# Overcoming Forgetting in Federated Learning on Non-IID Data

**Neta Shoham**
Edgify
neta.shoham@edgify.ai

**Tomer Avidor**
Edgify
tomer.avidor@edgify.ai

**Aviv Keren**
Edgify
aviv.keren@edgify.ai

**Nadav Israel**
Edgify
nadav.israel@edgify.ai

**Daniel Benditkis**
Edgify
daniel.benditkis@edgify.ai

**Liron Mor-Yosef**
Edgify
liron.moryosef@edgify.ai

**Itai Zeitak**
Edgify
itai.zeitak@edgify.ai

## Abstract

We tackle the problem of Federated Learning in the non i.i.d. case, in which local models drift apart, inhibiting learning. Building on an analogy with Lifelong Learning, we adapt a solution for catastrophic forgetting to Federated Learning. We add a penalty term to the loss function, compelling all local models to converge to a shared optimum. We show that this can be done efficiently for communication (adding no further privacy risks), scaling with the number of nodes in the distributed setting. Our experiments show that this method is superior to competing ones for image recognition on the MNIST dataset.

## 1 Introduction

Recent years have seen the advent of smart devices and sensors gathering data at the edge and being able to act on that data. The desire to keep data private and other considerations have led the machine learning community to study algorithms for distributed training that do not require sending the data out of the edge devices. Edge devices most often have low networking availability and capacity, which could prohibit training through standard SGD. The *Federated Averaging* (FedAvg) algorithm of McMahan et al. [1] lets the devices train on their local data for several epochs (using local SGD) before sending the trained model to a central server. The server then aggregates the models and sends the aggregated model back to the devices. This is done iteratively until convergence is achieved.

Federated Learning poses three challenges that make it different from traditional distributed learning. The first one is the number of computing stations, which can be in the hundreds of millions.[1] The second is much slower communication compared to the inter cluster communication found in data centers. The third difference, on which we focus in this work, is the highly non i.i.d. manner in which the data may be distributed among the devices.

In some real-life cases, Federated Learning has shown robustness to non i.i.d. distribution [2]. There are also recent theoretical results proving the convergence of Federated Learning algorithms [3] on

[1] In order to cope with this, it is common practice to select only a subset of devices at every training iteration [1]. For simplicity of presentation, we will ignore this method, for which our suggested algorithm can be easily adapted.

NeurIPS 2019 Workshop on Federated Learning for Data Privacy and Confidentiality, Vancouver, Canada.

有点在线学习的影子

non i.i.d. data. It is evident, however, that even in very simple scenarios, Federated Learning on non i.i.d. distributions has trouble achieving good results (in terms of accuracy and the number of communication rounds, as compared to the i.i.d. case) [1, 4].

### 1.1 Overcoming Forgetting in Sequential Lifelong Learning and in Federated Learning

There is a deep parallel between the Federated Learning problem and another fundamental machine learning problem called *Lifelong Learning* (and the related *Multi-Task Learning*). In Lifelong Learning, the challenge is to learn task $A$, and continue on to learn task $B$ using the same model, but without "forgetting", without severely hurting the performance on, task $A$; or in general, learning tasks $A_1, A_2 \ldots$ in sequence without forgetting previously-learnt tasks for which samples are not presented anymore. Besides learning tasks serially rather than in parallel, in Lifelong Learning each task is thus seen only once, whereas in Federated Learning there is no such limitation. But these differences aside, the paradigms share a common main challenge - how to learn a task without disturbing different ones learnt on the same model.

It is not surprising, then, that similar approaches are being applied to solve the Federated Learning and the Lifelong Learning problems. One such example is data distillation, in which representative data samples are shared between tasks [5, 6]. However, Federated Learning is frequently used in order to achieve data privacy, which would be broken by sending a piece of data from one device to another, or from one device to a central point. We therefore seek for some other type of information to be shared between the tasks.

