Towards Addressing GAN Training Instabilities: Dual-Objective GANs with Tunable Parameters

Kyle Otstot

Arizona State University

June 9, 2023

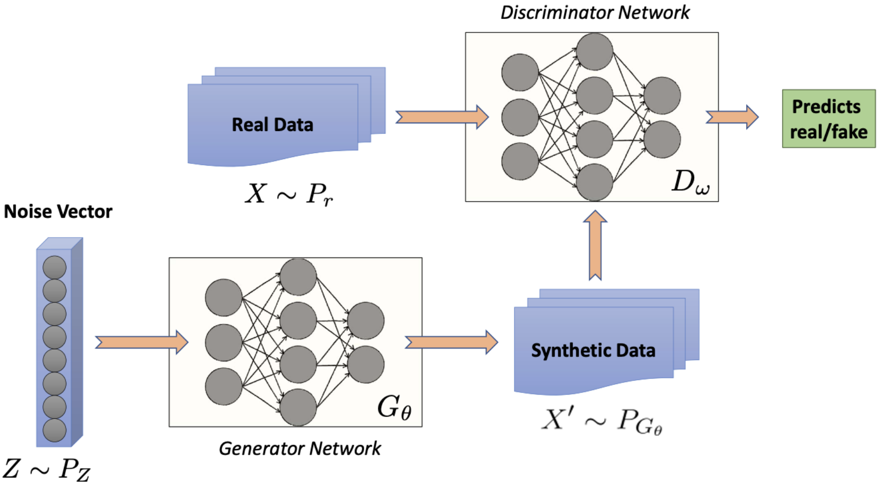
Graduate Supervisory Committee:
Lalitha Sankar, Chair
Oliver Kosut
Giulia Pedrielli

Thesis Outline

1. GAN Overview & Common Failures
2. Formulating the (α_D, α_G)-GAN
3. Experiments & Summary of Results

1. GAN Overview & Common Failures

Generative Adversarial Networks (GANs)



GAN: A Two Player Min-Max Game

- Adversarial min-max game between G_θ and D_ω

$$\inf_{\theta \in \Theta} \sup_{\omega \in \Omega} V(\theta, \omega)$$

- Goodfellow *et al.* (2014) introduced (now called) the *vanilla GAN*

$$V_{\mathrm{VG}}(\theta, \omega) = \mathbb{E}_{X \sim P_r}[\log D_\omega(X)] + \mathbb{E}_{Z \sim P_Z}[\log(1 - D_\omega(G_\theta(Z)))]$$

$D_\omega(x)$ is the probability that x is real, $x \in \mathcal{X}$

Vanilla GAN: Optimal Discriminator & Generator

$$V_{\mathrm{VG}}(\theta,\omega) = \mathbb{E}_{X\sim P_r}[\log D_\omega(X)] + \mathbb{E}_{Z\sim P_Z}[\log(1 - D_\omega(G_\theta(Z)))]$$

- Assuming sufficiently large Ω and fixed G_θ, the discriminator D_{ω^*} optimizing the sup of V_{VG} is given by

$$D_{\omega^*}(x) = \frac{p_r(x)}{p_r(x) + p_{G_\theta}(x)}$$

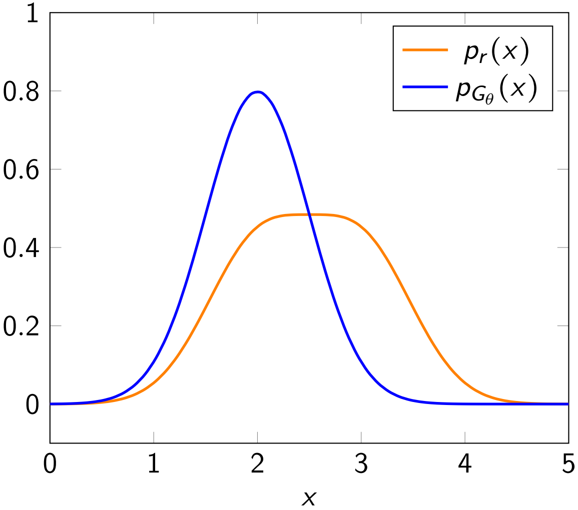
- Assuming sufficiently large Θ and optimal D_{ω^*}, the generator optimizing the inf of V_{VG} minimizes the **Jensen-Shannon Divergence** between p_r and p_{G_θ}
  - $p_r = p_{G_\theta}$ when $\forall_x D_\omega(x) = \frac{1}{2}$ and $D_{\mathrm{JS}}(p_r \| p_{G_\theta}) = 0$

Failures of the Vanilla GAN

- Although an elegant formulation, the vanilla GAN faces several challenges that threaten its training stability
  1. Exploding & vanishing gradients
  2. Mode collapse
  3. Model oscillation
- We illustrate these challenges with toy examples

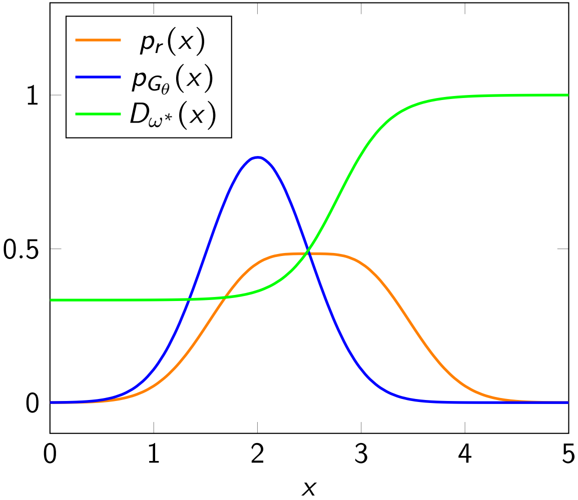
Exploding & Vanishing Gradients

- Cluster of generated data approaches real mode



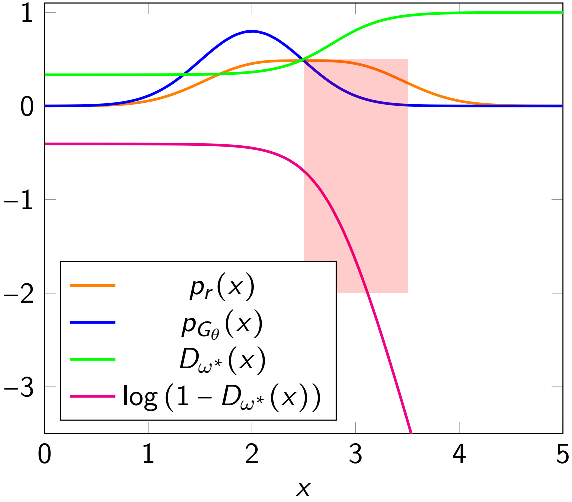
Exploding & Vanishing Gradients

- Discriminator updates to estimate $p_r(x)/\left(p_r(x) + p_{G_\theta}(x)\right)$

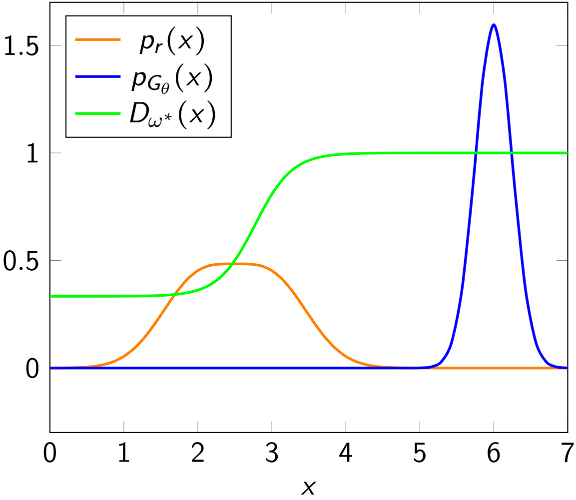


Exploding & Vanishing Gradients

- Rightmost generated samples receive steep gradients which heavily influence the next generator update

