SSII 2019

The 25th
Symposium on Sensing
via Image Information

Shall We GANs?

2019.6.12

高橋 智洋（オムロン）

＊この資料は動画未対応です．．．

自己紹介

- 高橋 智洋
  - 所属: オムロン (2018 年 6 月入社)
- 興味
  - 理論物理: 学生時代は一般相対論の研究をしてました.
  - 数理計画法: 離散最適について調査・実装.
  - 機械学習: 今の仕事.　最近はロボティクス関連も.

GANの研究例

理論面

Lossを工夫
- LSGAN
- WGAN
- Coulomb GAN
- WGAN-GP

計算の安定性向上
- SNGAN
- TTUR
- Relativistic GAN

収束性向上
- Numeric of GANs

Missing mode
- Unrolled GAN
- VEEGAN
- PacGAN
- BourGAN

応用例

画像生成
- DCGAN
- Progressive GAN
- BigGAN
- Style-Based GAN

(V)AEとの合わせ技
- AAE
- VAEGAN
- Image Compression

Super resolution
- SRGAN
- ESRGAN

Sequence to figure
- Stack GAN

disentanglement
- InfoGAN
- URDF

domain変換
- CycleGAN
- DiscoGAN

3D
- 3DRecGAN

Domain Adaptation
- ADDA
- CycADA
- DIRT-T

異常検知
- AnoGAN
- AnoVAEGAN
- Video anomaly detection

GAN の研究例

理論面

Lossを工夫

LSGAN WGAN

Coulomb GAN WGAN-GP

計算の安定性向上

SNGAN TTUR

Relativistic GAN

収束性向上

Numeric of GANs

Missing mode

Unrolled GAN VEEGAN

PacGAN BourGAN

応用例

画像生成

DCGAN

Pr ressive GAN

disentanglement

InfoGAN URDF

domain変換

CycleGAN

64x64 の画像の生成 （arXiv:1511.06434)

GAN の研究例

理論面

Lossを工夫

LSGAN　WGAN

Coulomb GAN　WGAN-GP

計算の安定性向上

SNGAN

Relativistic

収束性向上

Numeric of GANs

Missing mode

Unrolled GAN

PacGAN　Bo

応用例

画像生成

DCGAN

Progressive GAN

BigGAN

Styl　sed GAN

disentanglement

InfoGAN　URDF

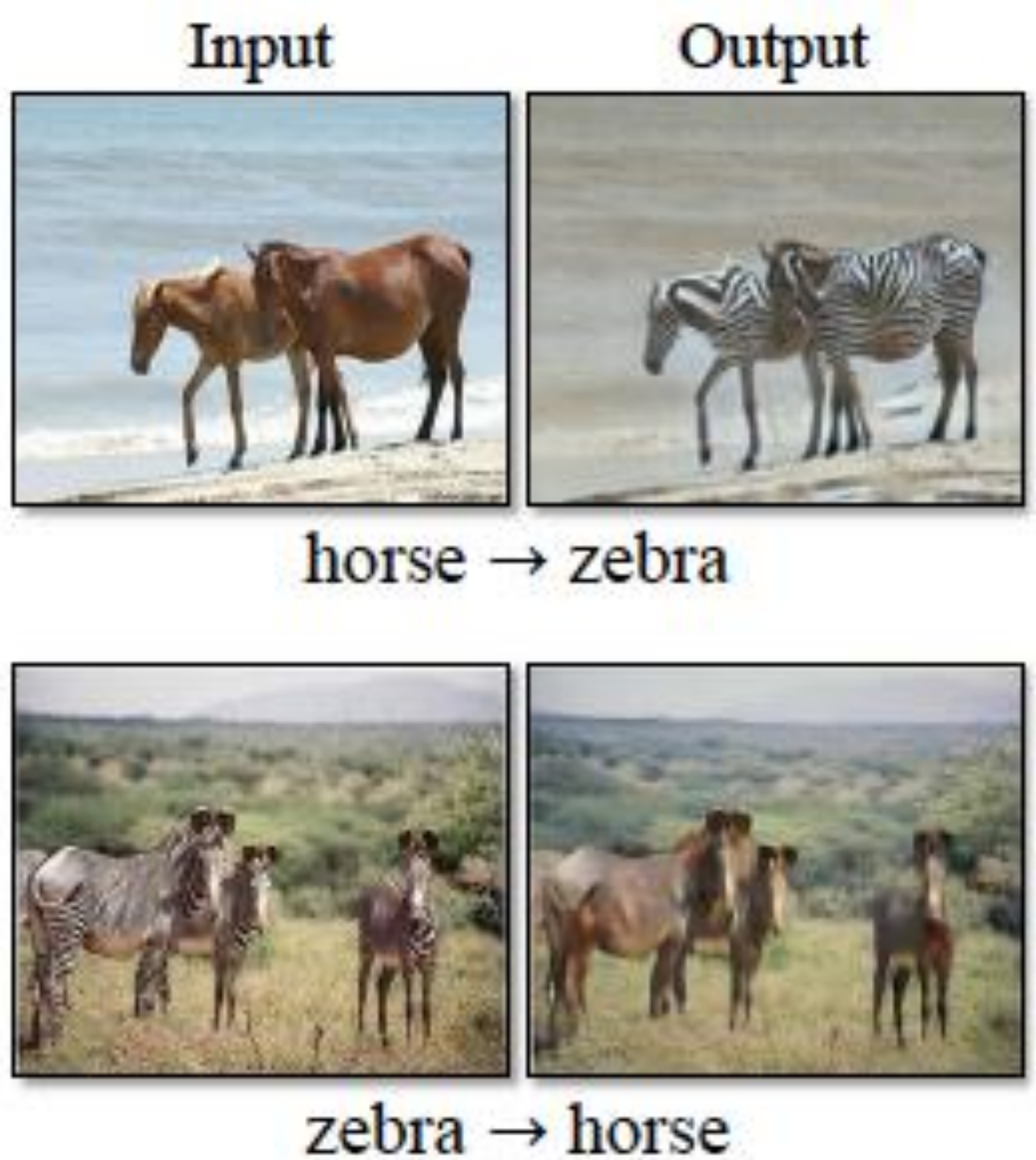
domain変換

CycleGAN

高解像度かつ本物と見分けのつかない画像生成

(arXiv:1809.11096)

GAN の研究例



画像の domain 変換 （arXiv:1703.10593）

応用例

画像生成

DCGAN

ive GAN

BourGAN

disentanglement

InfoGAN URDF

domain変換

CycleGAN

DiscoGAN

(V)AEとの合わせ技

AAE

VAEGAN

Image Compression

3D

3DRecGAN

Super resolution

SRGAN ESRGAN

Domain Adaptation

ADDA

CycADA

DIRT-T

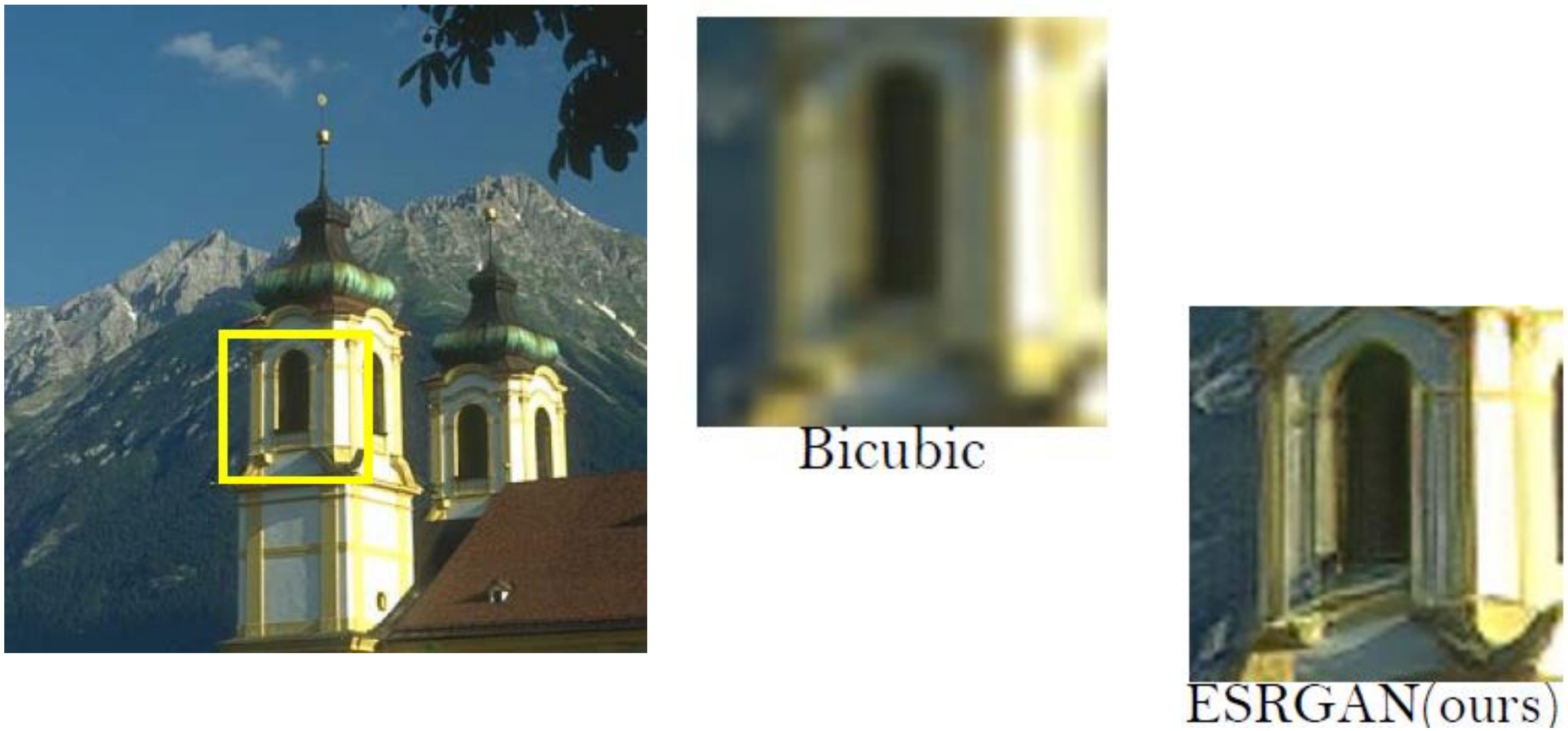
Sequence to figure

Stack GAN

異常検知

AnoGAN AnoVAEGAN

Video anomaly detection



超解像に利用. (arXiv: 180900219)

URDF

収束性向上

Numeric of GANs

Missing mode

Unrolled GAN VEEGAN

PacGAN BourGAN

Super resoluti

SRGAN **ESRGAN**

ADDA

CycADA

DIRT-T

Sequence to figure

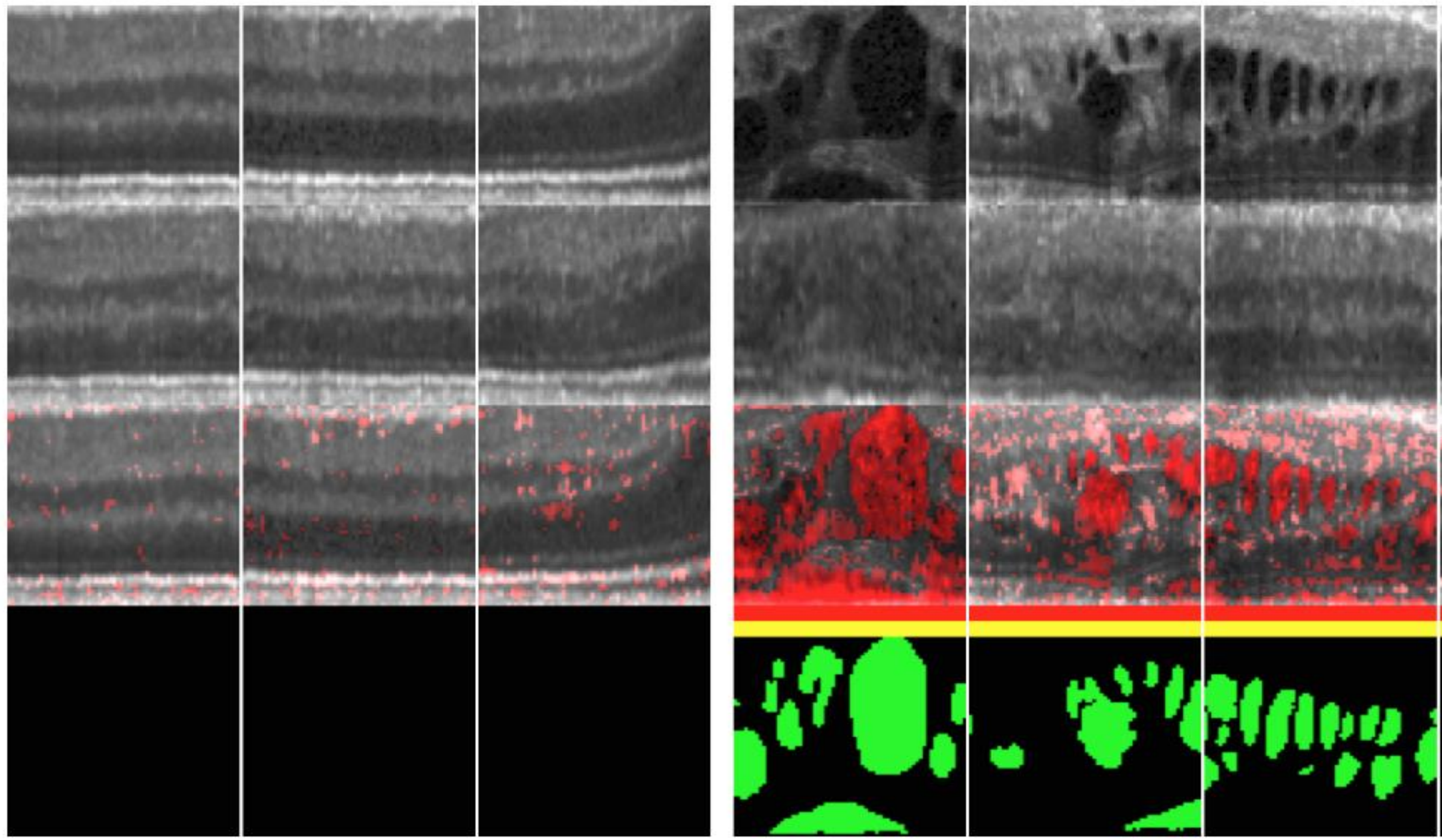
Stack GAN

異常検知

AnoGAN AnoVAEGAN

Video anomaly detection

GAN の研究例



異常検知に利用.

(arXiv:1703.05921)

理論面

Lossを工

LSGAN

Coulomb

計算の

SNGAN

Relati

収束性

Numer

Missing mode

Unrolled GAN VEEGAN

PacGAN BourGAN

Sequence to figure

Stack GAN

異常検知

AnoGAN AnoVAEGAN

Video anomaly detection

GAN の研究例

理論面

Lossを工夫
- LSGAN
- WGAN
- Coulomb GAN

計算の安定性向上
- SNGAN
- TTUR
- Relativistic GAN

収束性向上
- Numeric of GANs

Missing mode
- Unrolled GAN
- VEEGAN
- PacGAN
- BourGAN

✓ 計算が安定しない.
✓ 収束性が悪い.
✓ Missing mode が出る.
などの問題があり,
うまく学習できないことが多々ある.
それらを解決するような研究.

応用例

画像生成
- DCGAN

disentanglement
- InfoGAN
- URDF

- SRGAN
- ESRGAN
- DIRT-T

Sequence to figure
- Stack GAN

異常検知
- AnoGAN
- AnoVAEGAN
- Video anomaly detection

GAN の研究例

理論面

Lossを工夫
LSGAN
WGAN
Coulomb GAN
WGAN-GP

計算の安定性向上
SNGAN
TTUR
Relativistic GAN

収束性向上
Numeric of GANs

Missing mode
Unrolled GAN
VEEGAN
PacGAN
BourGAN

応用例

画像生成
DCGAN
Progressive GAN
BigGAN
Style-Based GAN

(V)AEとの合わせ技
AAE
VAEGAN
Image Compression

Super resolution
SRGAN
ESRGAN

Sequence to figure
Stack GAN

disentanglement
InfoGAN
URDF

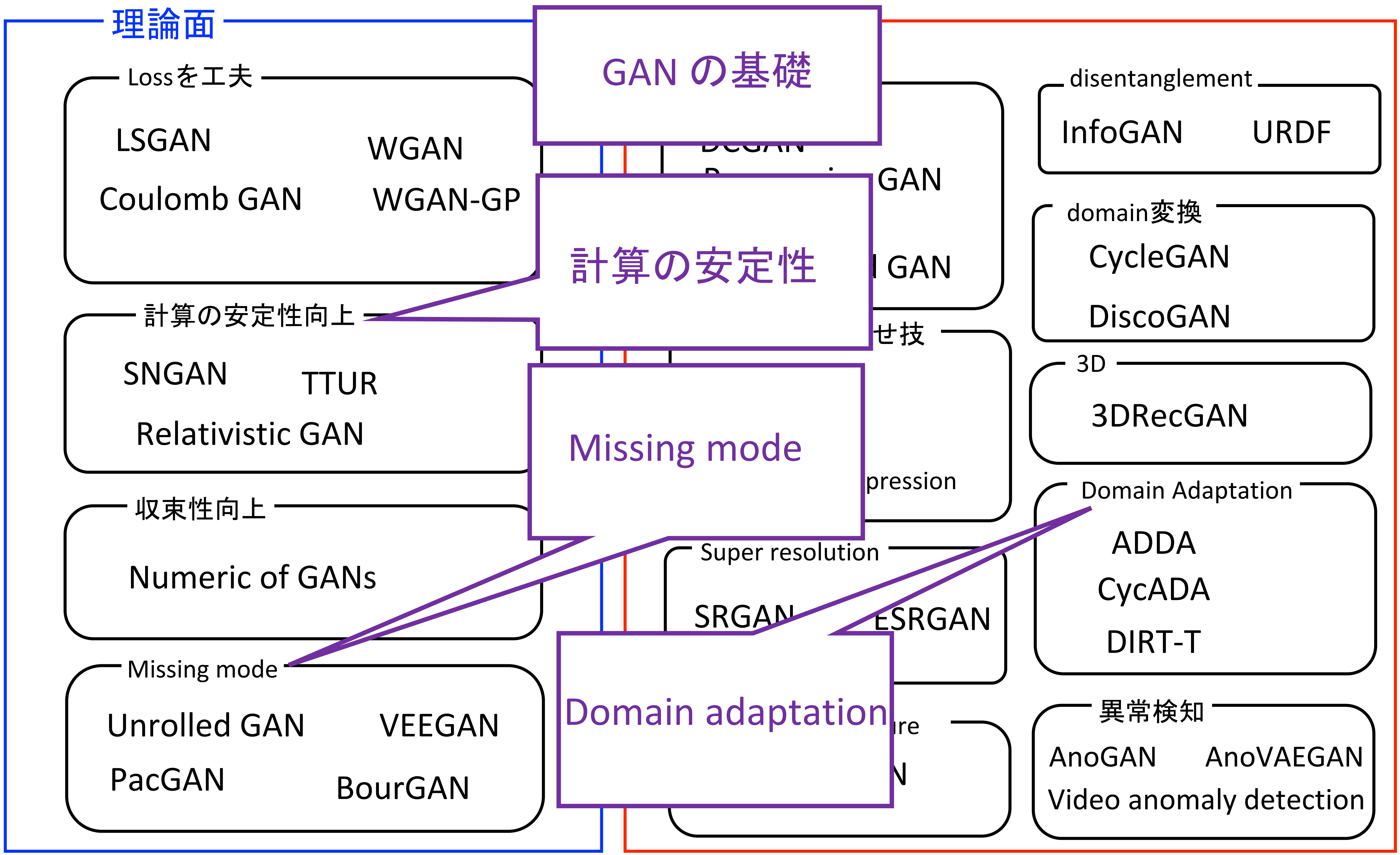
domain変換
CycleGAN
DiscoGAN

3D
3DRecGAN

Domain Adaptation
ADDA
CycADA
DIRT-T

異常検知
AnoGAN
AnoVAEGAN
Video anomaly detection

GAN の研究例



理論面
Lossを工夫
LSGAN
WGAN
Coulomb GAN
WGAN-GP
計算の安定性向上
SNGAN
TTUR
Relativistic GAN
収束性向上
Numeric of GANs
Missing mode
Unrolled GAN
VEEGAN
PacGAN
BourGAN
GAN の基礎
計算の安定性
Missing mode
Domain adaptation
GAN
GAN
せ技
pression
Super resolution
SRGAN
ESRGAN
disentanglement
InfoGAN
URDF
domain変換
CycleGAN
DiscoGAN
3D
3DRecGAN
Domain Adaptation
ADDA
CycADA
DIRT-T
異常検知
AnoGAN
AnoVAEGAN
Video anomaly detection

目次

発表用に作成したコードは大体ここにある.
https://github.com/takat0m0

*「Loss 変更」「力学系との関係」「異常検知」など，上記以外の話
- https://www.slideshare.net/TomohiroTakahashi2/miru-miru-gan
- https://www.slideshare.net/TomohiroTakahashi2/20171012-prmu-80820422

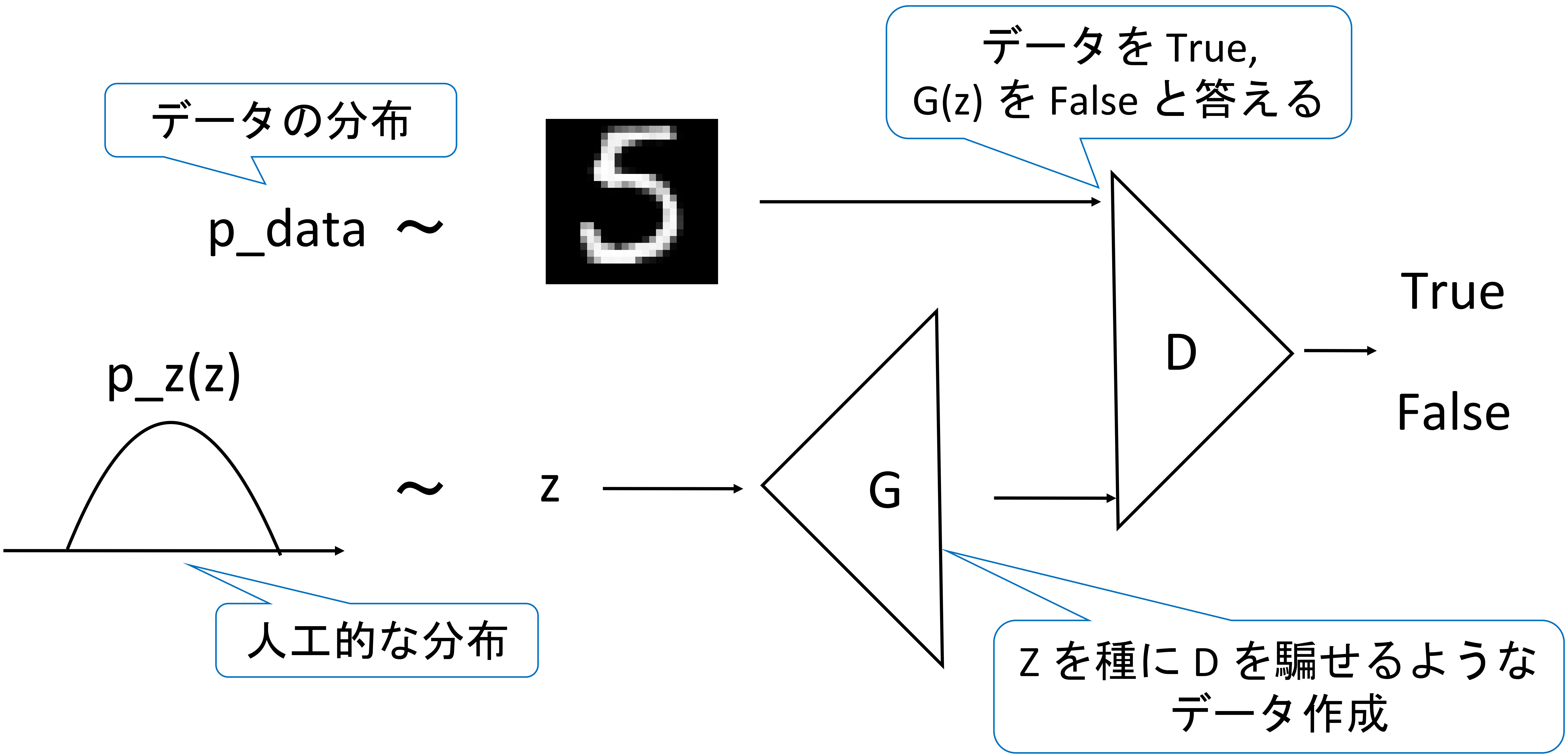
1. Original GAN

GAN 概要
どういう最適化問題を解けば良いの?
何故その最適化問題で良いの?
数値実験結果

参考文献
- arXiv:1406.2661
- arXiv:1511.06434
- arXiv:1511.05644

GAN 概要

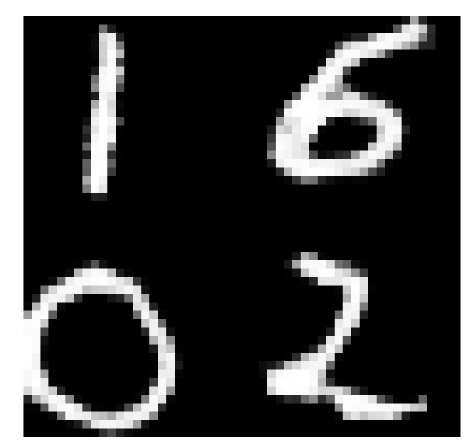
- 登場人物は， p_data, p_z, discriminator, generator の四人.



