QUANTIFYING CATASTROPHIC FORGETTING IN CONTINUAL FEDERATED LEARNING

Christophe Dupuy [1] *Jimit Majmudar* [1] *Jixuan Wang* [1] *Tanya G. Roosta* [1]
Rahul Gupta [1] *Clement Chung* [1] *Jie Ding* [1] *Salman Avestimehr* [2 *]

[1]Amazon Alexa AI [2]University of Southern California (USC), USA

ABSTRACT

The deployment of Federated Learning (FL) systems poses various challenges such as data heterogeneity and communication efficiency. We focus on a practical FL setup that has recently drawn attention, where the data distribution on each device is not static but dynamically evolves over time. This setup, referred to as Continual Federated Learning (CFL), suffers from catastrophic forgetting, i.e., the undesired forgetting of previous knowledge after learning on new data, an issue not encountered with vanilla FL. In this work, we formally quantify catastrophic forgetting in a CFL setup, establish links to training optimization and evaluate different episodic replay approaches for CFL on a large scale real-world NLP dataset. To the best of our knowledge, this is the first such study of episodic replay for CFL. We show that storing a small set of past data boosts performance and significantly reduce forgetting, providing evidence that carefully designed sampling strategies can lead to further improvements.

1. INTRODUCTION

Federated learning (FL) enables distributed devices or siloed data centers to collaboratively learn a shared global machine learning model, without egressing the data from the devices or data centers. Under this framework, the model is trained on potentially large volumes of data while enhancing users' privacy. Meanwhile, FL presents distinct challenges from standard machine learning settings such as communication efficiency and heterogeneous data distributions across devices.

Most existing work explore a standard FL setup where devices store a (static) collection of local data which is used to minimize a (static) joint objective. In a real-world application, users continuously generate new data over time for a given task. Most user-owned devices have limited capacity (e.g., voice assistants, mobile phones) and cannot save all historical data for model training. Moreover, the data distribution of collected streams may shift over time. This realistic FL setup is referred to as Continual Federated Learning (CFL) [1–7].

In the CFL setup, naively fine-tuning the model on the latest data may cause the model to forget the knowledge learned from the previous data, a phenomenon known as *catastrophic forgetting*. This prevents the model from generalizing across different data distributions and being improved continuously. Mitigation strategies – such as *episodic replay*, *model regularization*, or *parameter freezing* – have been proposed for Continual Learning (CL), a centralized setting where constraints on storage and compute are less strenuous. However, most CFL work do not provide a systematic study of catastrophic forgetting. Besides, existing CFL works have mostly focused on theoretical convergence guarantees and present performance results on small-scale datasets that are artificially converted into CFL datasets, thus not capturing properties of a real-world setting.

Our contributions: (1) We propose a new catastrophic forgetting metric in the same vein as [8, 9] and establish its link to training objective which justifies the presented replay strategies for storing past information. (2) We present the first study of episodic replay for CFL and run experiments on a large scale real-world NLP dataset. (3) We propose a practical CFL framework to train and evaluate CFL models. We show the first results both in terms of performance and forgetting for different replay strategies in CFL and show significant benefits of the proposed strategies.

2. RELATED WORK

FL was introduced in [10] as a machine learning framework to train a single global model without accumulating data from the devices at a central server. There have been various modifications to the initial framework such as tackling data and computation heterogeneity across devices [11–14], ensuring uniform learning across devices [15], introducing personalization [16], and accounting for noisy labels derived from both positive and negative user feedback [17].

In **CL**, a model is required to learn tasks in a sequential fashion [18, 19]. CL algorithms are prone to catastrophic forgetting, i.e., the model's deteriorating performance on previously-seen tasks as new tasks are learned. While there is no universal definition for it, [8, 9] proposed ad hoc metrics for catastrophic forgetting. We provide a rigorous definition – conceptually consistent with existing ones – and draw the (previously missing) connection to the training objective.