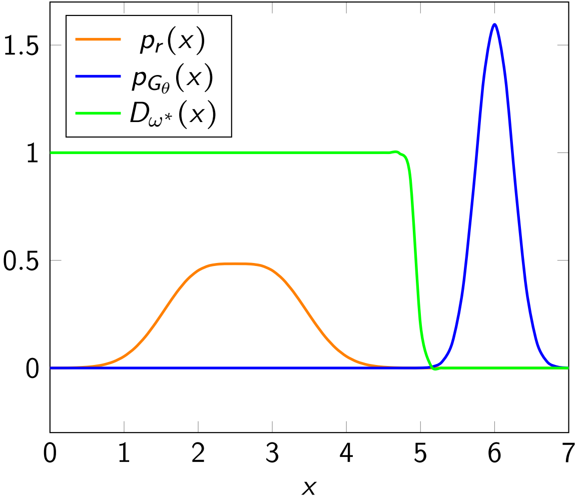


- Generated data overshoots mode toward the $D_{\omega^*}(x) \approx 1$ region



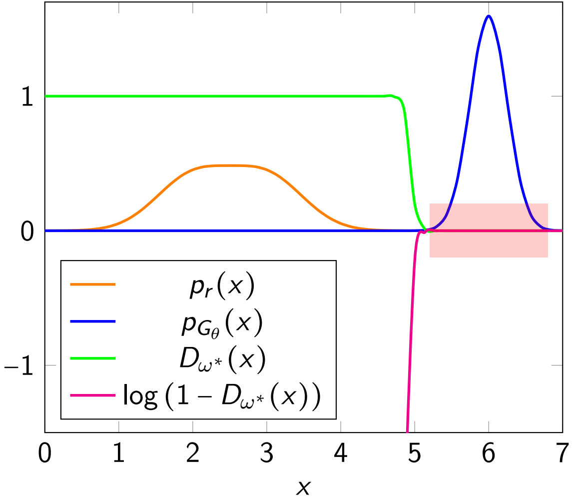
Exploding & Vanishing Gradients

- Discriminator updates with very confident predictions



Exploding & Vanishing Gradients

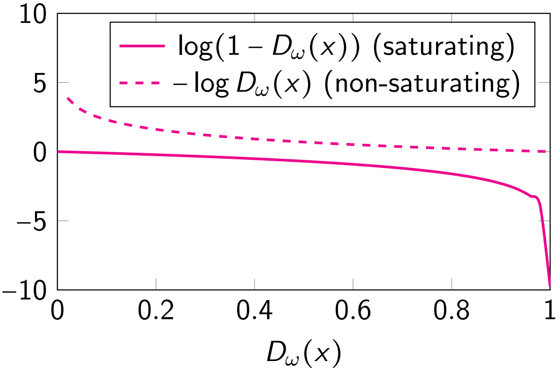
- Generated samples receive flat gradients, thus freezing G_θ



Non-Saturating Vanilla GAN

- To address exploding & vanishing gradients, Goodfellow *et al.* (2014) proposed the *non-saturating vanilla GAN* [1]

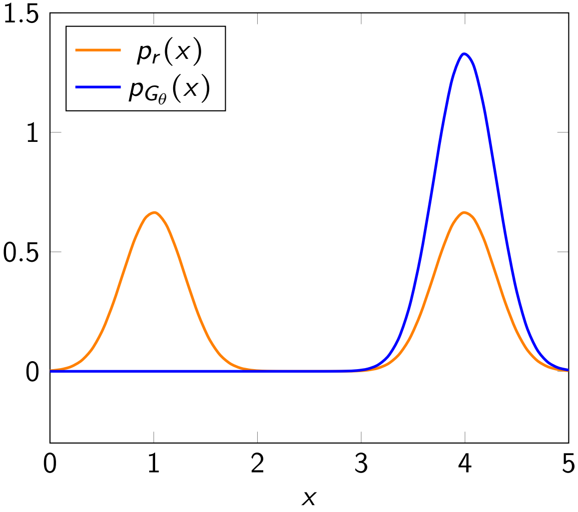
$$\sup_{\omega\in\Omega} V_{\mathrm{VG}}(\theta,\omega), \qquad \inf_{\theta\in\Theta} V_{\mathrm{VG}}^{\mathrm{NS}}(\theta,\omega) := \mathbb{E}_{X\sim P_{G_\theta}}\left[\log D_\omega(X)\right]$$



[1] First dual-objective GAN

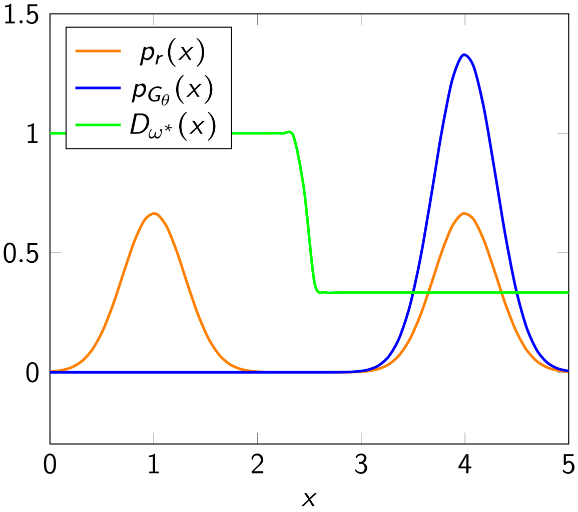
Mode Collapse

- Generated data fits onto real mode



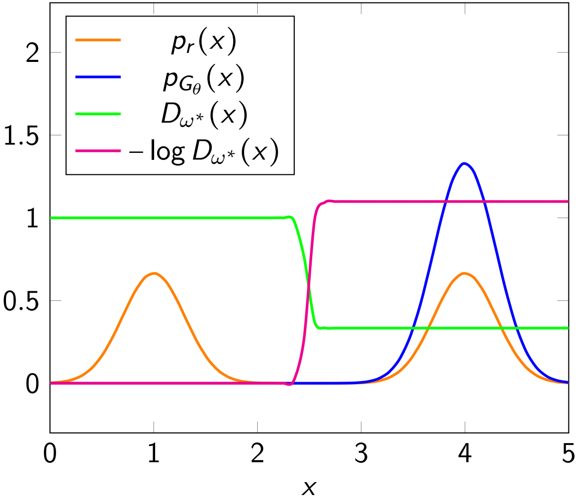
Mode Collapse

- Discriminator output is flat in dense p_{G_θ} region



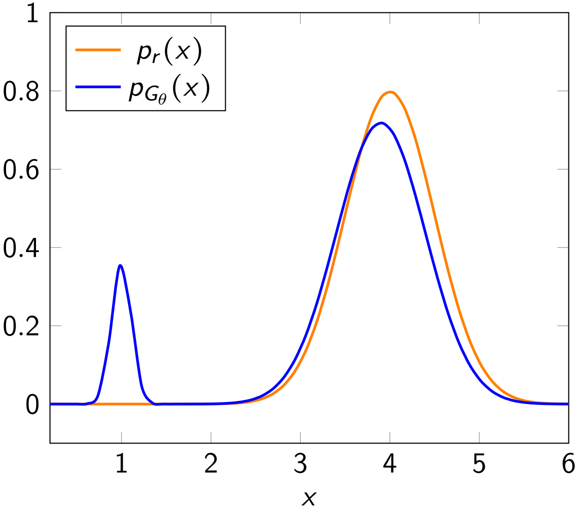
Mode Collapse

- Generator receives near-zero gradients from flat non-saturating (or saturating) loss, thus appearing to "collapse" on the real mode



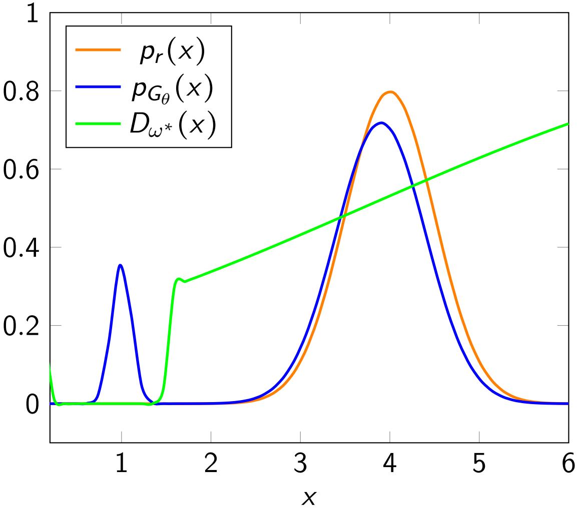
Model Oscillation

- Most generated data approach real mode, while some remain far away



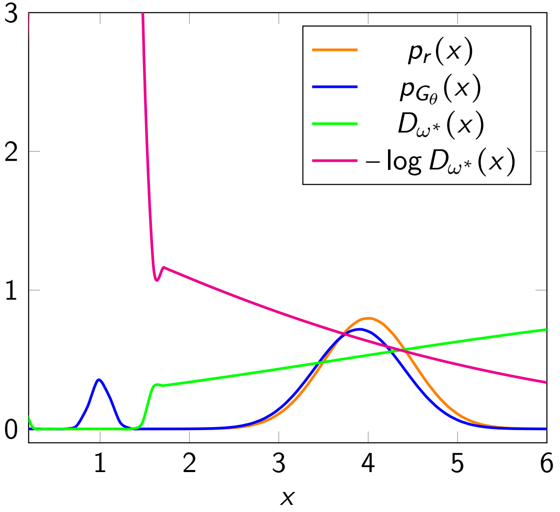
Model Oscillation

- Discriminator confidently classifies "outlier" generated mode, gives cautious predictions for remaining data



