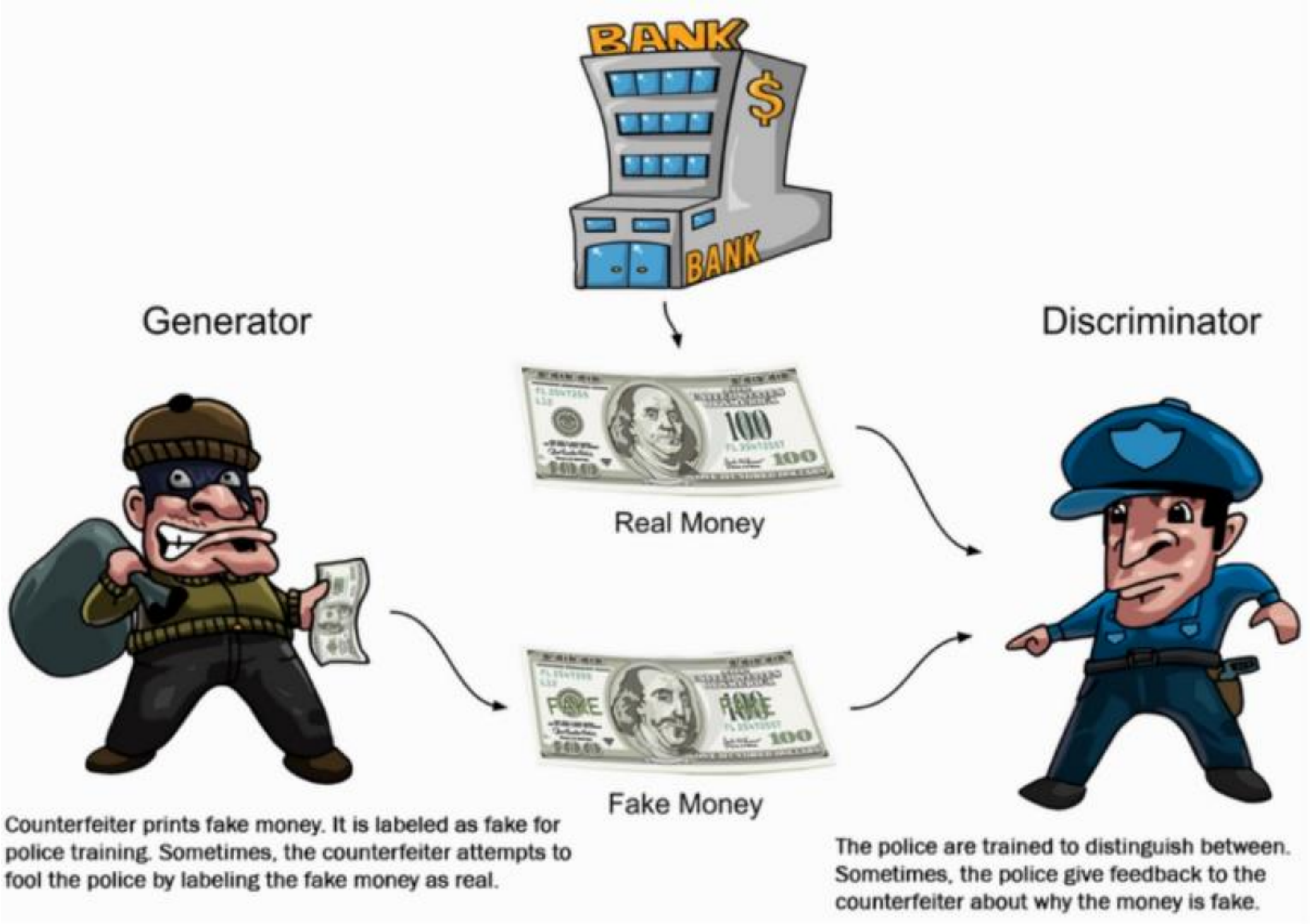