Under the name of **CFL**, there has recently been a grow-

*Work done while affiliated with Amazon

ing interest in the intersection of CL and FL [1–7] due to its practical utility. However, most of this work do not delve deeper into the problem of catastrophic forgetting and only provide results on small-scale synthetic datasets.

3. CONTINUAL FEDERATED LEARNING

3.1. Background and Notation

In a standard machine learning setting, we assume access to a collection of data examples $\mathbf{X} = \{x_i\}_{i\in[N]}$. We learn a model $\mathbf{w}$ to minimize a loss function $\mathcal{L}$ over those N examples: $\min_{\mathbf{w}} \mathcal{L}(\mathbf{X}, \mathbf{w}) \equiv \frac{1}{N}\sum_{i\in[N]} \ell(\mathbf{x}_i; \mathbf{w})$. The minimization is done with stochastic gradient descent (SGD) [20] where, at each iteration step, gradients calculated with a random batch of examples are used to update the current parameters $\mathbf{w}$. In an FL setting, the data is distributed across M devices and is never available to the central server. The objective above is rewritten as: $\mathcal{L}(\mathbf{X}, \mathbf{w}) = \frac{1}{N}\sum_{m\in[M]} |\mathbf{X}^m| \mathcal{L}^m(\mathbf{X}^m, \mathbf{w})$, where $\mathbf{X}^m$ are examples on device m and $\mathcal{L}^m(\mathbf{X}^m, \mathbf{w}) = \frac{1}{|\mathbf{X}^m|}\sum_{\mathbf{x}\in\mathbf{X}^m} \ell(\mathbf{x}; \mathbf{w})$. In `FedAVG` [10], this objective is minimized by first fixing a number of server-device communication rounds, for each round c sampling a subset of devices, $S_c \subset \{1, \dots, M\}$, and requiring the selected devices to optimize their own objective $\mathcal{L}^m(\mathbf{X}^m, \mathbf{w})$ locally using the latest version of the model $\mathbf{w}_{c-1}$ as the starting point for a fixed number of epochs. The updated local parameters $\mathbf{w}_c^m$ are sent to the central server for aggregation: $\mathbf{w}_c = \frac{1}{\sum_{m\in S_c} |\mathbf{X}^m|}\sum_{m\in S_c} |\mathbf{X}^m| \mathbf{w}_c^m$.

3.2. Proposed Setup

In practice, the data is not static as devices generate data over time. Devices used for FL are often user-owned (e.g., phones, voice assistants) and likely have limited capacity, preventing them from storing the whole history of generated data. We consider a CFL setting where devices continuously generate new data but only store a subset of the past data due to memory constraints. More formally, we consider a period $\mathcal{P}$ divided into T time windows. For each device $m \in [M]$ and time window $t \in [T]$, training data $\mathbf{X}_t^m$ is generated and stored on device which is used, along with a compressed representation of past data, to update the model at the end of t using FL. Subsequently, the compressed representation is updated to integrate the data $\mathbf{X}_t^m$ collected in window t and stored during $t+1$.

3.3. Catastrophic Forgetting and Training Objective

We quantify catastrophic forgetting in model $\mathbf{w}$ on data $\mathbf{X}$ w.r.t. a collection of models C as the performance gap between $\mathbf{w}$ and the best model in C when evaluated on $\mathbf{X}$:

$$\mathrm{FgT}(\mathbf{X}, C, \mathbf{w}) = \mathcal{L}(\mathbf{X}, \mathbf{w}) - \min_{\mathbf{w}'\in C} \mathcal{L}(\mathbf{X}, \mathbf{w}'). \quad (1)$$

