Federated Learning

Wang Xianyi

LZU

2024.2.3

Problem and Motivation

- Data is Privacy sensitive
- Data is Large in quantity

Therefore, restoring data in centralized localization has risk

Solution & Contributions

Solution(Federated Learning):

- Leaves training data in mobile devices.
- Shares locally-computed updates in mobile devices rather than the data with server.

Contributions:

- The identitfication of the problem of training on decentralized data from mobile devices as an important research direction;
- the selection of a straightforward and practical algorithm that can be applied to this setting
- an extensive empirical evaluation of the proposed approach.

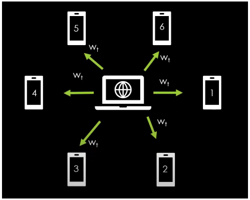
They have introduced the "FederatedAveraging[1]" algorithm, which combines local stochastic radient descent (SGD) on each client with a server that pertorms model averaging.

Introduction

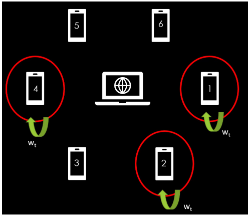
Federated optimization has several key properties compared to a typical distributed optimization problems;

- Non-IID
- Unbalanced
- Massively distributed
- Limited communication

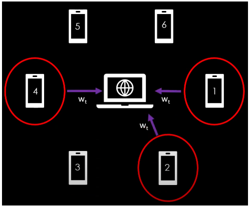
Introcduction



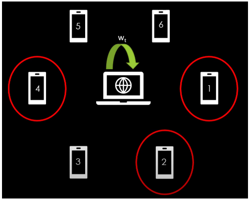
(a)



(b)



(c)



(d)

Problem

There are two primary aspects of the algorithm;

- How to compute global/local loss function?
- How to compute global/local gradient calculation and update state

Objective Function

The objective function:

$$\min_{\omega \in R^d} f(\omega) \tag{1}$$

$$f(\omega) = \frac{1}{n} \sum_{i=1}^{n} f_i(\omega) \tag{2}$$

Every client computes the loss function above with its local dataset
Then these local loss scores are weighted-averaged

$$f(\omega) = \sum_{k=1}^{K} \frac{n_k}{n} F_k(\omega) \tag{3}$$

$$F_k(\omega) = \frac{1}{n_k} \sum_{i \in P_k} f_i(\omega) \tag{4}$$

Optimize

FederatedAveraging(or FedAvg)

Each client k locally takes one step of gradient descent on the global model with its local dataset.

$$\forall k, \quad \omega_{t+1} \leftarrow \omega_t - \eta g_k \tag{5}$$

Then the server takes a weighted average of the resulting models.

$$\omega_{t+1} \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} \omega_{t+1}^{k} \tag{6}$$

Computation

(FedAvg) fits real world problems more.

$$\omega^k \leftarrow \omega^k - \eta \nabla F_k(\omega^k) \tag{7}$$

Computation is controlled by three key parameters for this approach

- C:The fraction of clients that perform computation on each round.
- E:Number of training passes each client makes over its local dataset on each round.
- B:Local minibatch size used for the client updates.

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

图 2: FederatedAveraging

MNIST Digit Recognition

Model

- 2NN model[2]
- Basic CNN Model

Data distribution

- IID
- Non-IID

The complete Works of William Shakespeare

Model

- LSTM[3]

Data distribution

- Balanced and IID version
- Unbalanced and non-IID version

Experiment Results

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×)

图 3: Effect of the client fraction C on the MNIST 2NN with E = 1 and CNN with E = 5.

C is set to 0.1 for all experiments.
Computation increased by increasing E, decreasing B.