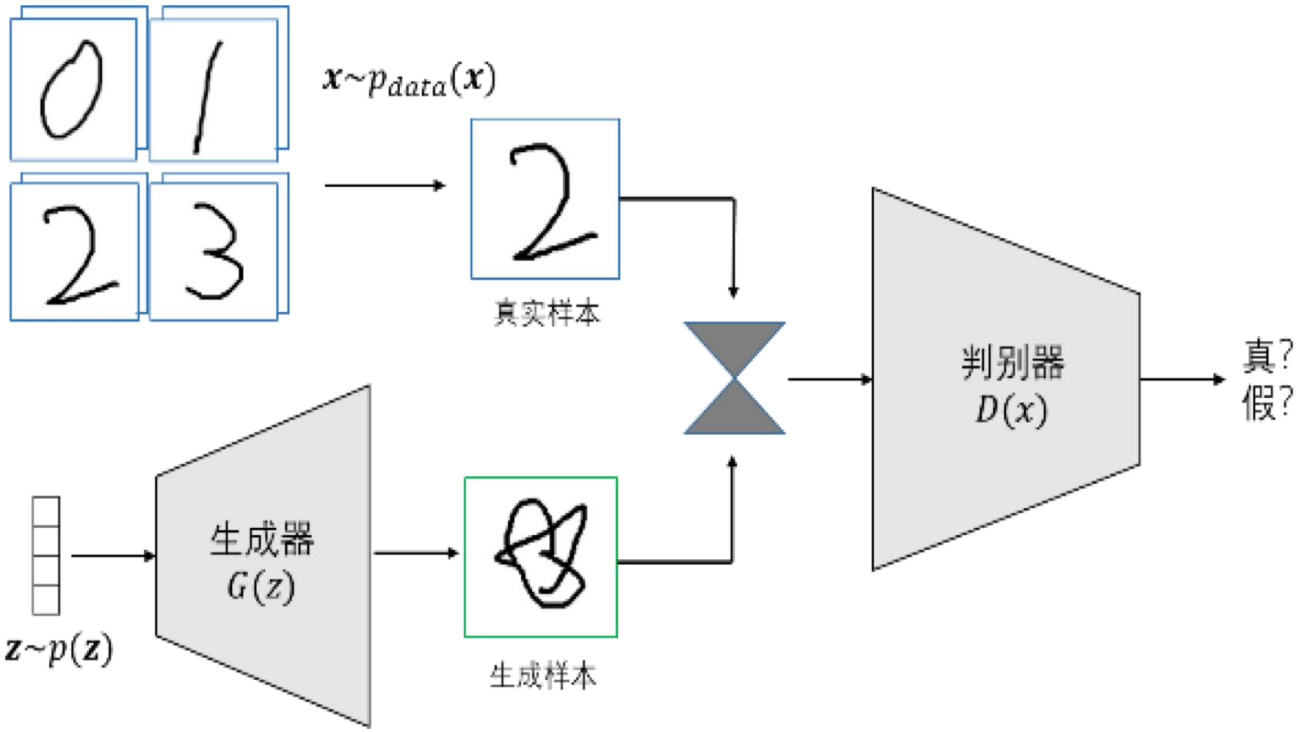
Generative Adversarial Networks