We adopt the convention $\mathrm{FgT}(\mathbf{X}, \varnothing, \mathbf{w}) = 0$ since there is no known historic performance on $\mathbf{X}$ to be forgotten. Note that $\mathrm{FgT}(\mathbf{X}, C, \mathbf{w}) < 0$ means that $\mathbf{w}$ outperforms all the models in C on data $\mathbf{X}$. For our CFL setup, we wish to learn a sequence of global models $\mathbf{w}_1, \dots, \mathbf{w}_T$ such that for each $t \in [T]$, model $\mathbf{w}_t$ is trained to: (1) perform well on the current task, and (2) minimize forgetting on all previously seen tasks. That is, we solve the following optimization problem:

$$\mathbf{w}_t = \arg\min_{\mathbf{w}} \mathcal{L}(\mathbf{X}_t, \mathbf{w}) + \lambda \sum_{t'<t} \mathrm{FgT}(\mathbf{X}_{t'}, C_t, \mathbf{w}), \quad (2)$$

where $C_t := \{\mathbf{w}_{t'}\}_{t'<t}$ and the tuning parameter λ controls the relative importance of past data with respect to current data. From Eq. (1), we rewrite the optimization in Eq. (2) as:

$$\mathbf{w}_t = \arg\min_{\mathbf{w}} \mathcal{L}(\mathbf{X}_t, \mathbf{w}) + \lambda \sum_{t'<t} \mathcal{L}(\mathbf{X}_{t'}, \mathbf{w}). \quad (3)$$

A practical challenge for this optimization is that the historical data ($t' < t$) is not entirely available in FL, motivating the need for a compressed representation of the past.

3.4. Representation of Past Data

We study two methods to represent past data and mitigate catastrophic forgetting, both utilizing a limited amount of storage on device: 1) *episodic replay*: a subset of past data is stored on device; 2) *model regularization*: model parameters from the previous time window are stored on device.

Episodic Replay: At the beginning of window t, each device m selects a subset, of size at most N_{past}, of the data from the previous window $t-1$ (potentially containing examples from before $t-1$) that is appended to $\mathbf{X}_t^m$ generated during t. The model $\mathbf{w}_t^m$ is trained at the end of t on the augmented set $\mathbf{A}_t^m = \mathbf{X}_t^m \cup S\left[\mathbf{A}_{t-1}^m\right]$, with $S\left[\mathbf{A}_{t-1}^m\right] \subseteq \mathbf{A}_{t-1}^m$ of size at most N_{past} and $\mathbf{A}_0^m = \varnothing$. Given Eq. (2) and (3), the best strategy to remember (in terms of optimization) would be to append a set $\mathbf{A}_t^m$ that is a uniform sample of the past time windows $t' < t$. In practice, the limited on-device storage prevents drawing uniformly random samples from past windows. At time t, only a subset of $\mathbf{A}_{t-1}^m$ of size N_{past} is available. We propose three selection strategies S below.

Naive uniform: At time window t, each device samples uniformly at random N_{past} examples from the *augmented set* $\mathbf{A}_{t-1}^m$. This method is "naive" because for an active device (i.e., such that $|\mathbf{X}_t^m| \geq N_{\text{past}}$), the likelihood of selecting examples from the earlier time windows decreases with time, which suggests higher vulnerability to catastrophic forgetting.

Approximation of uniform sampling (*approx. unif.*)**:** At the end of window $t-1$, device m contains $\mathbf{A}_{t-1}^m = \mathbf{X}_{t-1}^m \cup S\left[\mathbf{A}_{t-2}^m\right]$ and at time t we want the set of N_{past} examples to correspond to a uniform sample from the past examples up to $t-1$. To do so, we select $\lfloor N_{\text{past}} \frac{N_{t-1}^m}{N_{<t}^m} \rfloor$ examples from $\mathbf{X}_{t-1}^m$ and $\lfloor N_{\text{past}} \frac{N_{<t-1}^m}{N_{<t}^m} \rfloor$ examples from $S\left[\mathbf{A}_{t-2}\right]$, where $N_{t-1}^m =$

