf{x}, \mathbf{y})]$ where D_j are the distribution of clients other than client k. The TCE approach, modifies the original local objective function to avoid excessive pressure that drives up the loss of other client data so that classes not present at the current client are ignored by the local optimization.

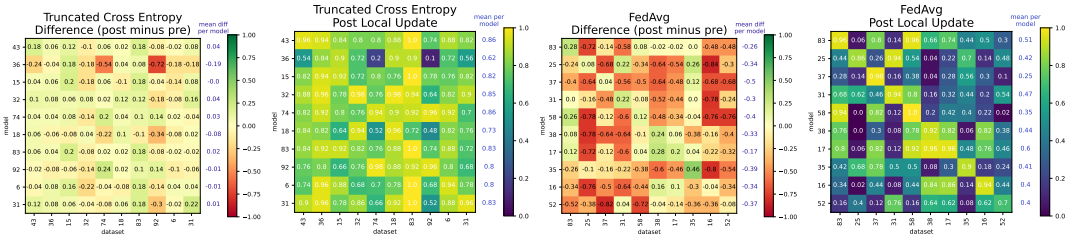

Figure 1: Prior to a local training round, each client model is an identical copy of the server model, after local training, local models have diverged according to their own objectives. We show each local model evaluated on each dataset of the clients selected for the round in question, the x, y axes contain the indices of the clients. The difference heat maps indicate F_{ik}, the forgetting of the model of the k^{th} client evaluated on the i^{th} client's data. The final column gives F_k, the average forgetting over all clients. The post update heat maps show the accuracy of each client's model on the other client's data. We see that TCE (First two heat maps) significantly reduces forgetting across clients.

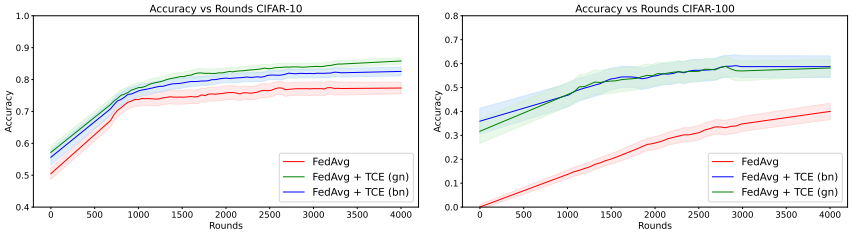

Figure 2: We show the performance over rounds in a highly heterogenous setting with FedAvg, FedAvg+TCE. We observe that TCE consistently gives the best performance.

This forces each client to learn by adapting the model's internal representation of the classes present in its training data, rather than abruptly shifting representations of classes outside its training set [11]. In what follows we will empirically demonstrate this leads to a reduction in the notion of forgetting defined below.

Local client forgetting We formalize the notion of local client forgetting discussed in Sec 1 for a classification problem. Denoting the accuracy on a client k's local test data $Acc_k(\mathbf{w})$, with $\mathbf{w}$ the model parameters. We can consider the local client forgetting $F_{ki} = Acc_k(\mathbf{w}_t^i) - Acc_k(\mathbf{w}_{t-1})$. Here $\mathbf{w}_t^i$ refers to the model of client i at round t (before the aggregation step) and $\mathbf{w}_{t-1}$ the global model at the end of round $t-1$. Furthermore, we can define an average forgetting for a client k's model $F_k = \frac{1}{K-1}\sum_{i \neq k} F_{ki}$. In the sequel we will study these quantities for a standard FL setting.

3 Experiments

In this section, we present the empirical results for the TCE framework. We analyze the notion of forgetting in the context of FedAvg and then show how the application of the TCE to the FedAvg objective can help resolve local client forgetting. Subsequently, we study how TCE can enhance standard FL algorithms like FedAvg, improving their overall performance over a range of key hyperparameters.