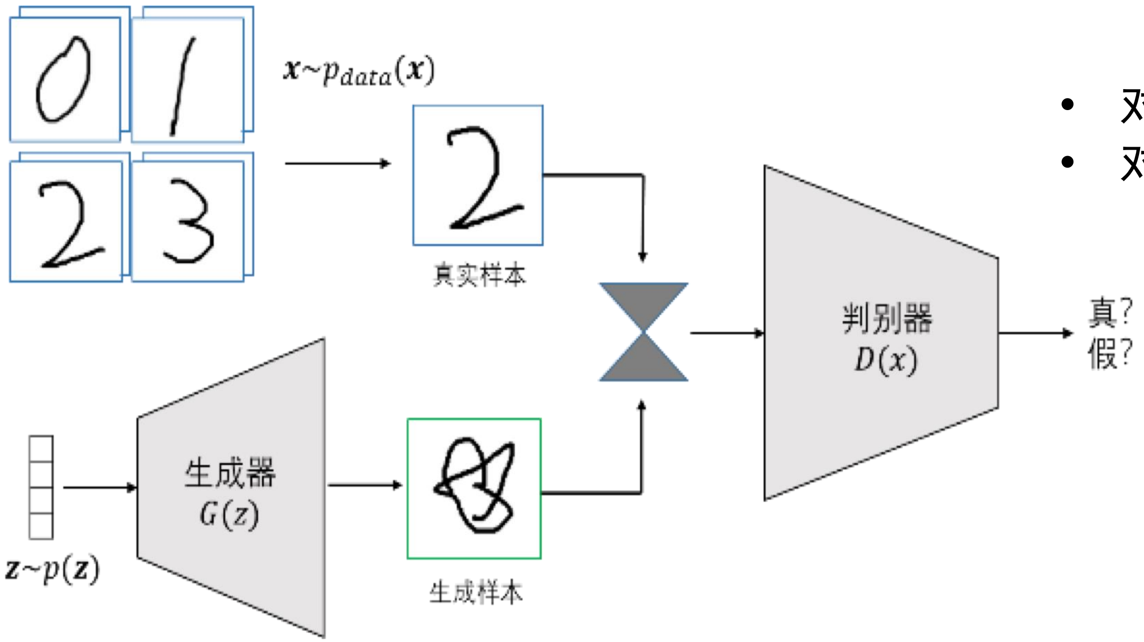


GAN



- 生成对抗网络包含两个神经网络，一个是生成器（generator），另一个是判别器（discriminator）
- 生成器的任务是在一定的隐变量控制下生成新样本，判别器的任务是对真实训练样本和生成器生成的“假样本”进行判别。
- “对抗”，就是指生成对抗网络在训练过程中，一方面训练判别器，使之尽可能准确地区分真样本和假样本；另一方面训练生成器，使之产生的假样本尽量不会被判别器识别出来。

GAN



- 对真样本x，希望D(x)判别为1
- 对假样本G(z)，希望D(G(z))判别为0

$$\min_G \max_D V(D,G) = E_{\boldsymbol{x} \sim p_{data}(\boldsymbol{x})}[log D(\boldsymbol{x})] + E_{\boldsymbol{z} \sim p_{\boldsymbol{z}}(\boldsymbol{z})}[log\left(1 - D(G(\boldsymbol{z}))\right)]$$

- maxD 使得分类器尽量准确的区分真实样本和生成样本， 即最大化log(D(x))和log(1-D(G(z))
- minG 使得生成器尽量骗过分类器，即log(1-D(G(z))最小

GAN

对固定的生成器：z->x的映射关系固定

$$V(D,G) = \int_x p_{data}(x)log\big(D(x)\big)dx + \int_z p_z(z)log\Big(1 - D\big(G(z)\big)\Big)dz$$

$$= \int_x p_{data}(x)log\big(D(x)\big)\,dx + \int_x p_g(x)log\big(1 - D(x)\big)\,dx$$

$$= \int_x \Big(p_{data}(x)log\big(D(x)\big) + p_g(x)log\big(1 - D(x)\big)\Big)\,dx$$

对判别器求V(D,G)最大，最优解需满足

$$\frac{\partial}{\partial D(x)}\Big(p_{data}(x)log\big(D(x)\big) + p_g(x)log\big(1 - D(x)\big)\Big) = 0$$

可得最优判别器D*(x) 为

$$D^*(x) = \frac{p_{data}(x)}{p_{data}(x) + p_g(x)}$$

GAN

对于固定的判别器，需要对生成器求V(D,G)最小

将 $D^*(x) = \frac{p_{data}(x)}{p_{data}(x) + p_g(x)}$ 代入

$$V(D^*, G) = E_{x\sim p_{data}}\left[log\frac{p_{data}(x)}{p_{data}(x) + p_g(x)}\right] + E_{x\sim p_g}\left[log\frac{p_g(x)}{p_{data}(x) + p_g(x)}\right]$$

$$V(D^*, G) = -log(4) + KL\left(p_{data}||\frac{p_{data} + p_g}{2}\right) + KL\left(p_g||\frac{p_{data} + p_g}{2}\right)$$

$$V(D^*, G) = -log(4) + 2JSD(p_{data}||p_g)$$

$JSD(p_{data}||p_g)$是$p_{data}(x)$和$p_g(x)$的J-S散度（Jensen-Shannon divergence）

对生成器最小化V(D*,G)就是最小化$p_{data}(x)$和$p_g(x)$的差异，最优解是$p_{data}(x) = p_g(x)$，
即生成样本与真实样本相同