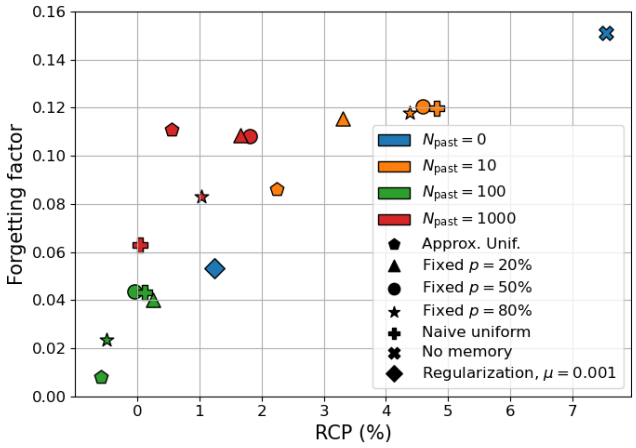

Fig. 1. Forgetting factor $\text{FgT}(\mathbf{X}_{all,test}, C_T, \mathbf{w}_T)$ against model performance (RCP); perplexity is computed using $\mathbf{w}_T$ on the combined test set $\mathbf{X}_{all,test}$. Colors indicate a value for N_{past} and symbols a memorization strategy.

$|\mathbf{X}^m_{t-1}|$ and $N^m_{<t} = \sum_{t'<t} N^m_{t'}$. A drawback of this method is that the amount of selected examples from $\mathbf{X}^m_{t-1}$ shrinks with time as $\frac{N^m_{t-1}}{N^m_{<t}}$ likely decreases with increasing t.

Fixed proportion $p \in (0, 1)$: At time window t, we select $\lfloor p \times N_{\text{past}} \rfloor$ examples from $\mathbf{X}^m_{t-1}$ and $\lfloor (1-p) \times N_{\text{past}} \rfloor$ examples from the previous augmented set $S[\mathbf{A}_{t-2}]$. The stored set of N_{past} examples contain less and less examples from the earliest periods, but the decrease is controlled by p (instead of depending on the device's activity as for the naive method).

Model Regularization: Another approach is to store model parameters (instead of data) from the previous time window $\mathbf{w}_{t-1}$ and use those for regularization during optimization: $\mathbf{w}_t = \arg\min_{\mathbf{w}} \mathcal{L}(\mathbf{X}_t, \mathbf{w}) + \mu \|\mathbf{w} - \mathbf{w}_{t-1}\|_2$. Note that in a constrained device, storing a few examples is much more efficient than a whole copy of the past model. We include this method here for a more comprehensive comparison.

4. EXPERIMENTAL SETTING

Model & Task: We use a BERT-*base* model [21] for the masked language modeling (MLM) task pre-trained on the Common Crawl dataset [22]. For each period t, parameters are initialized with the model from the previous period $t-1$.
Data: $\mathcal{P} = 50$ weeks of automated transcription of Alexa user request data, that is de-identified to remove information connecting the data to users. We divide the data in $T = 10$ time windows, each containing 5 weeks of data and use the first 4 weeks of each window as the training set, the last week for testing. We denote $\mathbf{X}_{t,test}$ the test set for window t and $\mathbf{X}_{all,test} = \bigcup_{t\in[T]} \mathbf{X}_{t,test}$. We randomly select 10,000 devices and their utterances. We generate 5 different datasets by repeating the random selection process.
FL method: We apply `FedAVG` [10] where we select 800 devices per round and each device performs one local epoch over their data, with standard central aggregation.
Replay Buffer: We consider $N_{past} \in \{10, 100, 1000\}$ for all our experiments and $p \in \{0.2, 0.5, 0.8\}$ for *fixed prop.*
Model regularization: We run experiments with $\mu \in \{0.0001, 0.001, 0.01, 0.1, 0.5, 1.0\}$ and only report the best results (for sake of space and clarity) obtained with $\mu = 0.001$.
Baselines: We train a "low-end" model with *no memory*, meaning that the model for time window t is simply fine-tuned on the data generated during t, starting from the model of period $t-1$. We also train a "high-end" (static) baseline FL model on the whole training data $\bigcup_{t\in[T]} \mathbf{X}_t$.
Evaluation: We present the relative change in perplexity (referred to as RCP from this point on) compared to the perplexity of the "high-end" baseline. We also measure the forgetting factor for each time period $\text{FgT}(\mathbf{X}_{t,test}, C_T, \mathbf{w}_T)$, corresponding to the forgetting factor of the final model $\mathbf{w}_T$ for the test data from period t. We also compute the overall forgetting on the whole test set $\text{FgT}(\mathbf{X}_{all,test}, C_T, \mathbf{w}_T)$. For FgT and RCP, a lower value is better.