Datasets and Setting We utilize CIFAR-10, CIFAR-100 [14] and FEMNIST [15] datasets in our experiments, each of these datasets come pre-separated into training and testing sets. Our primary evaluations consider 100 clients where each client requires their own training and validation sets according to their own unique distribution. To facilitate this, the entire training set is separated into equally sized non-iid partitions using the Dirichlet distribution parameterized by $\alpha = 0.1$, similar to the method of [16]. These client partitions are then further separated into training (90%) and validation (10%) sets for each client. For example, 100 clients being trained using CIFAR-10 which contains 50 000 training samples would each have 500 of these training samples. Of those 500 samples, 450 would be used for local model updates and 50 would be used exclusively for validation.

Validation Throughout the training process the global model is evaluated periodically on the aggregation of client validation sets to gauge overall training progress. A model is evaluated on the training set only once, after the completion of the entire 4000 rounds of training. This value is the accuracy reported in the Table 1. Since we focus our analysis on the highly heterogeneous case $\alpha = 0.1$, this can lead to higher variance in the validation statistics [16], particularly for smaller

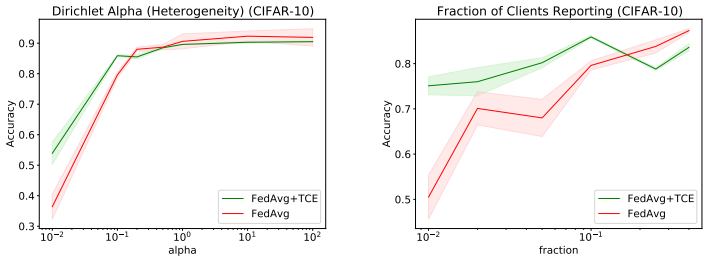

Figure 3: Ablations for (a) data heterogeniety (b) number of participating clients. We observe that (a) TCE gives improvements in cases where data is highly heterogenous (b) TCE is most useful with low number of partici

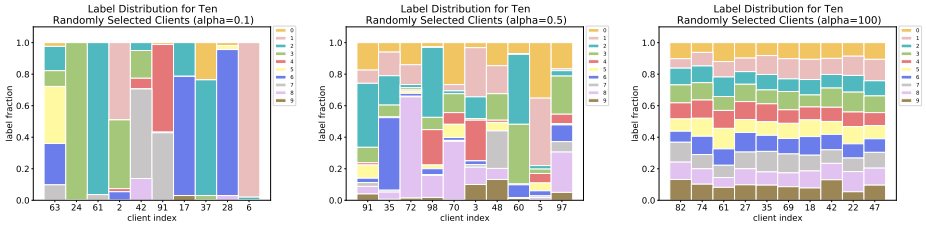

Figure 4: Percentages of each class label for ten randomly selected clients with $\alpha = 0.1, 0.5, 100$ from left to right

datasets such as CIFAR-10 and CIFAR-100. To combat this effect, we run each training three times to reduce variance of the final reported results.

3.1 Forgetting During a Federated Round

We study the local client forgetting at different optimization rounds of FL. In Figure 1 we show F_{ki} (the heatmap) and F_k (the last column) for the set of participating clients for the respective round for FedAvg using both TCE and CE. We observe forgetting is very high when using standard CE but TCE is able to control local client forgetting leading to better performance of the global model post aggregation. The results in Figure 1 are shown for round 2800 of training however, the observation persists for other rounds as shown in the Appendix Sec A.3. Having observed that TCE can indeed reduce the local client forgetting we now study its effect on the aggregated models.

3.2 Evaluations on CIFAR-10,CIFAR-100 and FEMNIST

Figure 2 shows the best performing models among FedAvg and FedAvg+TCE for CIFAR-10 and CIFAR-100. From these results, we observe FedAvg+TCE is robust to the normalization layer unlike vanilla FL methods which typically do poorly when using BN. In general we conclude that for both CIFAR-10 and CIFAR-100, TCE substantially improves performance for any training budget. We see a similar performance increase when using the FEMNIST dataset (Table 1) further supporting our claims.