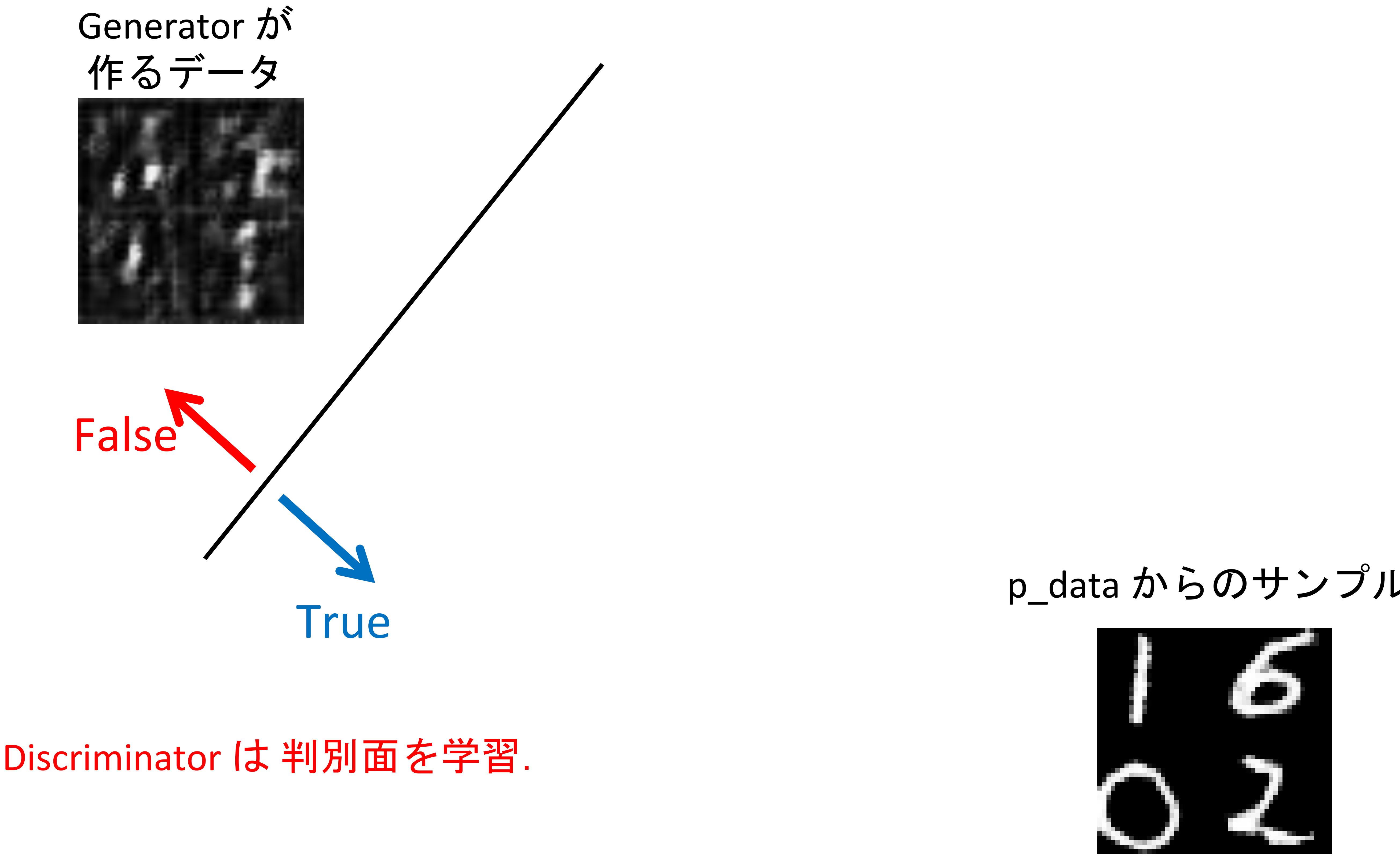
GAN 概要

Generator が
作るデータ

p_data からのサンプル



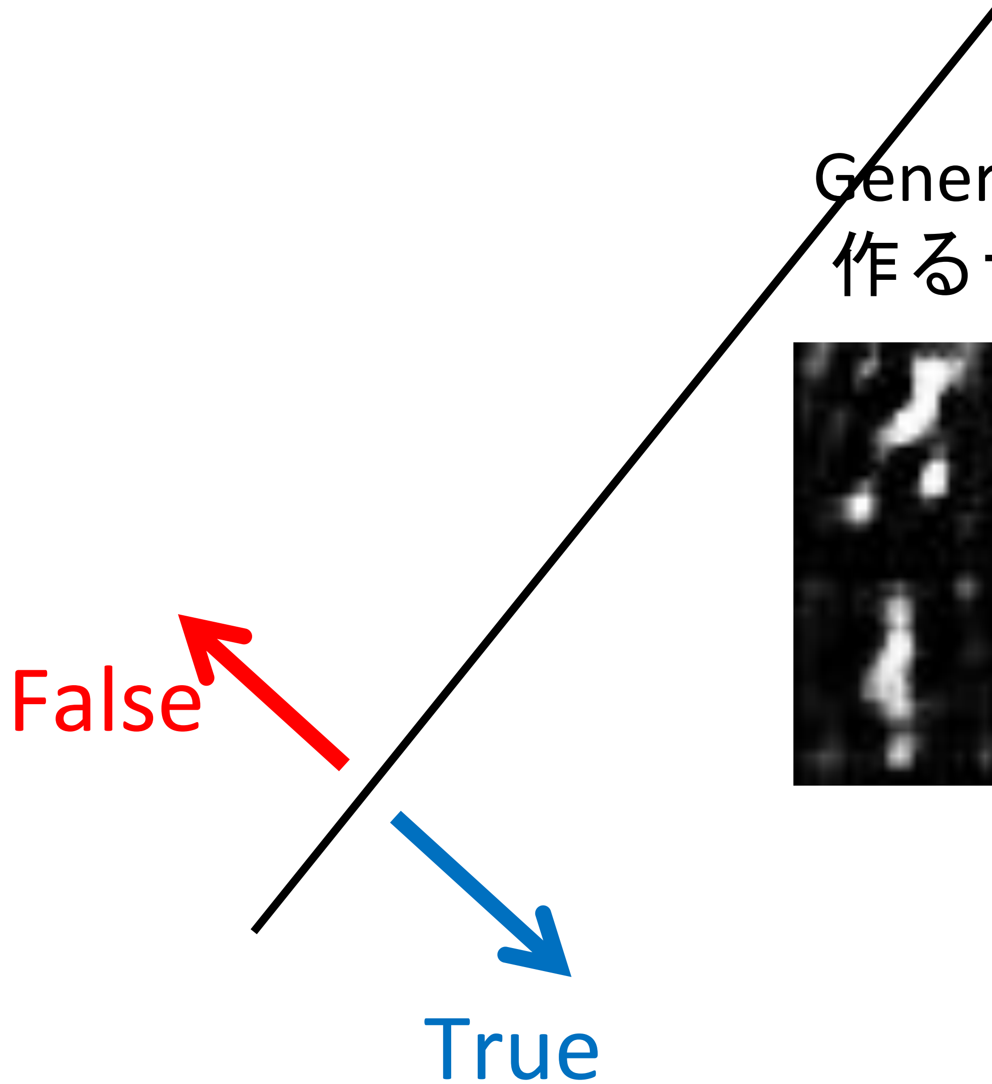
GAN 概要



Generator が
作るデータ
False
True
Discriminator は 判別面を学習.
p_data からのサンプル

GAN 概要

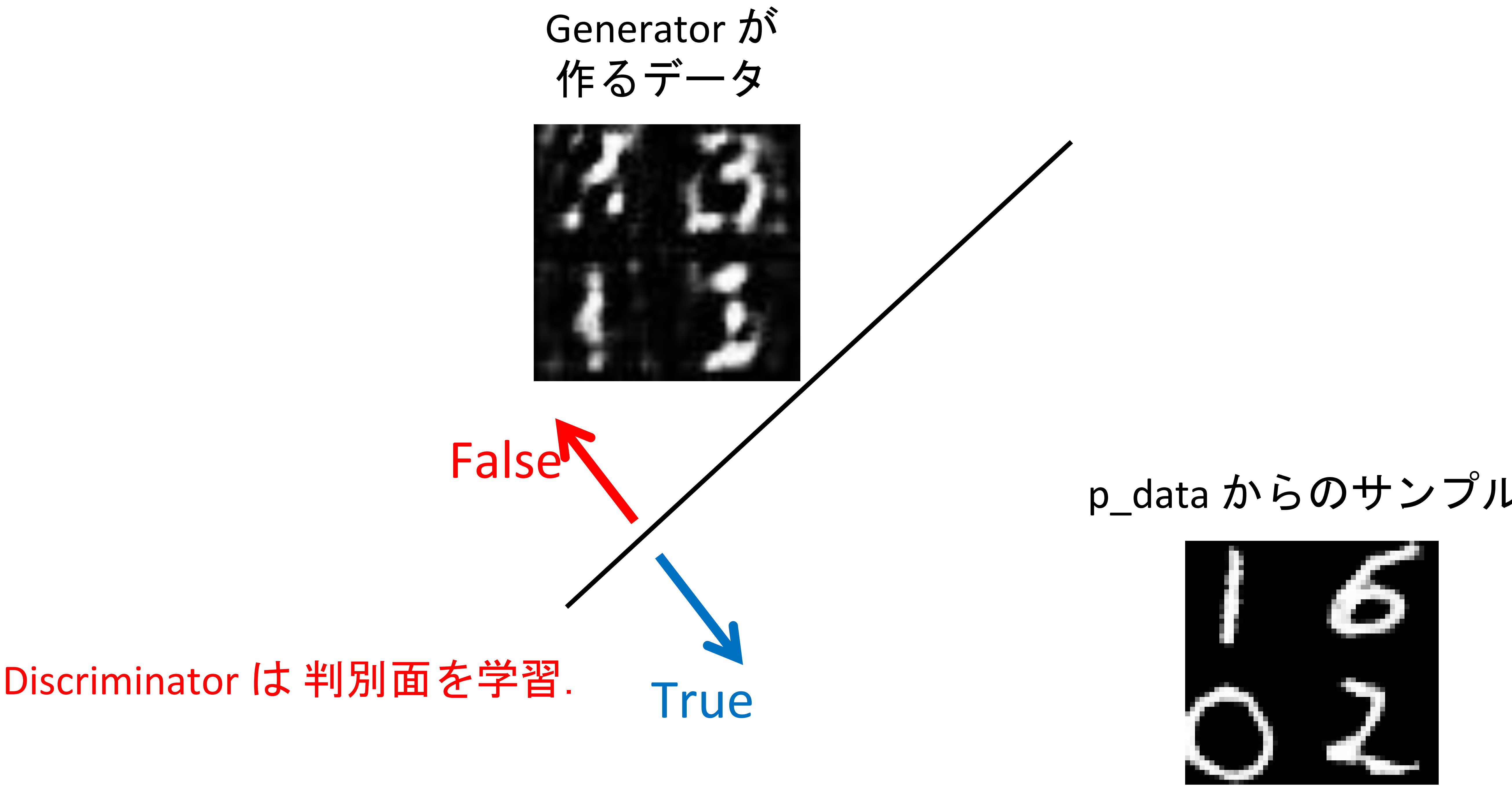
Generatorは，判別面を固定して
True と言われる様に学習



Generator が
作るデータ

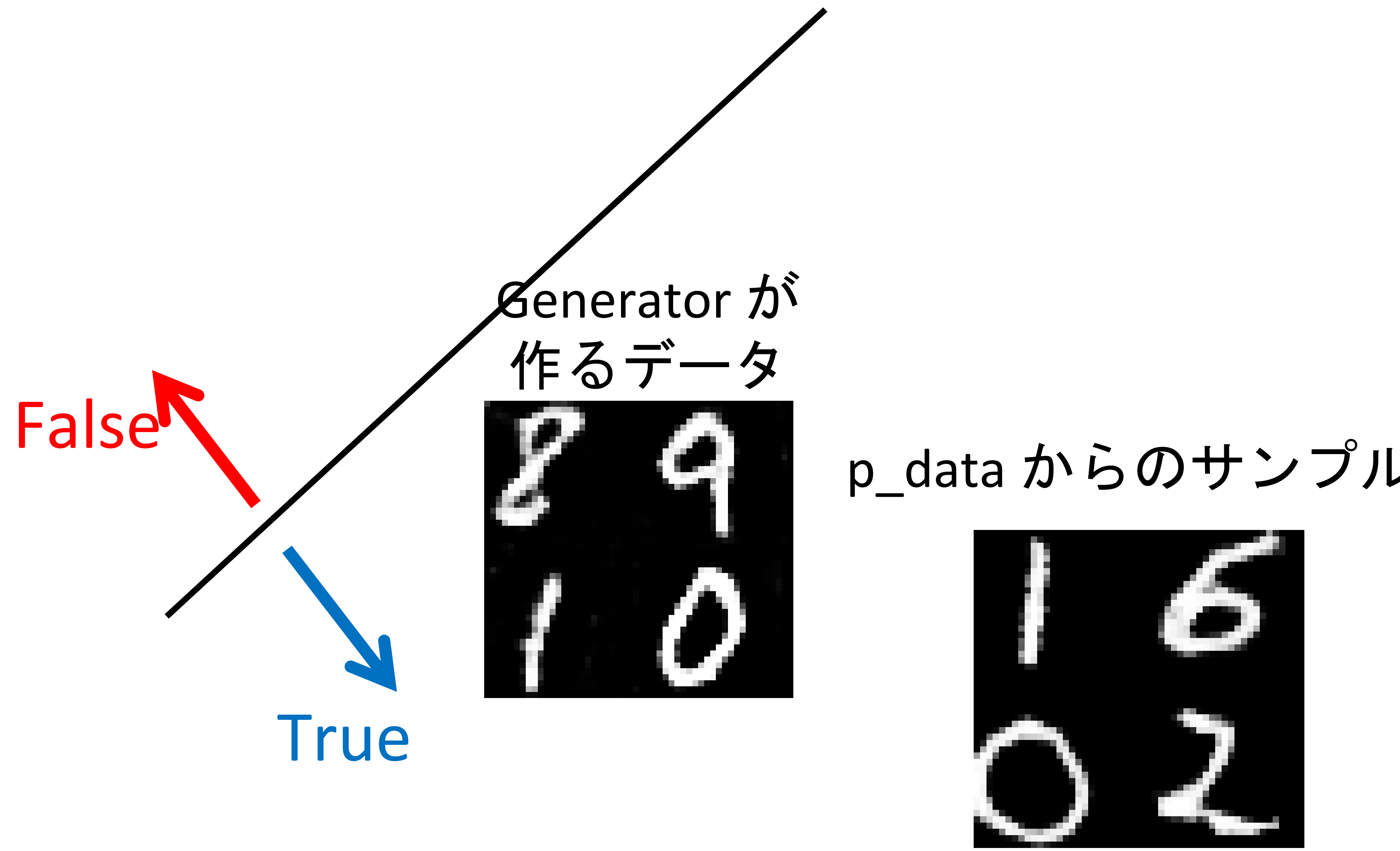
p_data からのサンプル

GAN 概要


Generator が
作るデータ
False
True
p_data からのサンプル
Discriminator は 判別面を学習.

GAN 概要

Generatorは，判別面を固定して
True と言われる様に学習



どういう問題を解けば良いの?

- 以下が前項に対応しそうな最適化問題.

$$\min_{G} \max_{D} \mathbb{E}_{x \sim p_{data}} \left[\log \left\{ D(x) \right\} \right] + \mathbb{E}_{z \sim p_z} \left[\log \left\{ 1 - D(G(z)) \right\} \right]$$

どういう問題を解けば良いの?

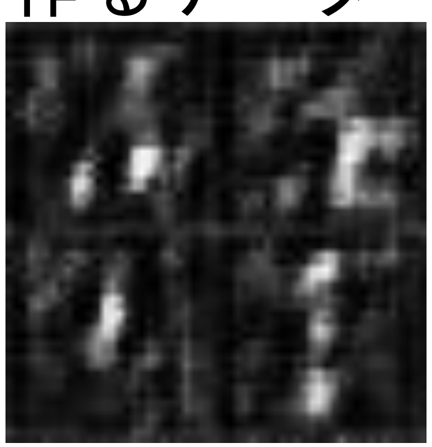
- 以下が前項に対応しそうな最適化問題.

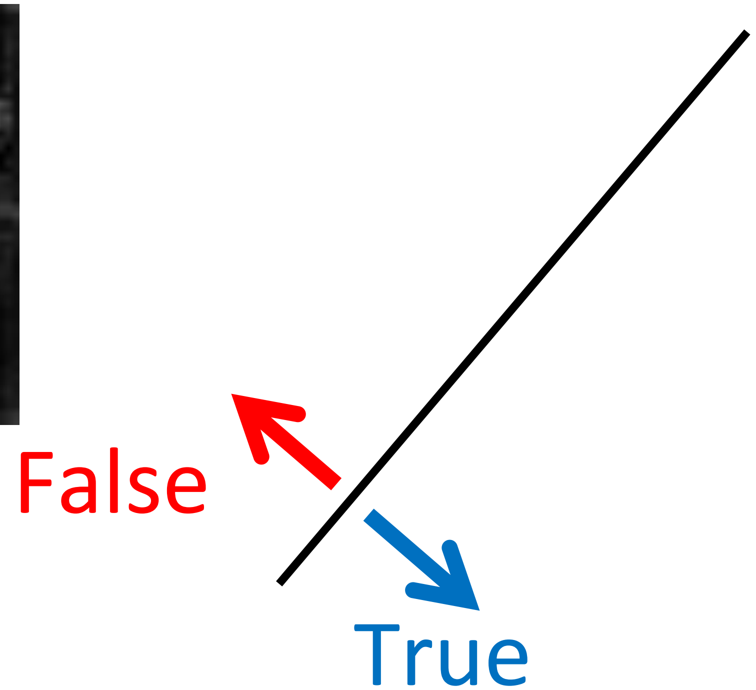
$$\min_{G} \boxed{\max_{D}} \mathbb{E}_{x\sim p_{data}}\left[\log\{D(x)\}\right] + \mathbb{E}_{z\sim p_z}\left[\log\{1 - D(G(z))\}\right]$$

P_data からの draw x に対して,
D(x) = 1 とすれば最大.

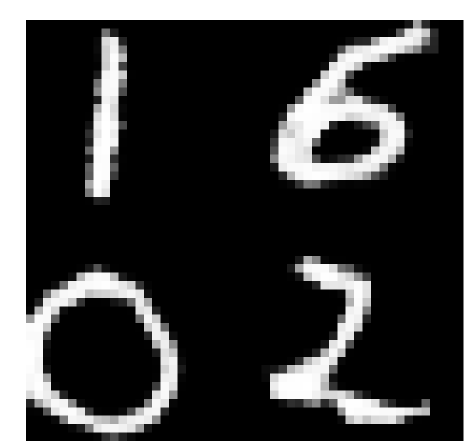
G(z) に対して,
D(G(z)) = 0 とすれば最大.

