Federate Learning
&
Other researches about it

Base Paper :

Communication-Efficient Learning of Deep Networks from Decentralized Data

H. Brendan McMahan etc

Google Inc.

International Conference on Artificial Intelligence and Statistics (AISTATS) 2017

Cite : 1920

2021.03.04

임진혁

Index

0 index ++ 발표 피드백

발표 피드백

1. 맨 처음에 이 논문이 하고자 하는 것을
   [meta path을 auxiliary task로 self-supervised learning 적용해서 hetergerous graph에서의 primary task 성능을 향상시키는 방법론]
   이라고 설명해준 다음에
   키워드 별로 개념 설명을 간단하게 해주고 맨 처음에 말한 개념을 각각의 키워드를 연결해서 설명함으로써 이해시키는 접근
   좋았던 것 같음 내 취향임.

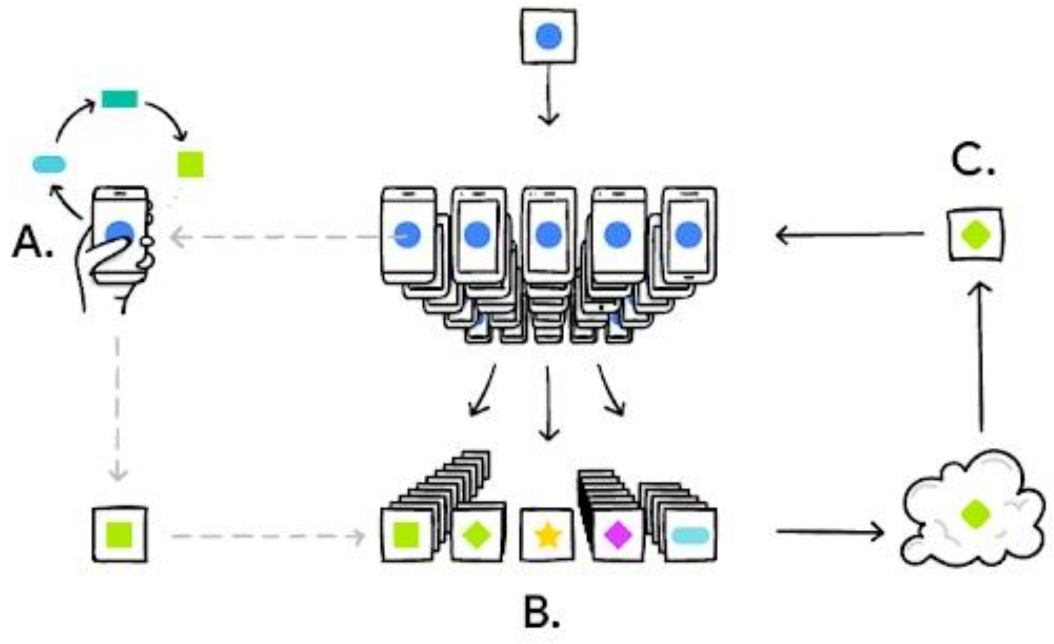
2. 논문 선택 배경
   - 추천시스템
   - self-supervised 에 대한 흥미도. 대단함 등을 어필해서 하면 좋을듯 (최근 트렌드라던가, 관심이 많다던가)
   vision에서 어쩌구 nlp에서 어쩌구 , 근데 gnn에도 적용을 한다는게 어떻게 하고 얼마나 좋은지 궁금해서
   (bootstrap on your 이런거 간단하게 말로 설명하면 될듯)

3. Meta-predict을 좀 더 명확하게 설명해주는게 좋을듯 (예시?)

4. 전체적으로 input vector을 그려서 명확하게 어떤 식으로 input => outpu이 되는지 설명하면 엄청 엄청 좋을듯
   (근데 이게 가능하면 구현도 가능할듯?)

Introduction: What is FL ?