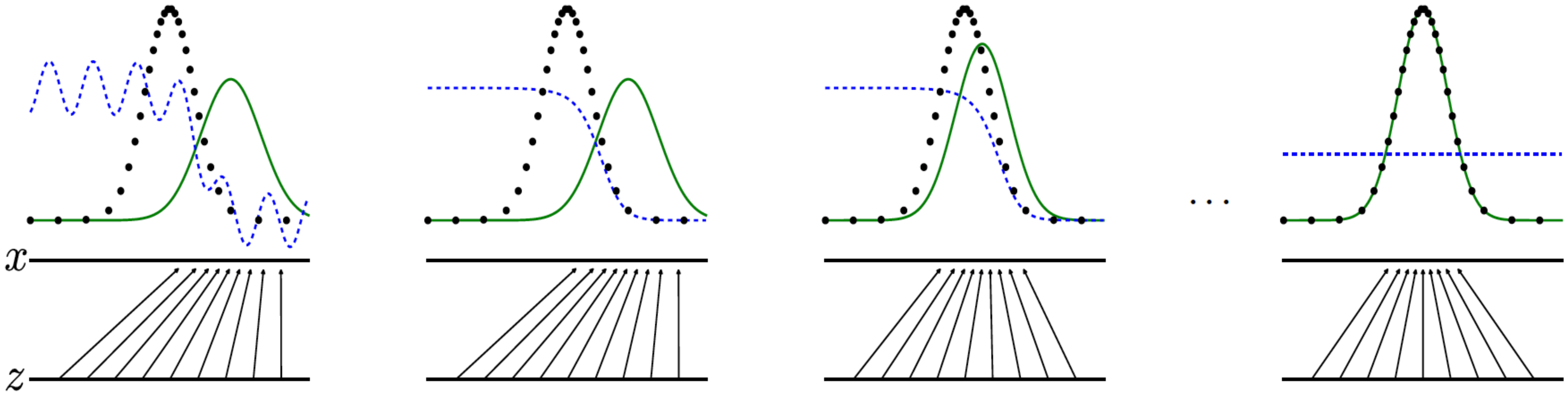
GAN

初始化的D和G

$$D^*(\boldsymbol{x}) = \frac{p_{\text{data}}(\boldsymbol{x})}{p_{\text{data}}(\boldsymbol{x}) + p_g(\boldsymbol{x})}$$