Robustness to learning rate and normalization In the Appendix Table 1 we summarize a detailed analysis of model performance over a range of learning rates. Additionally, we vary the normalization method between BN and GN for TCE. With regard to the normalization method, we observe comparable performance using either BN or GN indicating TCE helps to mitigate performance degradation typically observed when using BN in heterogeneous FL [17]. In general, we observe that across all client learning rates TCE provides performance improvement using either normalization setting. We particularly notice improvements by TCE even at higher learning rates, where vanilla FedAvg can collapse or under-perform. We also remark that not only is overall performance higher for FedAvg+TCE but it's performance deviates less from the case of it's best performing hyperparameters as various settings are changed. This analysis is further supported by Table 2 in the Appendix.

3.3 Ablations

We further study the behavior of TCE in combination with FedAvg under different data distributions and client participation settings. In Figure 3 ablations for the CIFAR-10 setting are shown where we ablate one setting at a time. We observe particular settings under which TCE provides vast improvement over vanilla FedAvg.

Parameter α of the Dirichlet Distribution The Dirichlet distribution is parameterized by α as $\alpha \to 0$ the client distributions will become increasingly heterogeneous and as $\alpha \to \infty$, each client data edges closer towards the same i.i.d distribution. In our base case we use $\alpha = 0.1$ and we now investigate how model performance is affected by changing the heterogenaity of the data partitions. Similar to the work of [16], we investigate $\alpha = \{0.01, 0.1, 0.2, 0.5, 1, 10, 100\}$ Figure 4 offers a practical illustration of how client partitions change as a function of α. From Figure 3 we observe TCE has a more significant effect as α decreases with the biggest margin of improvement over vanilla FedAvg occurring when $\alpha = 0.01$. As the data becomes increasingly homogeneous, the gap between cross entropy with and without TCE shrinks until the data is iid and the two methods are equivalent. From Figure 4 we see that by the time $\alpha = 0.5$ most of the clients possess all of the classes, albeit in very skewed proportions. This is also the point at which the performance of FedAvg and FedAvg+TCE become very close, from this observation, we conclude TCE is most advantageous when clients do not possess all possible class labels.

Fraction of Participating Clients For the fraction of participating clients C, we observe the largest performance gap between FedAvg+TCE and FedAvg when the number of participating clients is low. This result is significant since it mimics potential real world settings in which communication constraints can impede a user's ability to send its update to the central server. We hypothesize the performance gap as a function of client fraction between the FedAvg and FedAvg+TCE is due to the larger impact of local client forgetting when we limit communication capability. Unless we actively take steps to control forgetting, non-participating clients will have their data distributions forgotten because unlike participating clients, they will be unable to contribute their updates to the global model. As the fraction of clients selected at each round increases, we observe the performance gap between the two methods narrow since more clients will have the opportunity to be selected at each round and "remind" the model of their data distributions.

4 Conclusion and Future Directions

In this paper we took a deeper look at the *local client forgetting* problem, we show empirically that when a client performs local updates during FL, it risks over optimizing its local objective, which can lead to forgetting on other subsets of data which degrades performance of the global model. We demonstrate this phenomenon is especially severe in cases where there is a significant distribution mismatch across clients. We proposed a client level modification of the objective function which we call TCE, that allows us to mitigate client level forgetting. We demonstrate our method leads to improved performance when combined with a standard FL algorithm across a range of learning rates. particularly in the regime of highly heterogeneous client datasets and/or when a small percentage of clients are selected at each round. We also demonstrate TCE is more robust to a wide range of hyperparameter settings than vanilla FedAvg. In on-going and future work we are investigating soft versions of the truncation operation, which can effectively deal with cases where classes are present but in very low quantities.

5 Acknowledgements

This research was partially funded by NSERC Discovery Grant RGPIN-2021-04104. We also would like to acknowledge the support from Digital Research Alliance and Calcul Quebec.

References

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

[2] Jakub Konečnỳ, H Brendan McMahan, Daniel Ramage, and Peter Richtárik. Federated optimization: Distributed machine learning for on-device intelligence. *arXiv preprint arXiv:1610.02527*, 2016.

[3] Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. Scaffold: Stochastic controlled averaging for federated learning. In *International Conference on Machine Learning*, pages 5132–5143. PMLR, 2020.

[4] Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier, 1989.

[5] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

[6] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017.

[7] Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K Dokania, Philip HS Torr, and Marc'Aurelio Ranzato. On tiny episodic memories in continual learning. *arXiv preprint arXiv:1902.10486*, 2019.

