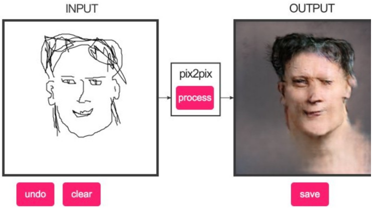
Gossiping GANs

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

*Technicolor & Inria **Inria

DIDL 2018

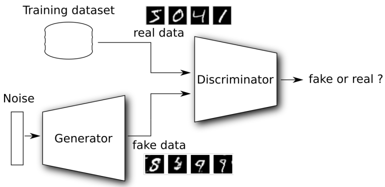
Applications related to GAN



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).

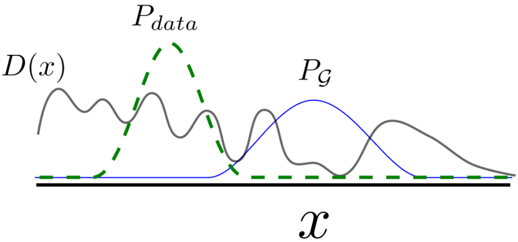


[1] Goodfellow *et al.* "Generative adversarial nets." (2014)

Adversarial loss functions

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

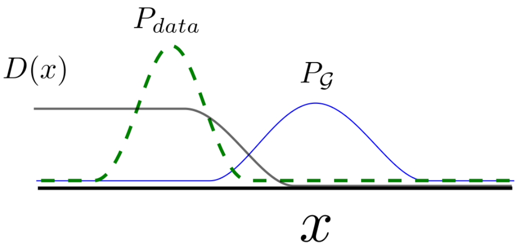
- the discriminator $\mathcal{D}$ tries to minimize: $L_D = \mathbb{E}_{x \sim P_{data}} [\log D(x)] + \mathbb{E}_{x \sim P_{\mathcal{G}}} [\log(1 - D(x))]$
- the generator $\mathcal{G}$ tries to maximize: $L_G = \mathbb{E}_{x \sim P_{\mathcal{G}}} [\log D(x)]$



Adversarial loss functions

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

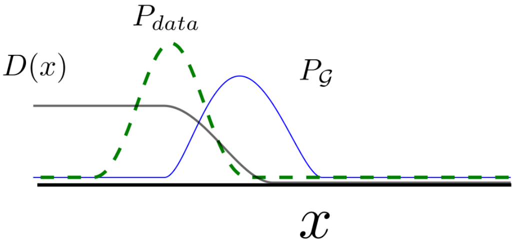
- the discriminator $\mathcal{D}$ tries to minimize:
  $L_D = \mathbb{E}_{x \sim P_{data}} [\log D(x)] + \mathbb{E}_{x \sim P_{\mathcal{G}}} [\log(1 - D(x))]$
- the generator $\mathcal{G}$ tries to maximize: $L_G = \mathbb{E}_{x \sim P_{\mathcal{G}}} [\log D(x)]$



Adversarial loss functions

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

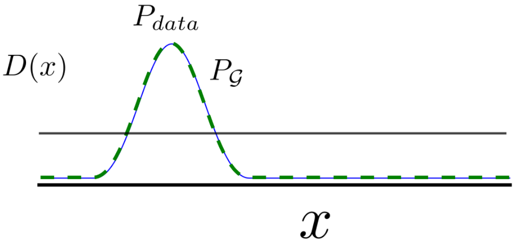
- the discriminator $\mathcal{D}$ tries to minimize: $L_D = \mathbb{E}_{x \sim P_{data}} [\log D(x)] + \mathbb{E}_{x \sim P_{\mathcal{G}}} [\log(1 - D(x))]$
- the generator $\mathcal{G}$ tries to maximize: $L_G = \mathbb{E}_{x \sim P_{\mathcal{G}}} [\log D(x)]$