$$p_g = p_{\text{data}}$$

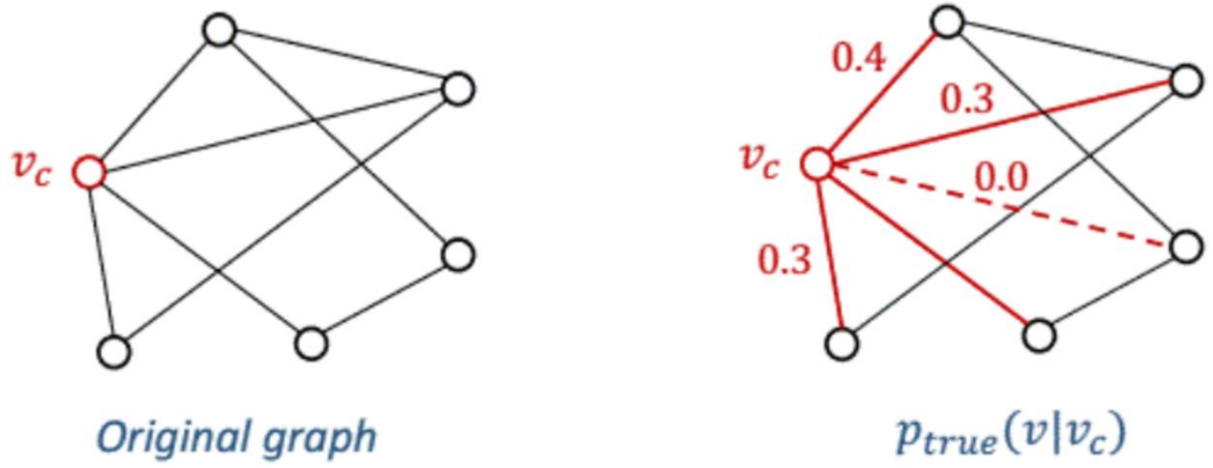
$$D(\boldsymbol{x}) = \frac{1}{2}$$



(a) (b) (c) (d)

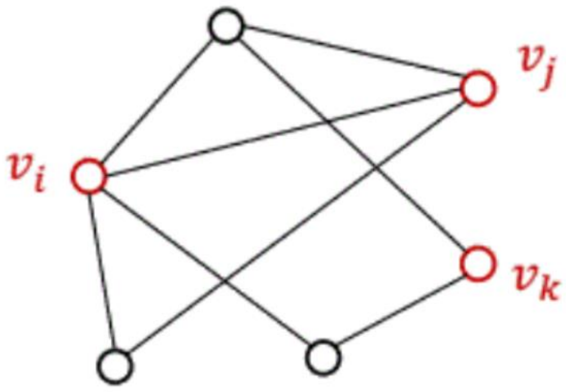
graphGAN

- Generative graph representation learning model assumes an **underlying true connectivity distribution** $p_{true}(v|v_c)$ for each vertex v_c
  - The edges can be viewed as observed samples generated by $p_{true}(v|v_c)$
  - Vertex embeddings are learned by maximizing the likelihood of edges
  - E.g., DeepWalk (KDD 2014) and node2vec (KDD 2016)



graphGAN

- Discriminative graph representation learning model aim to learn a **classifier** for predicting the existence of edges directly
  - Consider two vertices v_i and v_j jointly as features
  - Predict the probability of an edge existing between them, i.e., $p(edge|v_i, v_j)$
  - E.g., SDNE (KDD 2016) and PPNE (DASFAA, 2017)



$p(edge|v_i, v_j) = 0.8$

$p(edge|v_i, v_k) = 0.3$

……

Key point: positive and negative sample imbalance

graphGAN

- $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, $\mathcal{V} = \{v_1, \ldots, v_V\}$, $\mathcal{E} = \{e_{ij}\}_{i,j=1}^{V}$
- $\mathcal{N}(v_c)$: set of neighbors of v_c
- $p_{true}(v_c)$: underlying true connectivity distribution for v_c

- The objective of GraphGAN is to learn the following two models:
  - $G(v|v_c; \theta_G)$ which tries to approximate $p_{true}(v_c)$
  - $D(v, v_c; \theta_D)$ which aims to discriminate the connectivity for the vertex pair (v, v_c)
- The two-player minimax game:

$$\min_{\theta_G} \max_{\theta_D} V(G, D) = \sum_{c=1}^{V} \Big(\mathbb{E}_{v \sim p_{\mathrm{true}}(\cdot|v_c)} \big[\log D(v, v_c; \theta_D) \big] + \mathbb{E}_{v \sim G(\cdot|v_c; \theta_G)} \big[\log \big(1 - D(v, v_c; \theta_D)\big) \big] \Big)$$

Optimization of D

$$\min_{\theta_G} \max_{\theta_D} V(G, D) = \sum_{c=1}^{V} \Big(\mathbb{E}_{v \sim p_{\text{true}}(\cdot|v_c)} \big[\log D(v, v_c; \theta_D) \big] + \mathbb{E}_{v \sim G(\cdot|v_c;\theta_G)} \big[\log \big(1 - D(v, v_c; \theta_D)\big) \big] \Big) \tag{1}$$