Generator が
作るデータ





p_data からのサンプル

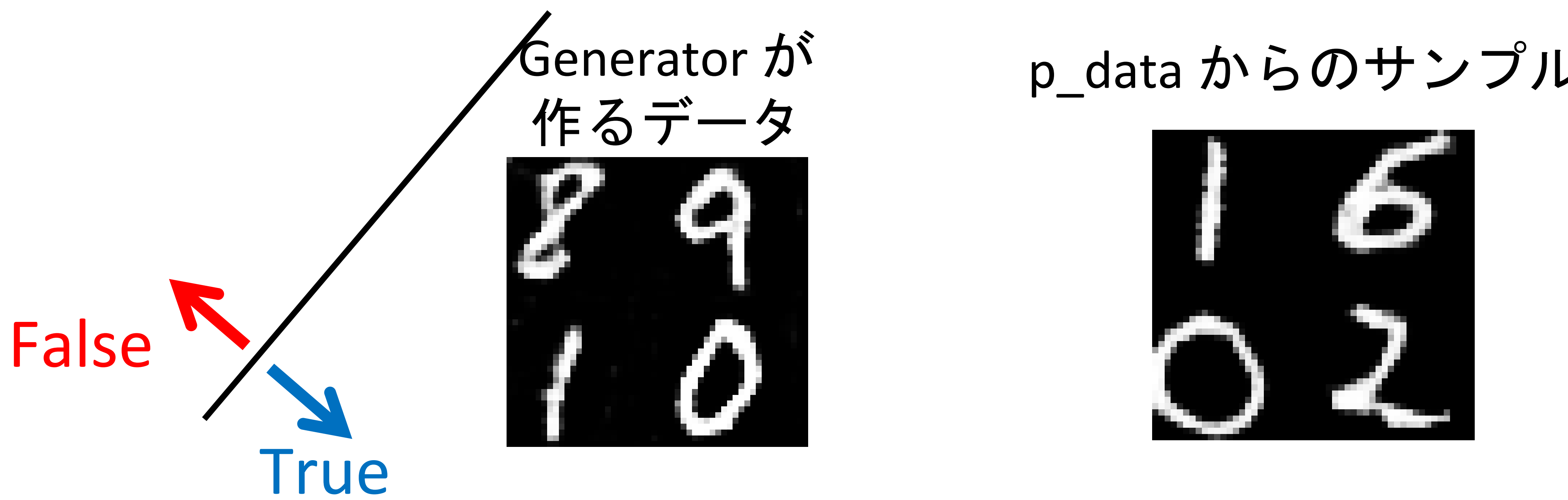


どういう問題を解けば良いの?

- 以下が前項に対応しそうな最適化問題.

$$\boxed{\min_{G}} \max_{D} \mathbb{E}_{x \sim p_{data}} \left[\log \left\{ D(x) \right\} \right] + \underline{\mathbb{E}_{z \sim p_z} \left[\log \left\{ 1 - D(G(z)) \right\} \right]}$$

G(z) に対して,
D(G(z)) = 1 とすれば最小.



どういう問題を解けば良いの?

- 以下が前項に対応しそうな最適化問題.

$$\min_G \max_D \mathbb{E}_{x\sim p_{data}}\left[\log\left\{D(x)\right\}\right] + \mathbb{E}_{z\sim p_z}\left[\log\left\{1 - D(G(z))\right\}\right]$$

- 次項以降で，何故これで良いのかを見る.
  - 結論は，上記の最適解が以下の2つの確率分布が一致するときだから.
    - データ分布 p_data
    - P_z と G から導出される確率分布 p_g (p_g(G(z)) = p_z(z)/(dG/dz))

何故その問題で良いの?

- まず， max_D を考えてみる.

$$\min_G \boxed{\max_D \mathbb{E}_{x\sim p_{data}}\left[\log\{D(x)\}\right] + \mathbb{E}_{z\sim p_z}\left[\log\{1 - D(G(z))\}\right]}$$

$$\mathbb{E}_{x\sim p_{data}}\left[\log\{D(x)\}\right] + \mathbb{E}_{z\sim p_z}\left[\log\{1 - D(G(z))\}\right]$$

$$= \int dx \left[\underline{p_{data}(x)\log\{D(x)\} + p_g(x)\log\{1 - D(x)\}}\right]$$

$$D_G(x) = \frac{p_{data}}{p_{data} + p_g}$$ の時に最大

何故その問題で良いの?

- 続いて min_G を考える.

$$\boxed{\max_D \mathbb{E}_{x\sim p_{data}}\left[\log\{D(x)\}\right] + \mathbb{E}_{z\sim p_z}\left[\log\{1 - D(G(z))\}\right]}$$

$$= \mathbb{E}_{x\sim p_{data}}\left[\log\left\{\frac{p_{data}}{p_g + p_{data}}\right\}\right] + \mathbb{E}_{z\sim p_z}\left[\log\left\{\frac{p_g}{p_g + p_{data}}\right\}\right]$$

$$= -\log(4) + \underline{JS(p_{data}\|p_g)}$$

p_data = p_g の時に最小

提案手法

- min max 最適化？ どうすれば良いか分からないから交互で.

$$\boxed{\min_G} \boxed{\max_D} \mathbb{E}_{x \sim p_{data}} \left[\log \{D(x)\}\right] + \mathbb{E}_{z \sim p_z} \left[\log \{1 - D(G(z))\}\right]$$

Gを止めて，Dについて以下を一回だけ勾配降下

$$\max_D \mathbb{E}_{x \sim p_{data}} \left[\log \{D(x)\}\right] + \mathbb{E}_{z \sim p_z} \left[\log \{1 - D(G(z))\}\right]$$

Dを止めて，Gについて以下を一回だけ勾配降下

$$\min_G \mathbb{E}_{z \sim p_z} \left[\log \{1 - D(G(z))\}\right]$$

提案手法

- min max 最適化？ どうすれば良いか分からないから交互で.

$$\boxed{\min_{G}}\boxed{\max_{D}} \mathbb{E}_{x\sim p_{data}}\left[\log\left\{D(x)\right\}\right] + \mathbb{E}_{z\sim p_z}\left[\log\left\{1 - D(G(z))\right\}\right]$$

Gを止めて，Dについて以下を一回だけ勾配降下

$$\max_{D} \mathbb{E}_{x\sim p_{data}}\left[\log\left\{D(x)\right\}\right] + \mathbb{E}_{z\sim p_z}\left[\log\left\{1 - D(G(z))\right\}\right]$$

Dを止めて，Gについて以下を一回だけ勾配降下

$$\min_{G} \mathbb{E}_{z\sim p_z}\left[\log\{1 - D(G(z))\}\right]$$

提案手法

- min max 最適化？ どうすれば良いか分からないから交互で.

$$\boxed{\min_G} \boxed{\max_D} \mathbb{E}_{x \sim p_{data}} [\log \{D(x)\}] + \mathbb{E}_{z \sim p_z} [\log \{1 - D(G(z))\}]$$

Gを止めて，Ｄについて以下を一回だけ勾配降下

$$\max_D \mathbb{E}_{x \sim p_{data}} [\log \{D(x)\}] + \mathbb{E}_{z \sim p_z} [\log \{1 - D(G(z))\}]$$

Dを止めて，Ｇについて以下を一回だけ勾配降下

$$\min_G \mathbb{E}_{z \sim p_z} [\log\{1 - D(G(z))\}]$$

ここまでのまとめ

- 以下の最適化問題を解くことで，p_data = p_g と学習できる.

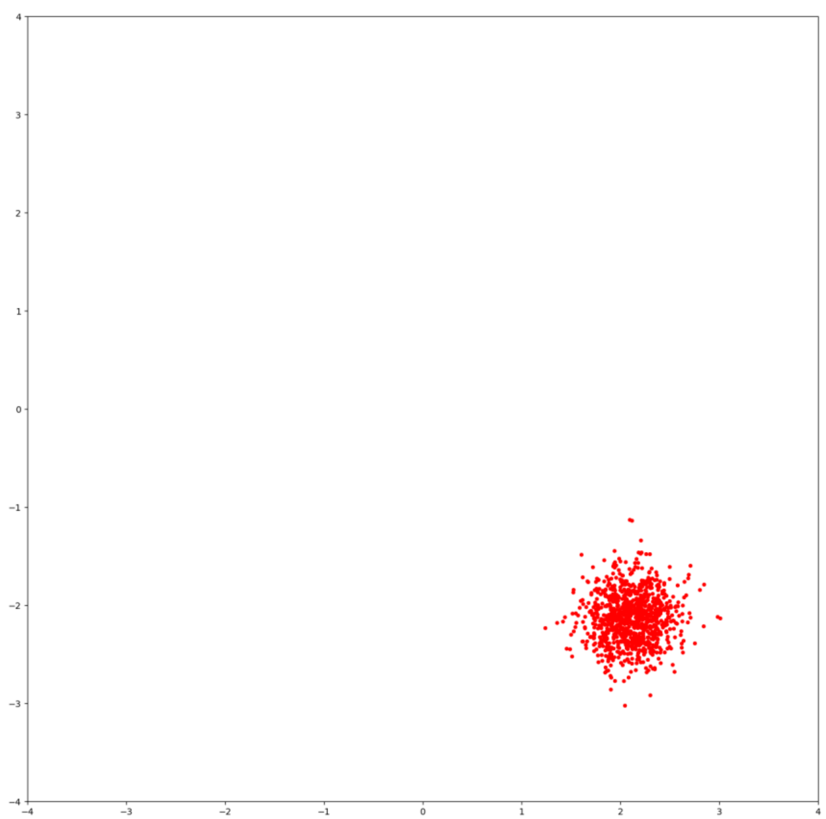
$$\min_G \max_D \mathbb{E}_{x \sim p_{data}} \left[\log \{ D(x) \} \right] + \mathbb{E}_{z \sim p_z} \left[\log \{ 1 - D(G(z)) \} \right]$$

- ちょっとした疑問：　本当に一致するの？

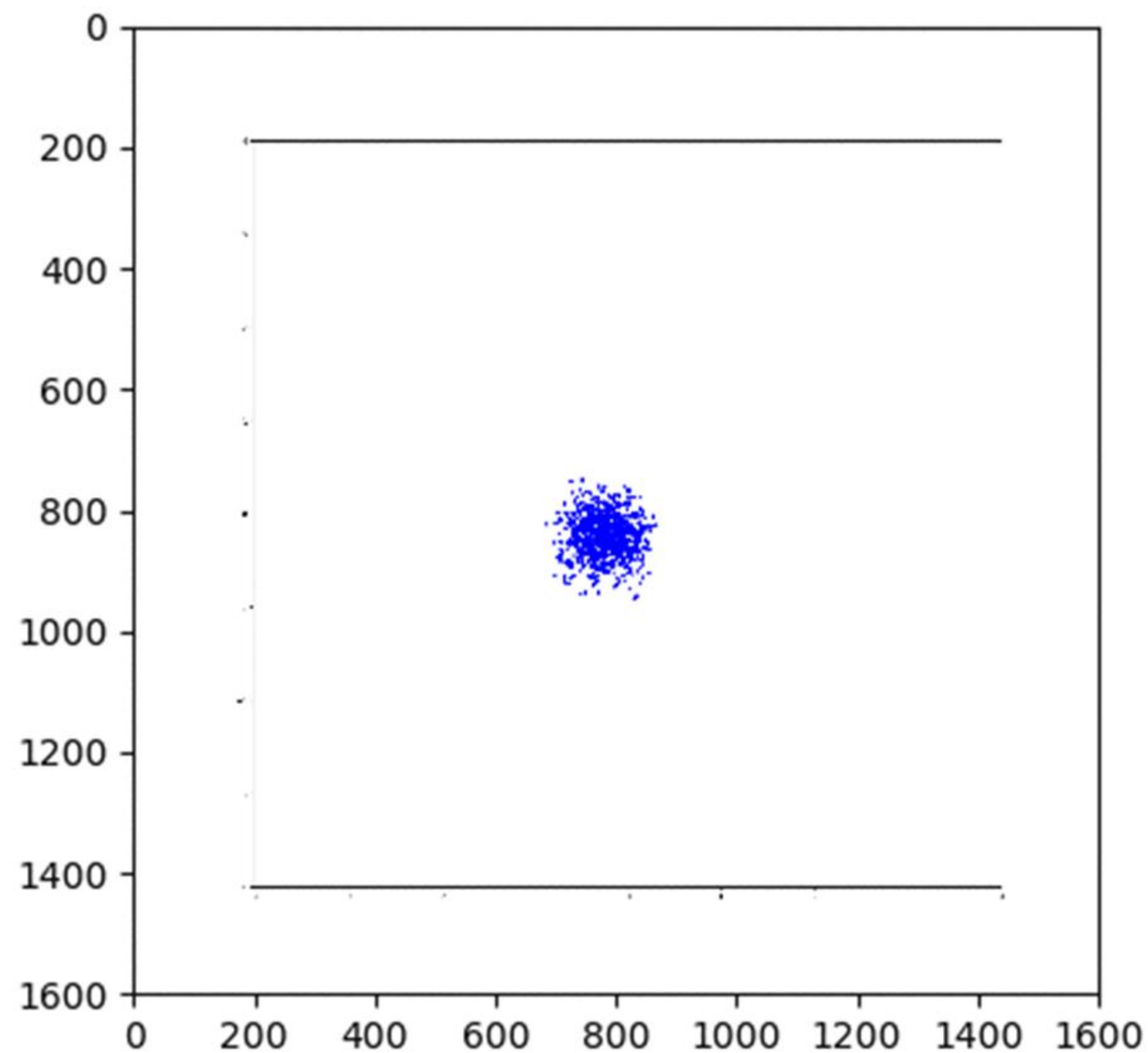
簡単な実験結果

- p_data: 2 次元の Gaussian $N\left(\left(\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}\right), 0.5I\right)$

- p_z: 256 次元[0,1]一様分布



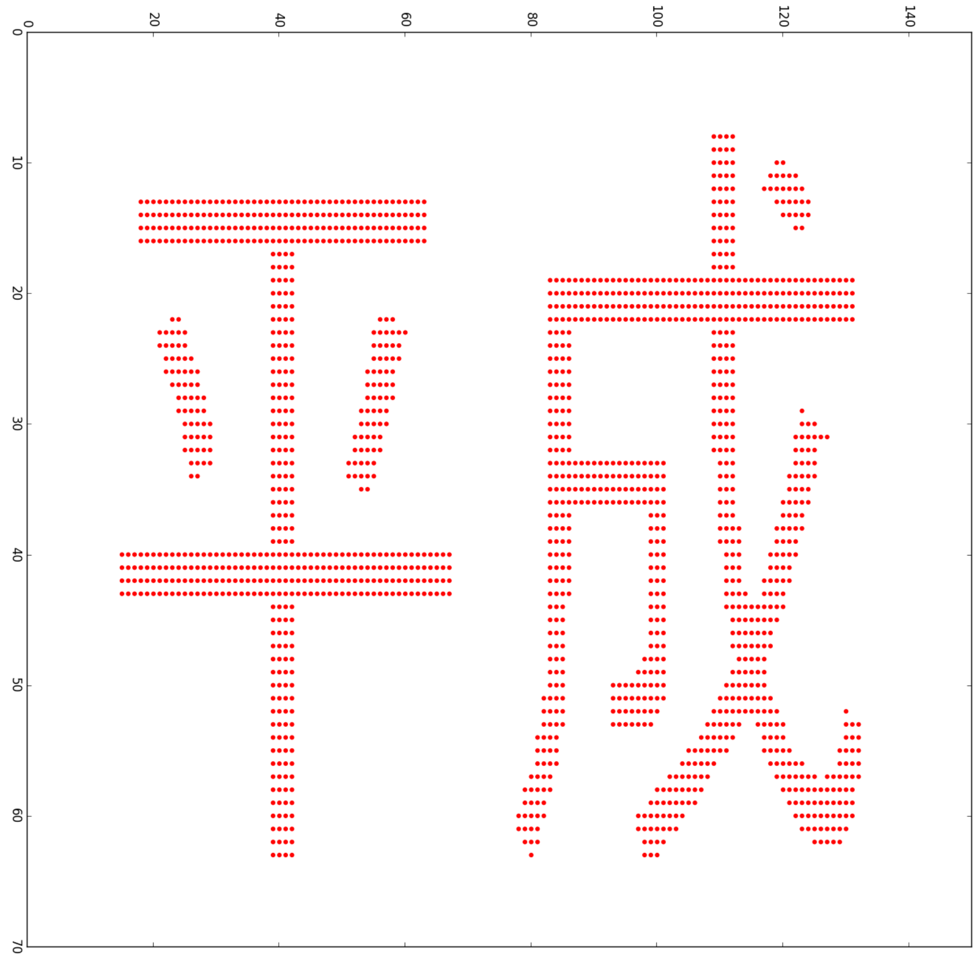
p_data

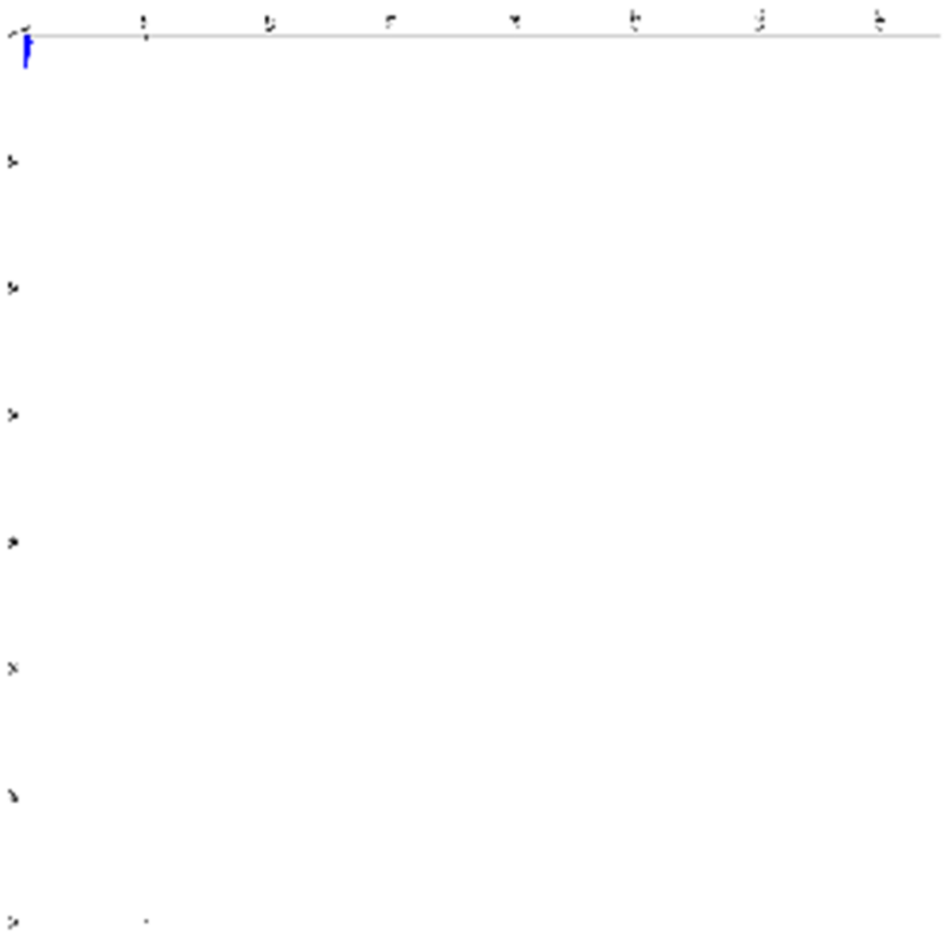


p_z から draw して G で送った先
≒p_g のランダムサンプル

p_data が一様分布な例

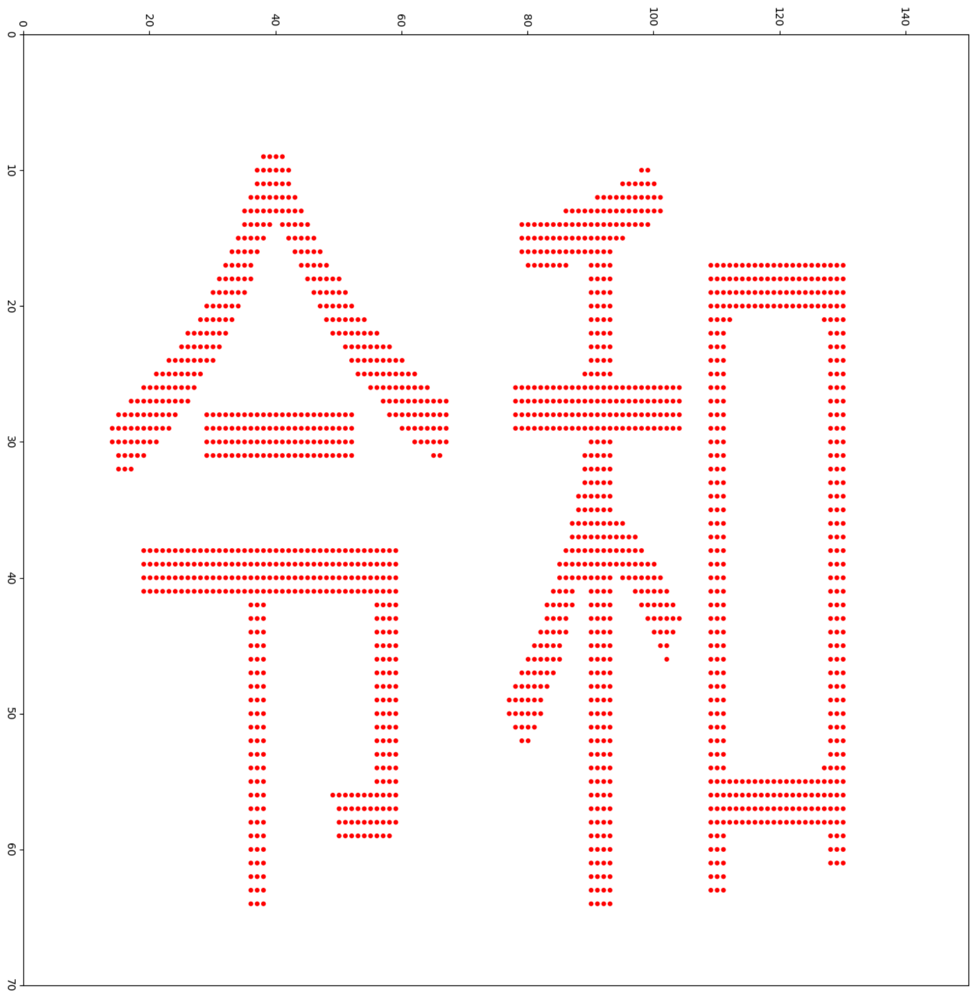
- p_data: 特定の二次元格子点のサンプリング（一様 平成 分布！）
- p_z: 192 次元正規分布 $N(0, I)$



