# Progressive Augmentation of GANs (PA-GANs)

arxiv github

Dan Zhang, Anna Khoreva
Bosch Center for Artificial Intelligence

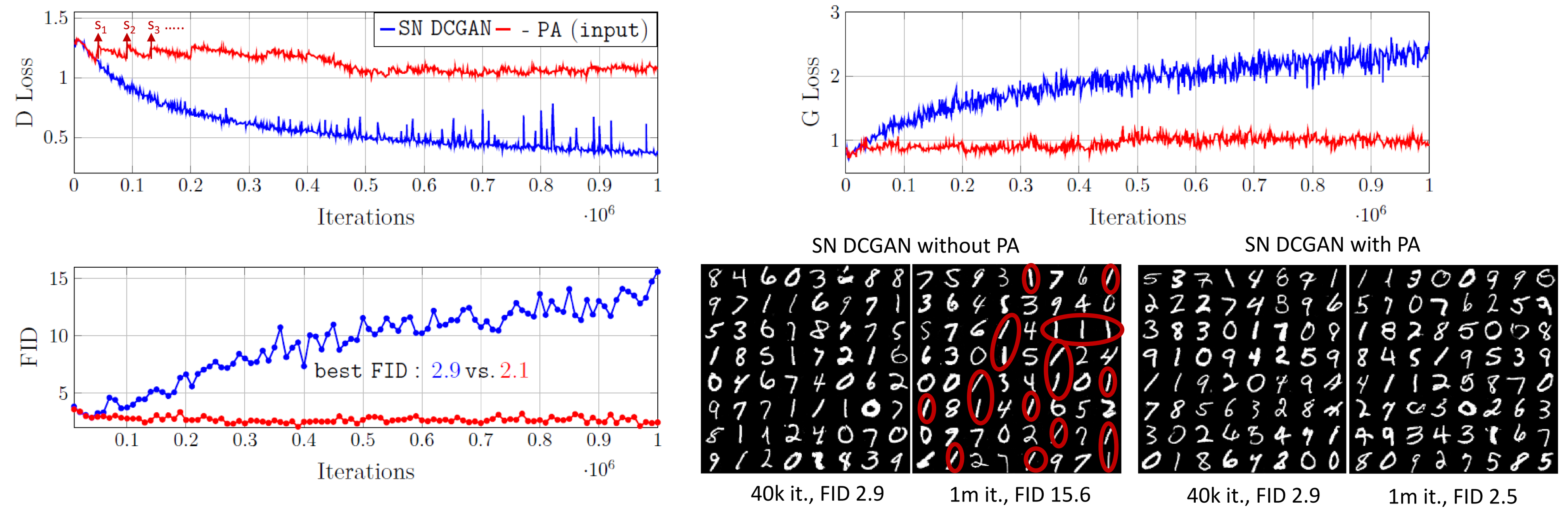
## 1 Motivation

**Generative adversarial network (GAN)** is powerful, but notoriously hard to train:

- Sensitive to learning parameters and training schedule;
- Suffers from mode collapse; …

**Commonly occurring problem**:
Discriminator becomes overconfident, providing a non-informative signal to train the generator.

**Our solution**:
*Progressive augmentation (PA)* of discriminator (D) to gradually increase its task difficulty in order to enable continuous learning of generator (G).




*PA prevents mode collapsing to a subset of classes (e.g., digit 1)*

✓ Novel mechanism to balance the two-player game;
✓ Theoretically justified (preserves the original GAN objective);
✓ Alleviates (partially) the need for fine hyper-parameter tuning;
✓ Improves generated sample diversity.

## 2 Key Idea

- **Augment** D's input (or its features) $\boldsymbol{x}$ with random bit $\boldsymbol{s}$ and cast binary classification of D into ***XOR*** computation between $\boldsymbol{x}$ and $\boldsymbol{s}$.
- **Progressively** add more bits $\boldsymbol{s}=\{s_1, \ldots, s_L\}$ to $\boldsymbol{x}$ in order to gradually complicate D's task during training.