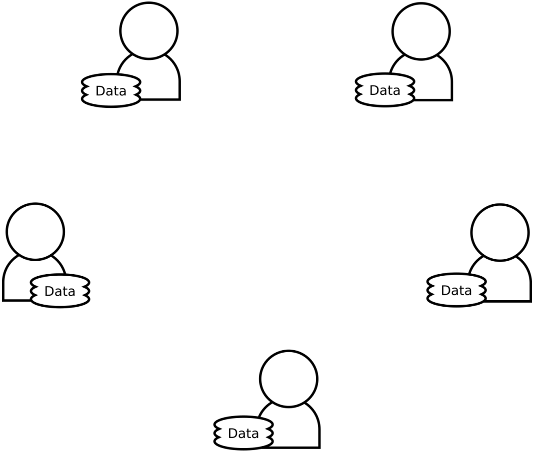
Adversarial loss functions

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

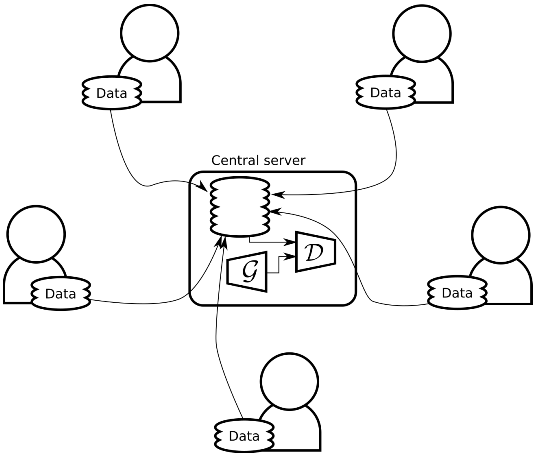
- the discriminator $\mathcal{D}$ tries to minimize:
  $L_D = \mathbb{E}_{x \sim P_{data}} [\log D(x)] + \mathbb{E}_{x \sim P_{\mathcal{G}}} [\log(1 - D(x))]$
- the generator $\mathcal{G}$ tries to maximize: $L_G = \mathbb{E}_{x \sim P_{\mathcal{G}}} [\log D(x)]$



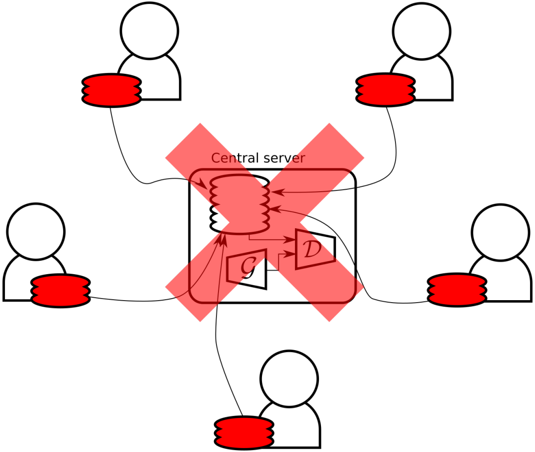
How train a GAN over a spread dataset ?



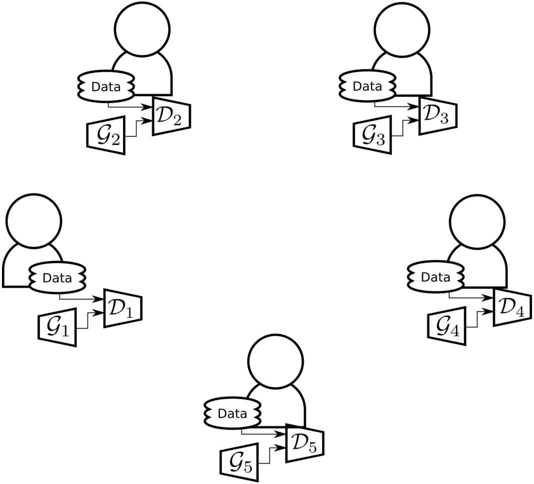
How train a GAN over a spread dataset ?