[8] Jonathan Schwarz, Wojciech Czarnecki, Jelena Luketina, Agnieszka Grabska-Barwinska, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. Progress & compress: A scalable framework for continual learning. In *International Conference on Machine Learning*, pages 4528–4537. PMLR, 2018.

[9] MohammadReza Davari, Nader Asadi, Sudhir Mudur, Rahaf Aljundi, and Eugene Belilovsky. Probing representation forgetting in supervised and unsupervised continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16712–16721, 2022.

[10] Rahaf Aljundi, Eugene Belilovsky, Tinne Tuytelaars, Laurent Charlin, Massimo Caccia, Min Lin, and Lucas Page-Caccia. Online continual learning with maximal interfered retrieval. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL `https://proceedings.neurips.cc/paper/2019/file/15825aee15eb335cc13f9b559f166ee8-Paper.pdf`.

[11] Lucas Caccia, Rahaf Aljundi, Nader Asadi, Tinne Tuytelaars, Joelle Pineau, and Eugene Belilovsky. New insights on reducing abrupt representation change in online continual learning. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=N8MaByOzUfb`.

[12] Hongjoon Ahn, Jihwan Kwak, Subin Lim, Hyeonsu Bang, Hyojun Kim, and Taesup Moon. Ss-il: Separated softmax for incremental learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 844–853, 2021.

[13] Timothée Lesort. Continual feature selection: Spurious features in continual learning. *arXiv preprint arXiv:2203.01012*, 2022.

[14] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical Report 0, University of Toronto, Toronto, Ontario, 2009.

[15] Sebastian Caldas, Sai Meher Karthik Duddu, Peter Wu, Tian Li, Jakub Konečnỳ, H Brendan McMahan, Virginia Smith, and Ameet Talwalkar. Leaf: A benchmark for federated settings. *arXiv preprint arXiv:1812.01097*, 2018.

[16] Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. Measuring the effects of non-identical data distribution for federated visual classification. *arXiv preprint arXiv:1909.06335*, 2019.

[17] Sashank Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečnỳ, Sanjiv Kumar, and H Brendan McMahan. Adaptive federated optimization. *arXiv preprint arXiv:2003.00295*, 2020.

[18] Sebastian U Stich. Local sgd converges fast and communicates little. *arXiv preprint arXiv:1805.09767*, 2018.

[19] Jianyu Wang and Gauri Joshi. Cooperative sgd: A unified framework for the design and analysis of communication-efficient sgd algorithms. *arXiv preprint arXiv:1808.07576*, 2018.

[20] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine Learning and Systems*, 2:429–450, 2020.

[21] Dezhong Yao, Wanning Pan, Yutong Dai, Yao Wan, Xiaofeng Ding, Hai Jin, Zheng Xu, and Lichao Sun. Local-global knowledge distillation in heterogeneous federated learning with non-iid data. *arXiv preprint arXiv:2107.00051*, 2021.

[22] Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. On the convergence of fedavg on non-iid data. *arXiv preprint arXiv:1907.02189*, 2019.

[23] Zhuangdi Zhu, Junyuan Hong, and Jiayu Zhou. Data-free knowledge distillation for heterogeneous federated learning. In *International Conference on Machine Learning*, pages 12878–12889. PMLR, 2021.

[24] Tao Lin, Lingjing Kong, Sebastian U Stich, and Martin Jaggi. Ensemble distillation for robust model fusion in federated learning. *Advances in Neural Information Processing Systems*, 33: 2351–2363, 2020.

[25] Irene Tenison, Sai Aravind Sreeramadas, Vaikkunth Mugunthan, Edouard Oyallon, Eugene Belilovsky, and Irina Rish. Gradient masked averaging for federated learning. *arXiv preprint arXiv:2201.11986*, 2022.

[26] Zhiyuan Chen and Bing Liu. Lifelong supervised learning. In *Lifelong Machine Learning*, pages 33–74. Springer, 2018.

[27] David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. *Advances in neural information processing systems*, 30, 2017.