The answer to what kind of information to use may be found in Kirkpatrick et al. [7]. In this work, the authors present a new algorithm for Lifelong Learning - *Elastic Weight Consolidation* (EWC). EWC aims to prevent catastrophic forgetting when moving from learning task $A$ to learning task $B$. The idea is to identify the coordinates in the network parameters $\theta$ that are the most informative for task $A$, and then, while task $B$ is being learned, penalize the learner for changing these parameters. The basic assumption is that deep neural networks are over-parameterized enough, so that there are good chances of finding an optimal solution $\theta^*_B$ to task $B$ in the neighborhood of previously learned $\theta^*_A$.

找到最富含另一个任务知识的参数，在对新任务训练的时候在这些参数上加惩罚项，减少知识的丢失

In order to control the stiffness of $\theta$ per coordinate while learning task $B$, the authors suggest to use the diagonal of the Fisher information matrix $\mathcal{I}^*_A = \mathcal{I}_A(\theta^*_A)$ to selectively penalize parts of the parameters vector $\theta$ that are getting too far from $\theta^*_A$. This is done using the following objective

是衡量观测所得的随机变量{X携带的关于未知参数theta的信息量，}X的概率分布依赖于参数θ 。

$$\tilde{L}(\theta) = L_B(\theta) + \lambda(\theta - \theta^*_A)^T \operatorname{diag}(\mathcal{I}^*_A)(\theta - \theta^*_A) \tag{1}$$

The formal justification they provide for (1) is Bayesian: Let $D_A$ and $D_B$ be independent datasets used for tasks $A$ and $B$. We have that

$$\log p(\theta | D_A \text{ and } D_B) = \log p(D_B|\theta) + \log p(\theta|D_A) - \log p(D_B)$$

$\log p(D_B|\theta)$ is just the standard likelihood maximized in the optimization of $L_B(\theta)$, and the posterior $p(\theta|D_A)$ is approximated with Laplace's method as a Gaussian distribution with expectation $\theta^*_A$ and covariance $\operatorname{diag}(\mathcal{I}^*_A)$.

It is also well known that under some regularity conditions, the information matrix approximates the Hessian $H_L$ of $L(\theta)$, at $\theta = \theta^*$ [8]. By this we get a non Bayesian interpretation of (1),

$$\tilde{L}(\theta) \approx L_B(\theta) + \frac{1}{2}(\theta - \theta^*_A)^T H_{L_A}(\theta - \theta^*_A) \approx L_B(\theta) + L_A(\theta), \tag{2}$$

where $L(\theta) = L_B(\theta) + L_A(\theta)$ is exactly the loss we want to minimize. In general, one can learn a sequence of tasks $A_1 \ldots A_T$. In section 3 we rely on the above interpretation as a second order approximation in order to construct an algorithm for Federated Learning. We will further show how to implement such an algorithm in a way that preserves the privacy benefits of the standard FedAvg algorithm.

## 2 Related Work

There are only a handful of works that directly try to cope with the challenge of Federated Learning with non i.i.d. distribution. One approach is to just give up the periodical averaging, and reduce the communication by sparsification and quantization of the updates sent to the central point after each local mini batch [9]. In Zhao et al. [6] it was shown that by sharing only a small portion of the data between different nodes, one can achieve a great improvement in model accuracy. However, sharing data is not acceptable in many Federated Learning scenarios.

Somewhat similar to our approach, MOCHA [10] links each task with a different parameter $w_i \in \mathbb{R}^{d\times n}$ and the relation between the tasks is modeled by adding a loss term $\operatorname{tr}(W\Omega W^T)$, where $W = [w_1, \ldots, w_n]$ and $\Omega \in \mathbb{R}^{n\times n}$. The optimization is done on both $W$ and $\Omega$. MOCHA uses a primal-dual formulation in order to solve the optimization problem and thus, unlike our algorithm, is not suitable for deep networks.

