Algorithms

1850245174

December 2024

1 Introduction

This paper introduces various algorithms used in our research.

2 Symbol Table

Below is a table of symbols used in the algorithms:

Symbol	**Description**
J	Number of classes
T	Global rounds
E	Edge rounds
K	Local epochs
L	Number of edge servers
N	Number of clients in each edge server
$\mathcal{N}_j^l$	Number of clients in edge l containing class j that have participated in aggregation
$\mathcal{N}_j$	Number of edge servers l containing class j that have participated in aggregation
S^l	Set of clients participating in training in the l-th edge server
$\bar{C}_j$	Aggregated prototype of class j in the cloud edge server
C_j^l	Aggregated prototype of class j from the l-th edge server
$C_j^{l,old}$	Last version of aggregated prototype of class j from the l-th edge server stored in the cloud server
$c_{i,j}^l$	Aggregated prototype of class j from the client i in the l-th edge server stored in the edge server
$c_{i,j}^{l,\text{old}}$	Last version of the aggregated prototype of class j from client i in the l-th edge server
B	The buffer of client server with static length

Table 1: Symbol Table

3 Pseudo-Code

The following algorithm demonstrates how to calculate the factorial of a number.

Algorithm 1 Hierarchical Federated Prototype Learning -Part 1

```
Input: nothing
Output: nothing
procedure cloud server executes
    Initialize weights for clients withheterogeneous models
    All edge servers execute in parallel
    for $t = 1, \ldots, T$ do
        Clear the buffer $B$
        while The buffer is not full do
            Receive aggregated prototypes $C^l$ from one edge node
            Fill the buffer $B$ with $C^l$
        end while
        $\bar{C} \leftarrow$ CloudAggregate($B$)
        Send $\bar{C}$ to $S^t_{edge}$
        $S^t_{edge}$ re-execute
    end for
end procedure
procedure edge server executes
    Receive $\bar{C}$ from the cloud server
    for $e = 1, \ldots, E$ dor
        Send $\bar{C}$ to client $i \in S^l$
        for each client $i$ in parallel do
            $c^l_i \leftarrow$ LocalUpdate($i, \bar{C}_i$)
        end for
        $C^l \leftarrow$ EdgeAggregate($\{c^l_i\}_{i \in S^l}$)
        $\bar{C} \leftarrow$ EdgeUpdate($\bar{C}, C^l$)
    end for
    Send $C^l$ to the cloud server
end procedure
```

Algorithm 2 Hierarchical Federated Prototype Learning -Part 2

```
procedure CloudAggregate(buffer)
    for $j = 1, \dots, J$ do
        $\bar{C}_j^{'} \leftarrow \mathcal{N}_j \cdot \bar{C}_j$                    ▷ Extend the aggregated prototypes $\bar{C}$
    end for
    for $\bar{C}_j^{'} \in \bar{C}^{'}$ do
        for $C^l \in$ buffer do
            if $C_j^{l,old}$ is not empty then
                $\bar{C}_j^{'} \leftarrow \bar{C}_j^{'} - C_j^{l,old} + C_j^l$
            else
                $\bar{C}_j^{'} \leftarrow \bar{C}_j^{'} + C_j^l$
                $\mathcal{N}_j \leftarrow \mathcal{N}_j + 1$
            end if
            $C^{l,old} \leftarrow C^l$
        end for
    end for
    $\bar{C}_j \leftarrow \frac{\bar{C}_j^{'}}{\mathcal{N}_j}$
    return $\bar{C}$
end procedure
procedure EdgeAggregate($l, \{c_i^l\}_{i \in S^l}$)
    for $j = 1, \dots, J$ do
        $C_j^{l^{'}} \leftarrow \mathcal{N}_j^l \cdot C_j^l$
    end for
    for each $C_j^{l^{'}}$ do
        for each $c_i^l$ do
            if $c_{i,j}^{l,old}$ is not empty then
                $C_j^{l^{'}} \leftarrow C_j^{l^{'}} - c_{i,j}^{l,old} + c_{i,j}^l$
            else
                $C_j^{l^{'}} \leftarrow C_j^{l^{'}} + C_j^l$
                $\mathcal{N}_j^l \leftarrow \mathcal{N}_j^l + 1$
            end if
            $C_i^{l,old} \leftarrow c_i^l$
        end for
    end for
     return $C^l$
end procedure
procedure LocalUpdate
    return $C_i^l$
end procedure
```