- Implementation of D:

$$D(v, v_c; \theta_D) = \sigma(\mathbf{d}_v^\top \mathbf{d}_{v_c}) = \frac{1}{1 + \exp(-\mathbf{d}_v^\top \mathbf{d}_{v_c})}, \tag{2}$$

where $\mathbf{d}_v, \mathbf{d}_{v_c} \in \mathbb{R}^k$ are the k-dimensional vectors of v and v_c for D

- Gradient of $V(G, D)$ w.r.t θ_D:

$$\nabla_{\theta_D} V(G, D) = \begin{cases} \nabla_{\mathbf{d}_v, \mathbf{d}_{v_c}} \log D(v, v_c; \theta_D), & if\ v \sim p_{\text{true}}; \\ \nabla_{\mathbf{d}_v, \mathbf{d}_{v_c}} (1 - \log D(v, v_c; \theta_D)), & if\ v \sim G. \end{cases} \tag{3}$$

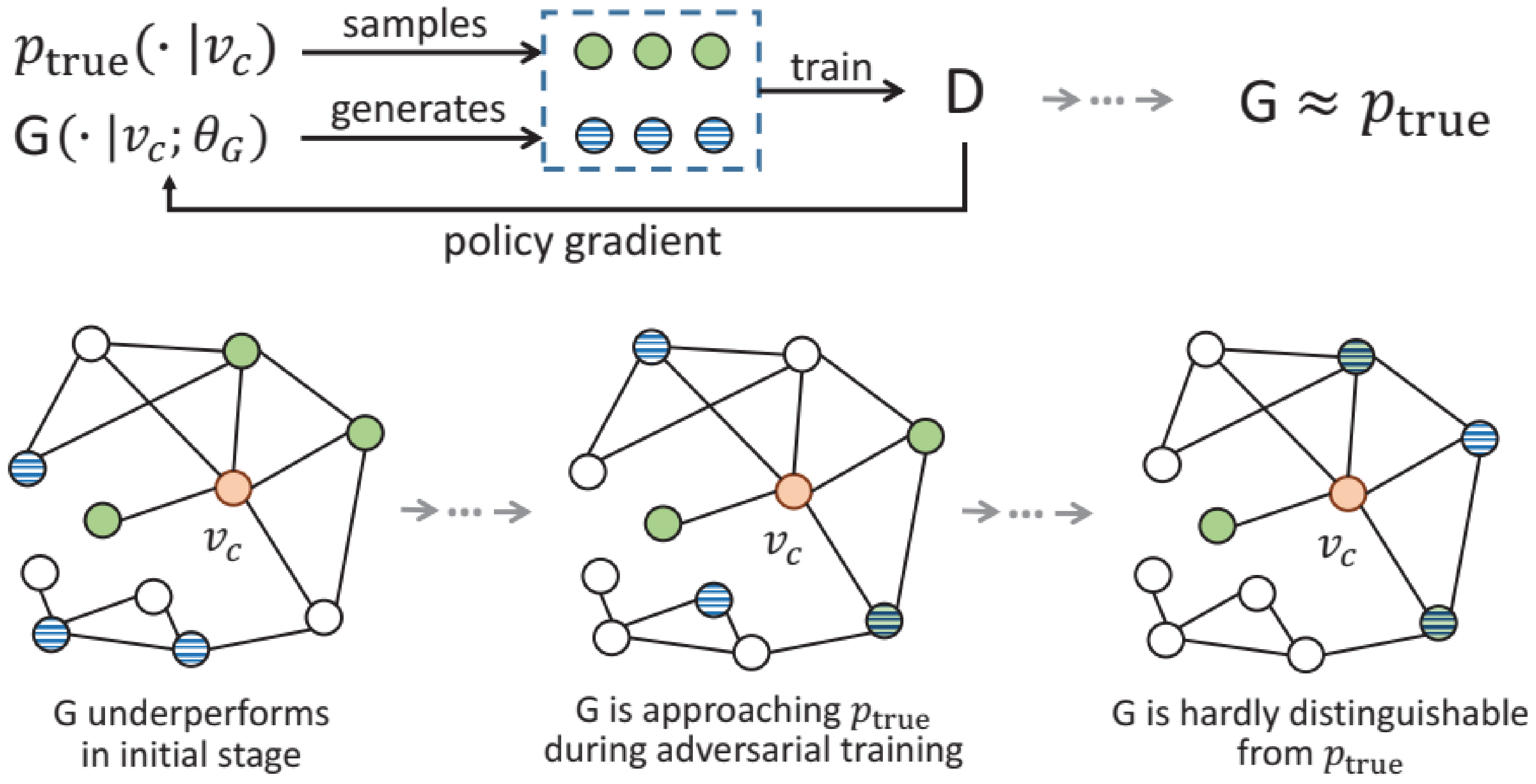
Optimization of G

$$\min_{\theta_G}\max_{\theta_D} V(G,D) = \sum_{c=1}^{V}\Big(\mathbb{E}_{v\sim p_{\mathrm{true}}(\cdot|v_c)}\big[\log D(v,v_c;\theta_D)\big]+$$
$$\mathbb{E}_{v\sim G(\cdot|v_c;\theta_G)}\big[\log\left(1-D(v,v_c;\theta_D)\right)\big]\Big) \tag{1}$$

- Gradient of $V(G,D)$ w.r.t θ_G (policy gradient):

$$\begin{aligned}
&\nabla_{\theta_G} V(G,D)\\
=&\nabla_{\theta_G}\sum_{c=1}^{V}\mathbb{E}_{v\sim G(\cdot|v_c;\theta_G)}\big[\log\left(1-D(v,v_c;\theta_D)\right)\big]\\
=&\sum_{c=1}^{V}\sum_{i=1}^{N}\nabla_{\theta_G} G(v_i|v_c;\theta_G)\log\left(1-D(v_i,v_c;\theta_D)\right)\\
=&\sum_{c=1}^{V}\sum_{i=1}^{N} G(v_i|v_c;\theta_G)\nabla_{\theta_G}\log G(v_i|v_c;\theta_G)\log\left(1-D(v_i,v_c;\theta_D)\right)\\