Federate Learning (연합학습) ∈ Decentralized approach



(i) Data Privacy

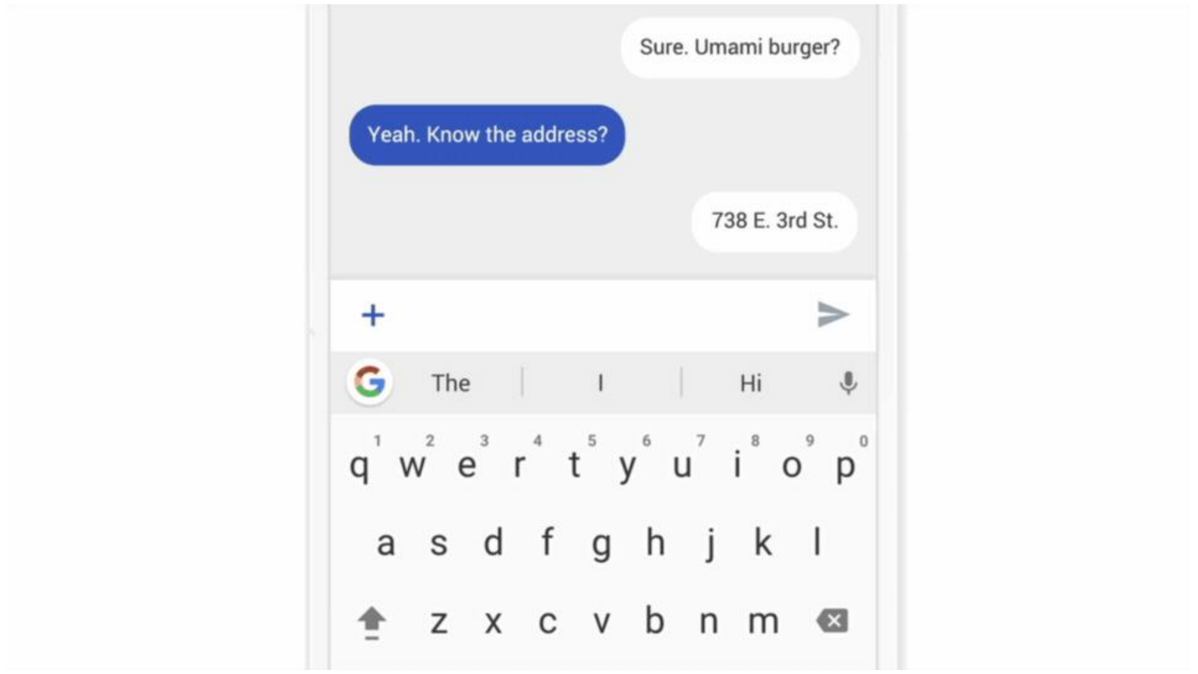
(ii) Communication Efficiency

Abstract

"We advocate an alternative that leaves the training data distributed on the mobile devices, and learns a shared model by aggregating locally-computed updates."

1 Introduction: What is FL ?

Background:


Sure. Umami burger?

Yeah. Know the address?

738 E. 3rd St.

Worldwide popularization of mobile devices
Devices’ powerful sensors & powerful computing power

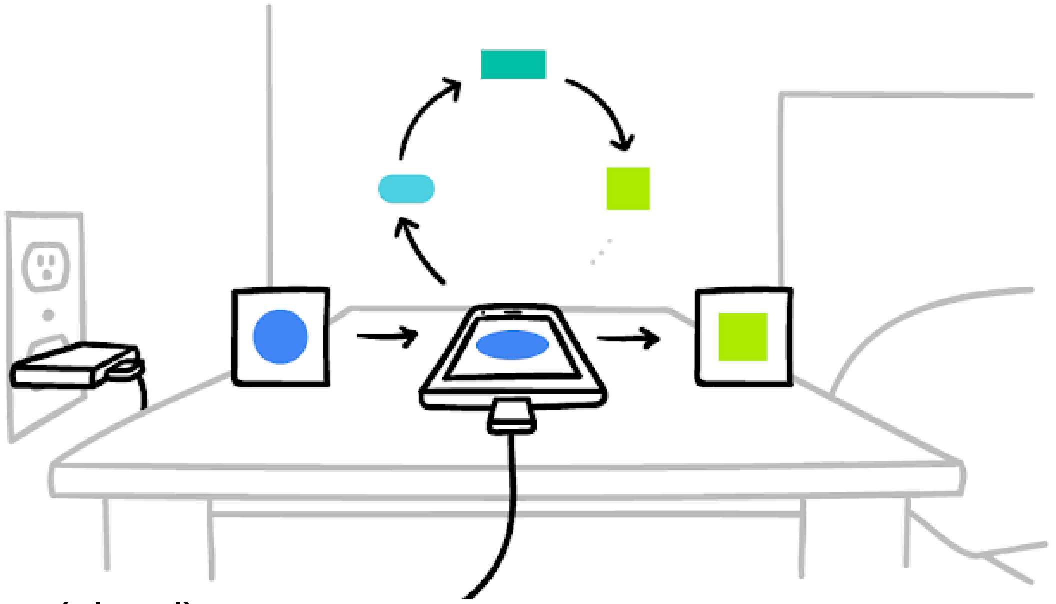
Conventional standard ML service :
Require Centralizing training data on one machine or in a datacenter(cloud)