[28] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[29] Jaehong Yoon, Wonyong Jeong, Giwoong Lee, Eunho Yang, and Sung Ju Hwang. Federated continual learning with weighted inter-client transfer. In *International Conference on Machine Learning*, pages 12073–12086. PMLR, 2021.

[30] Neta Shoham, Tomer Avidor, Aviv Keren, Nadav Israel, Daniel Benditkis, Liron Mor-Yosef, and Itai Zeitak. Overcoming forgetting in federated learning on non-iid data. *arXiv preprint arXiv:1910.07796*, 2019.

[31] Chencheng Xu, Zhiwei Hong, Minlie Huang, and Tao Jiang. Acceleration of federated learning with alleviated forgetting in local training. *arXiv preprint arXiv:2203.02645*, 2022.

[32] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015.

[33] Kevin Hsieh, Amar Phanishayee, Onur Mutlu, and Phillip Gibbons. The non-iid data quagmire of decentralized machine learning. In *International Conference on Machine Learning*, pages 4387–4398. PMLR, 2020.

[34] Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19, 2018.

[35] Enmao Diao, Jie Ding, and Vahid Tarokh. Heterofl: Computation and communication efficient federated learning for heterogeneous clients. *arXiv preprint arXiv:2010.01264*, 2020.

A Appendix

A.1 Related Work

Federated Learning The most commonly used baseline in FL is the FedAvg algorithm proposed by [1]. Communication efficiency in FL of upmost importance [2] and FedAvg reduces communication costs by allowing clients to train multiple iterations successively. Convergence of FedAvg has been widely studied for both i.i.d [18, 19] and non i.i.d settings [20]. Under non i.i.d settings, convergence deteriorates as a function of increasing heterogeneity [16]. [17] studied approaches for introducing adaptive algorithms into the server updates to accelerate FL algorithms.

Non-Heterogeneous Data Partitions and Client Drift One significant challenge of decentralized data is client sample heterogeneity. Different local distributions create differing local objectives, when clients progress too far towards minimizing their own objectives, local models drift apart. This degrades performance of the shared global model and slows convergence[21, 22, 3]. Attempts to alleviate client drift include approaches centered around knowledge distillation like [23] and [24] who ensemble information about the global data distribution and disseminate it to clients via models trained at the server. These methods possess the added risk of privacy attacks and while [23] take steps to mitigate this risk, their method requires the existence of an unlabeled dataset which may not be practical in all settings. Other approaches attempt to regularize updates at the client level, [3] propose SCAFFOLD, an algorithm to control client drift by use of control norms modifying client gradients. [20] add a proximal term to the local objective limiting variation in local updates and [25] propose a gradient masking technique that modifies server side update aggregation. Our method on the other hand, modifies the objective functions locally, this approach is compatible with any federated optimization method in the literature.

Continual Learning CL is a process by which tasks are learned sequentially over a period of time, the learner retains knowledge of previous tasks and leverages that prior knowledge to learn new tasks [26]. CL is made difficult by the fact that neural networks suffer from catastrophic forgetting, in which learning a new task overrides weights learned from past training, degrading performance on previously learned tasks [4]. Attempts to mitigate the catastrophic forgetting include architecture based approaches [8] which attempt to grow or modify an architecture over time to expand knowledge, regularization based methods which attempt to use regularization to solve the problem [5] and another class of methods which take the approach of storing some subset of old data for "rehearsal" [27, 7, 28]. A federated CL setting in which each client continuously collects data was proposed by [29], this differs from our work which considers the standard FL setting where each client maintains a fixed set of data and draws connections to a notion of forgetting across clients to motivate a modification of the loss function. [30] have used ideas from CL to propose FedCurv based on the EWC algorithm [5] from CL, FedCurv requires sending additional information and is not compatible with all FL methods. Along this line [31] proposed an approach inspired by rehearsal methods, generating pseudo data and adding an additional regularization term. This requires an expensive pseudo data generating procedure that increases local training time.