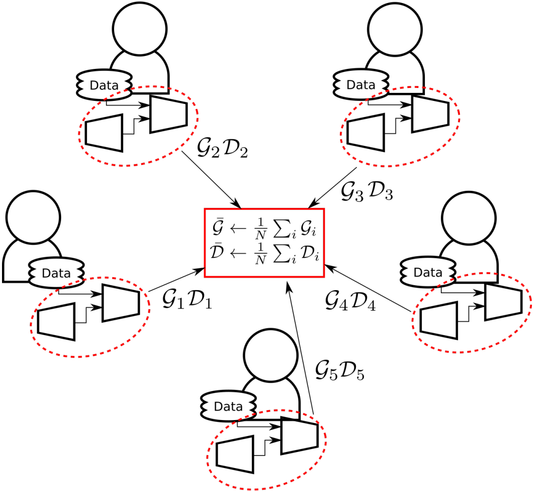
How train a GAN over a spread dataset ?



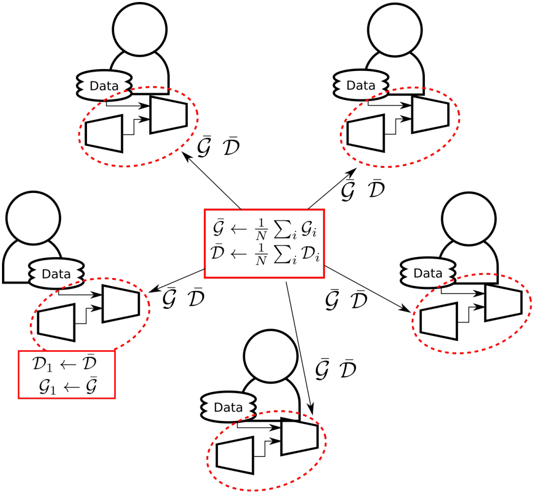
How train a GAN over a spread dataset ?



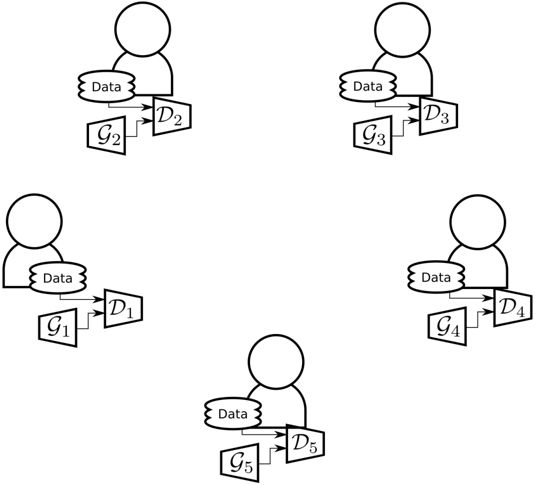
How train a GAN over a spread dataset ?



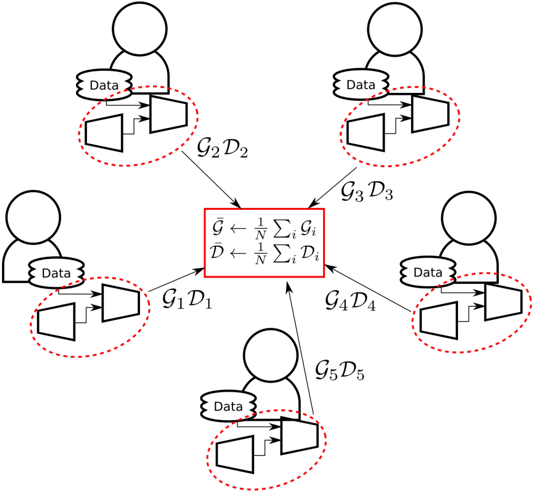
How train a GAN over a spread dataset ?