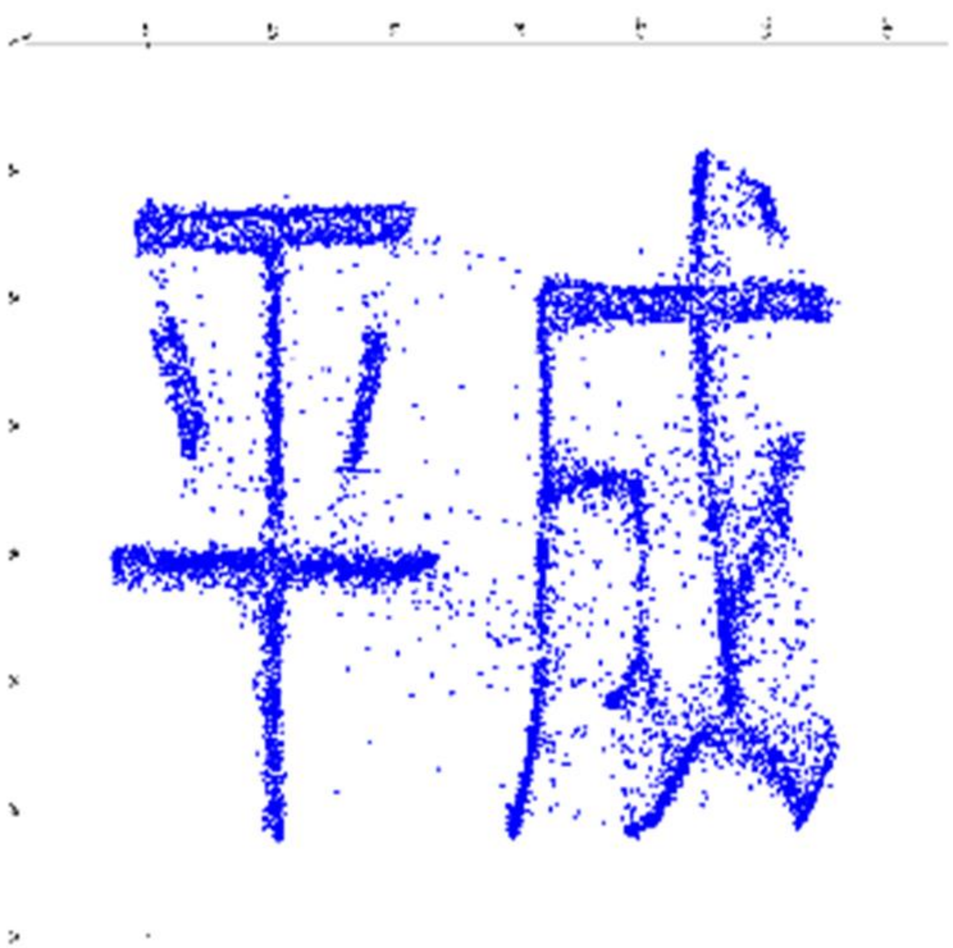


p_data が一様分布な例

- p_data: 特定の二次元格子点のサンプリング（一様 令和 分布！）
- p_z: 192 次元正規分布 $N(0, I)$



「平成」の近似結果を
initial condition に



画像の例

- p_data: 手持ち画像の一様サンプリング
- p_z: 256 次元正規分布 $N(0, I)$

（arXiv:1511.06434より)

GAN まとめ

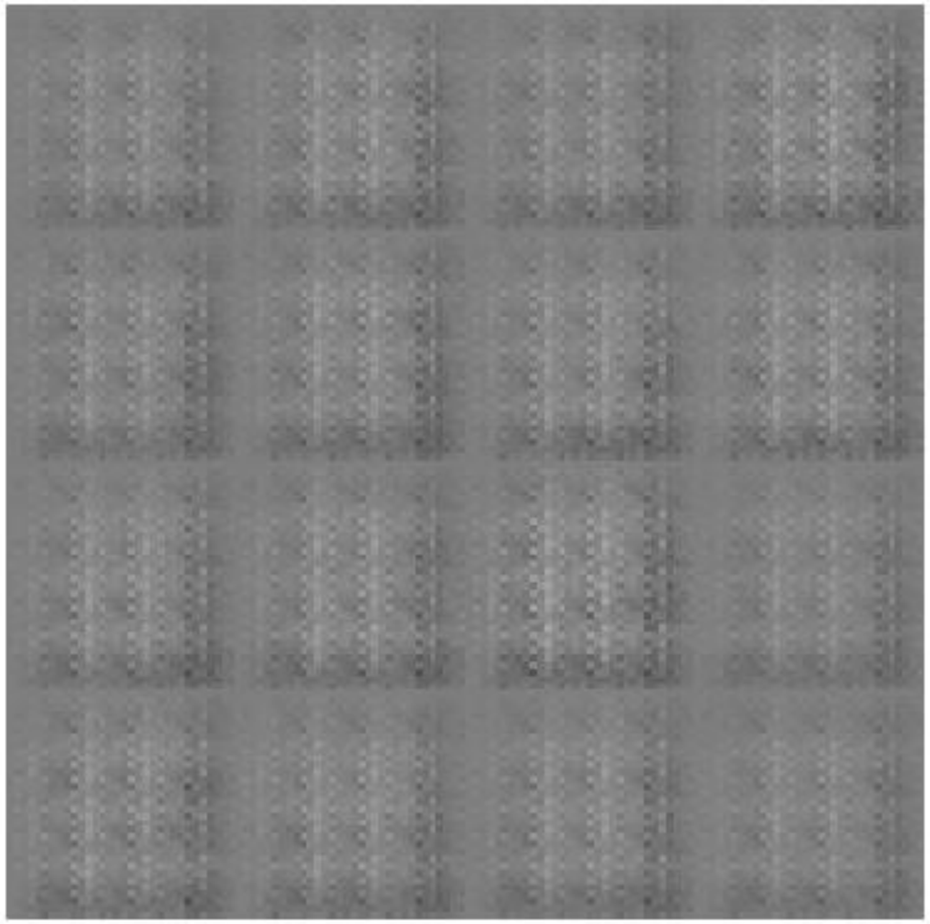
- GAN は二つの分布を一致させるもの.
  - 一様平成分布のように「数式に書けない分布」も近似できる.
- だから，P_data として「手持ち画像の一様サンプリング」を持ってくると，手持ち画像に近しい画像が生成できる.

Naïve にやってみると．．．

- ところが，Naïve に実装して学習してみると，様々な問題が起こる.

計算がクラッシュする．．．

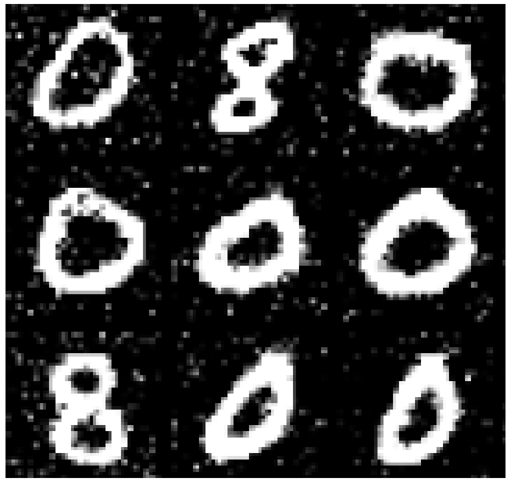
- p_data: MNIST一様サンプリング
- p_z: 256 次元正規分布 $N(0, I)$



Naïve にやってみると．．．

- ところが，Naïve に実装して学習してみると，様々な問題が起こる．

Missing mode



生成できる画像がやたら偏る．
= p_data で生成できない画像がある．

(この画像自体は arXiv: 1807.04015より)

Unwanted sample

P_data には無さそうな画像が
生成されてしまうことも．．．

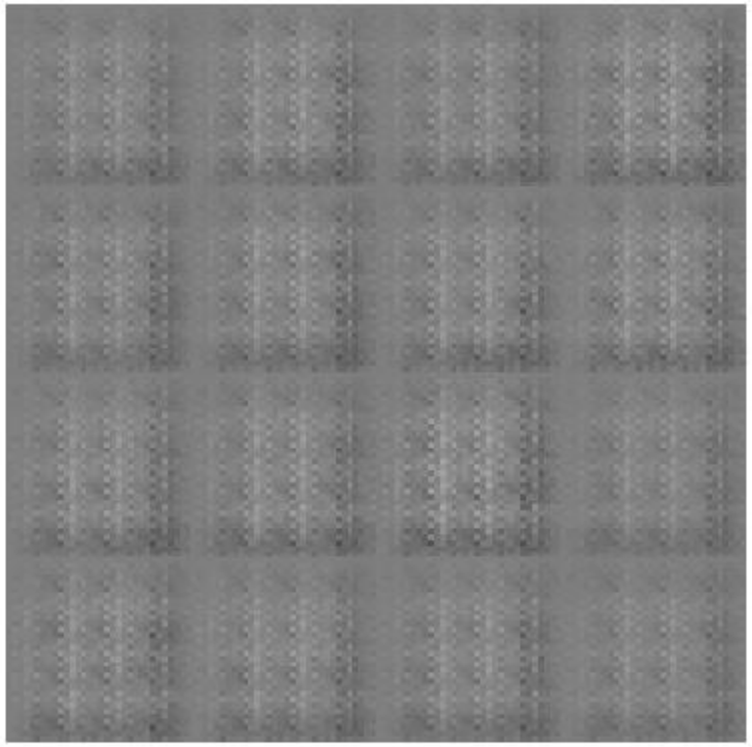
(この画像自体は arXiv: 1805.07674 より)

2. 安定性 – 高解像画像生成に向けて -

2.1 spectral normalization

高解像度画像生成

現在は，高解像度な画像も生成できるようになってきた．(arXiv: 1809.11096)



何があった？

GAN の training の安定化に向けた様々な技術の考案

安定化の技術例

例えば，arXiv: 1809.11096 では，以下の手法が使用されている．

- Hinge Loss
- Spectral normalization
- Self attention
- TTUR
- Large batch size
- Large channel
- Shared embedding
- Zero-centered gradient penalty
- Orthogonal regularization
- First singular value clamp
- Truncated Gaussian

安定化の技術例

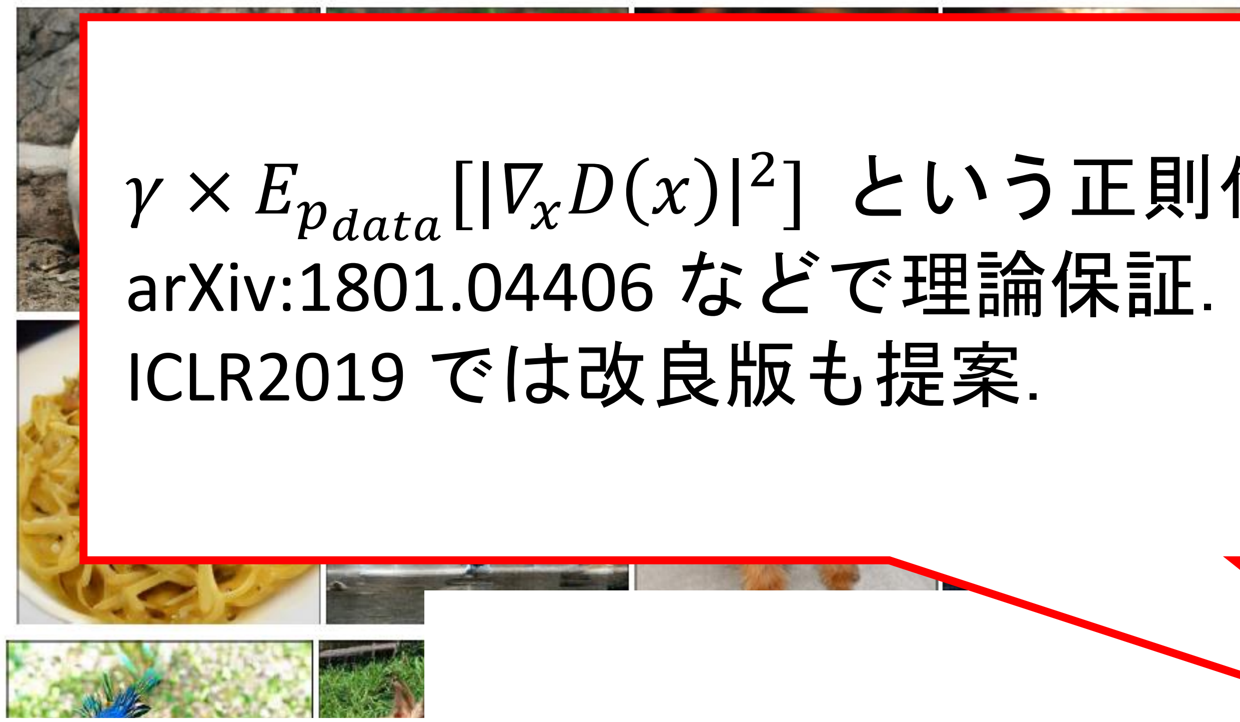
例えば，arXiv: 1809.11096 では，以下の手法が使用されている．

- Hinge Loss
- Spectral normalization
- Self attention
- **TTUR**
- Large batch size
- Large channel
- Shared embedding
- Zero-centered gradient penalty
- Orthogonal regularization
- First singular value clamp
- Truncated Gaussian

G と D で learning rate を変える．
D の方を大きくした方が収束性が良い．
arXiv: 1706.08500 で理論保証．

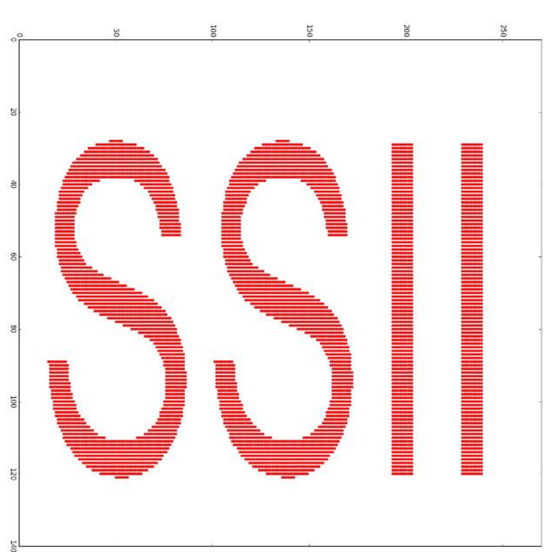
安定化の技術例

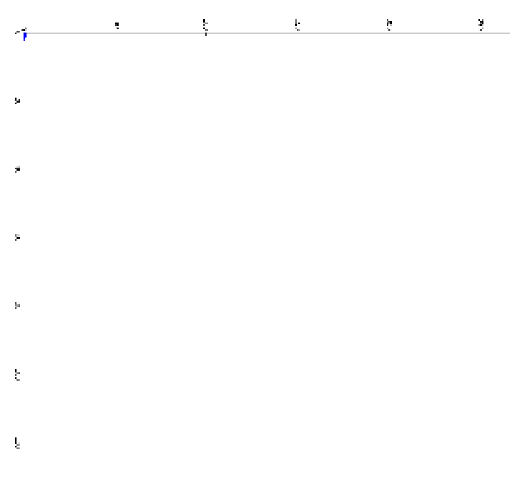
例えば，arXiv: 1809.11096 では，以下の手法が使用されている．



$\gamma \times E_{p_{data}}[|\nabla_x D(x)|^2]$ という正則化項.
arXiv:1801.04406 などで理論保証.
ICLR2019 では改良版も提案.

- Hinge Loss
- Spectral normalization
- Self attention
- TTUR
- Large batch size
- Large channel
- Shared embedding
- **Zero-centered gradient penalty**
- Orthogonal regularization
- First singular value clamp
- Truncated Gaussian





mainに紹介する手法

使用されている手法のうち，画像生成以外にも使えそうな SN を紹介.

- Hinge Loss
- **Spectral normalization**
- Self attention
- TTUR
- Large batch size
- Large channel
- Shared embedding
- Zero-centered gradient penalty
- Orthogonal regularization
- First singular value clamp
- Truncated Gaussian

2.1 spectral normalization

Motivation
- Original GAN の勾配消失
- 勾配消失のためによくやること
- 不安定性

対策

数値計算結果

参考文献
- arXiv:1701.04862
- arXiv:1802.05957
- arXiv:1805.08318
- arXiv:1809.11096

勾配消失

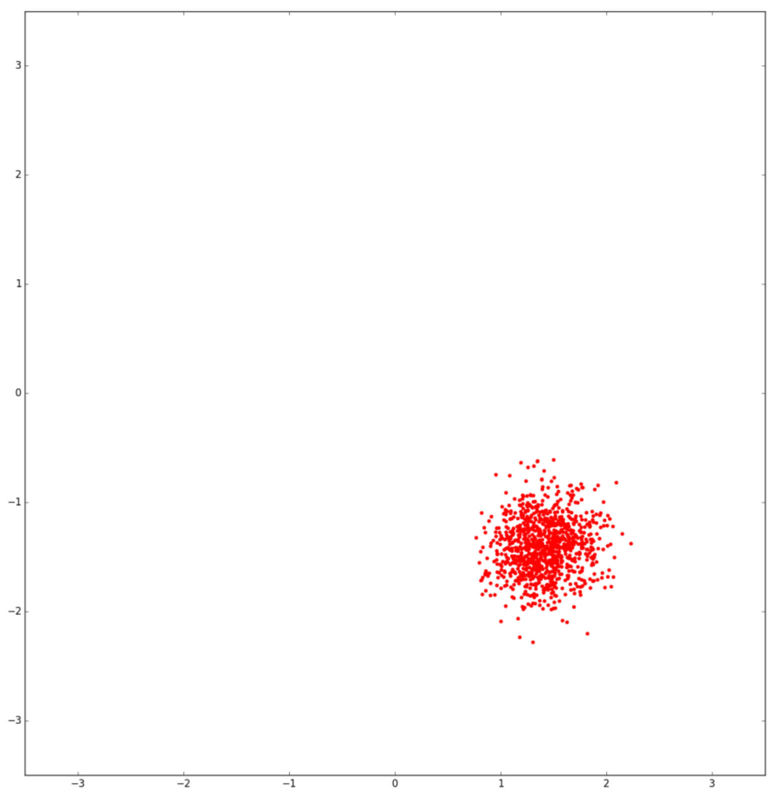
$D(x) = \sigma\big(f(x)\big)$ といつも通り sigmoid が最後にあるとする.

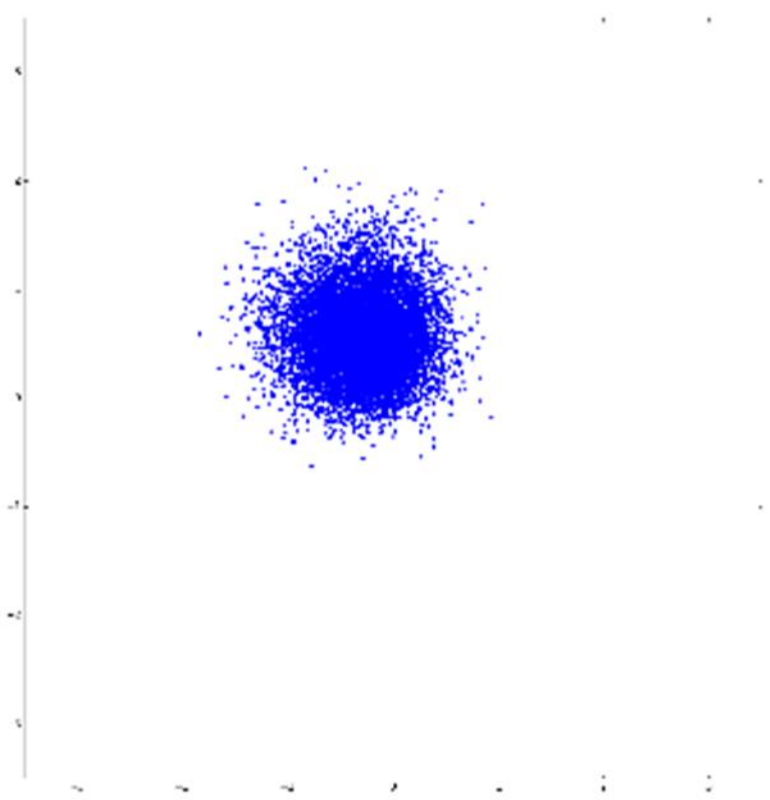
$$
\begin{aligned}
\frac{\partial \mathcal{L}_G}{\partial \phi} &= \frac{\partial}{\partial \phi} \log(1 - D(G_\phi(z))) \\
&= -\frac{\sigma'(f(G(z)))}{1-\sigma(f(G(z)))} f'(G(z)) \frac{\partial G_\phi(z)}{\partial \phi} \\
&= -\sigma(f(G(z))) f'(G(z)) \frac{\partial G_\phi(z)}{\partial \phi} \\
&= -\underline{D(G(z))} f'(G(z)) \frac{\partial G_\phi(z)}{\partial \phi}
\end{aligned}
$$

偽物を完全に偽物と言える状況だと勾配消失.

* 高画質なほど input の自由度が多く，discriminator の判断材料が増えるため，勾配消失が起きやすく学習が止まりやすいと言われている.

勾配消失の例





初期として「全てを false と答える discriminator」を用意した結果.