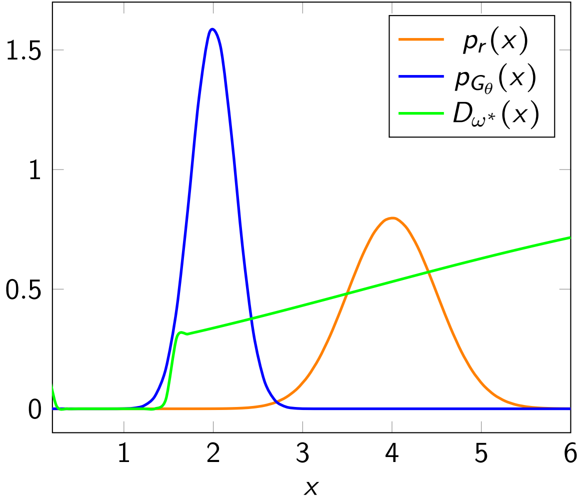
Model Oscillation

- Outlier data receive very steep gradients while local data receive relatively flat gradients



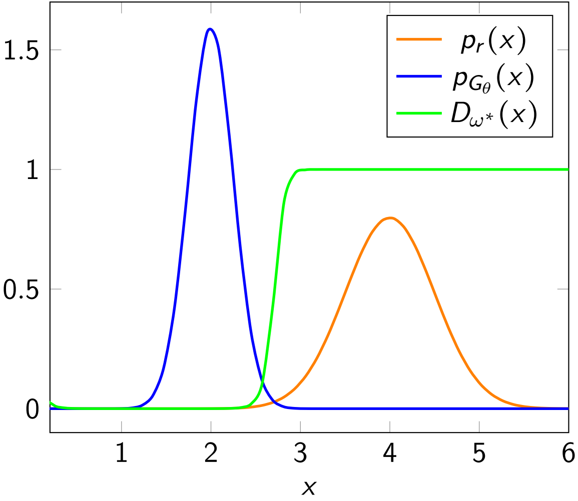
Model Oscillation

- Generator prioritizes correcting the outlier data at the expense of preserving the proximity of the local data



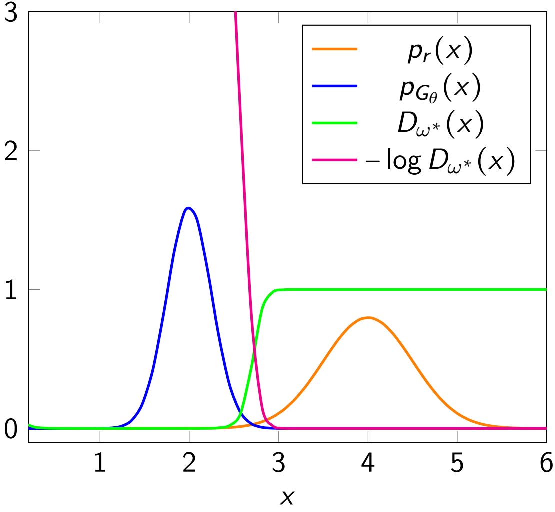
Model Oscillation

- Discriminator updates with confident predictions



Model Oscillation

- Generated samples receive steep gradients, which may lead to oscillations around the real mode



2. Formulating the (α_D, α_G)-GAN

CPE Loss Function Perspective of GANs

- Kurri *et al.* (2021) shows that $V(\theta,\omega)$ can be expressed with a *class probability estimation* (CPE) loss ℓ

$$V(\theta,\omega) = \mathbb{E}_{X\sim P_r}[-\ell(1, D_\omega(X))] + \mathbb{E}_{X\sim P_{G_\theta}}[-\ell(0, D_\omega(X))]$$

- $\ell(y,\hat{y})$ - any CPE loss
  - $\hat{y} \in [0,1]$ is a soft prediction of $y \in \{0,1\}$
- **Example:** α-GAN [Kurri *et al.* (2021)] uses the CPE loss function α-loss, $\alpha \in (0,1) \cup (1,\infty]$ [Sypherd *et al.* (2019)]:

$$\boxed{\ell_\alpha(y,\hat{y}) = \frac{\alpha}{\alpha-1}\left(1 - y\hat{y}^{\frac{\alpha-1}{\alpha}} - (1-y)(1-\hat{y})^{\frac{\alpha-1}{\alpha}}\right)}$$

(α_D, α_G)-GAN: A Generalization of α-GAN

- α-GAN uses value function V_α

$$V_\alpha(\theta,\omega) = \mathbb{E}_{X\sim P_r}[-\ell_\alpha(1, D_\omega(X))] + \mathbb{E}_{X\sim P_{G_\theta}}[-\ell_\alpha(0, D_\omega(X))]$$

  in the min-max game

$$\inf_{\theta\in\Theta} \sup_{\omega\in\Omega} V_\alpha(\theta,\omega)$$

- This formulation recovers a class of f-GANs that minimize the Arimoto f-divergence [2]
- Fails to address GAN challenges due to overly-convex generator loss with $\alpha < 1$, or overconfident discriminator with $\alpha > 1$

[2] Hellinger GAN ($\alpha = 1/2$), Vanilla GAN ($\alpha = 1$), Total Variation GAN ($\alpha = \infty$)

(α_D, α_G)-GAN: A Generalization of α-GAN

$$V_\alpha(\theta,\omega) = \mathbb{E}_{X\sim P_r}[-\ell_\alpha(1, D_\omega(X))] + \mathbb{E}_{X\sim P_{G_\theta}}[-\ell_\alpha(0, D_\omega(X))]$$

- To address the GAN challenges, we introduce (α_D, α_G)-GAN

$$\sup_{\omega\in\Omega} V_{\alpha_D}(\theta,\omega), \qquad \inf_{\theta\in\Theta} V_{\alpha_G}(\theta,\omega)$$

- Recovers α-GAN ($\alpha_D = \alpha_G$) and vanilla GAN ($\alpha_D, \alpha_G = 1$)
- Motivated by Goodfellow *et al.* (2014), we also introduce the *non-saturating* (α_D, α_G)-GAN

$$\sup_{\omega\in\Omega} V_{\alpha_D}(\theta,\omega), \qquad \inf_{\theta\in\Theta} V^{\mathrm{NS}}_{\alpha_G}(\theta,\omega) := \mathbb{E}_{X\sim P_{G_\theta}}[\ell_{\alpha_G}(1, D_\omega(X))]$$

Optimal Discriminator of (α_D, α_G)-GAN

- Assuming a sufficiently large Ω and fixed G_θ, the discriminator D_{ω^*} optimizing the sup of V_{α_D} is given by

$$D_{\omega^*}(x) = \frac{p_r(x)^{\alpha_D}}{p_r(x)^{\alpha_D} + p_{G_\theta}(x)^{\alpha_D}}$$

- Same optimal D_ω for both saturating and non-saturating cases

[Result 1] Discriminator Learns α_D-Tilted Posterior

Theorem 1

The optimal (α_D, α_G)-GAN discriminator D_{ω^*} is equivalent to the α_D-tilted version of the true posterior $P(Y = 1|X)$, namely $P_{\alpha_D}(Y = 1|X)$.

[Result 1] Discriminator Learns α_D-Tilted Posterior

Theorem 1

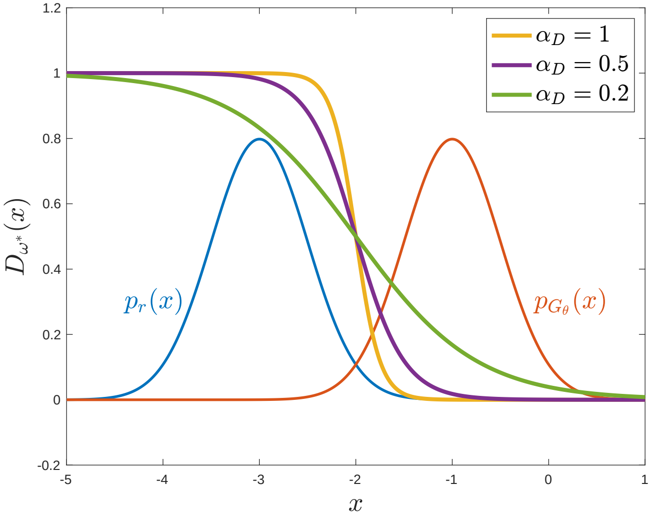
The optimal (α_D, α_G)-GAN discriminator D_{ω^*} is equivalent to the α_D-tilted version of the true posterior $P(Y=1|X)$, namely $P_{\alpha_D}(Y=1|X)$.