Normalization in FL Batch normalization (BN) is a commonly employed machine learning tactic that normalizes features by the mean and variance computed across a mini batch. It has been shown to help with generalization and stabilize optimization [32]. A limitation of BN is that under certain conditions, the mean and standard deviation used at test time may differ significantly from those used in training. Scenarios including small batch sizes or non-i.i.d batch distributions such as those in heterogeneous FL have been noted to suffer from significant performance degradation when using BN [17]. Group normalization (GN) is not dependent on batch statistics, and is effective at mitigating the effect of model performance degradation induced by skewed data partitions and small batch sizes [33, 34]. Recent works using FedAvg and any of its variants typically avoid using BN [35].

A.2 Evaluation of Hyperparameter Robustness

In Table A.2 we further confirm the robustness of FedAvg+TCE to learning rate. For each dataset used in the experiments we indicate accuracies of models trained using a different learning rate that fall within 2%, 3% and 5% of the best accuracy obtained in training. The best learning rate is indicated in column 2 for convenience.

Method	*Hyper-params* lr	norm	*Dataset* CIFAR-10	CIFAR-100	FEMNIST
FedAvg	0.7	group	NC	NC	NC
FedAvg+TCE (ours)			NC	NC	NC
FedAvg+TCE (ours)		batch	0.805 ± 0.039	0.350 ± 0.037	0.803
FedAvg	0.5	group	0.777 ± 0.033	0.223 ± 0.059	0.695
FedAvg+TCE (ours)			0.751 ± 0.039	0.382 ± 0.036	0.700
FedAvg+TCE (ours)		batch	0.832 ± 0.028	0.354 ± 0.019	0.831
FedAvg	0.3	group	0.77 ± 0.022	0.324 ± 0.148	0.801
FedAvg+TCE (ours)			0.836 ± 0.029	0.438 ± 0.041	0.787
FedAvg+TCE (ours)		batch	0.836 ± 0.010	0.309 ± 0.060	**0.850**
FedAvg	0.1	group	0.79 ± 0.017	0.365 ± 0.131	0.805
FedAvg+TCE (ours)			0.846 ± 0.010	0.451 ± 0.046	0.806
FedAvg+TCE (ours)		batch	0.841 ± 0.020	0.397 ± 0.044	**0.850**
FedAvg	0.07	group	0.751 ± 0.020	0.486 ± 0.036	0.805
FedAvg+TCE (ours)			0.832 ± 0.029	0.429 ± 0.024	0.774
FedAvg+TCE (ours)		batch	**0.859 ± 0.009**	0.478 ± 0.024	0.836
FedAvg	0.05	group	0.742 ± 0.087	0.456 ± 0.031	0.829
FedAvg+TCE (ours)			0.834 ± 0.011	0.484 ± 0.030	0.791
FedAvg+TCE (ours)		batch	0.852 ± 0.010	0.506 ± 0.008	0.830
FedAvg	0.03	group	0.787 ± 0.011	0.486 ± 0.013	0.735
FedAvg+TCE (ours)			**0.859 ± 0.004**	0.520 ± 0.006	0.732
FedAvg+TCE (ours)		batch	0.850 ± 0.018	0.513 ± 0.030	0.813
FedAvg	0.01	group	0.742 ± 0.065	0.429 ± 0.052	0.780
FedAvg+TCE (ours)			0.825 ± 0.004	**0.524 ± 0.010**	0.809
FedAvg+TCE (ours)		batch	0.845 ± 0.015	0.479 ± 0.015	0.791
FedAvg	0.007	group	0.739 ± 0.030	0.440 ± 0.041	0.780
FedAvg+TCE (ours)			0.751 ± 0.005	0.501 ± 0.018	0.814
FedAvg+TCE (ours)		batch	0.747 ± 0.013	**0.524 ± 0.010**	0.782