Federated Learning:
For smarter models, lower latency, and less power consumption, all while ensuring privacy (*Google AI say*)

1 Introduction: What is FL ?

Backgrounds:

Worldwide popularization of mobile devices
Devices’ powerful sensors & powerful computing power



Conventional standard ML service :
Require Centralizing training data on one machine or in a datacenter(cloud)

Federated Learning:
For smarter models, lower latency, and less power consumption, all while ensuring privacy (*Google AI say*)

Introduction

Paper’s Contribution:

(1) Identification of the problem of training on decentralized data from mobile devices

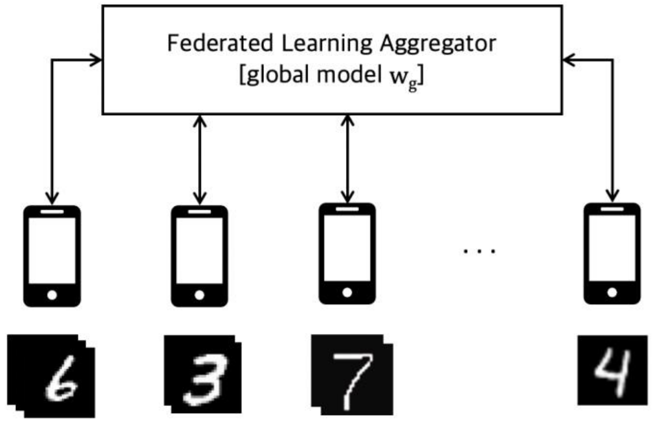
(2) Practical algorithm that can be applied to this setting (*FederatedAveraging* algorithm)

(3) Evaluation of the proposed approach

Problem Setting

Ideal problems for federated learning following properties

1) provides a distinct advantage over training on data that is generally in the data center.
2) data is privacy sensitive or large in size
3) labels on the data can be inferred naturally from user interaction



Federated optimization has several key properties ≠ Distributed optimization (분산학습)

1) Non-IID : 각 사용자가 만들어낸 데이터의 분포가 서로 다를 수 있다.
2) Unbalanced : 특정 사용자가 다른 사용자들에 비해 더 많은 비중을 차지할 수 있다.
3) Massively distributed : 사용자별 데이터보다 사용자가 더 많을 수 있다.
4) Limited communication : 기기가 오프라인이거나 과도하게 연결이 많을 수 있다.

2 Problem Setting

Fixed set K (clients)
Random fraction C (of clients)
Current global algorithm state to each of these clients

(1)

$$n_k = |\mathcal{P}_k|.$$

$$f(w) = \sum_{k=1}^{K} \frac{n_k}{n} F_k(w) \quad \text{where} \quad F_k(w) = \frac{1}{n_k} \sum_{i \in \mathcal{P}_k} f_i(w).$$

$$\mathbb{E}_{\mathcal{P}_k}[F_k(w)] = f(w)$$

(2)

Data center optimization
Communication costs low and computational costs more high

(3)

Simple one-shot averaging

each client’s model that minimizes loss on their local data
these models are averaged to produce the final **global model**

3 Proposed Approach: The *FederatedAveraging* Algorithm

Baseline: Large-batch synchronous SGD

FedSGD

Parameter: **C** (Data fraction of each clients)

$$w_{t+1} \leftarrow w_t - \eta \sum_{k=1}^{K} \frac{n_k}{n} g_k$$

(C = 1)