## 3 Method

**Min-Max game behind GANs [1]:**

$$\min_G \max_D \mathbb{E}_{\mathbf{x}\sim\mathbb{P}_d}\{\log[D(\mathbf{x})]\} + \mathbb{E}_{\mathbf{x}\sim\mathbb{P}_g}\{\log[1-D(\mathbf{x})]\}$$

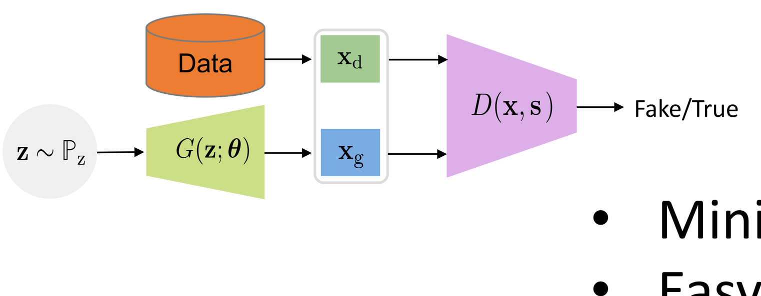
The discriminator (D) tells the samples from the data distribution $\mathbb{P}_d$ and generative model distribution $\mathbb{P}_g$. The generator (G) challenges D by making synthetic samples into data samples, i.e., $\mathbb{P}_g \to \mathbb{P}_d$.

**Theorem of PA-GANs.** The min-max optimization problem of GANs [1] is equivalent to

$$\min_G \max_D \mathbb{E}_{(\mathbf{x},\mathbf{s}_l)\sim\mathbb{P}}\{\log[D(\mathbf{x},\mathbf{s}_l)]\} + \mathbb{E}_{(\mathbf{x},\mathbf{s}_l)\sim\mathbb{Q}}\{\log[1-D(\mathbf{x},\mathbf{s}_l)]\} \quad \forall l \in \{1,2,\ldots,L\}$$

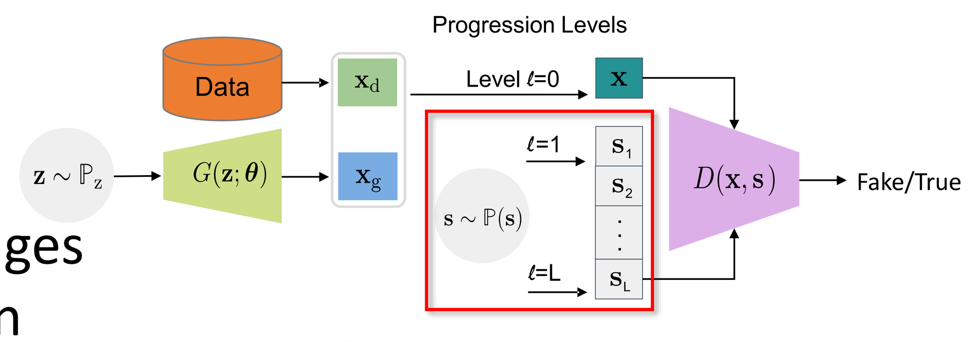
$$\mathbb{P} \triangleq \begin{cases}\prod_l \mathbb{P}_s(s_l)\mathbb{P}_d(\mathbf{x}) & \text{if } \mathrm{XOR}(\mathbf{s}_l)=0\\ \prod_l \mathbb{P}_s(s_l)\mathbb{P}_g(\mathbf{x}) & \text{if } \mathrm{XOR}(\mathbf{s}_l)=1\end{cases} \quad \mathbb{Q} \triangleq \begin{cases}\prod_l \mathbb{P}_s(s_l)\mathbb{P}_d(\mathbf{x}) & \text{if } \mathrm{XOR}(\mathbf{s}_l)=1\\ \prod_l \mathbb{P}_s(s_l)\mathbb{P}_g(\mathbf{x}) & \text{if } \mathrm{XOR}(\mathbf{s}_l)=0\end{cases} \quad \mathbb{P}_s(s_l)=0.5 \quad s_l\in\{0,1\}$$

