Asynchronous Federated Continual Learning

Donald Shenaj, Marco Toldo, Alberto Rigon, Pietro Zanuttigh

JUNE 18-22, 2023 CVPR VANCOUVER, CANADA

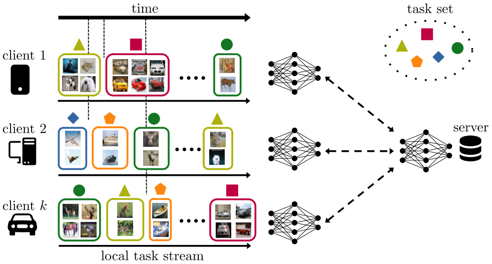
FedVision

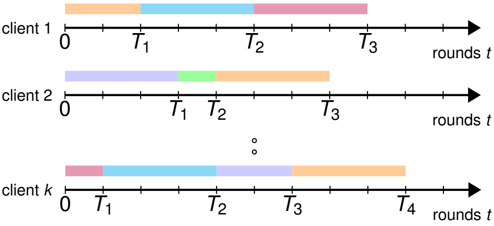
2nd Workshop on Federated Learning for Computer Vision

in Conjunction with CVPR 2023
(6/19 All Day)

Asynchronous Federated Continual Learning (AFCL)

- Continual learning (**CL**): learning happens in a sequence of steps each containing different tasks
- In many real-world **FL** scenarios users generate new data regularly
- Considering an aligned sequences of learning steps across different clients is not realistic
- In **AFCL** each client follows its own **CL** path asynchronously



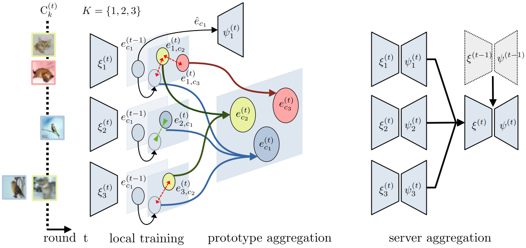


Federated learning System with Prototype Aggregation for Continual rEpresentation (FedSpace)

- Server fractal pre-training
- Local training:

$$\mathcal{L} = \mathcal{L}_{CE} + \lambda_p \mathcal{L}_p + \lambda_r \mathcal{L}_r$$

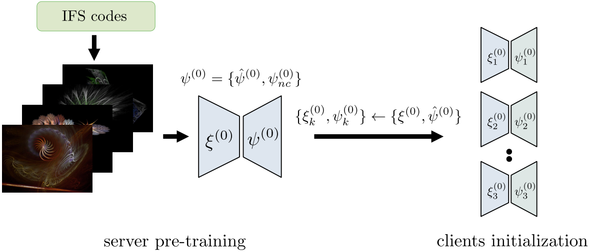
  - $\mathcal{L}_{CE}$: standard supervised cross-entropy on $\mathcal{D}_k^{(t)}$
  - $\mathcal{L}_p$: prototype-based loss
  - $\mathcal{L}_r$: representation loss



- Server aggregation

Server Fractal Pre-Training

- We employ the fractal pre-training strategy proposed in [1]



Why server pre-training?

- Improve performances for the downstream task
- Speeds up training in federated learning
- Limits client drift
- Better feature space alignment

Why fractals?

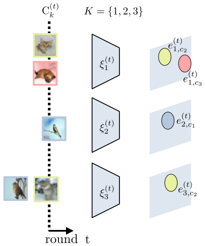
- Fractal images are generated from IFS codes (labels)
- Perfect label accuracy at zero cost
- No privacy issues
- No examples of classes to be learned

[1] C. Anderson and R. Farrell, Improving fractal pre-training, WACV 2022.