5. RESULTS

FgT vs. RCP. The performance of the latest model $\mathbf{w}_T$ on the combined test set $\mathbf{X}_{all,test}$ against the forgetting factor $\text{FgT}(\mathbf{X}_{all,test}, C_T, \mathbf{w}_T)$ is presented in Figure 1. We observe the number of stored inputs N_{past} directly impacts model performance (X-axis): for most strategies, $N_{\text{past}} = 100$ (green symbols) provides the best performance and the least forgetting, outperforming model regularization (which would require storing $\approx 10^8$ parameters on each device). We already observe significant benefits with $N_{\text{past}} = 10$ (orange symbols) compared to the *no memory* setting (blue "X"). However, since the stored information is too small, the latest model $\mathbf{w}_T$ still exhibits high levels of forgetting. On the other hand, storing a larger number of inputs ($N_{\text{past}} = 1000$, red symbols) is detrimental to performance. For a given period t, the training set for device m is $\mathbf{X}^m_t \cup S[\mathbf{A}^m(t-1)]$, with $|S[\mathbf{A}^m(t-1)]| = N_{\text{past}}$. High values of N_{past}, i.e., $N_{\text{past}} \gg |\mathbf{X}^m_t|$ for most devices means past data outnumbers data generated during t, leading to overfitting on the past data. Across the proposed strategies, *approx. unif.* provides the best results, closely followed by *fixed prop.* with $p = 80\%$. They both provide the best balance between past and new examples. They outperform (i.e., RCP<0) the (static) "high-end" baseline and display almost no forgetting on the combined test set.

FgT across periods. We present the forgetting factor on the individual test sets of $t \in [T]$ for a selected set of strategies in Figure 2. As expected, for all the strategies, the latest model shows more forgetting on earlier periods ($t \leq 5$). FgT goes below 0 for the latest periods ($t \in \{9, 10\}$), which means

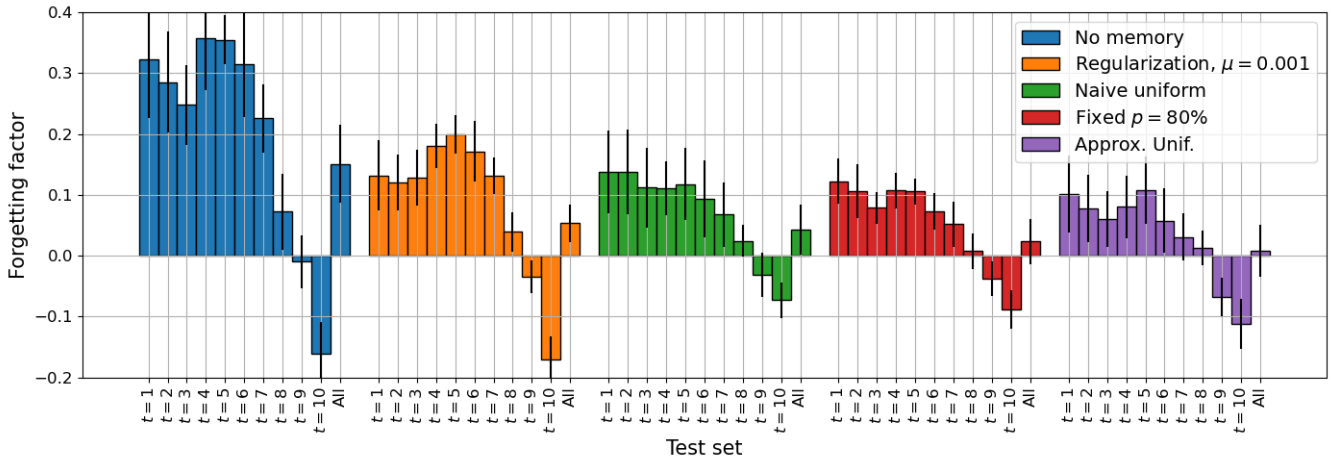

Fig. 2. Forgetting factor $\mathrm{FgT}(\mathbf{X}_{t,test}, C_T, \mathbf{w}_T)$ over the different periods $t \in [T] \cup \{all\}$ for different memory strategies with $N_{\text{past}} = 100$, sorted in decreasing order of $\mathrm{FgT}(\mathbf{X}_{all,test}, C_T, \mathbf{w}_T)$.

