Gossiping GANs

Corentin Hardy*
Erwan Le Merrer** — Bruno Sericola**

*Technicolor & Inria    **Inria

DIDL 2018
1 Introduction
   • Motivations
   • GAN over a spread dataset

2 Experiments
   • Competitors and experimental setup
   • Experimental setup
   • Results
   • Case of non i.i.d spread dataset

3 Discussion
1 Introduction
   • Motivations
   • GAN over a spread dataset

2 Experiments
   • Competitors and experimental setup
   • Experimental setup
   • Results
   • Case of non i.i.d spread dataset

3 Discussion
Applications related to GAN

Gossiping GANs
GAN in a nutshell

Generative adversarial network\(^1\) (GAN)

A GAN is composed of two components: a *generator* \(\mathcal{G}\) and a *discriminator* \(\mathcal{D}\).

The goal of a GAN is to generate new samples with the same distribution of a training dataset.

\(\mathcal{G}\) and \(\mathcal{D}\) are two ML models (DNNs).

---

\(^1\)Goodfellow *et al.* ”Generative adversarial nets.” (2014)
Training a GAN means learning $D$ and $G$ with adversary losses:

- the discriminator $D$ tries to minimize:
  $$L_D = \mathbb{E}_{x \sim P_{data}} [\log D(x)] + \mathbb{E}_{x \sim P_G} [\log(1 - D(x))]$$
- the generator $G$ tries to maximize: $$L_G = \mathbb{E}_{x \sim P_G} [\log D(x)]$$
Adversarial loss functions

Training a GAN means learning $D$ and $G$ with adversary losses:

• the discriminator $D$ tries to minimize:
  
  $L_D = \mathbb{E}_{x \sim P_{data}} [\log D(x)] + \mathbb{E}_{x \sim P_G} [\log(1 - D(x))]$

• the generator $G$ tries to maximize: $L_G = \mathbb{E}_{x \sim P_G} [\log D(x)]$

![Diagram of adversarial loss functions](image)
Training a GAN means learning $D$ and $G$ with adversary losses:

- the discriminator $D$ tries to minimize:
  $$L_D = \mathbb{E}_{x \sim P_{data}} [\log D(x)] + \mathbb{E}_{x \sim P_G} [\log(1 - D(x))]$$
- the generator $G$ tries to maximize:
  $$L_G = \mathbb{E}_{x \sim P_G} [\log D(x)]$$
Adversarial loss functions

Training a GAN means learning $\mathcal{D}$ and $\mathcal{G}$ with adversary losses:

- the discriminator $\mathcal{D}$ tries to minimize:
  $$L_D = \mathbb{E}_{x \sim P_{data}} [\log D(x)] + \mathbb{E}_{x \sim P_G} [\log(1 - D(x))]$$

- the generator $\mathcal{G}$ tries to maximize: $L_G = \mathbb{E}_{x \sim P_G} [\log D(x)]$
How train a GAN over a spread dataset?
How train a GAN over a spread dataset?

![Diagram of Gossiping GANs](image-url)
How train a GAN over a spread dataset?
How train a GAN over a spread dataset?
How train a GAN over a spread dataset?

\[ \bar{g} \leftarrow \frac{1}{N} \sum_i g_i \]

\[ \bar{D} \leftarrow \frac{1}{N} \sum_i D_i \]
How train a GAN over a spread dataset?

\[ \hat{g} \leftarrow \frac{1}{N} \sum \hat{g}_i \]

\[ \hat{D} \leftarrow \frac{1}{N} \sum \hat{D}_i \]

\[ \mathcal{D}_1 \leftarrow \mathcal{D} \]

\[ \hat{g}_1 \leftarrow \hat{g} \]
How train a GAN over a spread dataset?
How train a GAN over a spread dataset?

\[ \hat{g} \leftarrow \frac{1}{N} \sum_i g_i \]
\[ \hat{D} \leftarrow \frac{1}{N} \sum_i D_i \]
How train a GAN over a spread dataset?
How train a GAN over a spread dataset?
Federated Learning

All-reduce without PS

Corentin Hardy (Technicolor, Inria)

Gossiping GANs

DIDL 2018
All-reduce without PS

\[ G_5 \leftarrow \frac{1}{N} \sum_i G_i \]
\[ D_5 \leftarrow \frac{1}{N} \sum_i D_i \]
Gossip methods

\[ G_5 \leftarrow \frac{1}{2} (G_5 + G_2) \]
\[ D_5 \leftarrow \frac{1}{2} (D_5 + D_2) \]
Gossip methods

\[ G_5 \leftarrow \frac{1}{2} (G_5 + G_3) \]
\[ D_5 \leftarrow \frac{1}{2} (D_5 + D_3) \]
## Summary

<table>
<thead>
<tr>
<th>Methods</th>
<th>Communication per worker</th>
<th>Decentralized</th>
</tr>
</thead>
<tbody>
<tr>
<td>Federated Learning</td>
<td>$2(</td>
<td>G</td>
</tr>
<tr>
<td>All-reduce without PS</td>
<td>$N(</td>
<td>G</td>
</tr>
<tr>
<td>Gossip method</td>
<td>$</td>
<td>G</td>
</tr>
</tbody>
</table>

### Gossip-based method

- More scalable in term of communications.
- Should decreases the learning performances.

Question: In the case of GANs, does gossip-based method not decrease too much performances of the final model?

---

3Existing gossip method for classical DNN: M. Blot et al. ”Gossip training for deep learning” (2016)
1 Introduction
   - Motivations
   - GAN over a spread dataset

2 Experiments
   - Competitors and experimental setup
   - Experimental setup
   - Results
   - Case of non i.i.d spread dataset

3 Discussion
The different communications setups

Competitors:

a) Stand-alone (no communication)

b) Federated Learning (all-reduced)

c) Gossip DDL ($\mathcal{G}_i$ and $\mathcal{D}_i$ are dependents)

d) Gossip DDL_ind ($\mathcal{G}_i$ and $\mathcal{D}_i$ are independents)

Figure: Red and blue arrows represent $\mathcal{G}_i$ and $\mathcal{D}_i$ movement.
Experimental setup

We emulate up to 100 workers on a large server to evaluate performances of Gossip DDL against the competitors.

- $G$ and $D$ are two DNN models.
- Each worker performs 20,000 iterations during the training.
- All communications are synchronized every $K = 200$ iterations.
- Each machine hosts $\frac{1}{N}$ of the training dataset (MNIST) randomly i.i.d. split.
- The MNSIT score (Inception score adapted to MNIST) and the Fréchet Inception Distance (adapted to MNIST) of all generators is computed every 1,000 iterations.
Performances of GAN during the training

![Graphs showing FID and MNIST Score for different values of N (1, 2, 10, 100) during the training process.](image)
Final scores and generated samples
Experiment with non i.i.d data (N=10)

Gossiping GANs

Corentin Hardy (Technicolor, Inria)

DIDL 2018 17 / 18
Conclusion

• Gossip performances are closed to federated learning.
• Considering $G_i$ and $D_i$ independents slightly improves the final score.
• The distribution of data on machines is crucial for GANs!

Future works

• Explore solutions in the case of non i.i.d. spread dataset.
• Understand the potential of GAN trained on a spread dataset (data-augmentation?)