### Network Layout



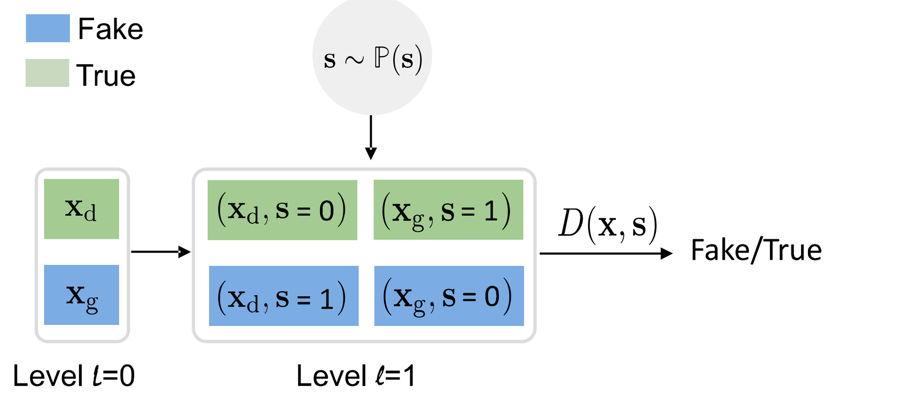

GAN

- Minimum changes
- Easy integration



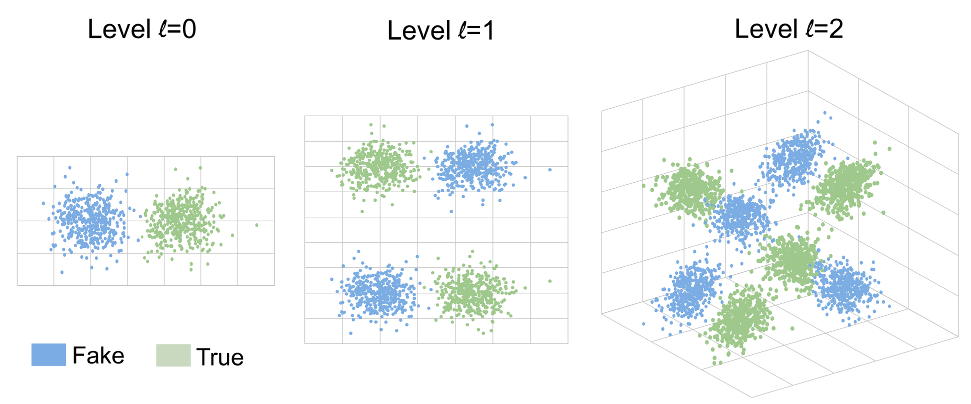

PA-GAN

### Augmentation principle:




- $x_d \sim P_d$ ⇔ 0, $x_g \sim P_g$ ⇔1, and s ∈ {0,1};
- XOR(x, **s**): 0 (1) ⇔ True(FAKE);
- D preserves the task of estimating the JSD between $P_d$ and $P_g$ under PA.

### Progression mechanism:




- Automatic scheduling;
- D's task difficulty gradually increases via augmentation progression.