$$w_{t+1} \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^k.$$

FedAvg

Parameter: **C , B , E**

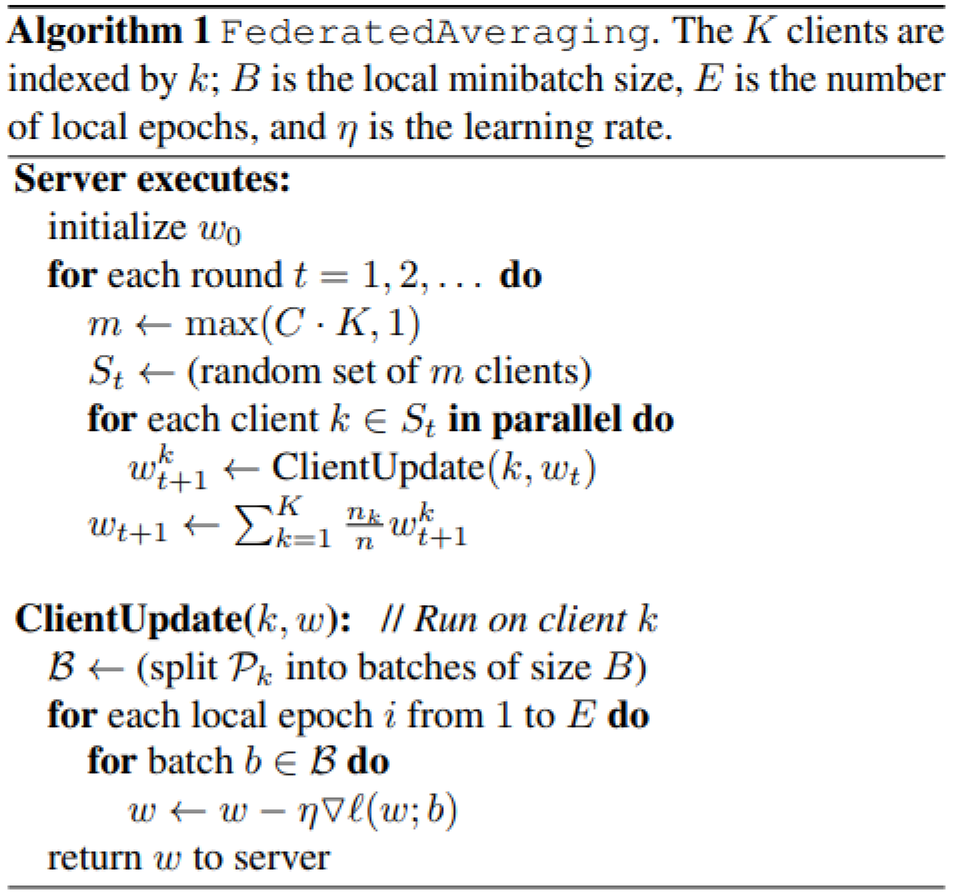
And it is related with the number of local updates per round

$$u_k = E\frac{n_k}{B}$$

Why SGD Based?

IF B = ∞ and E = 1?

3 Proposed Approach: The *FederatedAveraging* Algorithm

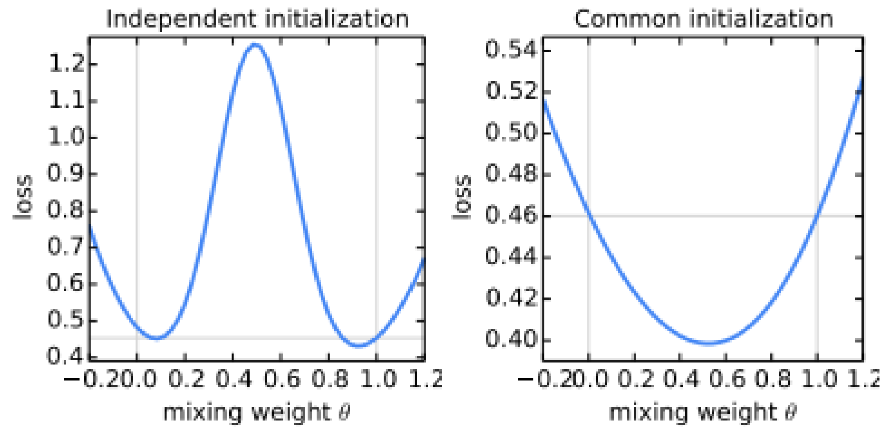


Algorithm 1 FederatedAveraging. The K clients are indexed by k; B is the local minibatch size, E is the number of local epochs, and η is the learning rate.

```
Server executes:
  initialize $w_0$
  for each round $t = 1, 2, \dots$ do
    $m \leftarrow \max(C \cdot K, 1)$
    $S_t \leftarrow$ (random set of $m$ clients)
    for each client $k \in S_t$ in parallel do
      $w_{t+1}^k \leftarrow \text{ClientUpdate}(k, w_t)$
    $w_{t+1} \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^k$

ClientUpdate($k, w$):   // Run on client $k$
  $\mathcal{B} \leftarrow$ (split $\mathcal{P}_k$ into batches of size $B$)
  for each local epoch $i$ from 1 to $E$ do
    for batch $b \in \mathcal{B}$ do
      $w \leftarrow w - \eta \nabla \ell(w; b)$
  return $w$ to server
```

단순히 모델을 평균내는 것은 전반적으로 성능하락으로 이어진다?



