TP 2: The Parameter-Server Architecture

Othmane MARFOQ, Chuan Xu 11/03/2021

Introduction

The purpose of this session is to implement and run distributed gradient descent using the parameter-server architecture [2].

Algorithm 1: Distributed Subgradient Descent

for iterations $t = 0, \dots, T$ **do**
  server broadcasts w^{t-1} to the **clients**;
  for client $r = 1, \dots, m$ **in parallel** *over* m **clients do**
    load a part of training data $\{y_{i_k}, x_{i_k}\}_{k=1}^{n_r}$;
    $g_r^{(t)} \leftarrow \sum_{k=1}^{n_r} \partial l(x_{i_k}, y_{i_k}, w_r^{(t)})$;
    push $g_r^{(t)}$ to server ;
  end
  $g^{(t)} \leftarrow \sum_{r=1}^{m} g_r^{(t)}$;
  $w^{(t+1} \leftarrow w^{(t)} - \eta g^{(t)}$;
end

Tasks

- First, clone / download the code corresponding to this session from the class repository.
- In this session, we will consider MNIST [1] dataset as example. We prepared a script that splits this dataset across clients (`data/generate_data.py`). Run this script in order to download and split the dataset.
- In `models.py`, we implemented a linear model. In the same way implement a deep network with two convolution and two dense layers.
- Complete `Server` class in `server.py`
- Complete `Client` class in `client.py`
- Complete the function `main` in `main.py` by precising the model, criterion, metric and device
- reserve resources using `oarsub`.
- run the code using `python -m torch.launch.distributed`

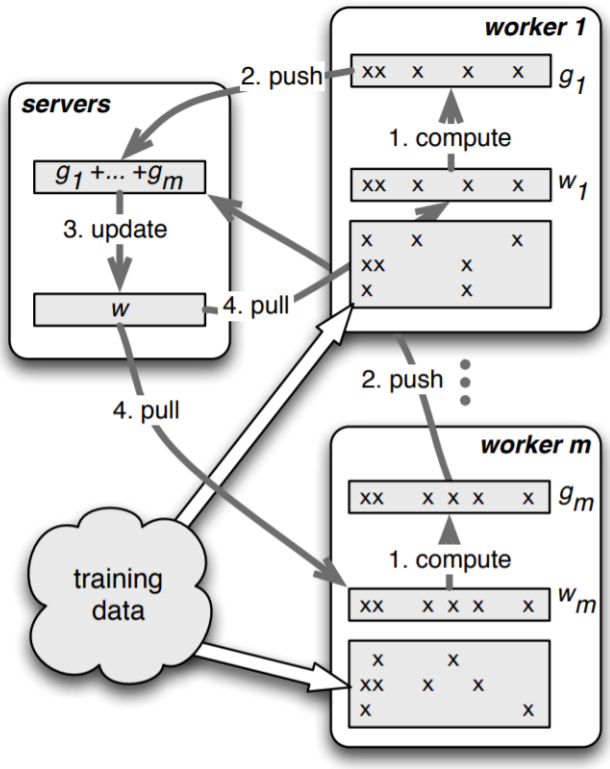

Figure 1: Steps required in performing distributed subgradient descent

References

[1] Yann LeCun and Corinna Cortes. “MNIST handwritten digit database”. In: (2010). URL: http://yann.lecun.com/exdb/mnist/.

[2] Mu Li. “Scaling Distributed Machine Learning with the Parameter Server”. In: *Proceedings of the 2014 International Conference on Big Data Science and Computing*. BigDataScience ’14. Beijing, China: Association for Computing Machinery, 2014. ISBN: 9781450328913. DOI: 10.1145/2640087.2644155. URL: https://doi.org/10.1145/2640087.2644155.