Perhaps the closest work to ours is Sahu et al. [11], where the authors present their FedProx algorithm, which, like our algorithm, also uses parameter stiffness. However, unlike our algorithm, in FedProx the penalty term is isotropic, $\frac{1}{2}\mu\|\theta - \theta_t\|$. DANE [12] augments FedProx by adding a gradient correction term $-(\nabla L_i(\theta_{t-1}) - \eta\nabla L(\theta_{t-1}))^T\theta$ to accelerate convergence, but is not robust to non i.i.d. data [11, 13]. AIDE [13] improves the ability of DANE to deal with non i.i.d. data. However, it does so by using an inexact version of DANE, through a limitation on the amount of local computations.

A recent work [3] proves convergence of FedAvg for the non i.i.d. case. It also provides a theoretical explanation for a phenomenon known in practice, of performance degradation when the number of local iterations is too high. This is exactly the problem that we tackle in this work.

## 3 Federated Curvature

In this section we present our adaptation of the EWC algorithm to the Federated Learning scenario. We call it FedCurv (for *Federated Curvature*, motivated by (2)). We mark by $S = \{1 \ldots N\}$ the $N$ nodes, with the tasks' local datasets $\{A_1, \ldots A_N\}$. We diverge from the FedAvg algorithm and in each round $t$ we use all the nodes in $S$ instead of randomly selecting a subset on them. (Our algorithm can easily be extended to select a subset.) At round $t$ each node $s \in S$ optimizes the following loss:

信息量越大的惩罚越大

$$\tilde{L}_{t,s}(\theta) = L_s(\theta) + \lambda \sum_{j\in S\setminus s} (\theta - \hat{\theta}_{t-1,j})^T \operatorname{diag}(\hat{\mathcal{I}}_{t-1,j})(\theta - \hat{\theta}_{t-1,j}), \tag{3}$$

On each round $t$, starting from initial point $\hat{\theta}_t = \frac{1}{N}\sum_{i=1}^{N} \hat{\theta}_{t-1,i}$, the nodes optimize their local loss by running SGD for $E$ local epochs. At the end of each round $t$, each node $j$ sends to the rest of the nodes the SGD result $\hat{\theta}_{t,j}$ and $\operatorname{diag}(\hat{\mathcal{I}}_{t,j})$ (where $\hat{\mathcal{I}}_{t,j} = \mathcal{I}(\hat{\theta}_{t,j})$). $\hat{\theta}_{t,j}$ and $\operatorname{diag}(\hat{\mathcal{I}}_{t,j})$ will be used for the loss of round $t+1$. We switched from $\theta^*$ to $\hat{\theta}$ to signify that local tasks are optimized for $E$ epochs and not until they converge (as was the case for EWC). However, (2) (its generalization to $N$ tasks) supports using large values of $E$, so $\hat{\theta}_{t,j} \approx \theta^*_{t,j}$ and then $\tilde{L}_{t,j} \approx L$.

### 3.1 Keeping Low Bandwidth and Preserving Privacy

At first glance, maintaining all the historical data required by FedCurv might look cumbersome and expensive to store and transmit. It also looks like a sensitive information is passed between nodes. However by careful implementation we can avoid these potential drawbacks. We note that (3) can also be rearranged as

$$\boxed{\tilde{L}_{t,s}(\theta) = L_s(\theta) + \lambda\theta^T \left[\sum_{j\in S\setminus s} \operatorname{diag}(\hat{\mathcal{I}}_{t-1,j})\right]\theta - 2\lambda\theta^T \sum_{j\in S\setminus s} \operatorname{diag}(\hat{\mathcal{I}}_{t-1,j})\hat{\theta}_{t-1,j} + \text{const}}$$

**Bandwidth** The central point needs only to maintain and transmit to the edge node two additional elements, besides $\theta$, of the same size as $\theta$,

$$\boxed{u_t = \sum_{j\in S} \operatorname{diag}(\hat{\mathcal{I}}_{t-1,j}) \quad \text{and} \quad v_t = \sum_{j\in S} \operatorname{diag}(\hat{\mathcal{I}}_{t-1,j})\hat{\theta}_{t-1,j}}$$

服务器需要传输上一个任务的Fisher信息矩阵

The device can then construct the data needed for the evaluation of $\tilde{L}$ from $u_t,\ v_t$ by subtraction. The device $j$ at time $t$ needs also two transmit only two additional element at the same size of $\operatorname{diag}(\hat{\mathcal{I}}_{t-1,j})$ and $\operatorname{diag}(\hat{\mathcal{I}}_{t-1,j})\hat{\theta}_{t-1,j}$.