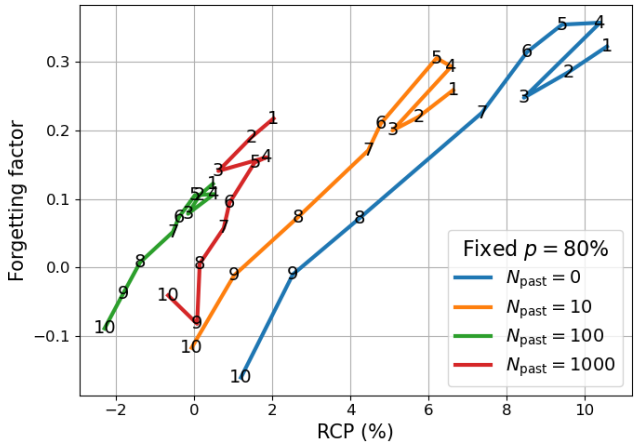

Fig. 3. Forgetting factor $\mathrm{FgT}(\mathbf{X}_{t,test}, C_T, \mathbf{w}_T)$ against performance (RCP) of $\mathbf{w}_T$ on $\mathbf{X}_{t,test}$, for $t \in [T]$ with *fixed prop.* ($p = 0.8$) and different values of N_{past}.

$\mathbf{w}_T$ outperforms all models in C_T on the latest test sets: $\mathbf{w}_T$ is trained on $\mathbf{X}_T$ while no model in C_T has seen this data. The methods with the highest forgetting on the combined test set (*no memory* and *model regularization*) have the lowest values of FgT on the last test set. For these strategies, the latest model fits the data generated during the latest period, quickly forgetting past information. On the other hand, the forgetting level for the three episodic replay strategies shows less variation across time periods, even though the forgetting factor remains negative for the latest periods. Among these strategies, *naive uniform* has the highest forgetting factor for most periods, especially for the earlier periods. Combined with performance values in Figure 1, we clearly observe the benefits of the proposed strategies *fixed prop.* and *approx. unif.*.

Detailed impact of N_{past}. We present the impact of N_{past} on the forgetting factor and performance for the different periods $t \in [T]$ in Figure 3, for *fixed prop.* with $p = 80\%$. The general trend is that performance (X-axis) improves with more data until a certain point ($N_{\text{past}} \leq 100$ in the figure) after which the latest model overfits the past data. This behavior is observed for every time period (while results in Figure 1 are aggregated over all periods). As expected, for any value of N_{past}, the latest model presents higher forgetting on earlier time periods ($t \leq 6$) than later periods. Both performance and forgetting are similar on all the early periods, suggesting that the model still partially forgets what was learned at the early stage. However, the forgetting factor is more uniform with higher N_{past}, so that with $N_{\text{past}} = 0$ the gap in forgetting factor between $t = 1$ and $t = 10$ is roughly 0.45, while it is around 0.2 for $N_{\text{past}} = 100$. Similarly, the performance gap is almost 10 RCP % points for $N_{\text{past}} = 0$ and reduces to slightly above 2 points in RCP for $N_{\text{past}} = 100$. The fact that both performance and forgetting factor are better and more uniform across time periods with $N_{\text{past}} = 100$ suggest a better generalizing model with $N_{\text{past}} = 100$.

6. CONCLUSION

We study FL in a continual setting where devices continuously generate data over time. We present different strategies to memorize past information and mitigate catastrophic forgetting caused by data distribution shift, allowing CFL on devices with limited storage capacity. We also propose a metric for catastrophic forgetting in such scenario. Through our experiments on a real-world NLP dataset, we show the benefits of storing (even a small amount) of past data on both performance and forgetting reduction. The presented strategies can outperform a static model trained on the whole data at once.