勾配消失のためによくやること

- Loss をちょっと違ったものに置き換えてしまう.

$$\mathcal{L}_D = \mathbb{E}_{x \sim p_{data}} [\log \{D(x)\}] + \mathbb{E}_{z \sim p_z} [\log \{1 - D(G(z))\}]$$

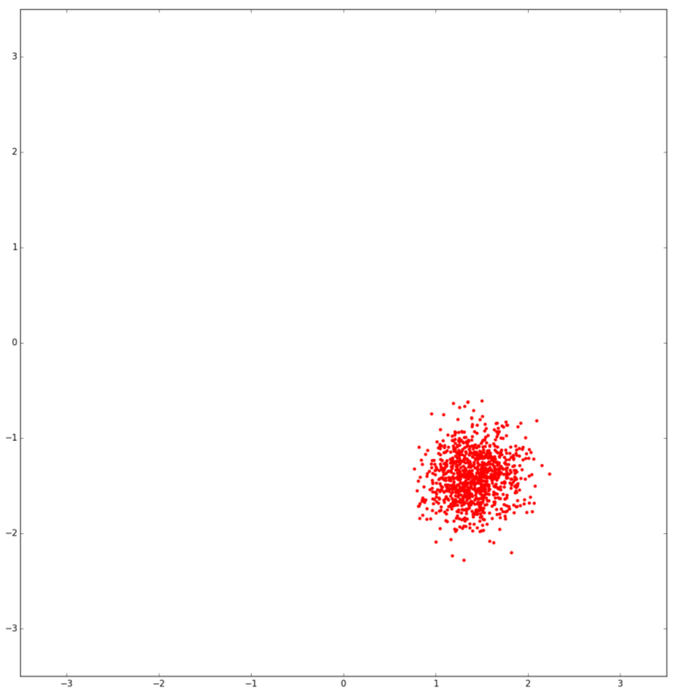
$$\mathcal{L}_G = \mathbb{E}_{z \sim p_z} [\log\{1 - D(G(z))\}]$$

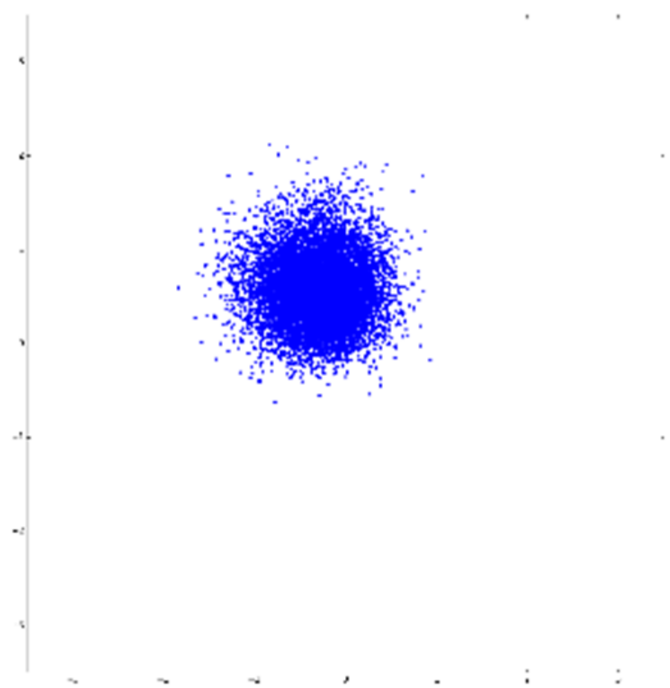
↓

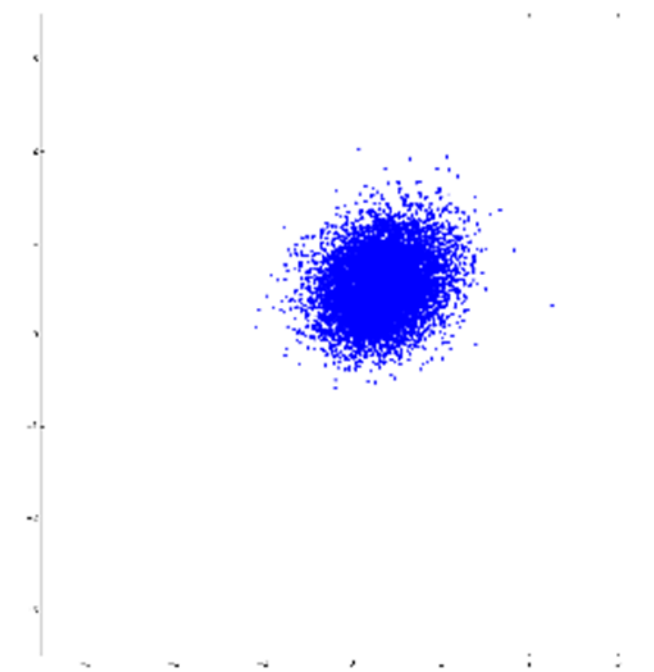
$$\mathcal{L}_D = \mathbb{E}_{x \sim p_{data}} [\log \{D(x)\}] + \mathbb{E}_{z \sim p_z} [\log \{1 - D(G(z))\}]$$

$$\mathcal{L}_G = \mathbb{E}_{z \sim p_z} [- \log \{D(G(z))\}]$$

置き換え結果







初期として「全てを false と答える discriminator」を用意した場合の結果比較.

*置き換えると，discriminatorがgeneratorが作ったものを True と言うと勾配消失.
が，discriminator は False と言いたがると信じれば，こちらの方が安定しそう.

置き換え後の不安定性

- 以下のような不安定性が生じうる.

$$
\begin{aligned}
\frac{\partial \mathcal{L}_G}{\partial \phi} &= \frac{\partial}{\partial \phi} \log(D(G_\phi(z))) \\
&= \underline{\left. \frac{\nabla_x D(x)}{D(x)} \right|_{x=G(z)}} \frac{\partial G_\phi(z)}{\partial \phi}
\end{aligned}
$$

Discriminator 強いと，分母 ≒ 0.
分子の大きさが普通くらいでも微分が大きくなりうる.

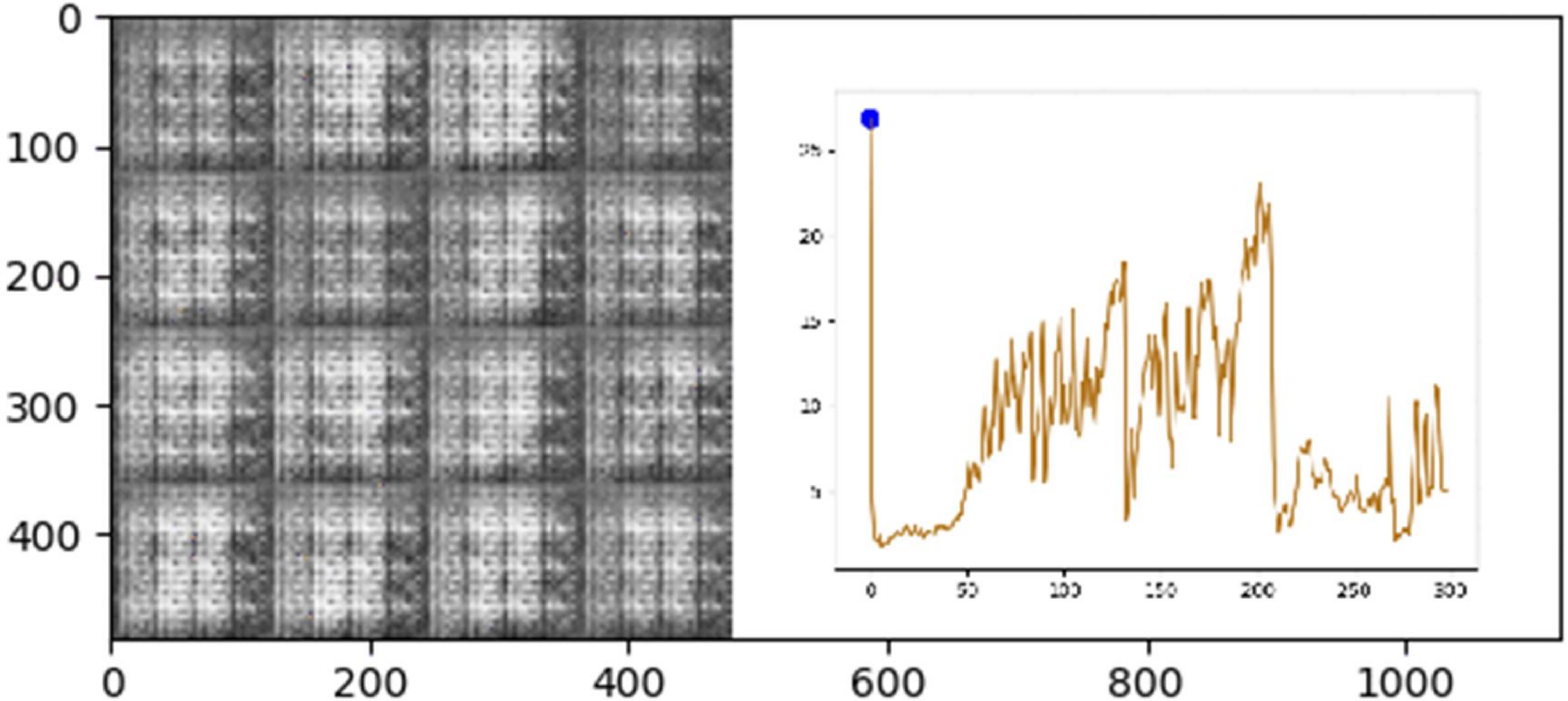
* arXiv:1701.04862 では，上記のように議論しているが，
arXiv:1802.05957 では，$D(x) = \sigma(f(x))$として，

$$
{}^{\nabla D}/_{D} = \Big(1 - \sigma\big(f(x)\big)\Big) \nabla f
$$

と計算されるが∇f が発散しうる，という形で議論している.

不安定性の例

- 以下のように不安定さと $|\nabla D/D|$ は関係があるケースも.



各 epoch での生成画像　　各 epoch の $|\nabla D/D|$の最大値

対策 : ∇Dが大きくならないように

- Spectral Normalization が注目されている (arXiv:1802.05957)

$$\frac{|NN(x+\epsilon)-NN(x)|}{|\epsilon|} \leq \prod_l SN(W^l)$$

NeuralNet の Lipschitz norm は
weight matrix の最大特異値の積で抑えられる.

上式を利用して，NeuralNet の変化量なり微分なりを抑える手法.
具体的には，NN の各層を以下の様に変更.

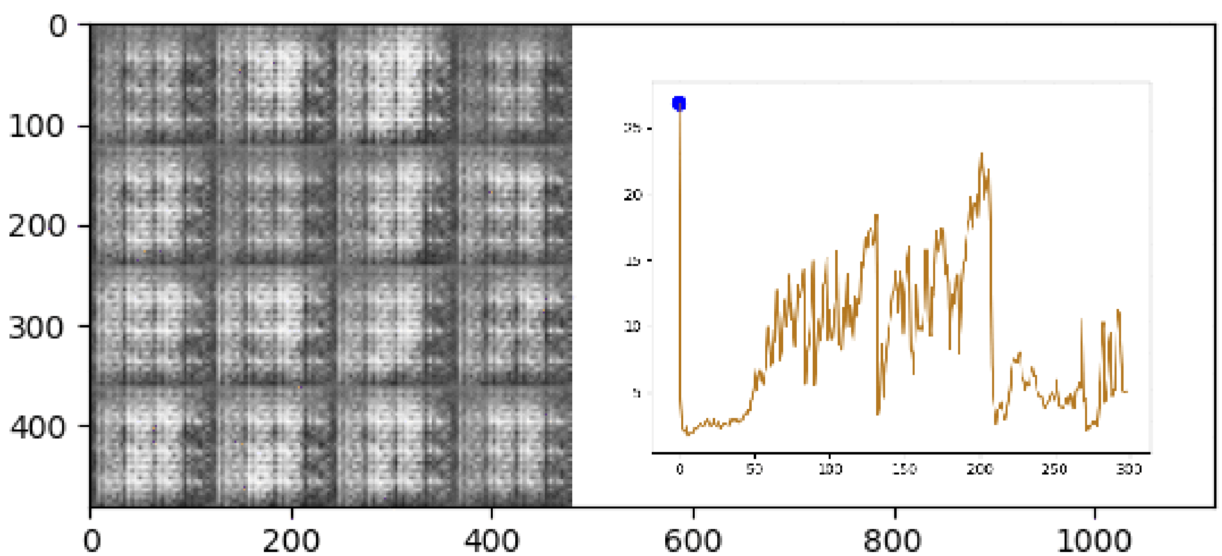
$$h^{l+1} = Activation(W^l h^l + b)$$

$SN(W^l)$ を見積もって以下を出力とする.

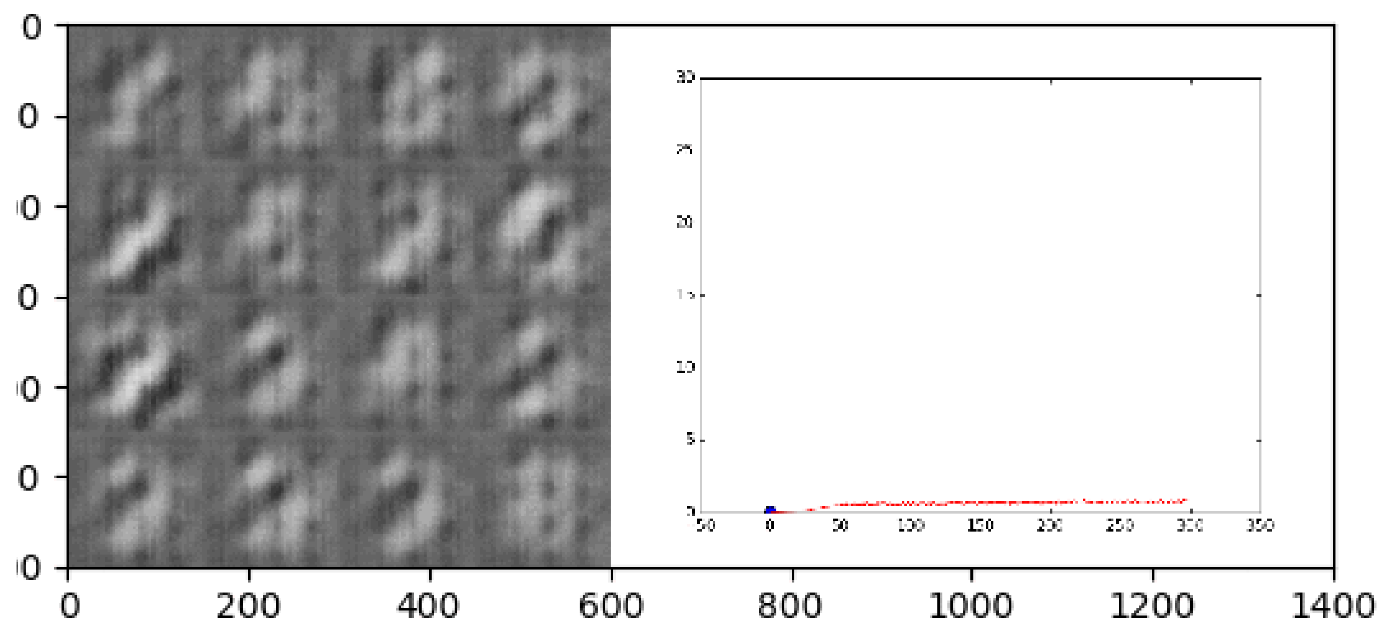
$$h^{l+1} = Activation(\frac{W^l}{SN(W^l)} h^l + b)$$

対策 : ∇Dが大きくならないように

- Discriminator に Spectral Normalization 入れた例.



無い時



ある時

SN まとめ

- Original のままだと勾配消失の問題があった.
- 勾配消失回避のトリックをすると例えば $\nabla D/D$ が原因で不安定.
- ∇D を小さく抑える手法として，spectral normalization がある.
  - arXiv:1805.08318 では，実験的には G にも，と指摘されている.
- なんか計算がうまく行かないなぁ，というときには是非お試しを！

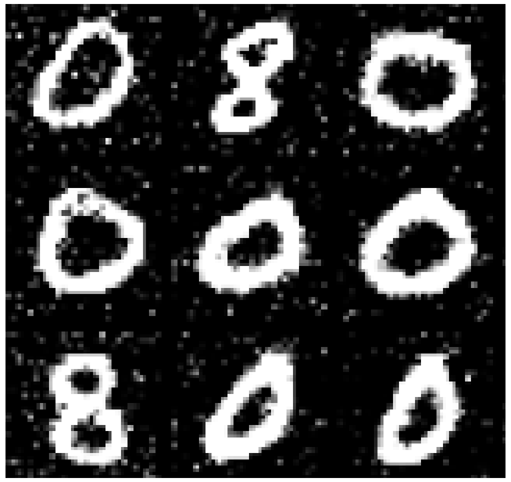
3. Missing mode と unwanted sample

P_z は single Gaussian で良いか？
対策
数値計算結果

参考文献
- arXiv: 1805.07674
- arXiv: 1902.02934
- arXiv: 1809.11096

GAN 学習すると以下が頻繁に起こる.

Missing mode



生成できる画像がやたら偏る.
= p_data で生成できない画像がある.

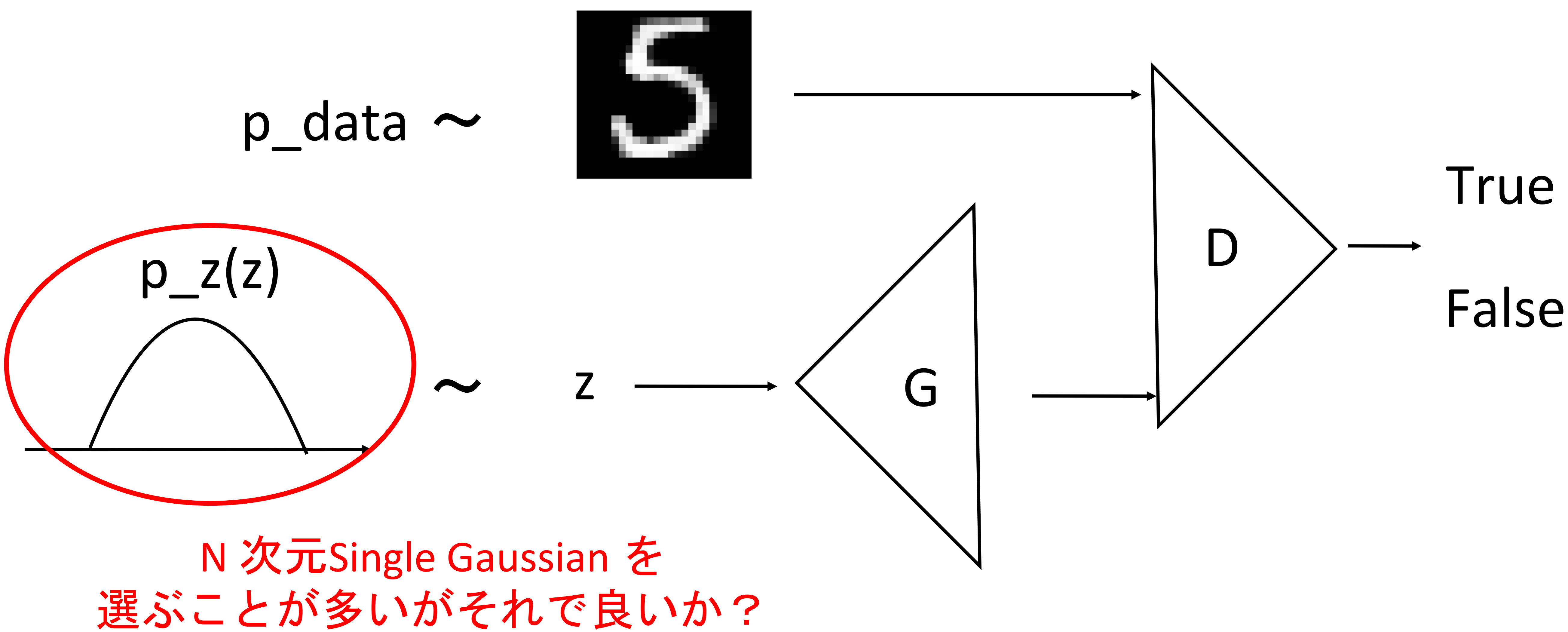
(この画像自体は arXiv: 1807.04015より)

Unwanted sample

P_data には無さそうな画像が
生成されてしまうことも．．．

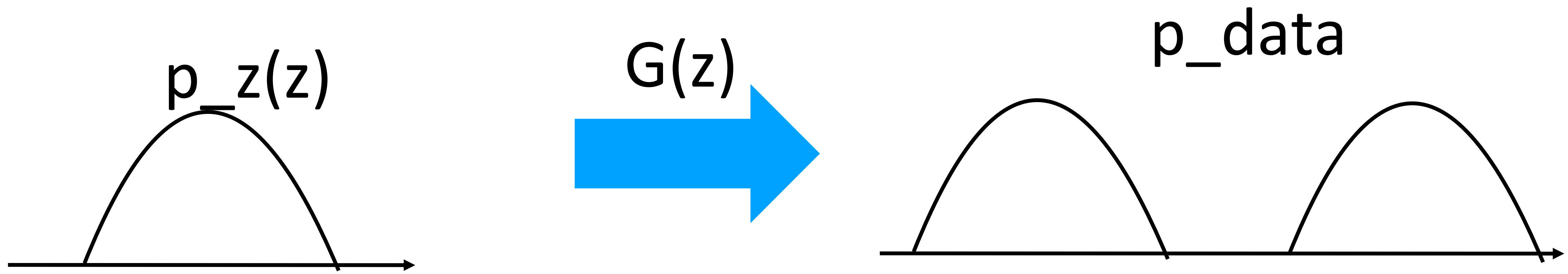
(この画像自体は arXiv: 1805.07674 より)