Proof sketch:

- The vanilla $(1,1)$-GAN discriminator learns $P(Y=1|X)$, the probability that sample $X \sim \frac{1}{2}P_r + \frac{1}{2}P_{G_\theta}$ is real $(Y=1)$ or generated $(Y=0)$, which is equivalent to $P_r(X)/(P_r(X)+P_{G_\theta}(X))$
- Using this equality, we can show that

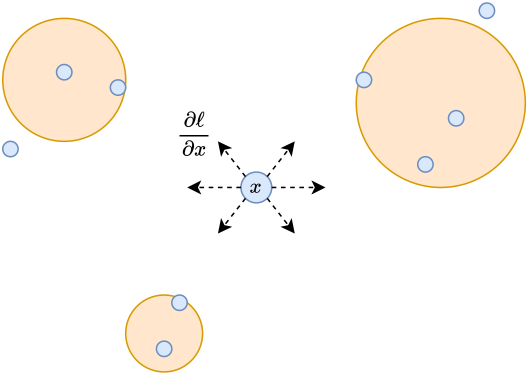
$$P_{\alpha_D}(Y=1|X) = \frac{P(Y=1|X)^{\alpha_D}}{P(Y=1|X)^{\alpha_D} + P(Y=0|X)^{\alpha_D}} = \frac{P_r(X)^{\alpha_D}}{P_r(X)^{\alpha_D} + P_{G_\theta}(X)^{\alpha_D}} = D_{\omega^*}(x)$$

[Result 1] Discriminator Learns α_D-Tilted Posterior



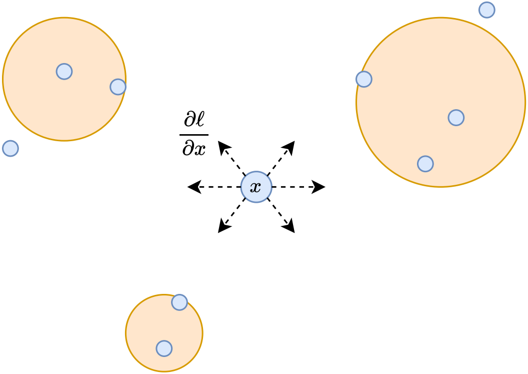
Generator Optimization of (α_D, α_G)-GAN

- During backpropagation, the gradient vector $\partial\ell/\partial x$ is computed for each generated sample x in the batch
  - **Interpretation:** which direction and magnitude should x move in order to reduce the generator loss?



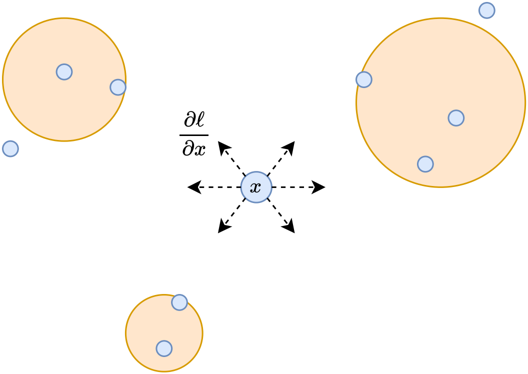
Generator Optimization of (α_D, α_G)-GAN

- **Our question:** how would tuning $(\alpha_D, \alpha_G) \in [0, \infty)^2$ influence this gradient vector?



Generator Optimization of (α_D, α_G)-GAN

- **Our question:** how would tuning $(\alpha_D, \alpha_G) \in [0, \infty)^2$ influence this gradient vector?



- **Our claim:** Tuning α_D and α_G only affects the magnitude, not direction, for *both* saturating/non-saturating (α_D, α_G)-GANs

[Result 2] Impact of (α_D, α_G) on Saturating Loss

Theorem 2

Let x be a sample generated by G_θ and D_{ω^*} be optimal with respect to V_{α_D}. Then the direction of the **saturating** gradient $-\partial \ell_{\alpha_G}\left(0, D_{\omega^*}(x)\right)/\partial x$ is independent of α_D and α_G.

[Result 2] Impact of (α_D, α_G) on Saturating Loss

Theorem 2

Let x be a sample generated by G_θ and D_{ω^*} be optimal with respect to V_{α_D}. Then the direction of the **saturating** gradient $-\partial \ell_{\alpha_G}\left(0, D_{\omega^*}(x)\right)/\partial x$ is independent of α_D and α_G.

Proof sketch:

- The saturating gradient can be simplified to

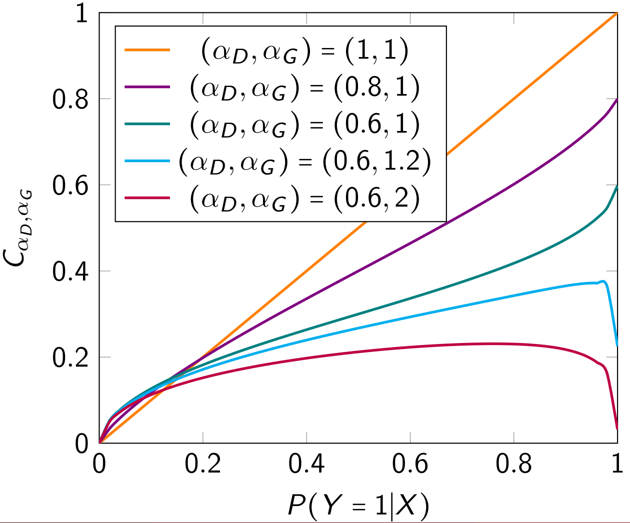
$$-\frac{\partial \ell_{\alpha_G}\left(0, D_{\omega^*}(x)\right)}{\partial x} = C_{\alpha_D,\alpha_G}\left(\frac{1}{p_{G_\theta}(x)}\frac{\partial p_{G_\theta}}{\partial x} - \frac{1}{p_r(x)}\frac{\partial p_r}{\partial x}\right)$$

where C_{α_D,α_G} is a scalar defined as

$$\boxed{C_{\alpha_D,\alpha_G} = \alpha_D P_{\alpha_D}\left(Y=1|X=x\right)\left(1 - P_{\alpha_D}\left(Y=1|X=x\right)\right)^{1-1/\alpha_G}}$$

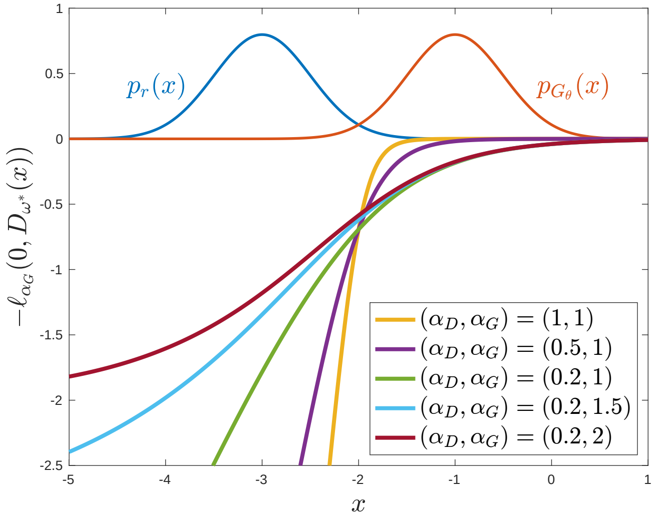
[Result 2] Impact of (α_D, α_G) on Saturating Loss

- Tuning $\alpha_D < 1$ increases gradient for samples far from real data
- Tuning $\alpha_G > 1$ decreases gradient for samples close to real data



[Result 2] Impact of (α_D, α_G) on Saturating Loss

- Tuning $\alpha_D < 1$ helps combat vanishing gradients
- Tuning $\alpha_G > 1$ helps combat exploding gradients



[Result 3] Impact of (α_D, α_G) on Non-Saturating Loss

Theorem 3

Let x be a sample generated by G_θ and D_{ω^*} be optimal with respect to V_{α_D}. Then the direction of the **non-saturating** gradient $\partial \ell_{\alpha_G}\left(1, D_{\omega^*}(x)\right) / \partial x$ is independent of α_D and α_G.