## 4 Experiments

### Comparison across different architectures and datasets

| Method | PA | F-MNIST | CIFAR10 | CELEBA-HQ | T-ImageNet | $\overline{\Delta\text{PA}}$ |
|---|---|---|---|---|---|---|
| SN DCGAN [2] | ✗ | 10.6 | 26.0 | 24.3 | - | 4.2 |
| | input | **6.2** | **22.2** | 20.8 | - | |
| | feat | **6.2** | 22.6 | **18.8** | - | |
| SA GAN (sBN) [3] | ✗ | - | 18.8 | 17.8 | 47.6 | 2.6 |
| | input | - | **16.1** | **15.4** | 44.8 | |
| | feat | - | 16.3 | 15.8 | **44.7** | |

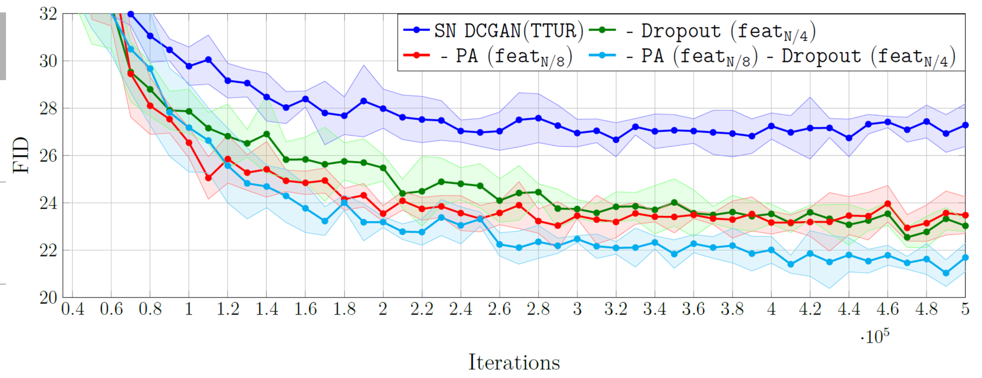
*On average PA achieves ~3 point improvement of FID score*

| $\mathrm{lr_g}$ \ $\mathrm{lr_d}$ | PA ($\mathrm{feat}_{N/8}$) | $10^{-4}$ | $2\times10^{-4}$ | $4\times10^{-4}$ | $10^{-3}$ | $\overline{\Delta\text{PA}}$ |
|---|---|---|---|---|---|---|
| $10^{-4}$ | ✗ | 27.0 | 25.8 | *25.3* | 27.0 | 3.5 |
| | ✓ | 23.3 | **22.2** | 22.6 | 22.9 | |
| $2\times10^{-4}$ | ✗ | 26.7 | 26.0 | 26.2 | 27.2 | 3.1 |
| | ✓ | 24.8 | 22.6 | 22.3 | 24.0 | |
| $4\times10^{-4}$ | ✗ | 28.7 | 26.1 | 26.3 | 28.2 | 3.6 |
| | ✓ | 24.7 | 23.3 | 22.9 | 24.2 | |
| $10^{-3}$ | ✗ | 28.5 | 27.0 | 26.4 | 27.4 | 2.9 |
| | ✓ | 25.7 | 23.6 | 23.4 | 25.0 | |