P_z の選び方が問題かも



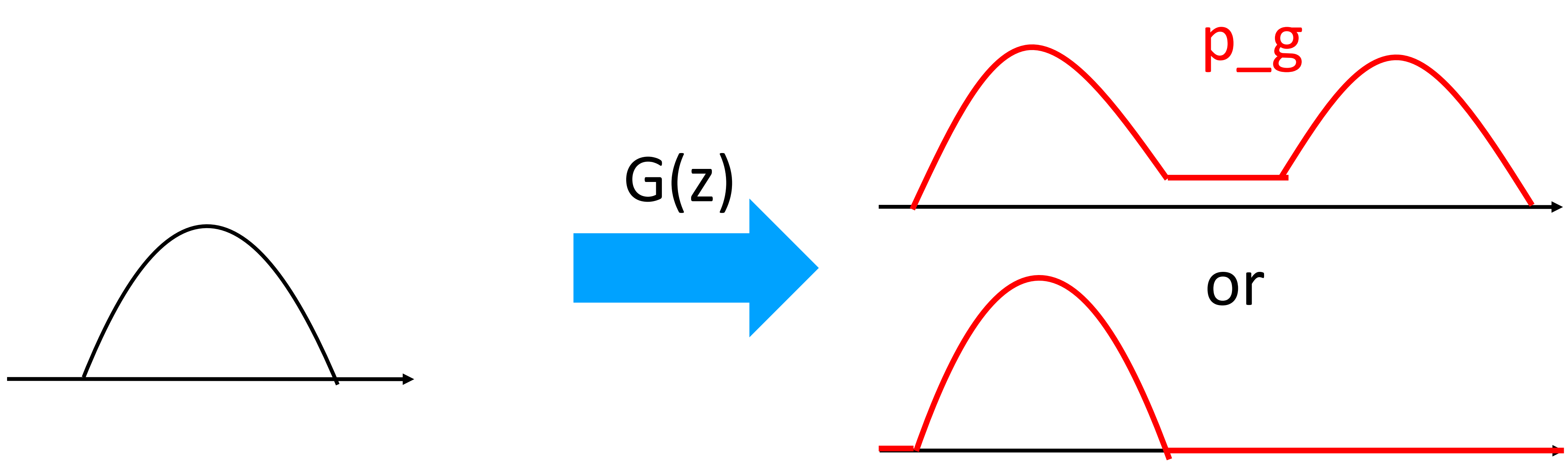
Single Gaussian を選んだ場合

- 複数の連結成分がある p_data を表現できるか？



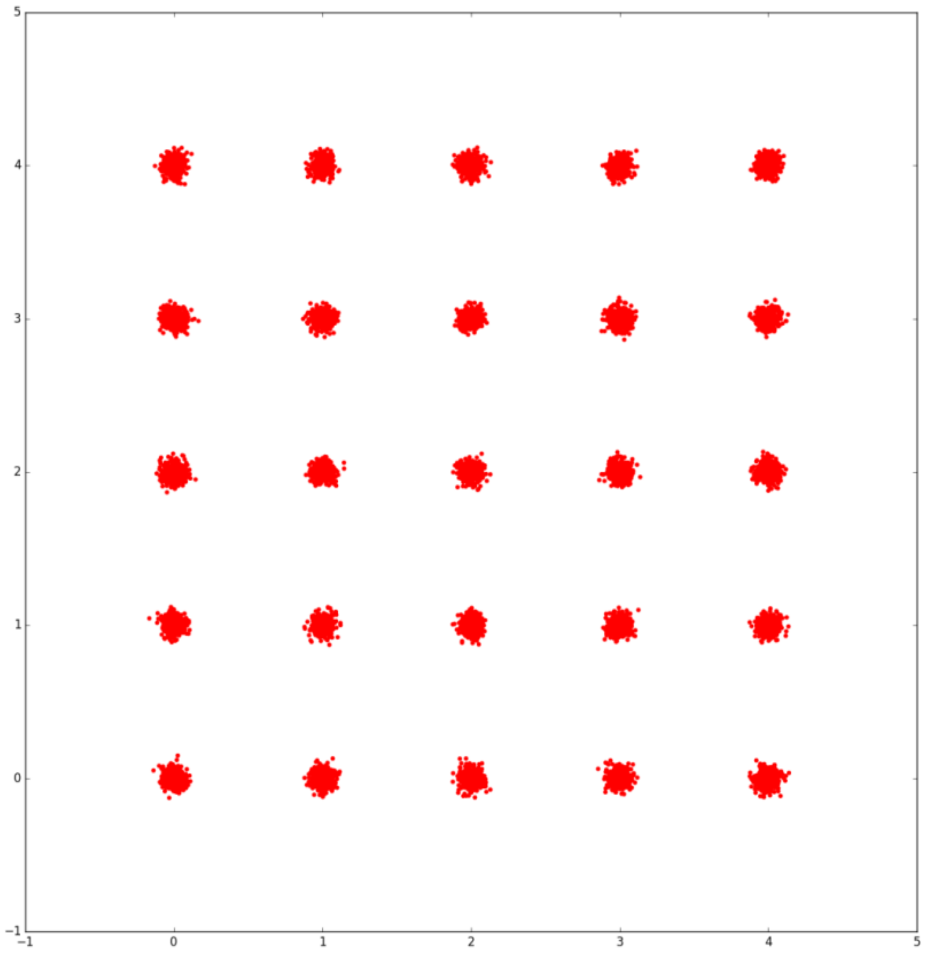
- Exact な一致はできない． NN で近似した G は連続関数だから．

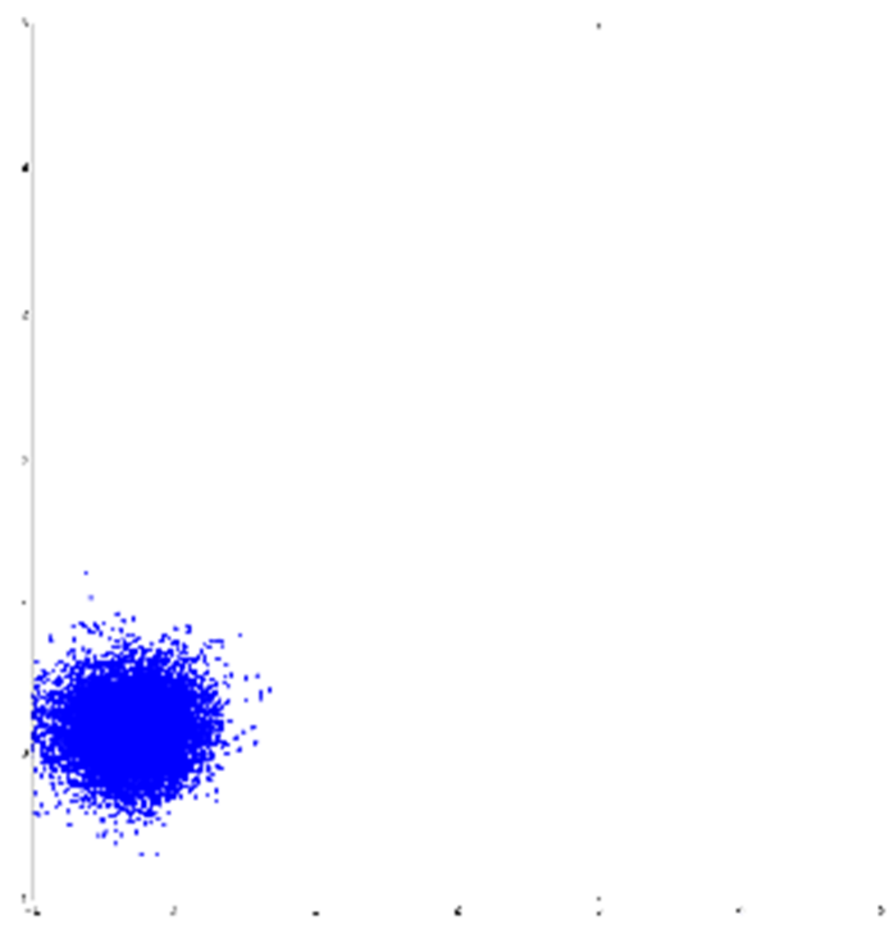
(arXiv: 1805.07674, arXiv: 1902.02934)



Single Gaussian を選んだ場合

- p_data: 5x5 の 2 次元混合 Gaussian
- p_z: 50 次元正規分布 $N(0, I)$





対策

- Neural net を使うとした場合，G の連続性はどうしようも無さそう.

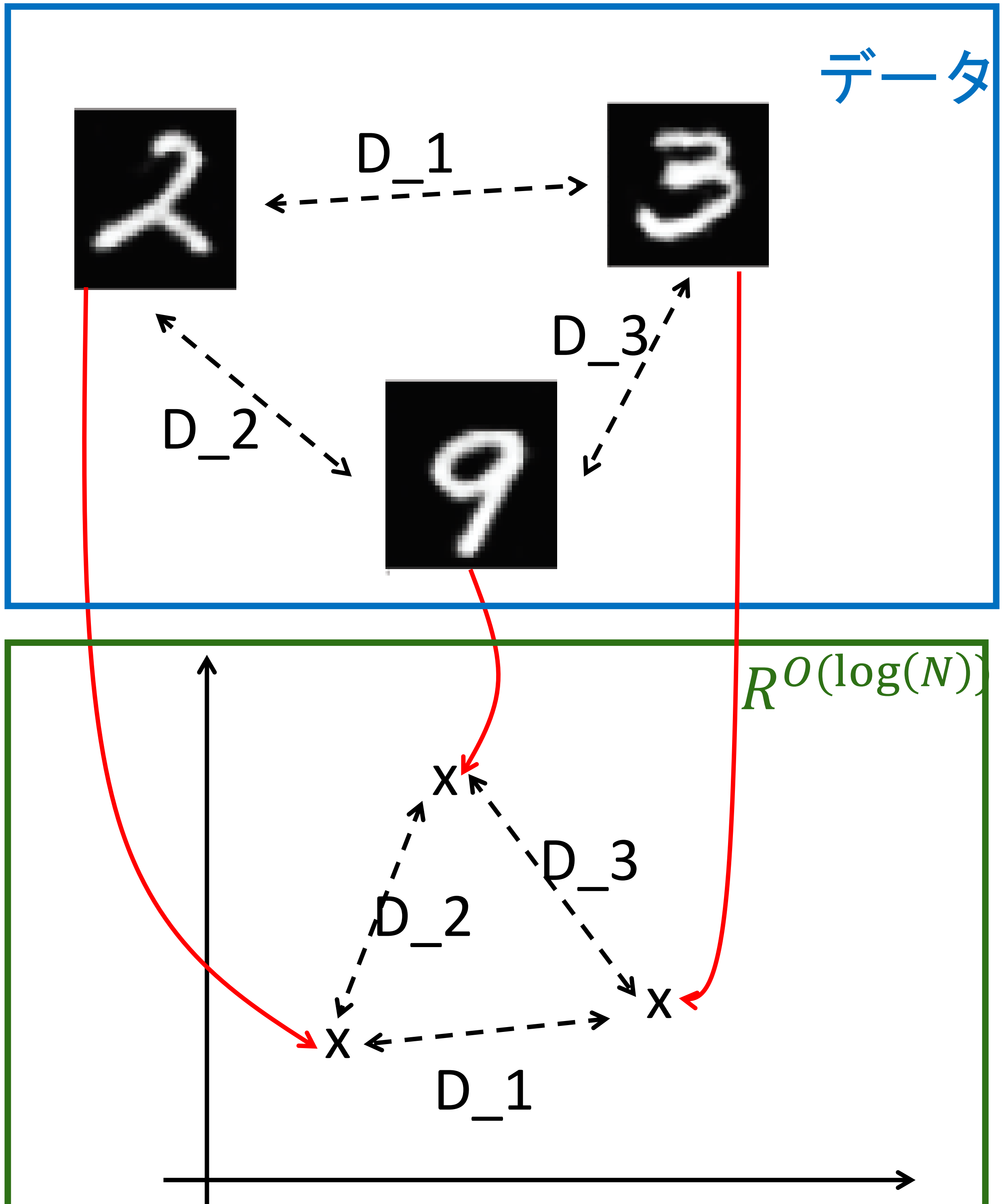
- p_z を p_data をもっと反映できるようなものを選ぶのはどうか？

<u>一例として，Bourgain Embedding を利用した方法を紹介</u>

(arXiv: 1805.07674)

Bourgain Embedding

- N 点のデータの距離を保ったまま O(log(N)) 次元に埋め込む algorithm.



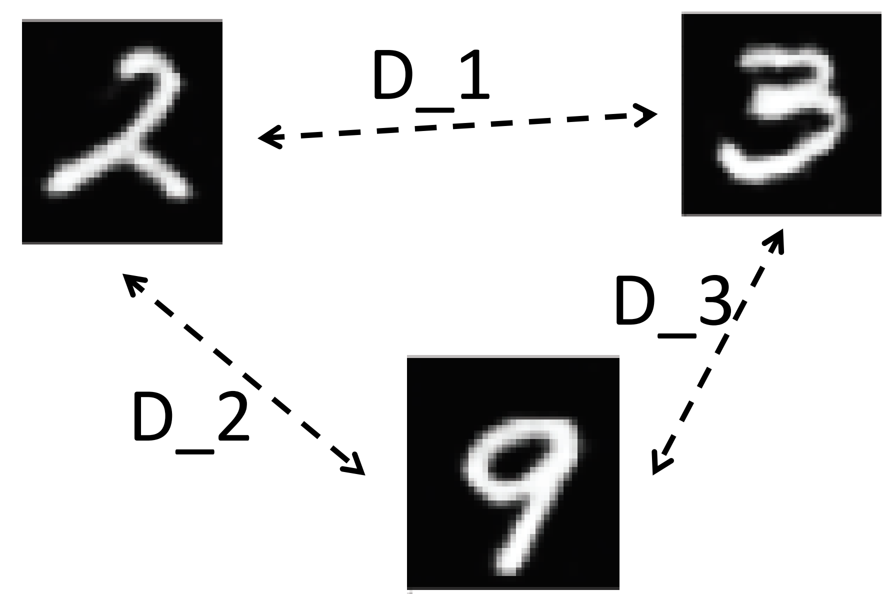
(詳細はarXiv: 1805.07674)

Algorithm 1 Improved Bourgain Embedding

Input: A finite metric space (Y, d).
Output: A mapping $f : Y \to \mathbb{R}^{O(\log |Y|)}$.
//Bourgain Embedding:
Initialization: $m \leftarrow |Y|$, $t \leftarrow O(\log m)$, and $\forall i \in [\lceil \log m \rceil]$, $j \in [t]$, $S_{i,j} \leftarrow \emptyset$.
for $i = 1 \to \lceil \log m \rceil$ **do**
  for $j = 1 \to t$ **do**
    For each $x \in Y$, independently choose x in $S_{i,j}$, i.e. $S_{i,j} = S_{i,j} \cup \{x\}$ with probability 2^{-i}.
  end for
end for
Initialize $g : Y \to \mathbb{R}^{\lceil \log m \rceil \cdot t}$.
for $x \in Y$ **do**
  $\forall i \in [\lceil \log m \rceil]$, $j \in [t]$, set the $((i-1) \cdot t + j)$-th coordinate of $g(x)$ as $\mathrm{d}(x, S_{i,j})$.
end for
//Johnson-Lindenstrauss Dimentionality Reduction:
Let $d = O(\log m)$, and let $G \in \mathbb{R}^{d \times (\lceil \log m \rceil \cdot t)}$ be a random matrix with entries drawn from i.i.d. $\mathcal{N}(0, 1)$.
Let $h : \mathbb{R}^{\lceil \log m \rceil \cdot t} \to \mathbb{R}^d$ satisfy $\forall x \in \mathbb{R}^{\lceil \log m \rceil \cdot t}$, $h(x) \leftarrow G \cdot x$.
//Rescaling:
Let $\beta = \min_{x,y \in Y : x \neq y} \frac{\|h(g(x)) - h(g(y))\|_2}{\mathrm{d}(x,y)}$.
Initialize $f : Y \to \mathbb{R}^d$. For $x \in Y$, set $f(x) \leftarrow h(g(x))/\beta$.
Return f.

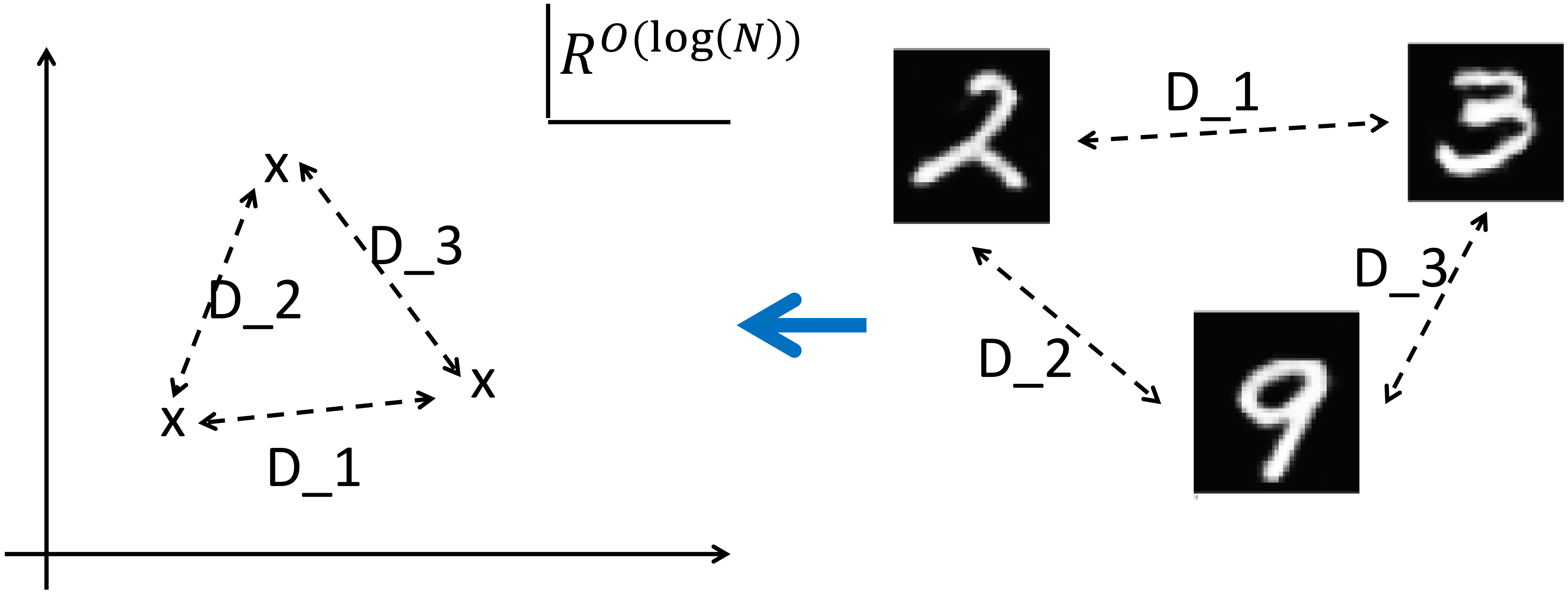
Bourgain Embedding を利用した p_z

p_data ～



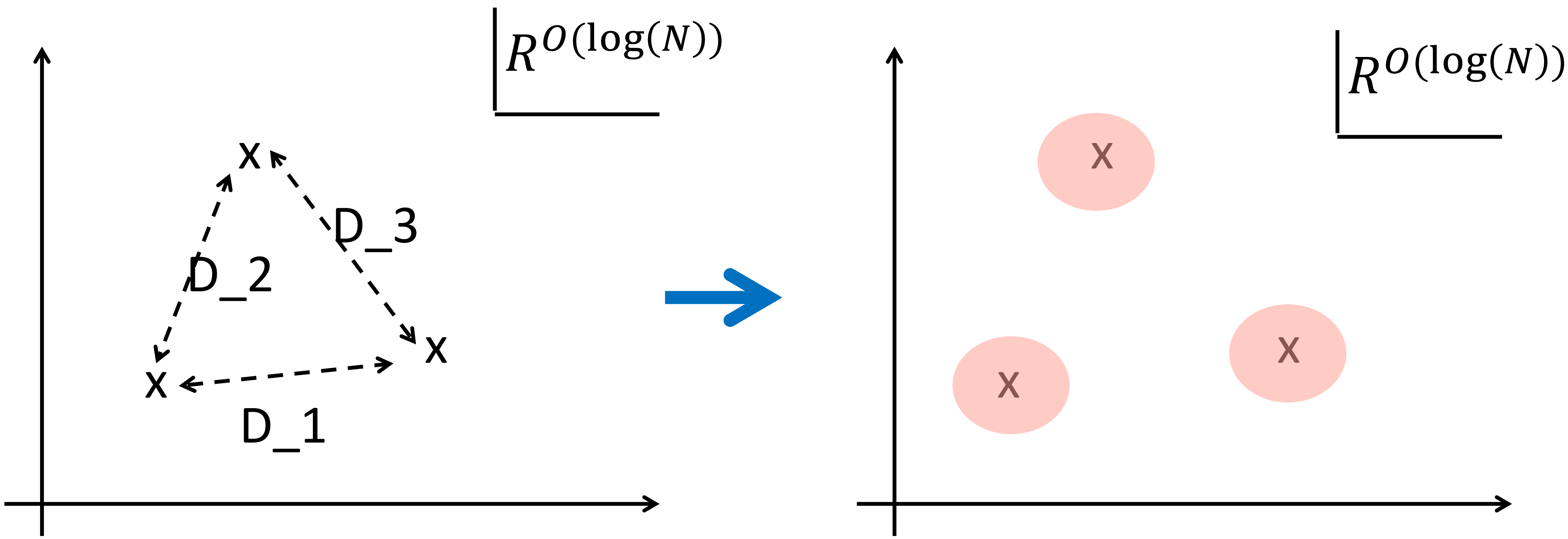
P_data (例えば M 個の画像の一様サンプル) から
N 点 draw する.

Bourgain Embedding を利用した p_z



Bourgain Embedding で O(log(N)) 次元に埋め込む

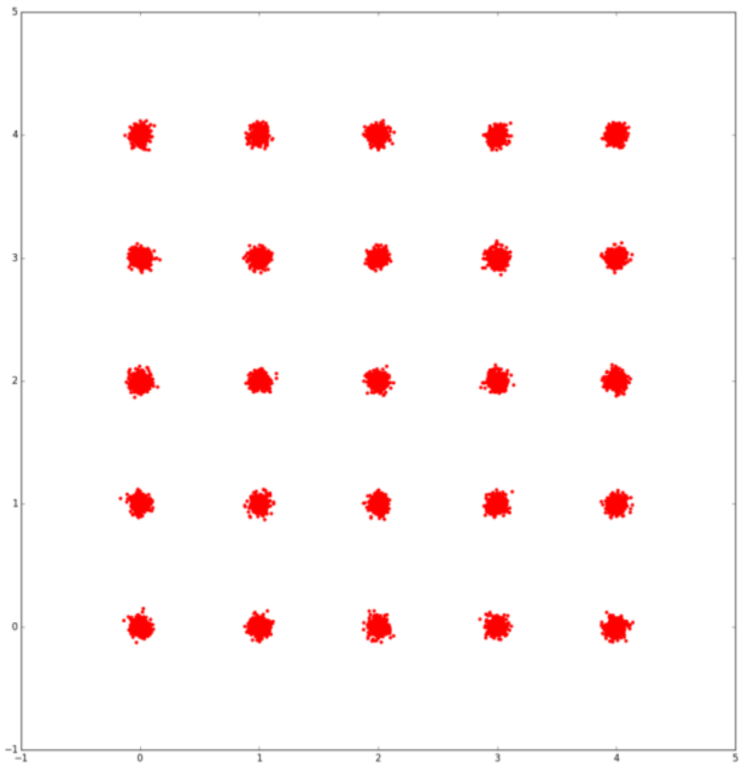
Bourgain Embedding を利用した p_z

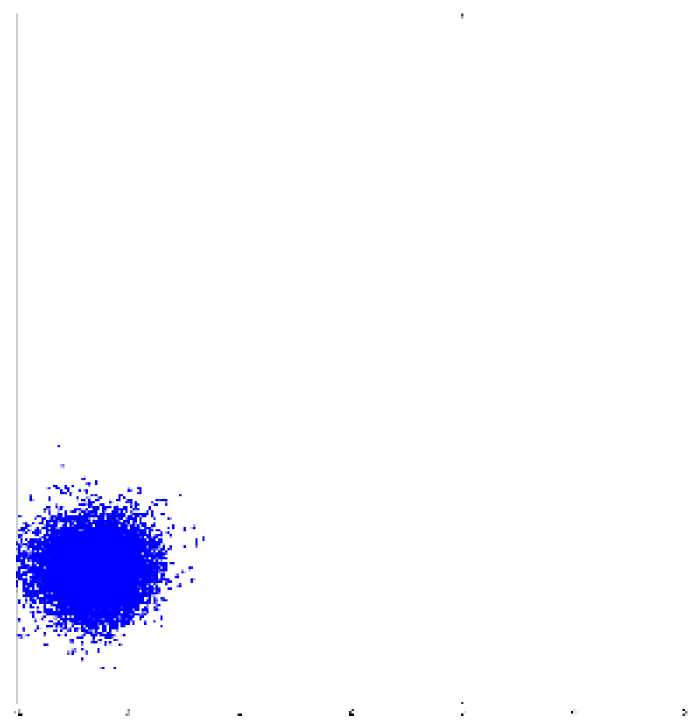


それぞれの点を中心とする，混合 Gaussian を p_z とする．
（都合，O(log(N)) 次元の N 混合 Gaussian を p_z としている．）