[Result 3] Impact of (α_D, α_G) on Non-Saturating Loss

Theorem 3

Let x be a sample generated by G_θ and D_{ω^*} be fixed and optimal with respect to V_{α_D}. Then the direction of the **non-saturating** gradient $\partial \ell_{\alpha_G}\left(1, D_{\omega^*}(x)\right)/\partial x$ is independent of α_D and α_G.

Proof sketch:

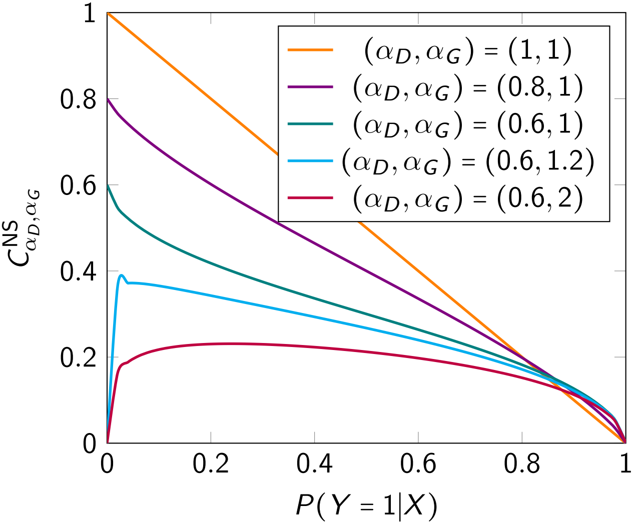
- The non-saturating gradient can be simplified to

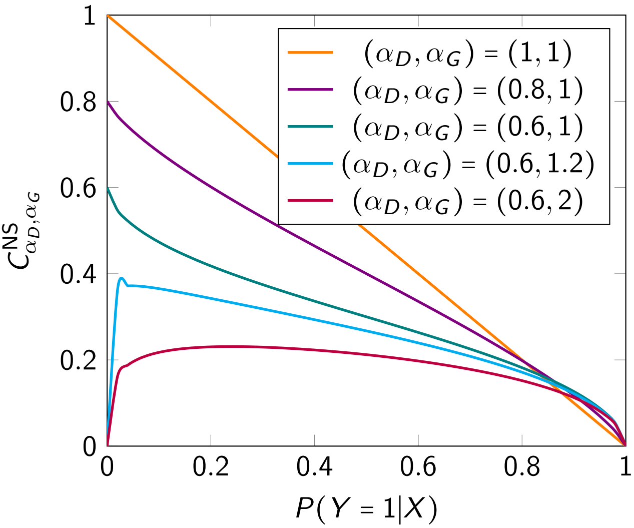
$$\frac{\partial \ell_{\alpha_G}\left(1, D_{\omega^*}(x)\right)}{\partial x} = C^{\mathrm{NS}}_{\alpha_D,\alpha_G}\left(\frac{1}{p_{G_\theta}(x)}\frac{\partial p_{G_\theta}}{\partial x} - \frac{1}{p_r(x)}\frac{\partial p_r}{\partial x}\right)$$

  where $C^{\mathrm{NS}}_{\alpha_D,\alpha_G}$ is a scalar defined as

$$\boxed{C^{\mathrm{NS}}_{\alpha_D,\alpha_G} = \alpha_D\left(1 - P_{\alpha_D}\left(Y = 1 | X = x\right)\right) P_{\alpha_D}\left(Y = 1 | X = x\right)^{1-1/\alpha_G}}$$

[Result 3] Impact of (α_D, α_G) on Non-Saturating Loss





- Can't we just decrease the learning rate for smaller gradients?

[Result 3] Impact of (α_D, α_G) on Non-Saturating Loss

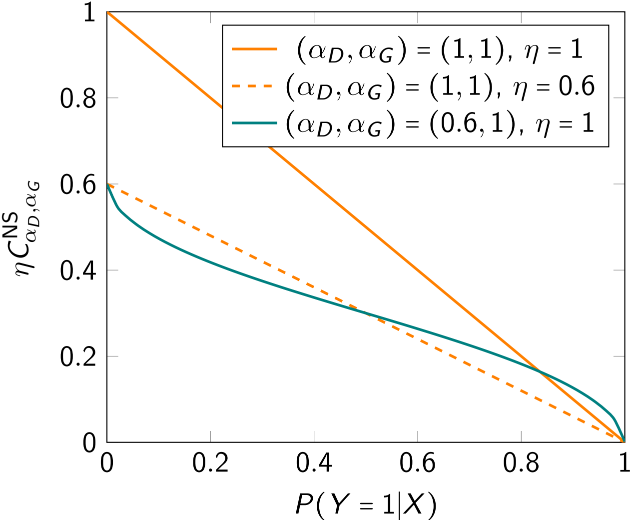
- Generator weight update with learning rate η

$$\begin{aligned}
\theta^{(i+1)} &:= \theta^{(i)} - \eta \frac{\partial \ell}{\partial \theta^{(i)}} \\
&:= \theta^{(i)} - \eta \frac{1}{|\mathcal{X}|} \sum_{x \in \mathcal{X}} \frac{\partial \ell}{\partial x} \frac{\partial x}{\partial \theta^{(i)}} \\
&:= \theta^{(i)} - \eta \frac{1}{|\mathcal{X}|} \sum_{x \in \mathcal{X}} \left[C^{\mathrm{NS}}_{\alpha_D, \alpha_G} (\cdots) \right] \frac{\partial x}{\partial \theta^{(i)}} \\
&:= \theta^{(i)} - \left(\eta C^{\mathrm{NS}}_{\alpha_D, \alpha_G} \right) \frac{1}{|\mathcal{X}|} \sum_{x \in \mathcal{X}} (\cdots) \frac{\partial x}{\partial \theta^{(i)}}
\end{aligned}$$

- More accurately, $\eta C^{\mathrm{NS}}_{\alpha_D, \alpha_G}$ can be considered the gradient scalar

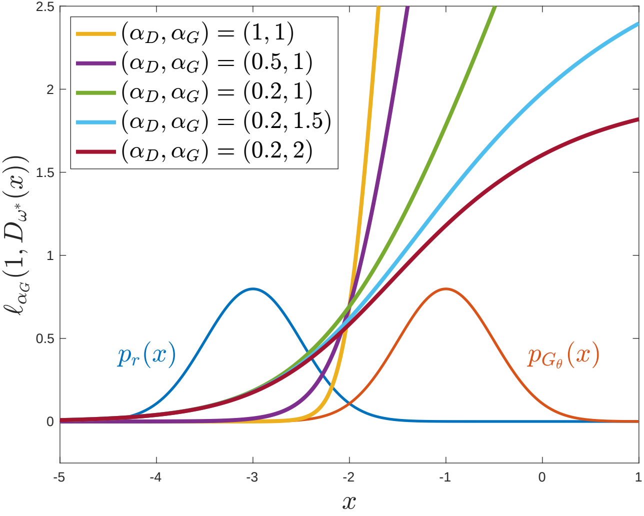
[Result 3] Impact of (α_D, α_G) on Non-Saturating Loss

- Tuning $\alpha_D < 1$ decreases (increases) gradients received by samples far from (close to) real data: helps combat **model oscillation**



[Result 3] Impact of (α_D, α_G) on Non-Saturating Loss

- Tuning $\alpha_G > 1$ may immobilize samples very far from real data



Advantages of Tuning (α_D, α_G)

- Saturating (α_D, α_G)-GAN
  - Tuning $\alpha_D < 1$ helps combat vanishing gradients
  - Tuning $\alpha_G > 1$ helps combat exploding gradients
- Non-saturating (α_D, α_G)-GAN
  - Tuning $\alpha_D < 1$ helps combat model oscillation
  - Tuning $\alpha_G > 1$ reduces the gradients received by outlier samples even more, but may cause generator to ignore outliers

3. Experiments & Summary of Results

Overview of Experiments

- GANs
  - Vanilla GAN (+ non-saturating)
  - (α_D, α_G)-GAN (+ non-saturating)
  - Least Squares GAN (LSGAN) [Mao *et al.* (2017)]

Overview of Experiments

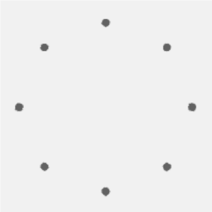
- GANs
  - Vanilla GAN (+ non-saturating)
  - (α_D, α_G)-GAN (+ non-saturating)
  - Least Squares GAN (LSGAN) [Mao *et al.* (2017)]
- Datasets
  - 2D Gaussian Mixture Ring [Srivastava *et al.* (2017)]
  - Celeb-A Image Dataset [Liu *et al.* (2015)]
  - LSUN Classroom Image Dataset [Yu *et al.* (2015)]