Table 1: Performance of FedAvg with and without TCE for different settings of client learning rates and normalization layer settings. We observe TCE consistently improves performance for many learning rate settings as well as having the highest overall accuracy by a large margin. Note results of FedAvg with BN are not included since this setting is known to yield poor performance [17], however, TCE combined with BN based models is competitive with group norm. NC indicates cases for which the algorithm did not converge

	best acc (best lr)	lr for acc within 2%	lr for acc within 3%	lr for acc within 5%
CIFAR-10				
FedAvg	0.796 (0.1)	0.1, 0.03	0.3, 0.1, 0.03	0.3, 0.1, 0.07, 0.03
FedAvg+TCE (group)	0.860 (0.03)	0.1, 0.03	0.3, 0.1, 0.07, 0.05, 0.03	0.3, 0.1, 0.07, 0.05, 0.03, 0.01
FedAvg+TCE (batch)	0.859 (0.07)	0.1, 0.07, 0.05, 0.01	0.3, 0.1, 0.07, 0.05, 0.01	0.3, 0.1, 0.07, 0.05, 0.01
CIFAR-100				
FedAvg	0.486 (0.07)	0.07	0.07, 0.03	0.07, 0.05, 0.03, 0.007
FedAvg+TCE (group)	0.524 (0.01)	0.01	0.05, 0.03, 0.01, 0.007	0.05, 0.03, 0.01, 0.007
FedAvg+TCE (batch)	0.524 (0.007)	0.05, 0.03, 0.007	0.05, 0.03, 0.007	0.07, 0.05, 0.03, 0.01, 0.007
FEMNIST				
FedAvg	0.850	0.05	0.05,0.07, 0.1, 0.3	0.007, 0.01, 0.05, 0.07, 0.1, 0.3
FedAvg+TCE (group)	0.814	0.007, 0.01, 0.1	0.007, 0.01,0.05, 0.07, 0.1, 0.3	0.007, 0.01, 0.05, 0.07, 0.1, 0.3
FedAvg+TCE (batch)	0.850	0.05, 0.07, 0.1, 0.3, 0.5	0.05, 0.07, 0.1, 0.3, 0.5	0.05, 0.07, 0.1, 0.3, 0.5, 0.7

Table 2: learning rates where accuracy is within a specified tolerance of the best accuracy. We observe that not only does TCE provide the best accuracy, this accuracy is less sensitive to hyperparameters

A.3 Additional Forgetting studies

In Figures 5, 6, 7 we show additional results for other rounds of training for forgetting. Figure 5 is evaluated after the first round of training, Figure 7 is evaluated after the 4000^{th} round of training and Figure 6 is evaluated right in the middle of training, after the 2000^{th} round. We observe in very early rounds since performance is still very low for many client datasets, there is not as much accuracy to destroy, however we still observe several cases where initial accuracy of the model is substantial enough that forgetting is observably more extreme without TCE. By the middle of training at round 2000, we see clear indications of forgetting, the bottom row of Figure 6 corresponding to FedAvg without TCE shows substantially better performance on its own dataset (indicated along the diagonal values). At the completion of training (Figure 7) we see FedAvg+TCE doing much better than FedAvg at overcoming the local client forgetting problem.

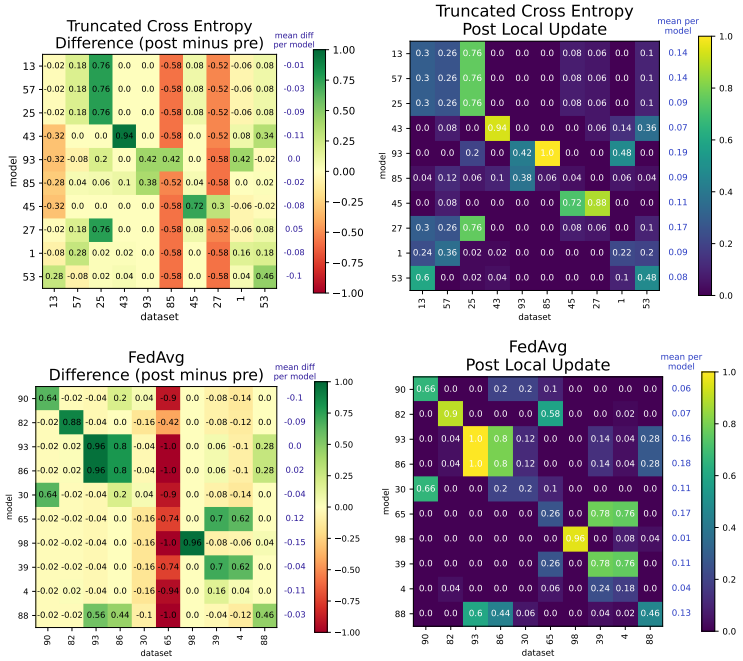

Figure 5: After round 1: The model of each client is evaluated on its own dataset and the datasets of each other client selected for the round both prior to training (right) and after training (center). The difference between the post and prior accuracies is presented on the left.