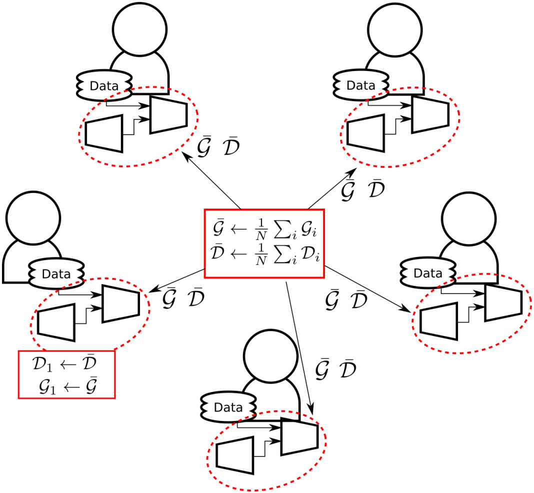
How train a GAN over a spread dataset ?



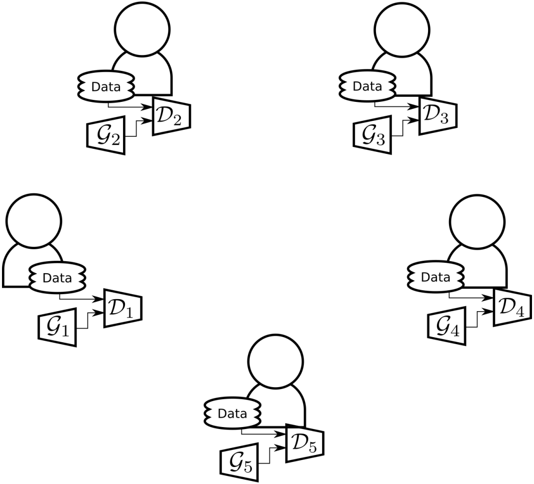
How train a GAN over a spread dataset ?



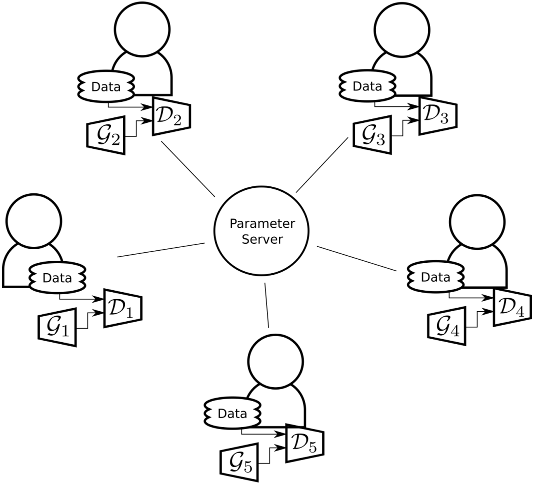
How train a GAN over a spread dataset ?



How train a GAN over a spread dataset ?

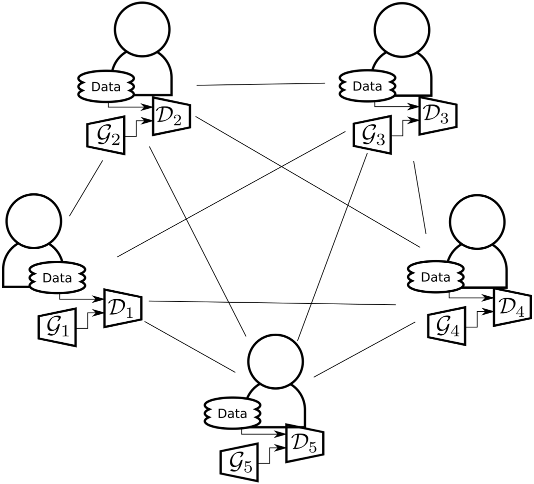


Federated Learning[2]