Same Random Initialization and Different Subset of data

4 Experiments

For **the MNIST digit** recognition task and **CIFAR-10 dataset**

MNIST 2NN

simple multilayer-perceptron with 2-hidden layers with 200 units
each using ReLu activations
(199,210 total parameters)

MNIST CNN

2CNN with two 5x5 convolution
fully connected layer with 512 units and ReLu activation
final softmax output layer
(1,663,370 total parameters).

how the data is distributed over the clients?

non-IID partition
sort the data by digit label, divide it into 200 shards of size 300, and assign each of 100 clients 2 shards

--> **only have examples of two digits**.

4 Experiments

Table 1: Effect of the client fraction C on the MNIST 2NN with $E = 1$ and CNN with $E = 5$. Note $C = 0.0$ corresponds to one client per round; since we use 100 clients for the MNIST data, the rows correspond to 1, 10 20, 50, and 100 clients. Each table entry gives the number of rounds of communication necessary to achieve a test-set accuracy of 97% for the 2NN and 99% for the CNN, along with the speedup relative to the $C = 0$ baseline. Five runs with the large batch size did not reach the target accuracy in the allowed time.

2NN	IID		Non-IID	
C	$B = \infty$	$B = 10$	$B = \infty$	$B = 10$
0.0	1455	316	4278	3275
0.1	1474 (1.0×)	87 (3.6×)	1796 (2.4×)	664 (4.9×)
0.2	1658 (0.9×)	77 (4.1×)	1528 (2.8×)	619 (5.3×)
0.5	— (—)	75 (4.2×)	— (—)	443 (7.4×)
1.0	— (—)	70 (4.5×)	— (—)	380 (8.6×)
CNN, $E = 5$				
0.0	387	50	1181	956
0.1	339 (1.1×)	18 (2.8×)	1100 (1.1×)	206 (4.6×)
0.2	337 (1.1×)	18 (2.8×)	978 (1.2×)	200 (4.8×)
0.5	164 (2.4×)	18 (2.8×)	1067 (1.1×)	261 (3.7×)
1.0	246 (1.6×)	16 (3.1×)	— (—)	97 (9.9×)

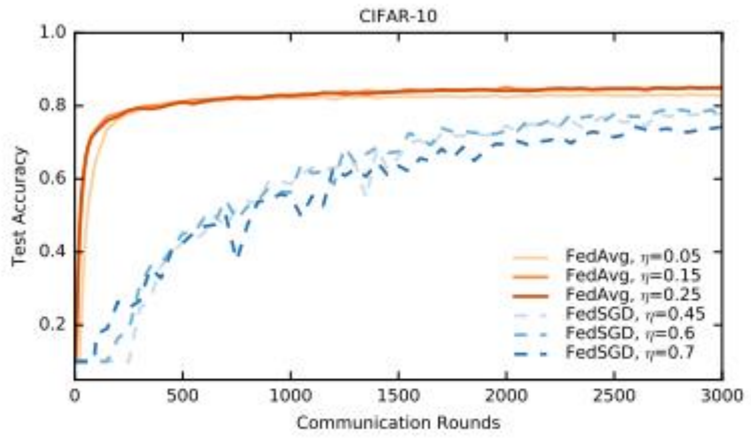


Figure 4: Test accuracy versus communication for the CIFAR10 experiments. `FedSGD` uses a learning-rate decay of 0.9934 per round; `FedAvg` uses $B = 50$, learning-rate decay of 0.99 per round, and $E = 5$.

REF

https://tootouch.github.io/research/federated_learning/
http://ki-it.com/_PR/view/?aidx=27730&bidx=2369

https://brunch.co.kr/@synabreu/100

https://ai.googleblog.com/2017/04/federated-learning-collaborative.html

https://arxiv.org/abs/1602.05629

https://medium.com/curg/%EC%97%B0%ED%95%A9-%ED%95%99%EC%8A%B5-federated-learning-%EA%B7%B8%EB%A6%AC%EA%B3%A0-%EC%B1%8C%EB%A6%B0%EC%A7%80-b5c481bd94b7

감사합니다