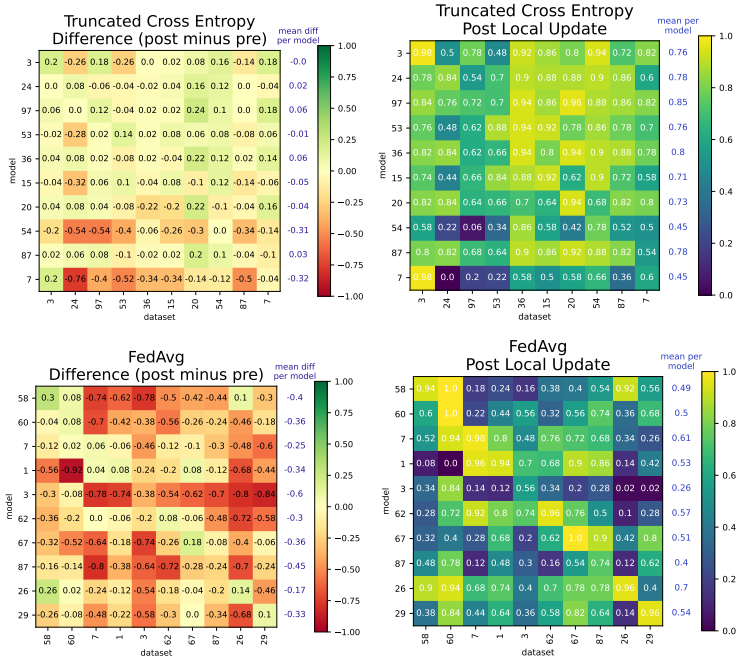

Figure 6: After round 2000: The model of each client is evaluated on its own dataset and the datasets of each other client selected for the round both prior to training (right) and after training (center). The difference between the post and prior accuracies is presented on the left.

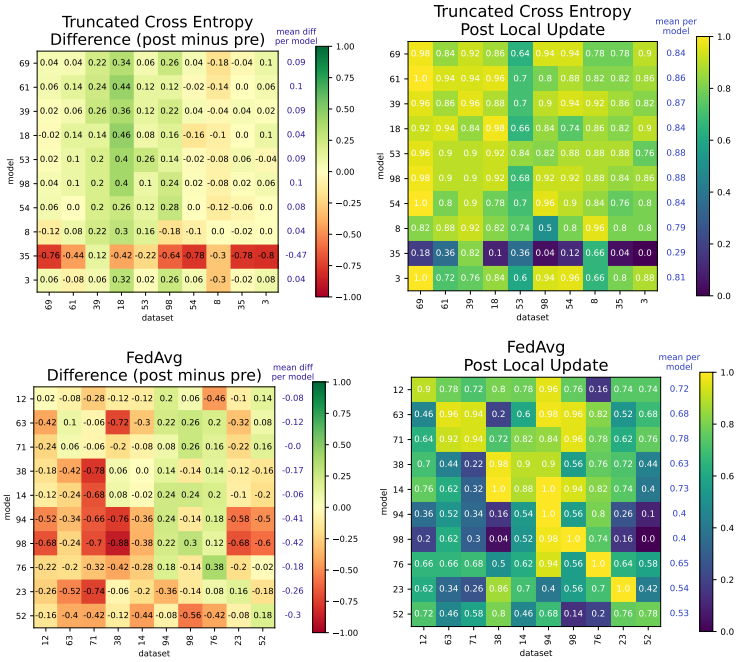

Figure 7: After round 4000: The model of each client is evaluated on its own dataset and the datasets of each other client selected for the round both prior to training (right) and after training (center). The difference between the post and prior accuracies is presented on the left.