Local Prototypes

- At each round t, every selected client $k \in K$, for each class $c \in C_k^{(t)}$, computes the mean feature vector, i.e., prototype $e_{k,c}^{(t)}$ with radius $r_k^{(t)}$, following [2] :

$$\begin{cases} e_{k,c}^{(t)} = \dfrac{1}{|\mathcal{D}_k^{(t)}|} \displaystyle\sum_{\forall c \in C_k^{(t)}} E(\mathbf{X_k^{(t)}}; \xi_{\mathbf{k}}^{(\mathbf{t})}) \\ r_k^{(t)} = \sqrt{\dfrac{1}{|\mathcal{D}_k^{(t)}|} \displaystyle\sum_{\forall c \in C_k^{(t)}} \dfrac{Tr(\Sigma_c^{(t)})}{d}} \end{cases}$$



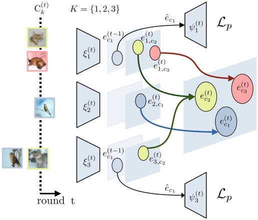
- $E(\cdot)$: output of the encoder with weights $\xi_k^{(t)}$ given $\mathbf{X_k^{(t)}}$ in input
- $Tr(\cdot)$: trace operator of a matrix
- $\Sigma_c^{(t)}$ covariance matrix for the features of class c at round t for client k
- d: dimension of the feature space

[2] F. Zhu et al., Prototype augmentation and self-supervision for incremental learning, CVPR 2021.

Prototypes Aggregation

- Local prototypes are aggregated into global prototypes:

$$e_c^{(t)} = \begin{cases} \sum_{k=1}^{K} \frac{|\mathcal{D}_k|}{\mathcal{D}} e_{k,c}^{(t)} & \text{if } c \notin C^{(t-1)} \\ \beta \left(\sum_{k=1}^{K} \frac{|\mathcal{D}_k|}{\mathcal{D}} e_{k,c}^{(t)} \right) + (1-\beta) e_c^{(t-1)} & \text{otherwise} . \end{cases}$$



- Each client k performs prototype augmentation [2] on the (old) global classes not present at round t

$$\hat{e}_c = e_c^{(t-1)} + r^{(t-1)} * \mathcal{N}(0,1)$$

- The loss is computed between a vector of augmented prototypes $\hat{\mathbf{e}}$ of the same length of the batch size:

$$\mathcal{L}_p = \sum_n \mathcal{L}_{CE}(D(\hat{\mathbf{e}}[n]; \psi_k^{(t)}), \mathcal{Y}[n])$$

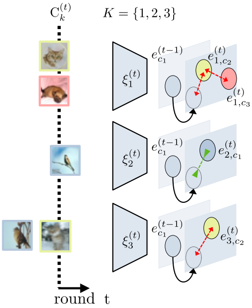
[2] F. Zhu et al., Prototype augmentation and self-supervision for incremental learning, CVPR 2021.

Representation Loss

- Feature clustering consistent across clients and time slots:

$$\mathcal{L}_r = -\frac{1}{|\mathcal{D}_k^{(t)}|} \sum_{c=1}^{|\mathcal{C}_k^{(t)}|} \frac{1}{N_c(N_c-1)} \sum_{i \neq j} \log \frac{e^{s_{i,j}^{+}}}{e^{s_{i,j}^{+}} + \sum_{k=1, k \neq c}^{C} e^{s_{i,k}^{-}}},$$

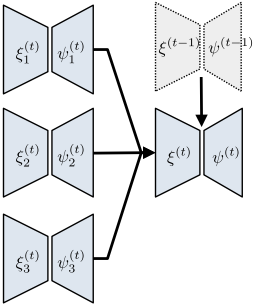
- N_c: number of samples of the class c
- $s_{i,j}^{+}$: cosine similarity between the i-th and j-th feature vectors in the positive set
- $s_{i,k}^{-}$: cosine similarity between the i-th feature vector in the positive set and the k-th vector in the negative set
- $\mathcal{C}_k^{(t)}$: classes discovered at round t



Server Aggregation

- Stable aggregation avoiding client drift and forgetting of old classes

$$\theta^{(t)} = \rho \left(\sum_{k=1}^{K} \frac{|\mathcal{D}_k|}{\sum_{i=1}^{k} |\mathcal{D}_i|} \theta_k^{(t)} \right) + (1-\rho)\theta^{(t-1)}$$