Table 1: Number of rounds to achieve a certain accuracy on Non-IID MNIST

| | $E = 50$ | | | $E = 10$ | | |
|---|---|---|---|---|---|---|
| Algorithm | 0.95 | 0.90 | 0.85 | 0.95 | 0.90 | 0.85 |
| FedCurv, $\lambda = 1.0$ | 38 | 9 | 6 | 99 | 35 | 27 |
| FedProx, $\mu = 0.00025$ | 140 | 22 | 16 | | | |
| FedProx, $\mu = 0.00001$ | | | | 115 | 46 | 33 |
| FedAvg | 76 | 30 | 22 | 106 | 51 | 43 |

**Privacy** It should be noted that we only need to send local gradient-related aggregated information (aggregated per local data sample) from the devices to the central point. In terms of privacy, it is not significantly different from the classical FedAvg algorithm. The central point itself, like in FedAvg, needs only to keep globally aggregated information from the devices. We see no reason why secure aggregation methods [14] which were successfully applied to FedAvg could not be applied to FedCurv.

**Further potential bandwidth reduction** The diagonal of the Fisher information has been used successfully for parameter pruning in neural networks [15]. This gives us a straightforward way to save bandwidth by using sparse versions of $\mathrm{diag}(\hat{\mathcal{I}})$ $\mathrm{diag}(\hat{\mathcal{I}})\hat{\theta}$ and even $\Delta\theta$, as $\mathrm{diag}(\hat{\mathcal{I}})$ provides a natural evaluation for the importance measure of the parameters of $\hat{\theta}$. The sparse versions are achieved by keeping only a fraction $0 < q \leq 1$ of indices that are related to the $q$ largest elements of the diagonal of the Fisher information matrix. We have not explored this idea in practice.

## 4 Experiments

We conducted our experiments on a group of 96 simulated devices. We divided the MNIST dataset [16] into $96 \times 2$ blocks of homogeneous labels (discarding a small amount of data). We randomly assigned two blocks to each device. We used the CNN architecture from the MNIST PyTorch example [17].

We explored two factors: (1) Learning method - we considered three algorithms, FedAvg, FedProx, and FedCurv (our algorithm); (2) $E$, the number of epochs in each round, which is of special interest in this work, as our algorithm is designed for large values of $E$. $C$, the fraction of devices that participate in each iteration, and $B$, the local batch size, were kept fixed at $C = 1.0, B = 256$. For all the experiments, we have also used a constant learning rate of $\eta = 0.01$.

FedProx's $\mu$ and FedCurv's $\lambda$ values were chosen in the following way: We looked for values that reached 90% test-accuracy in the smallest number of rounds. We did it by searching on a multiplicative grid using a factor of 10 and then a factor of 2 in order to ensure a minimum. Table 1 shows the number of rounds required in order to achieve 95%, 90% and 85% test-accuracy with these chosen parameters. We see that for $E = 50$, FedCurv achieved 90% test-accuracy three times as fast as the vanila FedAvg algorithm. FedProx also reached 90% faster than FedAvg. However, while our algorithm achieved 95% twice as fast as FedAvg, FedProx achieved it two times slower. For $E = 10$, the improvement of both FedCurv and FedProx is less significant, with FedCurv still outperforming FedProx and FedAvg.

In Figure 1 and Figure 2 we can see that both FedProx and FedCurv are doing well at the beginning of the training process. However, while FedCurv provides enough flexibility with $\theta$ that allows for reaching high accuracy at the end of the process, the stiffness of the parameters in FedProx comes at the expense of accuracy. FedCurv gives more significant improvements for higher values of $E$ (as does FedProx), as expected by the theory.