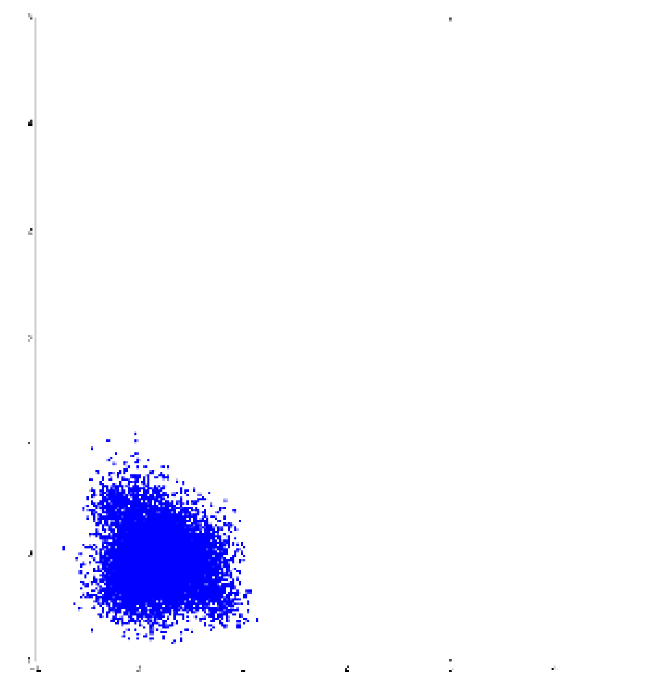
計算結果

- p_data: 5x5 の 2 次元混合 Gaussian





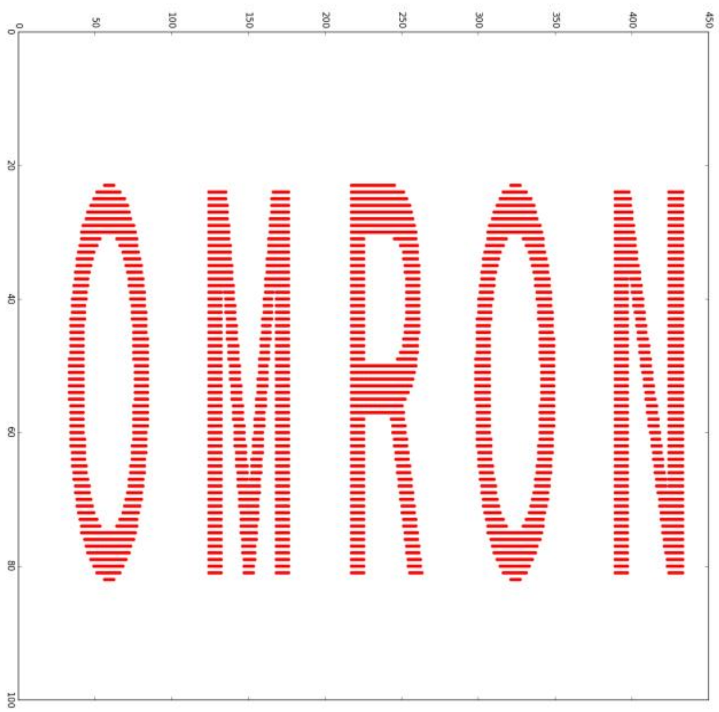
p_z = 50 次元正規分布$N(0, I)$

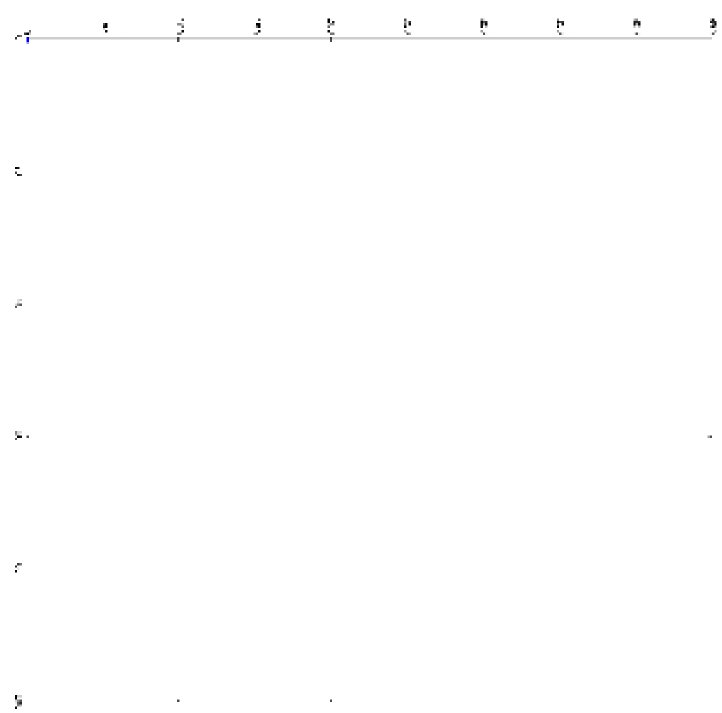


p_data から 1000 点 draw して作った p_z
(50 次元 1000 混合 Gaussian)

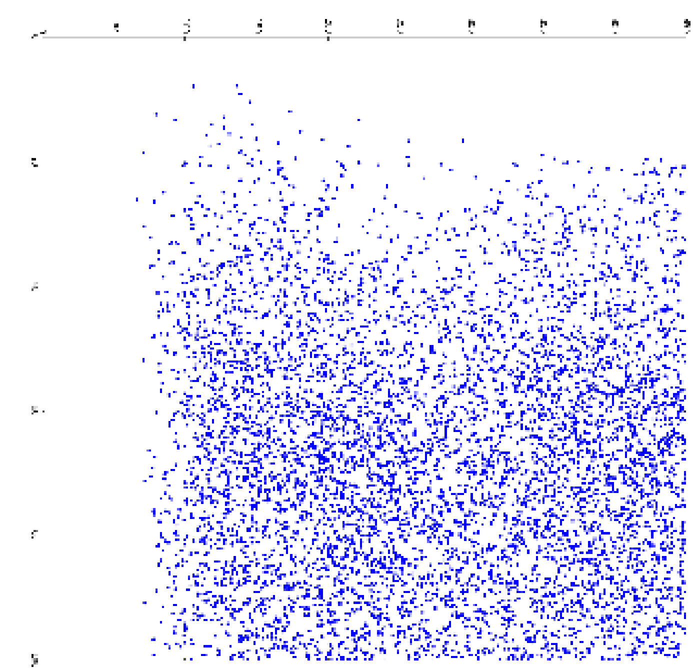
計算結果(一様 random sampling)

- p_data: 一様 OMRON 分布（約 6,000 点くらい）



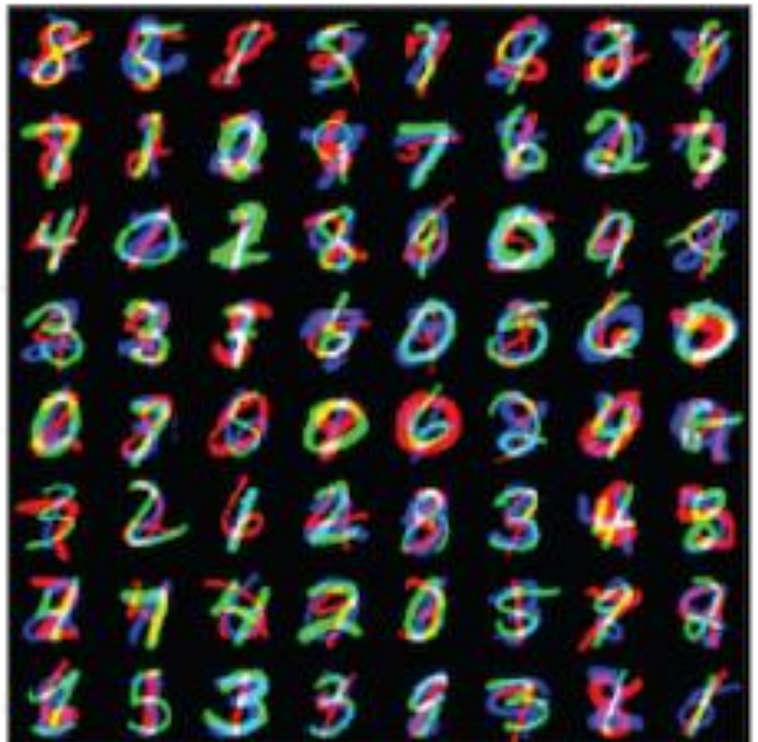


p_z = 55 次元正規分布$N(0, I)$

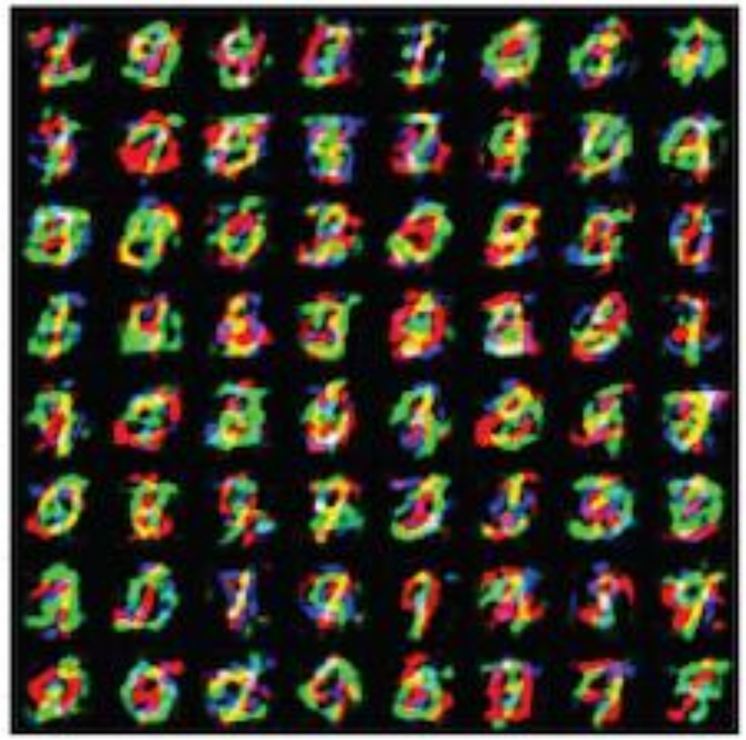


p_data から 2000 点 draw して作った p_z
(55 次元 2000 混合 Gaussian)

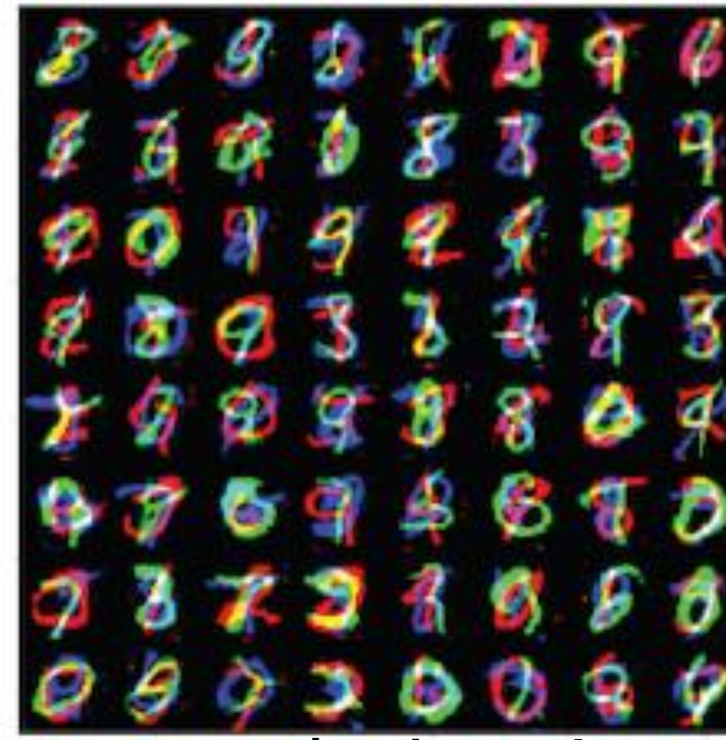
計算結果（stacked MNISTの例）



P_data



P_z = single Gaussian



P_z = 提案手法

	D is 1/4 size of G		D is 1/2 size of G		D is same size as G	
	# class covered (max 1000)	KL	# class covered (max 1000)	KL	# class covered (max 1000)	KL
DCGAN	92.2	5.02	367.7	4.87	912.3	0.65
BourGAN	715.2	1.84	936.1	0.61	1000.0	0.08

青：大きい程 missing mode 少ない.

赤：小さい程 missing mode 少ない.

(arXiv: 1805.07674)

この章のまとめ

- G が連続なので，p_z が single Gaussian だと辛いかも．．
- 色々試しても計算がうまく行かないときには，p_z の変更も視野に！
    - Motivation は違うが，先の高解像度画像生成論文でも「p_z は何が良い？」という疑問から truncated Gaussian を使用している.
- ただ，問題は山積み．．．
  - 最良の p_z の作成方法は謎.
  - P_z ≠ Single Gaussian 以外で安定に training できるかは謎.

4. Domain Adaptation

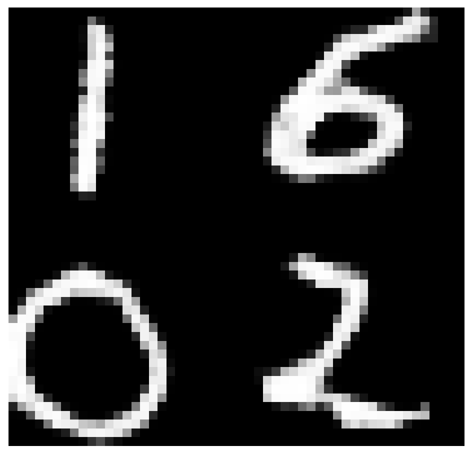
motivation
手法例の紹介と数値計算結果
最近の研究

参考文献

- arXiv: 1702.05464
- S.Xie, et al., ICML2018
- arXiv: 1711.03213
- arXiv: 1810.00045
- arXiv: 1812.04798
- arXiv: 1903.04064

motivation

MNIST で test accuracy 99% の Neural Net でも USPS で 70% 程度.



MNIST

USPS

手元の環境でデータを集めて label 付けし学習したが,
運用環境ではちょっと違う domain のデータで，精度が出ない可能性．．．

Source image (GTA5)

Target image (CityScapes)

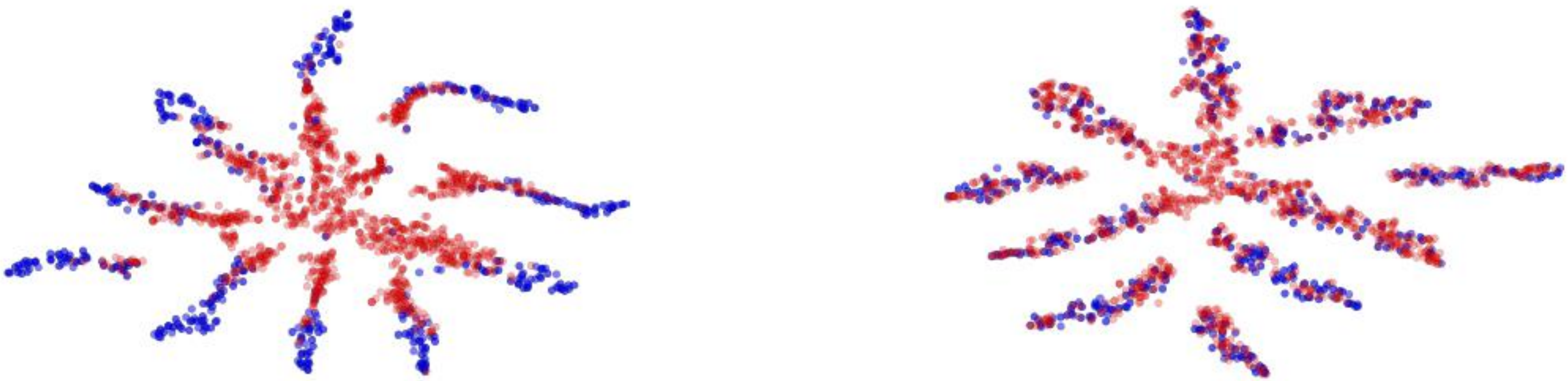
(arXiv:1711.03213より引用)

motivation

- 時間的な問題などで，運用環境のデータに label 付けできなかったら？
- このときは，以下だけで，運用環境で精度の良いものを作る必要あり.
  - 手元(source domain) : 画像とラベル情報
  - 運用環境(target domain): 画像

よくやる手法

Domain 間で「特徴量ベクトル分布」が重なるように学習を行う.



(a) Non-adapted　　(b) Adapted

赤：source データ群を source 特徴量抽出器にかけた結果

青：target データ群を target 特徴量抽出器にかけた結果

（arXiv: 1505.07818）

<u>分布が重なると，source 側で作る識別機が target 側でも有効.</u>

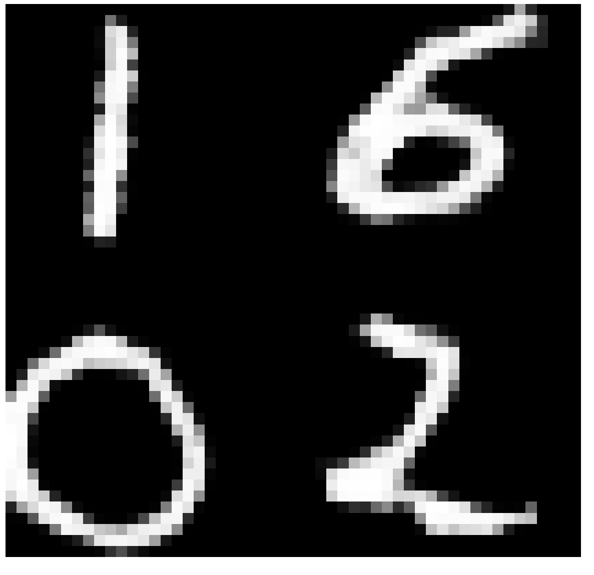
今回は以下の例

紹介手法

色々と流儀があるが，ADDA(arXiv: 1702.05464) を具体例で紹介

具体例で使うデータ

Source domain



MNIST 画像 + label

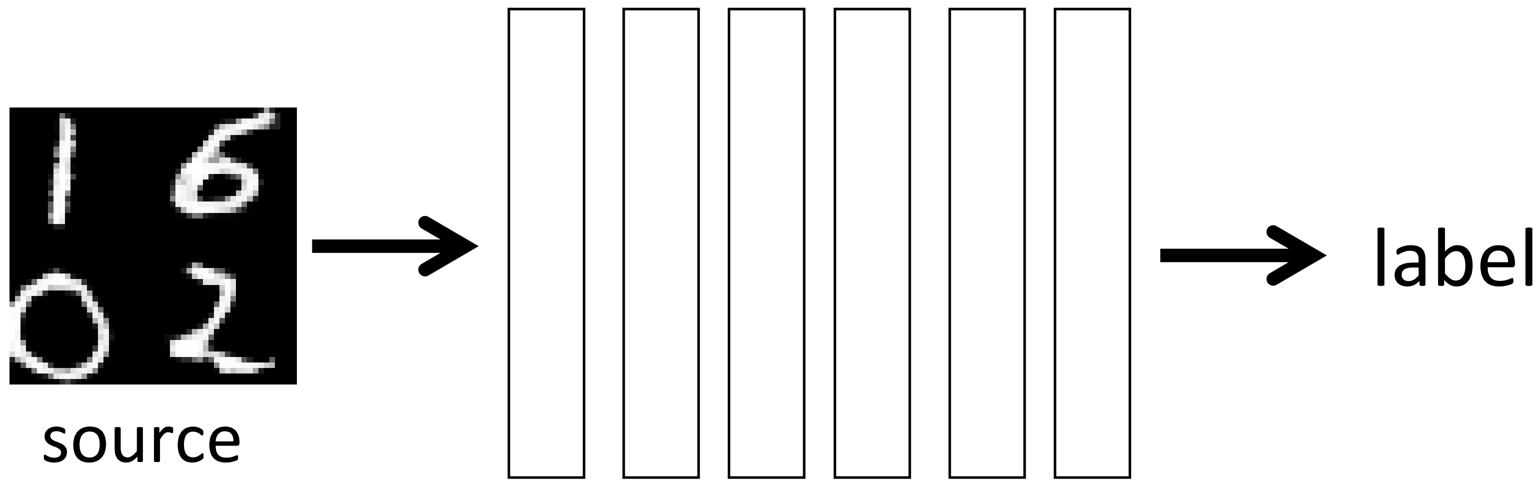
Target domain

USPS 画像

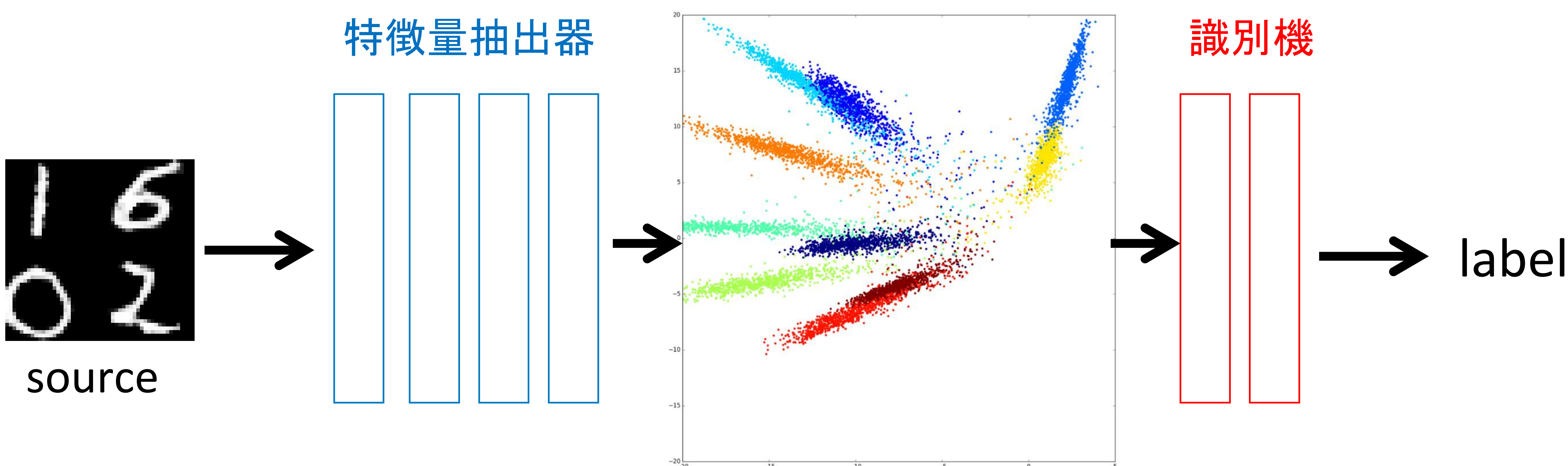
* 精度や可視化のため，testデータのみ USPS の label を使用.