Overview of Experiments

- GANs
  - Vanilla GAN (+ non-saturating)
  - (α_D, α_G)-GAN (+ non-saturating)
  - Least Squares GAN (LSGAN) [Mao *et al.* (2017)]
- Datasets
  - 2D Gaussian Mixture Ring [Srivastava *et al.* (2017)]
  - Celeb-A Image Dataset [Liu *et al.* (2015)]
  - LSUN Classroom Image Dataset [Yu *et al.* (2015)]
- Hypothesis
  - Tuning $\alpha_D < 1$ and $\alpha_G > 1$ improves the training stability of (α_D, α_G)-GAN
  - In particular, it robustifies the GAN training to random model weight initializations

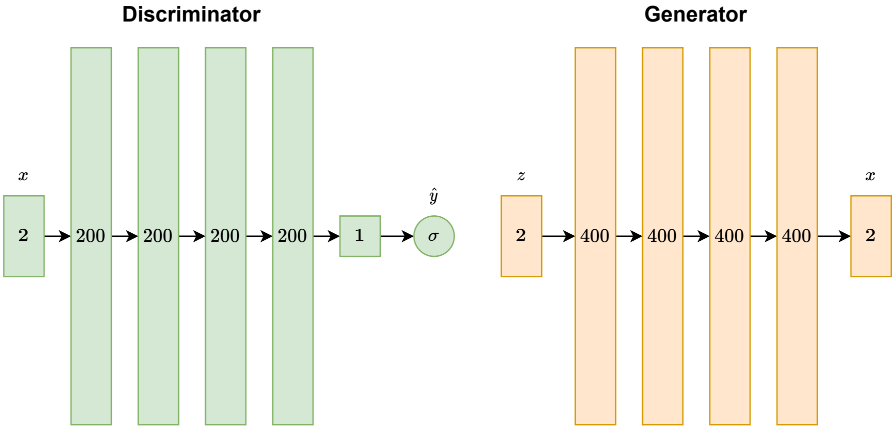
[2D-Ring] Data Preparation

- We draw samples from 8 equal-prior Gaussian distributions
- Each mode $i \in \{1, 2, \cdots, 8\}$ has mean $(\cos(2\pi i/8), \sin(2\pi/8))$ and variance 10^{-4}
- We generate 50k training samples and 25k testing samples
- We also generate the same amount of 2D Gaussian noise vectors for training/testing



[2D-Ring] Model Architecture

- Both D_ω and G_θ networks have 4 fully-connected layers with 200 and 400 units, respectively



[2D-Ring] GANs & Hyperparameters

- GANs
  - Vanilla GAN (+ non-saturating)
  - (α_D, α_G)-GAN (+ non-saturating)
    - $(\alpha_D, \alpha_G) \in [0.5, 1] \times [0.9, 1.2]$
  - LSGAN with 0-1 binary coding scheme

$$\inf_{\omega \in \Omega} \mathbb{E}_{X \sim P_r} \left[\frac{1}{2} (D_\omega(x) - 1)^2 \right] + \mathbb{E}_{X \sim P_{G_\theta}} \left[\frac{1}{2} (D_\omega(x))^2 \right]$$

$$\inf_{\theta \in \Theta} \mathbb{E}_{X \sim P_{G_\theta}} \left[\frac{1}{2} (D_\omega(x) - 1)^2 \right]$$

- Hyperparameters
  - Adam optimization with learning rate 10^{-4}
  - 400 training epochs

[2D-Ring] Evaluation Metrics

1. Mode coverage
   - Number of modes that contain a sample within three standard deviations of its mean
2. High-quality samples
   - Percentage of samples that are within three standard deviations of any modes' mean
3. KL Divergence
   - Assign each real and generated sample to its closest mode
   - Creates two distributions (real/generated) across the 8 modes

- We find that **mode coverage reported over 200 seeds** is the best indicator of GAN training stability

[2D-Ring] Quantitative Results

- Table of **saturating** (α_D, α_G)-GAN success rates

% of success (8/8 modes)		α_D					
		0.5	0.6	0.7	0.8	0.9	1.0
α_G	0.9	73	**79**	69	60	46	34
	1.0	**80**	**79**	74	68	54	47
	1.1	**79**	77	68	70	59	47
	1.2	75	74	71	65	57	46

- Top 4 results emboldened, vanilla GAN
- $\alpha_D < 1$ has more impact than $\alpha_G > 1$

[2D-Ring] Quantitative Results

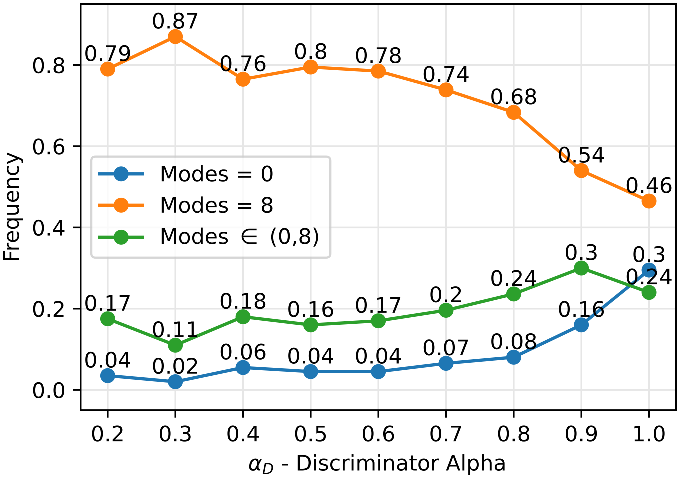
- Table of **saturating** ($\alpha_D, \alpha_G = 1$)-GAN failure rates

% of failure (0/8 modes)		α_D 0.5	0.6	0.7	0.8	0.9	1.0
α_G	0.9	11	10	12	13	29	49
	1.0	**5**	**5**	7	8	16	30
	1.1	7	9	13	12	13	26
	1.2	9	**5**	9	12	17	31

- Top 3 results emboldened, vanilla GAN
- $\alpha_D < 1$ has more impact than $\alpha_G > 1$

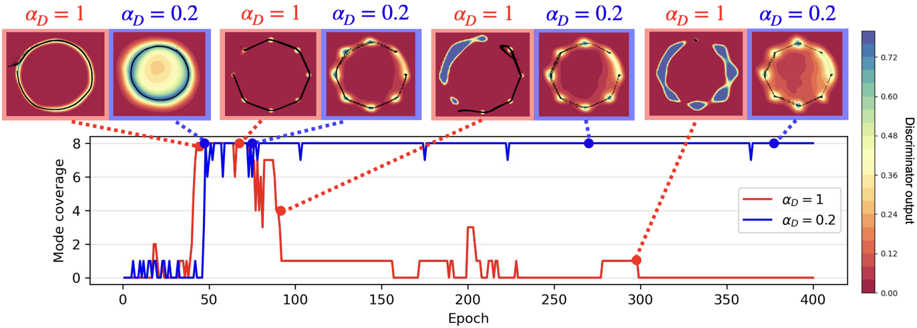
[2D-Ring] Quantitative Results

- Plot of **saturating** $(\alpha_D, 1)$-GAN results



[2D-Ring] Qualitative Results

- **Saturating:** Vanilla $(1, 1)$-GAN vs. $(0.2, 1)$-GAN



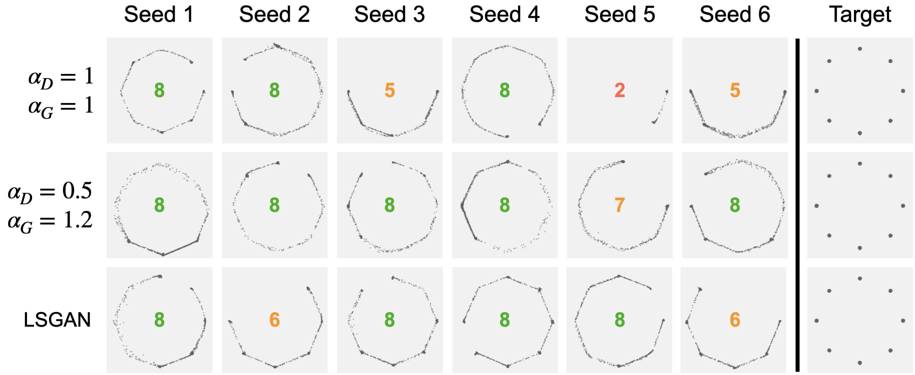
[2D-Ring] Quantitative Results

- Table of **non-saturating** (α_D, α_G)-GAN success rates

% of success (8/8 modes)		α_D 0.5	0.6	0.7	0.8	0.9	1.0	1.1	1.2
α_G	0.8	35	24	19	19	14	16	18	10
	0.9	**39**	37	19	22	16	20	19	21
	1.0	34	35	29	28	26	22	20	32
	1.1	**40**	36	31	22	24	15	23	25
	1.2	**45**	38	34	25	26	28	20	22
	1.3	**44**	**39**	26	28	28	25	31	29