## 5 Conclusion

This work has provided a novel approach to the problem of Federated Learning on non i.i.d. data. It built on a solution from Lifelong Learning, which uses the diagonal of the Fisher information matrix in order to protect the parameters that are important to each task. The adaptation required modifying

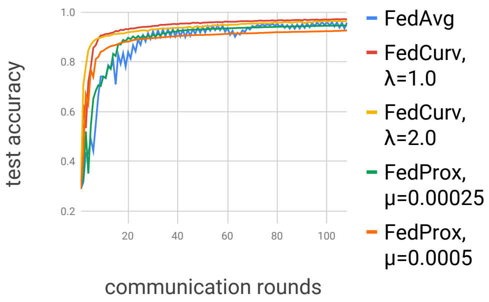

Figure 1: Learning curves, $E$=50

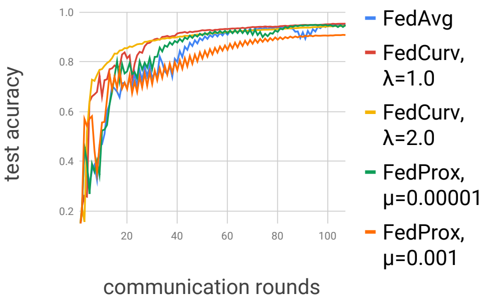

Figure 2: Learning curves, $E$=10

that sequential solution (from Lifelong Learning) into a parallel form (of Federated Learning), which a priori involves excessive sharing of data. We showed that this can be done efficiently, without substantially increasing bandwidth usage and compromising privacy. As our experiments have demonstrated, our FedCurv algorithm guards the parameters important to each task, improving convergence.

## References

[1] H Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, et al. Communication-efficient learning of deep networks from decentralized data. *arXiv preprint arXiv:1602.05629*, 2016.

[2] Swaroop Ramaswamy, Rajiv Mathews, Kanishka Rao, and Françoise Beaufays. Federated learning for emoji prediction in a mobile keyboard. *arXiv preprint arXiv:1906.04329*, 2019.

[3] 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.

[4] Shiqiang Wang, Tiffany Tuor, Theodoros Salonidis, Kin K Leung, Christian Makaya, Ting He, and Kevin Chan. Adaptive federated learning in resource constrained edge computing systems. *IEEE Journal on Selected Areas in Communications*, 37(6):1205–1221, 2019.

[5] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Lifelong learning via progressive distillation and retrospection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 437–452, 2018.

[6] Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582*, 2018.

[7] 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.

[8] Luc Pronzato and Andrej Pázman. Design of experiments in nonlinear models. *Lecture notes in statistics*, 212, 2013.

[9] Felix Sattler, Simon Wiedemann, Klaus-Robert Müller, and Wojciech Samek. Robust and communication-efficient federated learning from non-iid data. *arXiv preprint arXiv:1903.02891*, 2019.

[10] Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet S Talwalkar. Federated multi-task learning. In *Advances in Neural Information Processing Systems*, pages 4424–4434, 2017.

[11] Anit Kumar Sahu, Tian Li, Maziar Sanjabi, Manzil Zaheer, Ameet Talwalkar, and Virginia Smith. Federated optimization for heterogeneous network. *arXiv preprint arXiv:1812.06127*, 2018.

[12] Ohad Shamir, Nati Srebro, and Tong Zhang. Communication-efficient distributed optimization using an approximate newton-type method. In *International conference on machine learning*, pages 1000–1008, 2014.

[13] Sashank J Reddi, Jakub Konečnỳ, Peter Richtárik, Barnabás Póczós, and Alex Smola. Aide: Fast and communication efficient distributed optimization. *arXiv preprint arXiv:1608.06879*, 2016.

[14] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical secure aggregation for federated learning on user-held data. *arXiv preprint arXiv:1611.04482*, 2016.

[15] Yann LeCun, John S Denker, and Sara A Solla. Optimal brain damage. In *Advances in neural information processing systems*, pages 598–605, 1990.

[16] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[17] Pytorch mnist example. https://github.com/pytorch/examples/tree/master/mnist.