7. REFERENCES

[1] C. Dupuy, T. G. Roosta, L. Long, C. Chung, R. Gupta, and S. Avestimehr, "Learnings from federated learning in the real world," in *ICASSP*, 2022.

[2] Y. Guo, T. Lin, and X. Tang, "Towards federated learning on time-evolving heterogeneous data," *arXiv preprint arXiv:2112.13246*, 2021.

[3] X. Yao and L. Sun, "Continual local training for better initialization of federated models," in *IEEE ICIP*, 2020, pp. 1736–1740.

[4] Y. Chen, Y. Ning, M. Slawski, and H. Rangwala, "Asynchronous online federated learning for edge devices with non-iid data," in *IEEE International Conference on Big Data*, 2020, pp. 15–24.

[5] Jaehong Yoon, Wonyong Jeong, Giwoong Lee, Eunho Yang, and Sung Ju Hwang, "Federated continual learning with weighted inter-client transfer," in *ICML*, 2021.

[6] A. Usmanova, F. Portet, P. Lalanda, and G. Vega, "A distillation-based approach integrating continual learning and federated learning for pervasive services," *arXiv preprint arXiv:2109.04197*, 2021.

[7] F. E. Casado, D. Lema, R. Iglesias, C. V. Regueiro, and S. Barro, "Federated and continual learning for classification tasks in a society of devices," *arXiv preprint arXiv:2006.07129*, 2020.

[8] A. Chaudhry, N. Khan, P. Dokania, and P. Torr, "Continual learning in low-rank orthogonal subspaces," in *NeurIPS*, 2020.

[9] S. I. Mirzadeh, M. Farajtabar, R. Pascanu, and H. Ghasemzadeh, "Understanding the role of training regimes in continual learning," in *NeurIPS*, 2020.

[10] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *AISTATS*, 2017, pp. 1273–1282.

[11] T.-M. H. Hsu, H. Qi, and M. Brown, "Measuring the effects of non-identical data distribution for federated visual classification," *arXiv preprint arXiv:1909.06335*, 2019.

[12] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *Proceedings of Machine Learning and Systems*, vol. 2, pp. 429–450, 2020.

[13] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, "Scaffold: Stochastic controlled averaging for federated learning," in *ICML*, 2020, pp. 5132–5143.

[14] E. Diao, J. Ding, and V. Tarokh, "HeteroFL: Computation and communication efficient federated learning for heterogeneous clients," in *ICLR*, 2021.

[15] T. Li, M. Sanjabi, A. Beirami, and V. Smith, "Fair resource allocation in federated learning," *arXiv preprint arXiv:1905.10497*, 2019.

[16] T. Li, S. Hu, A. Beirami, and V. Smith, "Ditto: Fair and robust federated learning through personalization," in *ICML*, 2021, pp. 6357–6368.

[17] R. Sharma, A. Ramakrishna, A. MacLaughlin, A. Rumshisky, J. Majmudar, C. Chung, S. Avestimehr, and R. Gupta, "Federated learning with noisy user feedback," *arXiv preprint arXiv:2205.03092*, 2022.

[18] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, et al., "Overcoming catastrophic forgetting in neural networks," *Proceedings of the national academy of sciences*, vol. 114, no. 13, pp. 3521–3526, 2017.

[19] M. De Lange, R. Aljundi, M. Masana, S. Parisot, X. Jia, A. Leonardis, G. Slabaugh, and T. Tuytelaars, "A continual learning survey: Defying forgetting in classification tasks," *IEEE transactions on pattern analysis and machine intelligence*, vol. 44, no. 7, pp. 3366–3385, 2021.

[20] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *COMPSTAT*, 2010, pp. 177–186.

[21] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *NAACL*, 2019.

[22] Common Crawl Foundation, "Common crawl data," `https://commoncrawl.org/`, 2010.