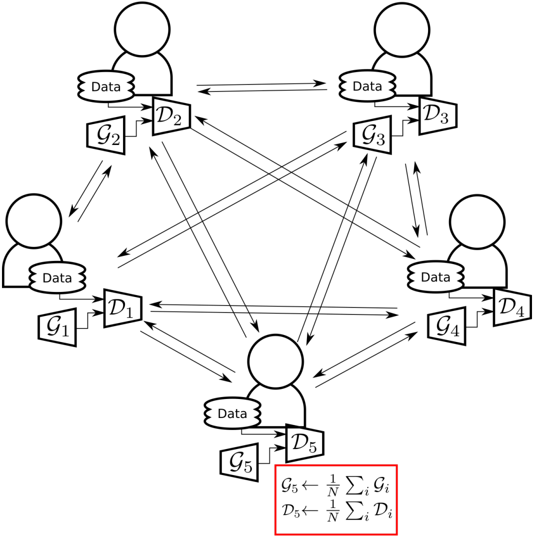


[2] McMahan, H. Brendan, et al. "Communication-efficient learning of deep networks from decentralized data." (2016)

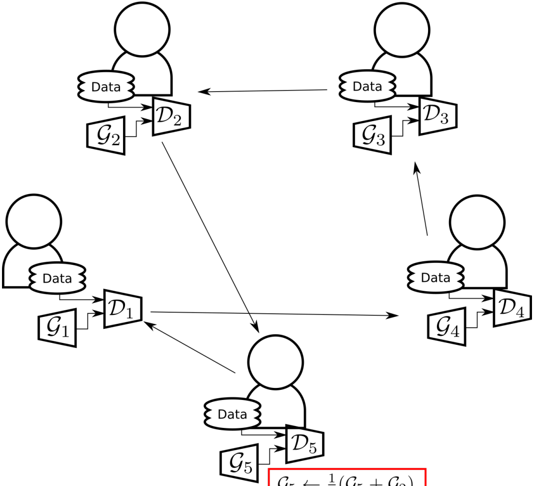
All-reduce without PS



All-reduce without PS

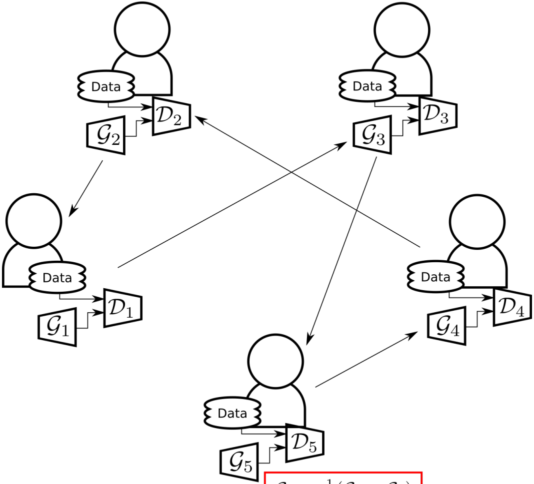


Gossip methods



$$\mathcal{G}_5 \leftarrow \frac{1}{2}(\mathcal{G}_5 + \mathcal{G}_2)$$
$$\mathcal{D}_5 \leftarrow \frac{1}{2}(\mathcal{D}_5 + \mathcal{D}_2)$$

Gossip methods



$$\boxed{\begin{array}{l}\mathcal{G}_5 \leftarrow \frac{1}{2}(\mathcal{G}_5 + \mathcal{G}_3) \\ \mathcal{D}_5 \leftarrow \frac{1}{2}(\mathcal{D}_5 + \mathcal{D}_3)\end{array}}$$

Summary

Methods	Communication per worker	Decentralized
Federated Learning	$2(\|\mathcal{G}\| + \|\mathcal{D}\|)$	No (PS)
All-reduce without PS	$N(\|\mathcal{G}\| + \|\mathcal{D}\|)$	Yes
Gossip method	$\|\mathcal{G}\| + \|\mathcal{D}\|$	Yes

Gossip-based method [3]

- 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 ?

