Decentralized Learning made Practical

Martijn de Vos
martijn.devos@epfl.ch

In collaboration with Anne-Marie Kermarrec, Rishi Sharma, Akash Dhasade, Johan Pouwelse and Erick Lavoie.

Grenoble, 14.12.2023
Distributed Learning with Decentralized Data

Federated Learning (FL)

Global model

Local data

Local updates

Data stay where it is generated. Learning happens by model exchange.

Decentralized Learning (DL)

Local model aggregation
Decentralized Learning

Train

Share
Decentralized Learning

Receive
Aggregate
Improving Decentralized Learning

1. **Churn** impacts convergence
2. **Global aggregation** improves convergence
3. Not all nodes need to work each round (theory in [1])

---

The Design of Plexus

**High-level algorithm**

1. Each round, a unique subset of online nodes, or a *sample*, train the model.

2. These nodes send their trained model to an *aggregator*.

3. The aggregator aggregates models and sends the model to the next sample.

*Resembles Federated Learning, but without server.*
Three Technical Challenges (TCs)

**TC1:** How can nodes derive samples?

**TC2:** How can Plexus avoid selecting offline nodes during sampling?

**TC3:** How can Plexus ensure system progression when nodes go offline during training or aggregation?
TC1: How can nodes derive samples?

- Important that different nodes derive the same sample.
- Each peer stores all peer IDs in a **local view**.
- Each peer samples peer for the next round using its population view.
- Aggregator selected like this as well, based on bandwidth.
TC1: How can nodes derive samples?

**Algorithm 1** Sampling by node $i$ where $k$ denotes the round number and $s$ is the requested sample size.

1: **Require:** Ping timeout $\Delta t_p$

2: 

3: **procedure** $\text{SAMPLE}(k, s)$

4: /* $\text{ACTIVES}()$ are the online nodes in local views */

5: $H \leftarrow \text{SORT}([\text{HASH}(j + k) \text{ for } j \text{ in } \text{ACTIVES}()])$

6: $C \leftarrow [j \text{ for } h_j \text{ in } H]$ $\triangleright$ Candidate identifiers

7: **return** the first $s$ in $C$ that answer a ping within $\Delta t_p$
TC2: How can Plexus avoid selecting offline nodes during sampling?

- Nodes send a join or leave message to other random nodes.
- Nodes keep track of the membership status of other nodes in their local view.
- Local views are gossiped and merged between nodes.
TC3: How can Plexus ensure system progression when nodes go offline during training or aggregation?

**Participant failure**
- Aggregator proceeds when
  - Received $f < s$ trained models
  - After some aggregation timeout

**Aggregator failure**
- Aggregator sends ACK message to previous participants when finished.
- Participants await ACK message
  - Retry with another aggregator after some timeout.
Experiment Setup (1/2)

- Implemented Plexus in Python 3 using PyTorch.
- Evaluation on the DAS6 compute cluster.

- Metrics:
  1. Time-to-accuracy
  2. Communication-to-accuracy
  3. Training-resources-to-accuracy
Experiment Setup (2/2)

- Four datasets:

<table>
<thead>
<tr>
<th>DATASET</th>
<th>TASK</th>
<th>NODES</th>
<th>LEARNING PARAMETERS</th>
<th>MODEL</th>
<th>MODEL SIZE</th>
</tr>
</thead>
<tbody>
<tr>
<td>CIFAR10</td>
<td>Image classification</td>
<td>1000</td>
<td>$\eta = 0.002, \text{momentum} = 0.9$</td>
<td>CNN (LeNet [20])</td>
<td>346 KB</td>
</tr>
<tr>
<td>CelebA</td>
<td>Image classification</td>
<td>500</td>
<td>$\eta = 0.001$</td>
<td>CNN</td>
<td>124 KB</td>
</tr>
<tr>
<td>FEMNIST</td>
<td>Image classification</td>
<td>355</td>
<td>$\eta = 0.004$</td>
<td>CNN</td>
<td>6.7 MB</td>
</tr>
<tr>
<td>MovieLens</td>
<td>Recommendation</td>
<td>610</td>
<td>$\eta = 0.2$, embedding dim = 20</td>
<td>Matrix Factorization</td>
<td>827 KB</td>
</tr>
</tbody>
</table>

- Three baselines
  - D-PSGD (sparsely connected topology)
  - D-PSGD (k-regular topology)
  - Gossip Learning
Plexus Compared to DL Baselines (FEMNIST)

Plexus shows significant performance improvements over baselines!
Plexus Overhead

[Diagram showing the percentage of communication volume and messages for datasets CIFAR10, CelebA, FEMNIST, and MovieLens.]
Conclusions

● Plexus is a practical and efficient DL system.

● Significant savings in time-to-accuracy (1.2-8.3x), communication-to-accuracy (2.4-15.3x) and resource-to-accuracy (6.4-370x).

● Future work: dealing with Byzantine nodes.
Thank you!

martijn.devos@epfl.ch