- Top 5 results emboldened, vanilla GAN
- LSGAN success rate: 33%
- $\alpha_D < 1$ and $\alpha_G > 1$ both improve performance

- **Non-Saturating:** Vanilla $(1,1)$-GAN vs. $(0.5,1.2)$-GAN vs. LSGAN

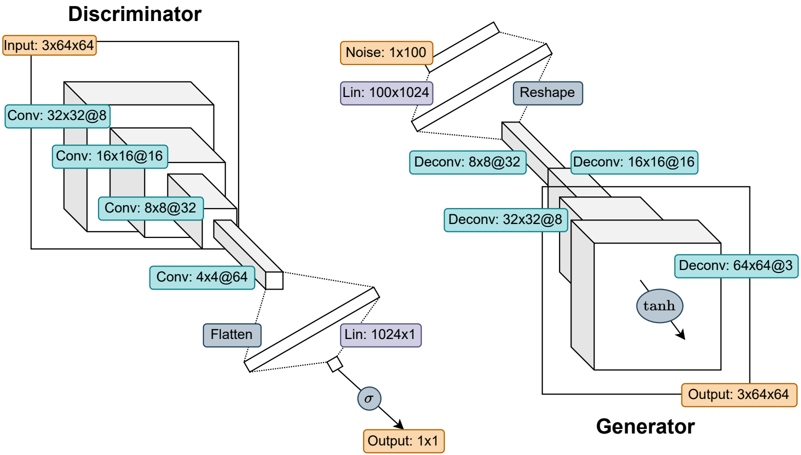


[Celeb-A] Data Preparation

- **Celeb-A Dataset:** collection of $\approx$ 200k celebrity headshots

- Resize & center crop all images to size 64×64
- Generate $\approx$ 200k Gaussian noise vectors of size 100
- 80%-20% train-validation split for both images & noise vectors

- Deep Convolutional GAN (DCGAN) [Radford *et al.* (2015)]

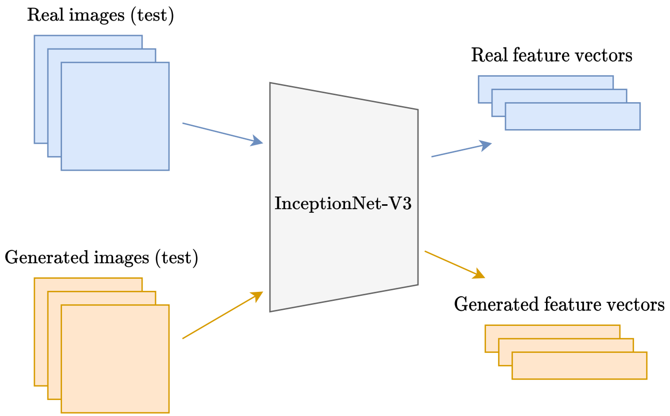


[Celeb-A] GANs & Hyperparameters

- GANs
  - Non-saturating vanilla GAN
  - Non-Saturating (α_D, α_G)-GAN
    - $(\alpha_D, \alpha_G) \in [0.5, 1] \times \{1\}$
  - LSGAN with 0-1 binary coding scheme
- Hyperparameters
  - Adam optimization with learning rates $\in [10^{-4}, 10^{-3}]$
  - Number of train epochs $\in \{10, 20, \cdots, 100\}$

[Celeb-A] Evaluation Metrics

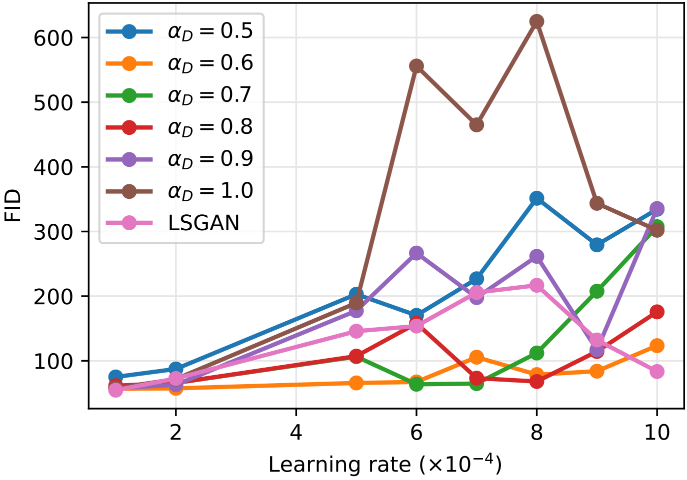
- **Fréchet Inception Distance (FID)** [Heusel *et al.* (2017)] averaged over 50 seeds



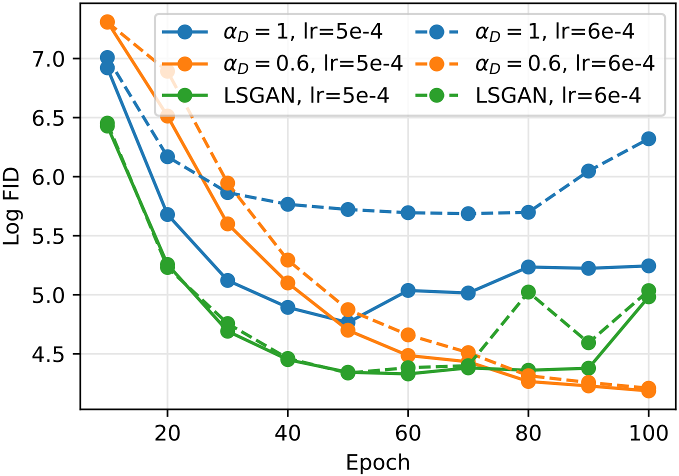
- FID = $\|\mu_r - \mu_g\|^2 + \mathrm{Tr}\left(\Sigma_r + \Sigma_g - 2\left(\Sigma_r\Sigma_g\right)^{1/2}\right)$

[Celeb-A] Quantitative Results

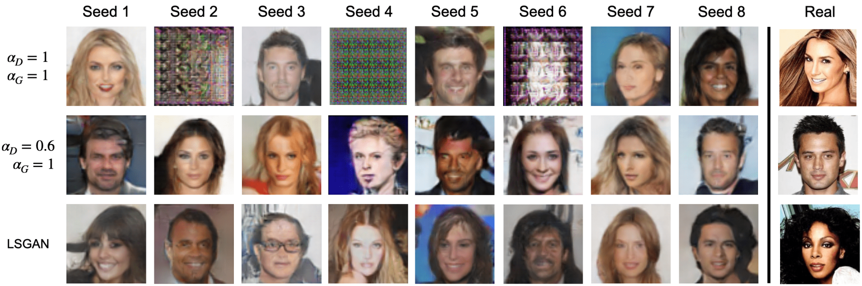
- Plot of mean FID over learning rate for 6 $(\alpha_D, 1)$-GANs and LSGAN
- $\alpha_D = 0.6$ appears to be most robust to learning rate



- Log-scale plot of mean FID over epoch for three GANs and two learning rates: $\alpha_D = 0.6$ appears to converge over time



- Generated samples across 8 seeds for three GANs trained with 5×10^{-4} learning rate for 100 epochs
- $(0.6, 1)$-GAN and LSGAN appear to be most stable & highest quality



[LSUN Classroom] Data Preparation

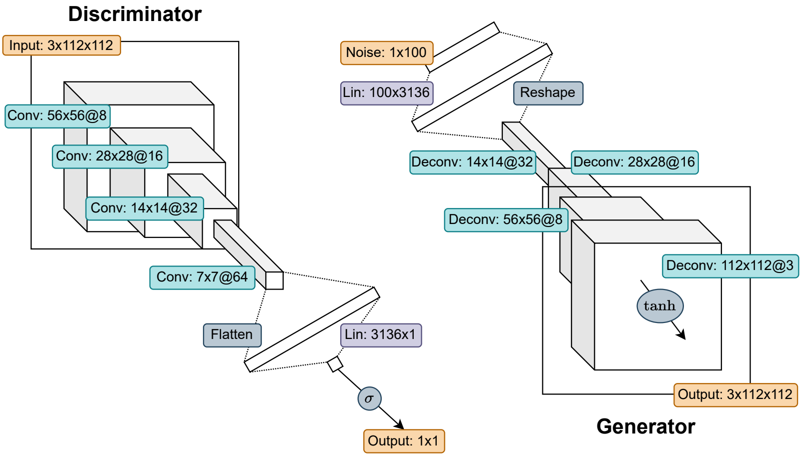
- **LSUN Classroom Dataset:** collection of $\approx$ 170k classroom images