[3] Existing gossip method for classical DNN : M. Blot et al. "Gossip training for deep learning" (2016)

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)

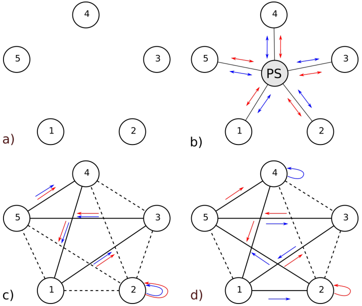


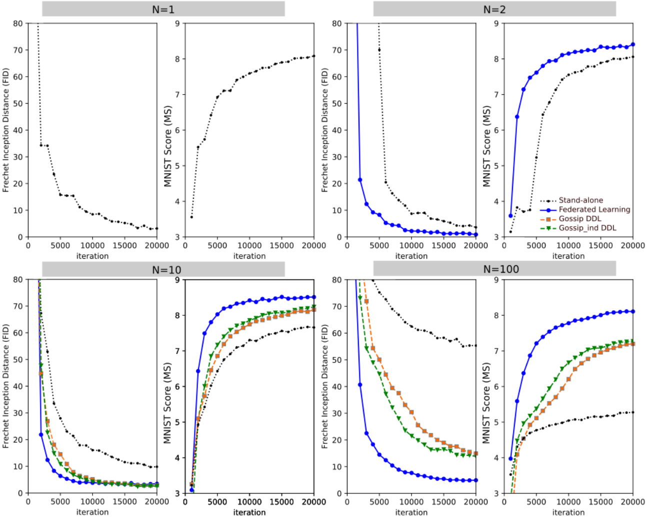
Figure: Red and blue arrows represent $\mathcal{G}_i$ and $\mathcal{D}_i$ movement.

Experimental setup

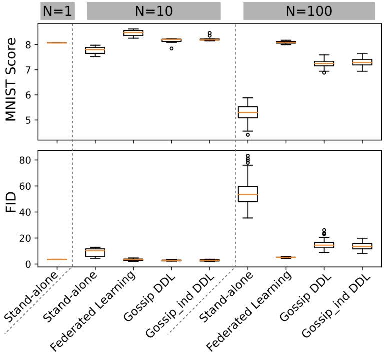
We emulate up to 100 workers on a large server to evaluate performances of Gossip DDL against the competitors.

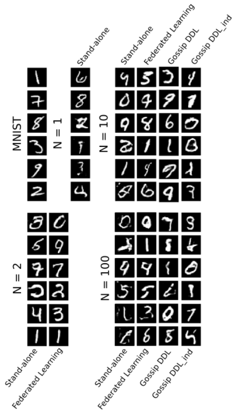
- $\mathcal{G}$ and $\mathcal{D}$ are two DNN models.
- Each worker performs $20,000$ iterations during the training.
- All communications are synchronized every $K = 200$ iterations.
- Each machine hosts $\frac{1}{N}$ of the training dataset (MNIST) randomly i.i.d. split.
- The MNSIT score (Inception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

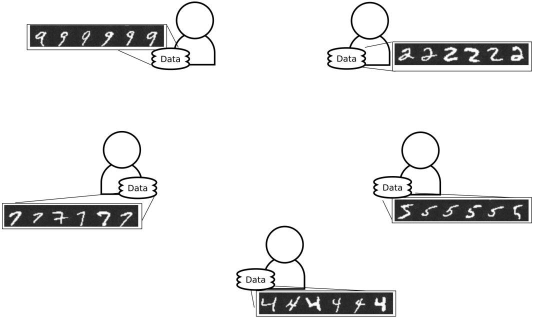


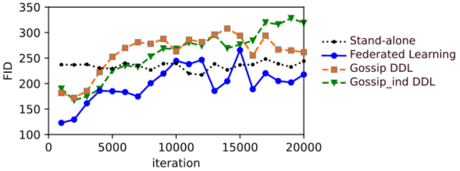
Final scores and generated samples





Experiment with non i.i.d data (N=10)





Conclusion

- Gossip performances are closed to federated learning.
- Considering $\mathcal{G}_i$ and $\mathcal{D}_i$ 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?)