*Insensitive to hyper-parameter settings*

### Combination/Comparison with other regularization techniques

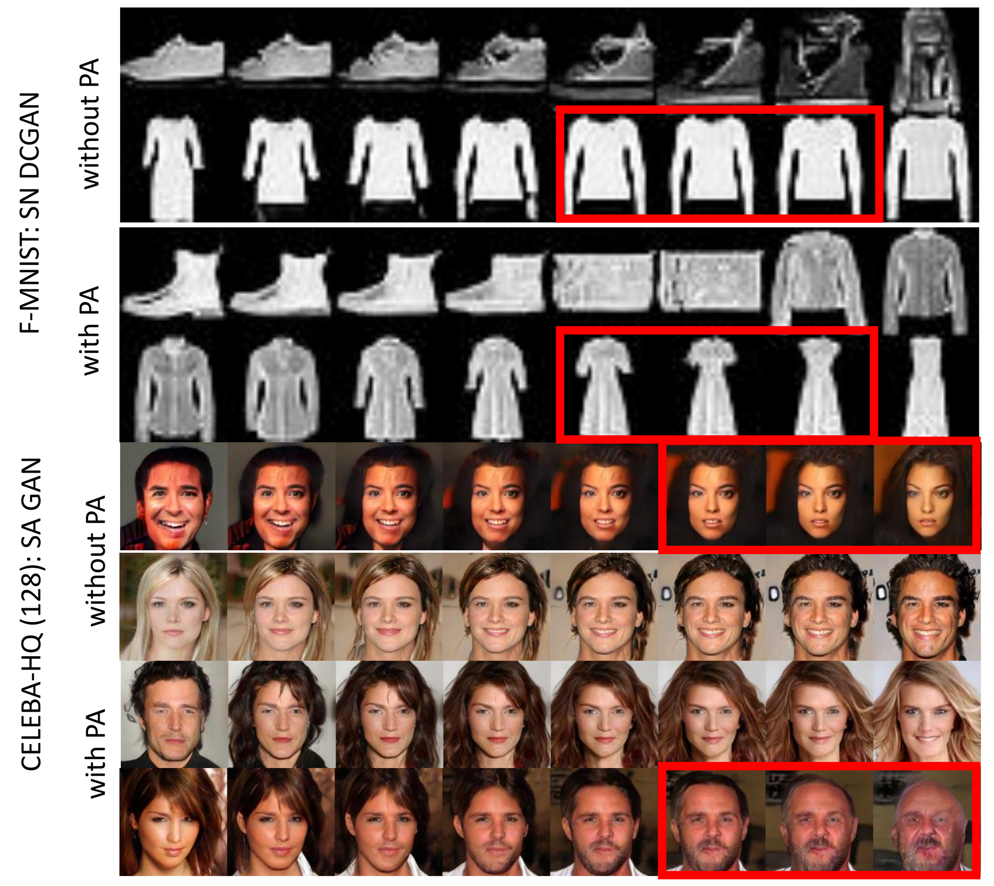
| Method | PA | GAN | -Label smooth. [4] | -GP [5] | -GP$_{\text{zero-cent}}$ [6] | -Dropout [7] | -SS [8] | $\overline{\Delta\text{PA}}$ |
|---|---|---|---|---|---|---|---|---|
| SN DCGAN [2] | ✗ | 26.0 | 25.8 | 26.7 | 26.5 | 22.1 | – | |
| | input | 22.2 | 23.1 | 21.8 | 22.3 | 21.9 | – | 3.0 |
| | feat | 22.6 | 22.3 | 22.7 | 23.0 | **20.6** | – | 3.1 |
| SA GAN (sBN) [3] | ✗ | 18.8 | – | 17.8 | 17.8 | 16.2 | 15.7 | |
| | input | 16.1 | – | 15.8 | 16.1 | 15.5 | **14.7** | 1.3 |
| | feat | 16.3 | – | 16.1 | 15.9 | 15.6 | 14.9 | 1.3 |
| | $\overline{\Delta\text{PA}}$ | 3.1 | 3.1 | 3.2 | 2.8 | 0.8 | 0.9 | 2.3 |




*Consistent and complementary performance gain*

### Qualitative results

- Images generated via latent space interpolation




*No close-by ones are look alike.*
*PA improves sample variation.*

- Randomly sampled images


*High-fidelity human face synthesis*

## 5 References

[1] Goodfellow, et al.: Generative adversarial nets, NIPS 2014
[2] Miyato, et al.: Spectral normalization for generative adversarial networks, ICLR 2018
[3] Zhang, et al.: Self-attention generative adversarial networks, ICML 2019
[4] Salimens, et al.: Improved techniques for training GANs, NIPS 2016
[5] Gulrajani, et al.: Improved training of Wasserstein GANs, NIPS 2017
[6] Roth, et al.: Stabilizing training of generative adversarial networks through regularization, NIPS 2017
[7] Srivastava, et al.: Dropout: a simple way to prevent neural networks from overfitting, JMLR 2014
[8] Chen, et al.: Self-supervised GANs via auxiliary rotation loss, CVPR 2019