- Resize & center crop all images to size 112×112
- Generate $\approx$ 170k Gaussian noise vectors of size 100
- 80%-20% train-validation split for both images & noise vectors

- Deep Convolutional GAN (DCGAN)

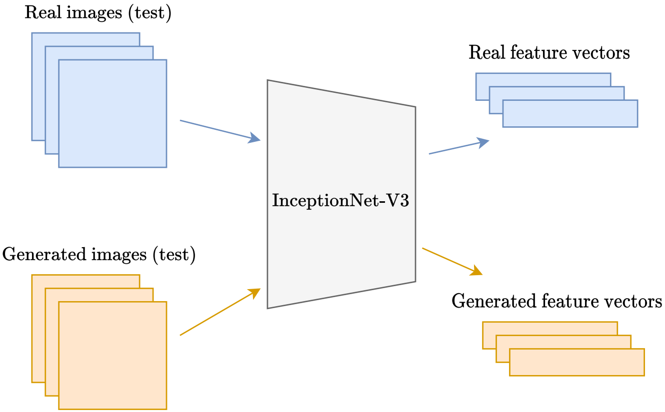


[LSUN Classroom] GANs & Hyperparameters

- GANs
  - Non-saturating vanilla GAN
  - Non-Saturating (α_D, α_G)-GAN
    - $(\alpha_D, \alpha_G) \in [0.5, 1] \times \{1\}$
  - LSGAN with 0-1 binary coding scheme
- Hyperparameters
  - Adam optimization with learning rates $\in [10^{-4}, 10^{-3}]$
  - Number of train epochs $\in \{10, 20, \cdots, 100\}$

[LSUN Classroom] Evaluation Metrics

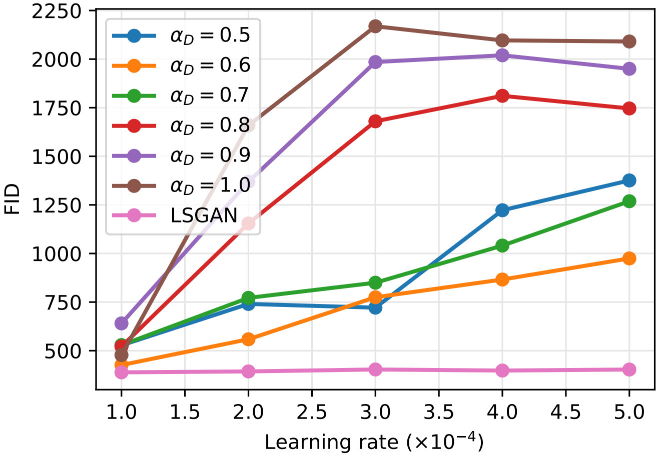
- **Fréchet Inception Distance (FID)** averaged over 50 seeds



- $\text{FID} = \left\|\mu_r - \mu_g\right\|^2 + \text{Tr}\left(\Sigma_r + \Sigma_g - 2\left(\Sigma_r \Sigma_g\right)^{1/2}\right)$

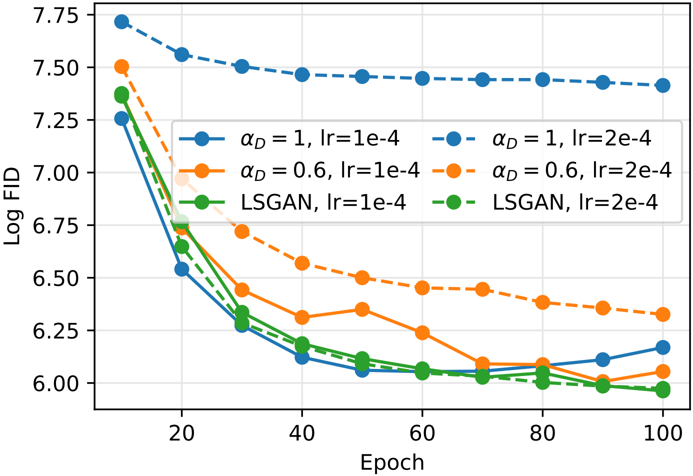
[LSUN Classroom] Quantitative Results

- Plot of mean FID over learning rate for 6 $(\alpha_D, 1)$-GANs and LSGAN
- Tuning $\alpha_D < 1$ is more robust to learning rate, but LSGAN greatly outperforms all tested (α_D, α_G)-GANs

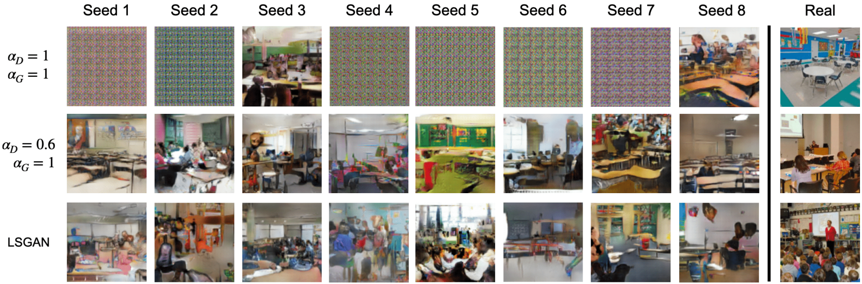


[LSUN Classroom] Quantitative Results

- Log-scale plot of mean FID over epoch for three GANs and two learning rates: vanilla GAN extremely sensitive to learning rate



- Generated samples across 8 seeds for three GANs trained with 2×10^{-4} learning rate for 100 epochs
- $(0.6, 1)$-GAN and LSGAN are much more stable than vanilla GAN



Summary of Results

- 2D-ring (saturating)
  - Vanilla GAN showed instability due to exploding & vanishing gradients
  - Tuning α_D down to 0.3 decreased failure rate 30% → 2%
  - Tuning α_G had no significant impact on stability

Summary of Results

- 2D-ring (saturating)
  - Vanilla GAN showed instability due to exploding & vanishing gradients
  - Tuning α_D down to 0.3 decreased failure rate 30% $\rightarrow$ 2%
  - Tuning α_G had no significant impact on stability
- 2D-ring (non-saturating)
  - Tuning α_D down to 0.5 and α_G up to 1.2 *doubled* success rate compared to vanilla GAN (22% $\rightarrow$ 45%)
  - $(0.5, 1.2)$-GAN performed more stable than LSGAN (45% vs. 33%)

Summary of Results

- 2D-ring (saturating)
  - Vanilla GAN showed instability due to exploding & vanishing gradients
  - Tuning α_D down to 0.3 decreased failure rate 30% $\to$ 2%
  - Tuning α_G had no significant impact on stability
- 2D-ring (non-saturating)
  - Tuning α_D down to 0.5 and α_G up to 1.2 *doubled* success rate compared to vanilla GAN (22% $\to$ 45%)
  - $(0.5, 1.2)$-GAN performed more stable than LSGAN (45% vs. 33%)
- Celeb-A
  - Fixing $\alpha_G = 1$ gave the best performance
  - Tuning α_D down to 0.6 gave the most robust GAN to learning rate

Summary of Results

- 2D-ring (saturating)
  - Vanilla GAN showed instability due to exploding & vanishing gradients
  - Tuning α_D down to 0.3 decreased failure rate 30% $\rightarrow$ 2%
  - Tuning α_G had no significant impact on stability
- 2D-ring (non-saturating)
  - Tuning α_D down to 0.5 and α_G up to 1.2 *doubled* success rate compared to vanilla GAN (22% $\rightarrow$ 45%)
  - $(0.5, 1.2)$-GAN performed more stable than LSGAN (45% vs. 33%)
- Celeb-A
  - Fixing $\alpha_G = 1$ gave the best performance
  - Tuning α_D down to 0.6 gave the most robust GAN to learning rate
- LSUN Classroom
  - Fixing $\alpha_G = 1$ gave the best performance
  - Tuning α_D down to 0.6 gave the most robust (α_D, α_G)-GAN
  - However, LSGAN significantly outperformed the (α_D, α_G)-GANs