=&\sum_{c=1}^{V}\mathbb{E}_{v\sim G(\cdot|v_c;\theta_G)}\big[\nabla_{\theta_G}\log G(v|v_c;\theta_G)\log\left(1-D(v,v_c;\theta_D)\right)\big].
\end{aligned} \tag{4}$$

graphGAN



[GitHub - liutongyang/GraphGAN-pytorch: (Still in coding)A pytorch implementation of GraphGAN (Graph Representation Learning with Generative Adversarial Nets) from hwwang55]

[如何进行优化：论文解读-GraphGan_哔哩哔哩_bilibili]

Graph Generative Adversarial Networks for Sparse Data Generation in High Energy Physics

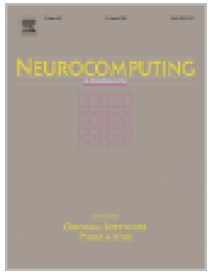
Raghav Kansal, Javier Duarte
University of California San Diego
La Jolla, CA 92093, USA

Breno Orzari, Thiago Tomei
Universidade Estadual Paulista
São Paulo/SP - CEP 01049-010, Brazil

Maurizio Pierini, Mary Touranakou*
European Organization for Nuclear Research (CERN)
CH-1211 Geneva 23, Switzerland

Jean-Roch Vlimant
California Institute of Technology
Pasadena, CA 91125, USA

Dimitrios Gunopulos
National and Kapodistrian University of Athens
Athens 15772, Greece



Neurocomputing

Volume 450, 25 August 2021, Pages 242-252

Molecular generative Graph Neural Networks for Drug Discovery

Pietro Bongini [a] [b], Monica Bianchini [a], Franco Scarselli [a]

Show more

Add to Mendeley Share Cite

https://doi.org/10.1016/j.neucom.2021.04.039

Get rights and content