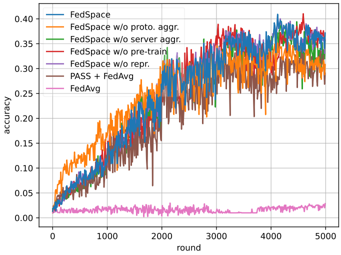


- $|\mathcal{D}_k|$: # of samples for client k
- $\theta_k^{(t)}$: parameters of client k
- $\theta_k^{(t-1)}$: parameters of the server
- sperimentally best results with $\rho = 0.5$ (simple mean)

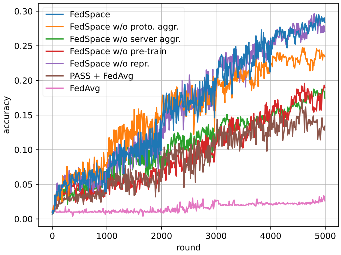
Results

Method	Top-1 Accuracy (%) 50	100	500
FedAvg [3]	2.86	2.01	2.63
PASS[2]+ FedAvg [3]	29.08	22.97	13.39
Ours (FedSpace)	**37.18**	**29.99**	**28.42**

Proto Aggr.	Pre-train	Repr. Loss	Server Aggr.	Accuracy (%) 50	100	500
✓				28.76	29.58	23.45
	✓	✓	✓	30.82	25.89	23.45
✓		✓	✓	35.72	25.89	19.01
✓	✓		✓	35.96	26.84	27.05
✓	✓	✓		35.60	31.90	17.46
✓	✓	✓	✓	37.18	29.99	28.42



(a) 50 clients.



(b) 500 clients.

Conclusion

- We introduce AFCL a new **challenging** and **realistic setting** for **CL** in **FL**
- We propose **FedSpace**, which makes use of prototype-based learning, representation loss, fractal pre-training, and a modified aggregation policy
- We provide **3 new splits** of CIFAR-100 dataset with 50, 100 and 500 clients
- Our approach reached state-of-the-art results when compared with competing methods adapted to our setting

Contacts

[1] C. Anderson and R. Farrell, Improving fractal pre-training, WACV 2022.

[2] F. Zhu et al., Prototype augmentation and self-supervision for incremental learning, CVPR 2021.

[3] B. McMahan et al., Communication-efficient learning of deep networks from decentralized data, AISTATS 2017.

Paper

Code

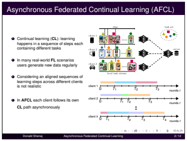
2nd Workshop on Federated Learning for Computer Vision

in Conjunction with CVPR 2023 (6/19 All Day)

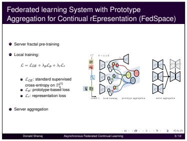
donald.shenaj@dei.unipd.it

Any questions?

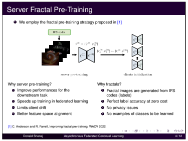
Frame 1



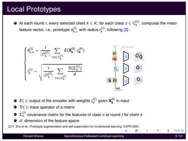
Asynchronous Federated Continual Learning (AFCL)



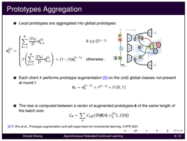
Federated learning System with Prototype Aggregation for Continual rEpresentation (FedSpace)



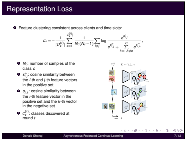
Server Fractal Pre-Training



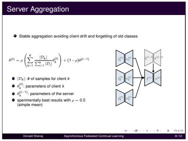
Local Prototypes



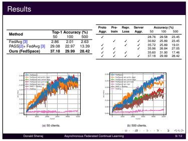
Prototypes Aggregation



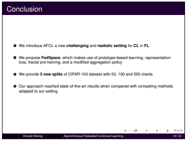
Representation Loss



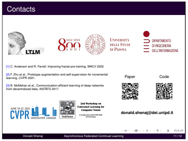
Server Aggregation



Results



Conclusion



Contacts