Experiment Results

MNIST CNN, 99% ACCURACY					
CNN	E	B	u	IID	NON-IID
FEDSGD	1	∞	1	626	483
FEDAVG	5	∞	5	179 (3.5×)	1000 (0.5×)
FEDAVG	1	50	12	65 (9.6×)	600 (0.8×)
FEDAVG	20	∞	20	234 (2.7×)	672 (0.7×)
FEDAVG	1	10	60	34 (18.4×)	350 (1.4×)
FEDAVG	5	50	60	29 (21.6×)	334 (1.4×)
FEDAVG	20	50	240	32 (19.6×)	426 (1.1×)
FEDAVG	5	10	300	20 (31.3×)	229 (2.1×)
FEDAVG	20	10	1200	18 (34.8×)	173 (2.8×)
SHAKESPEARE LSTM, 54% ACCURACY					
LSTM	E	B	u	IID	NON-IID
FEDSGD	1	∞	1.0	2488	3906
FEDAVG	1	50	1.5	1635 (1.5×)	549 (7.1×)
FEDAVG	5	∞	5.0	613 (4.1×)	597 (6.5×)
FEDAVG	1	10	7.4	460 (5.4×)	164 (23.8×)
FEDAVG	5	50	7.4	401 (6.2×)	152 (25.7×)
FEDAVG	5	10	37.1	192 (13.0×)	41 (95.3×)

图 4: Effect of the client fraction C on the MNIST 2NN with E = 1 and CNN with E = 5.

Experiment Results

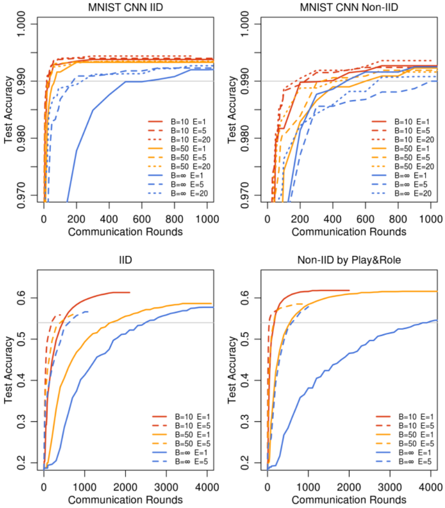


图 5: Test set accuracy vs. communication rounds for the MNIST CNN and Shakespeare LSTM with C = 0.1 and optimized η. The gray lines show the target accuracies used in Table 4.

Experiment Results

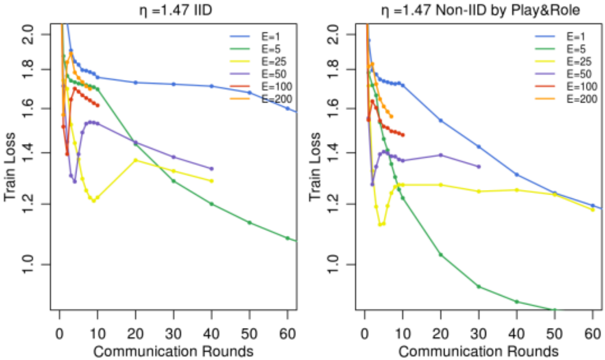


图 6: The effect of training for many local epochs between averaging steps, fixing B = 10 and C = 0.1 for the Shakespeare LSTM with a fixed learning rate $\eta = 1.47$.

Experiment Conclusion

- Using extreme non IID data to train the model, FedAvg can also perform better than FedSGD, indicating that the FedAvg algorithm may have strong robustness.
- FedAvg has higher accuracy than FedSGD on the test set, and the author speculates that FedAvg produces a dropout like regularization effect.
- When larger, the accuracy of FedAvg algorithm convergence will decrease, or even not converge.

Conclusion

Fedavg trains high-quality models using relatively few rounds of communication.

It offers many practical privacy benefits;

- Differential privacy
- Secure multi-party computation

[1] B. McMahan, E. Moore, D. Ramage, et al. Communication-efficient learning of deep networks from decentralized data[C]. Artificial intelligence and statistics, 2017, 1273-1282

[2] Y. LeCun, L. Bottou, Y. Bengio, et al. Gradient-based learning applied to document recognition[J]. Proceedings of the IEEE, 1998, 86(11): 2278-2324

[3] Y. Kim, Y. Jernite, D. Sontag, et al. Character-aware neural language models[C]. Proceedings of the AAAI conference on artificial intelligence, 2016, 2741-2749

Thanks!