Source 特徴量抽出器と識別機

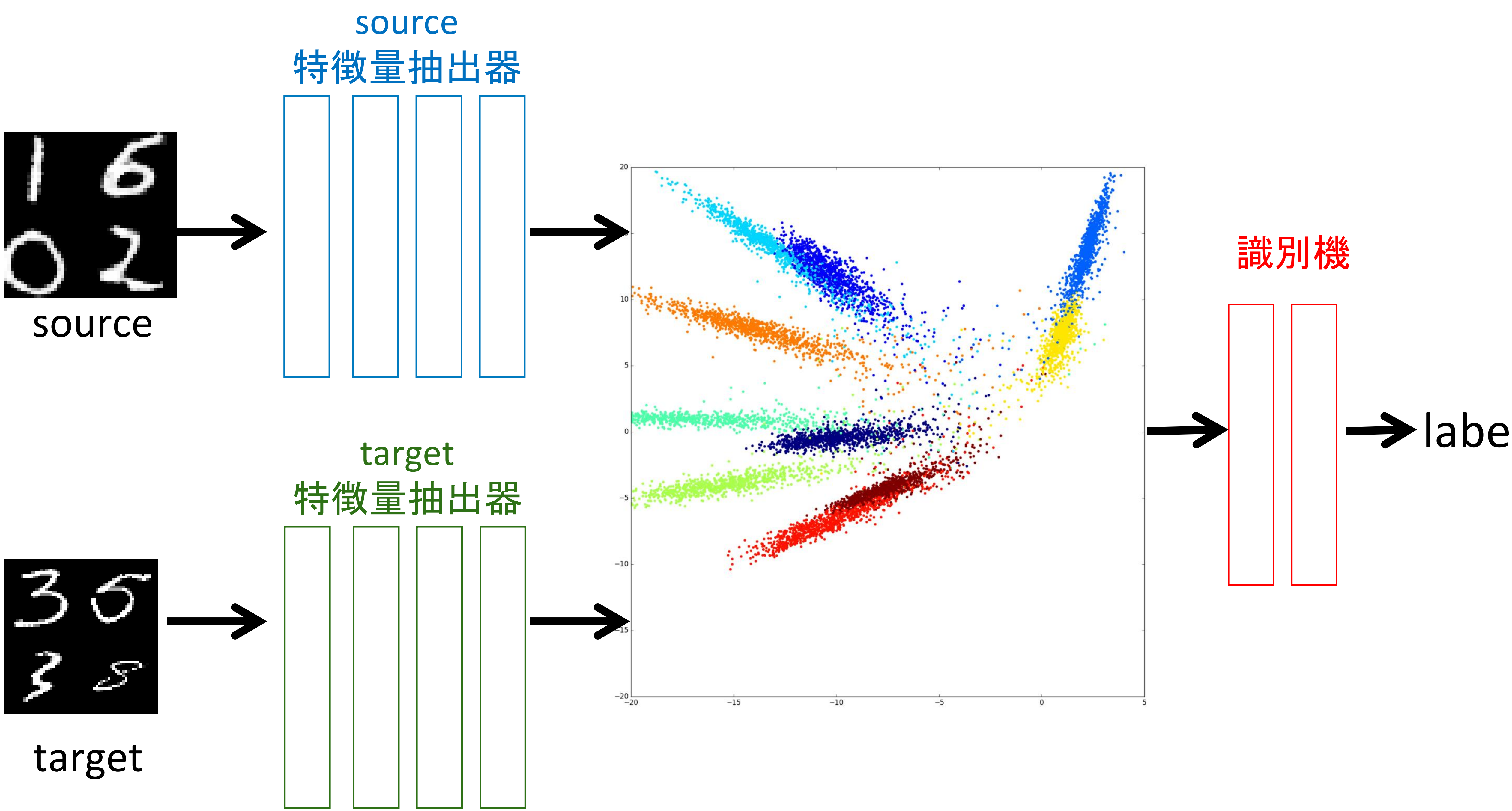
1. source を使って，普通に deep neural network を学習する．



2. 二つに割って，source 特徴量抽出器と識別機とする．



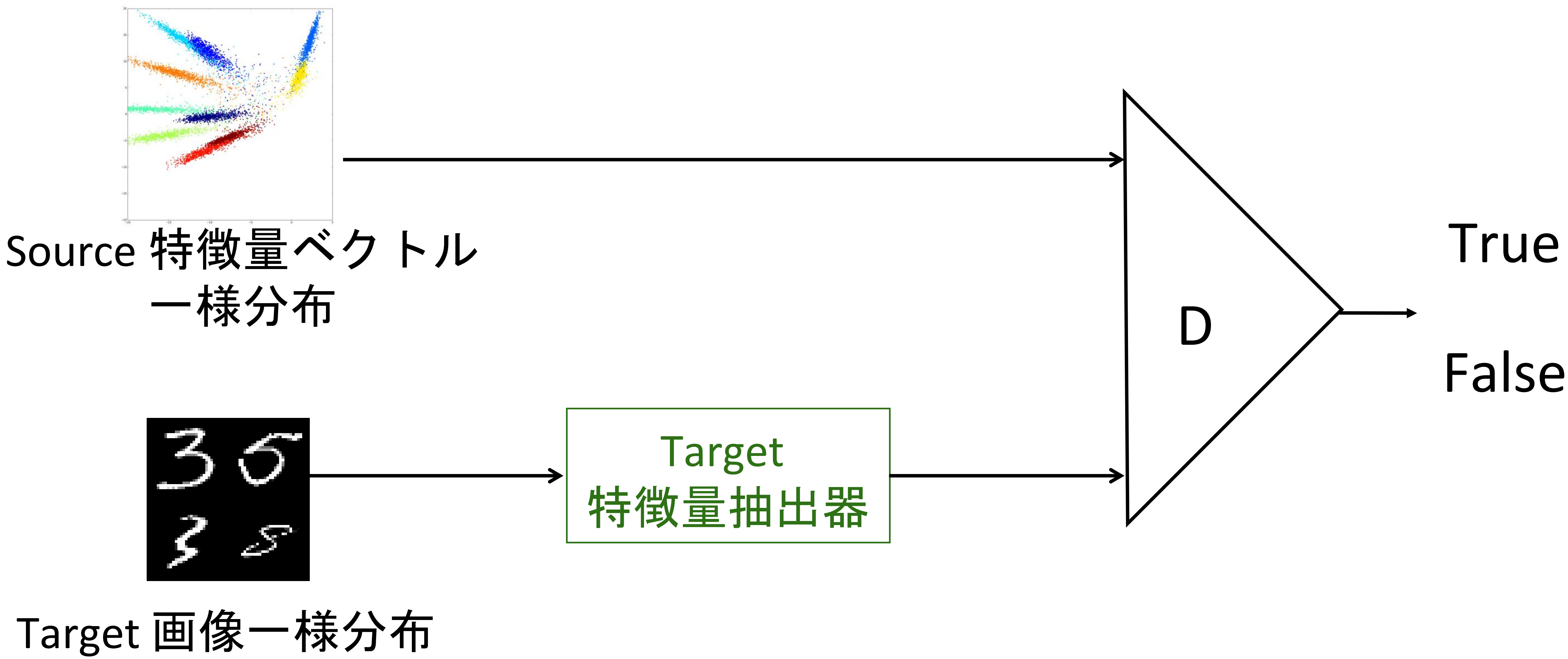
分布を合わせる



- Source 特徴量抽出器と識別機は固定
- Target 特徴量抽出器は分布を合わせるように学習する.

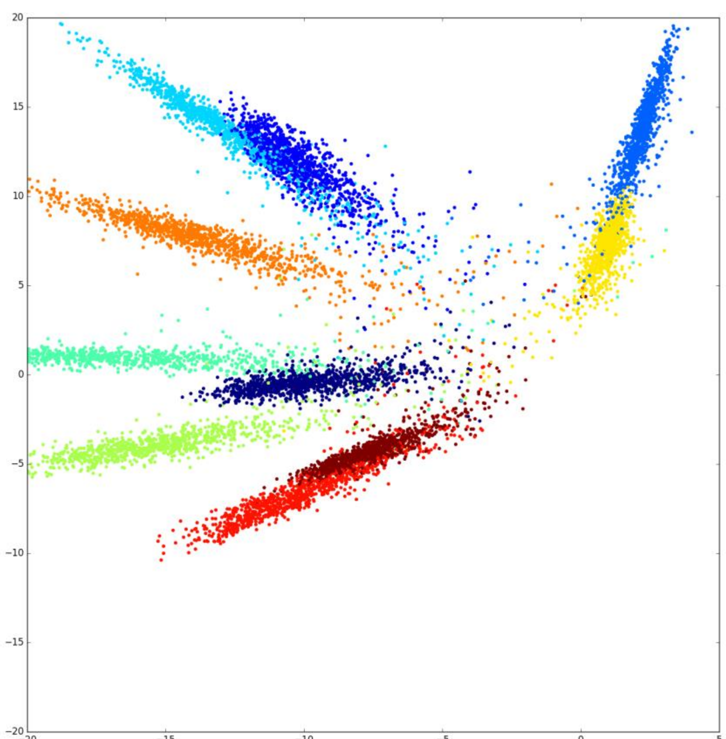
分布を合わせる

- P_data: Source 特徴量ベクトルの一様ランダムサンプリング
- P_z: TargetFig の一様ランダムサンプリング

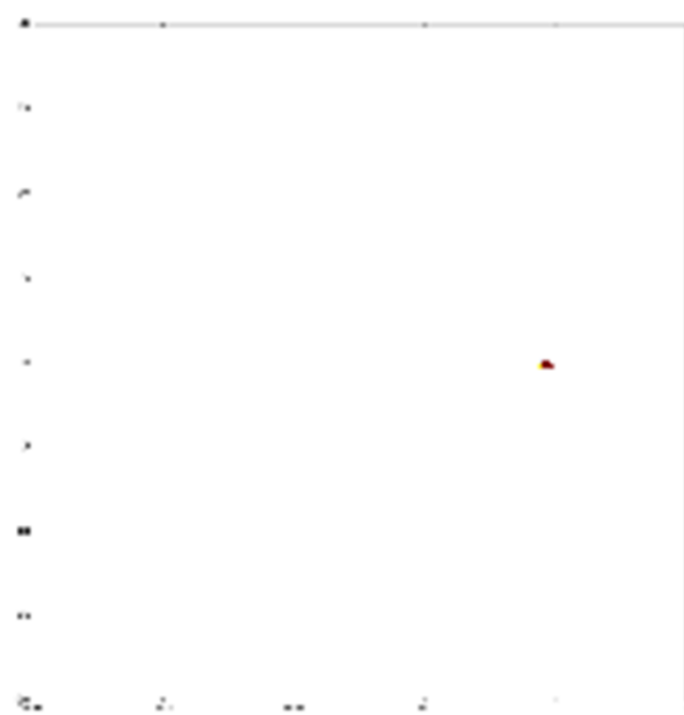


Naïve にやってみると．．．

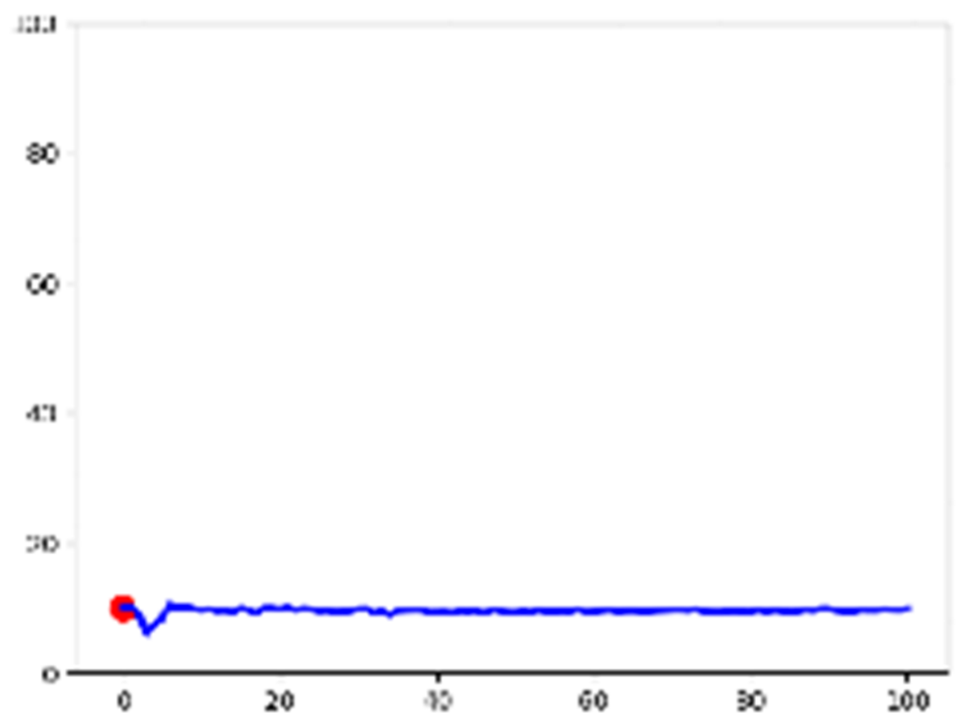
- P_data: Source 特徴量ベクトルの一様ランダムサンプリング
- P_z: TargetFig の一様ランダムサンプリング



P_data



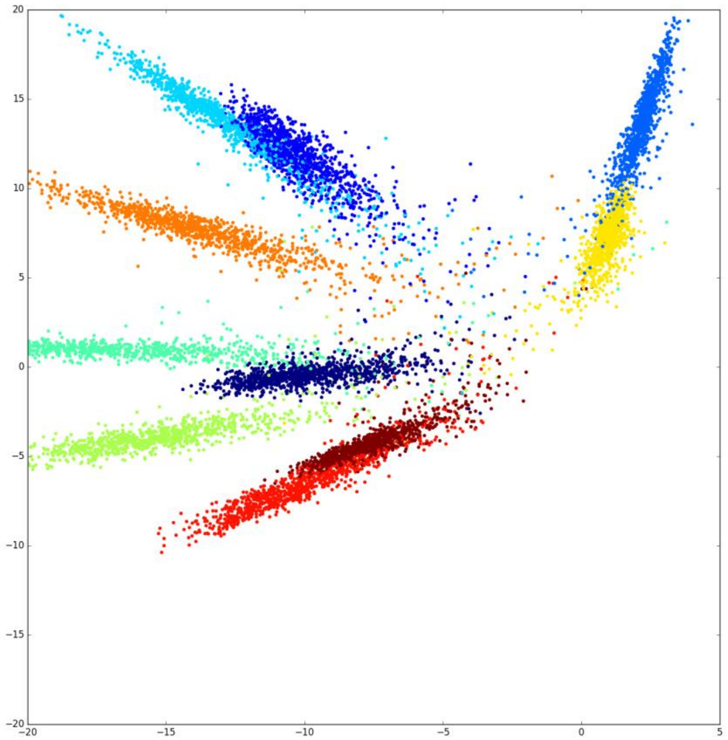
Target test 画像の
特徴量ベクトル



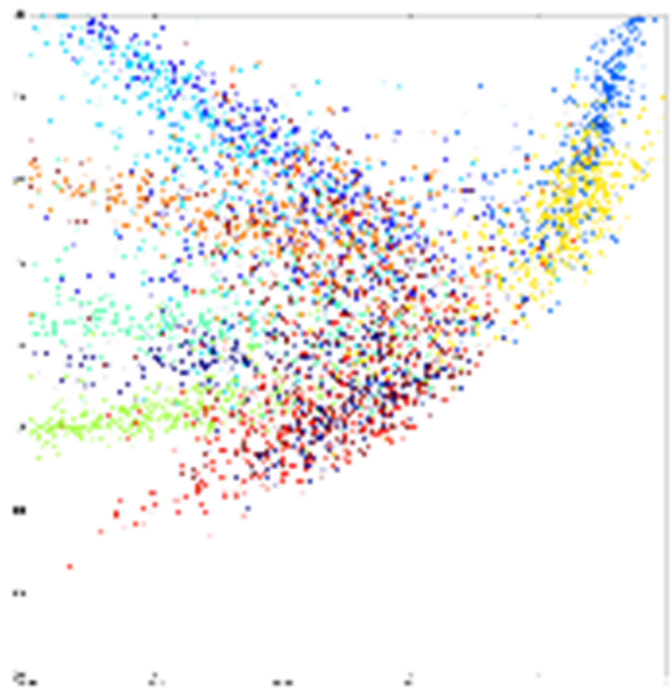
Target test accuracy

分布を合わせる（改）

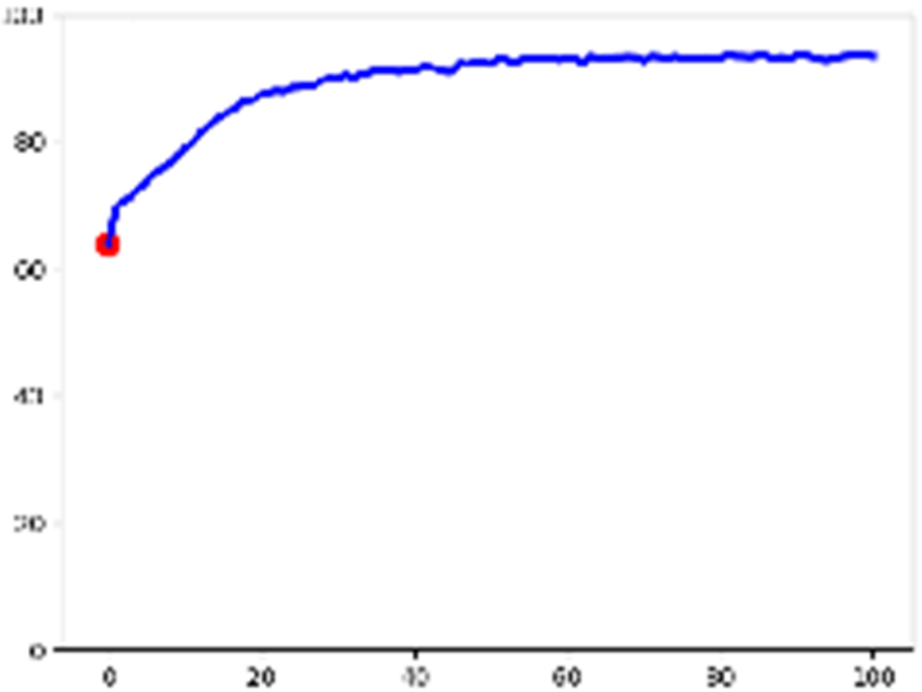
- ADDA(arXiv: 1702.05464) でなされている工夫を紹介.
- Target 特徴量抽出器 を source のもののコピーから学習を開始する.



P_data



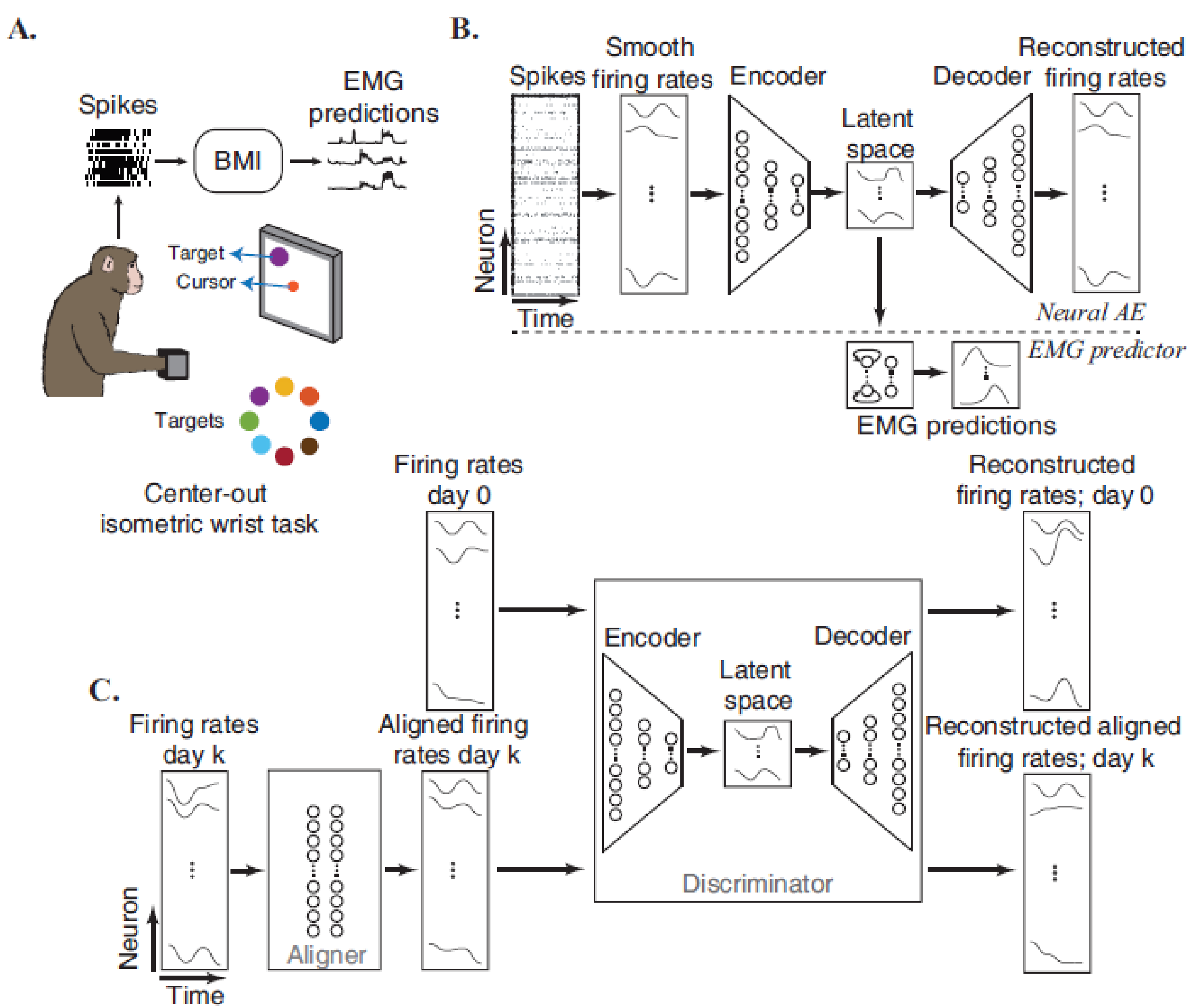
Target test 画像の
特徴量ベクトル

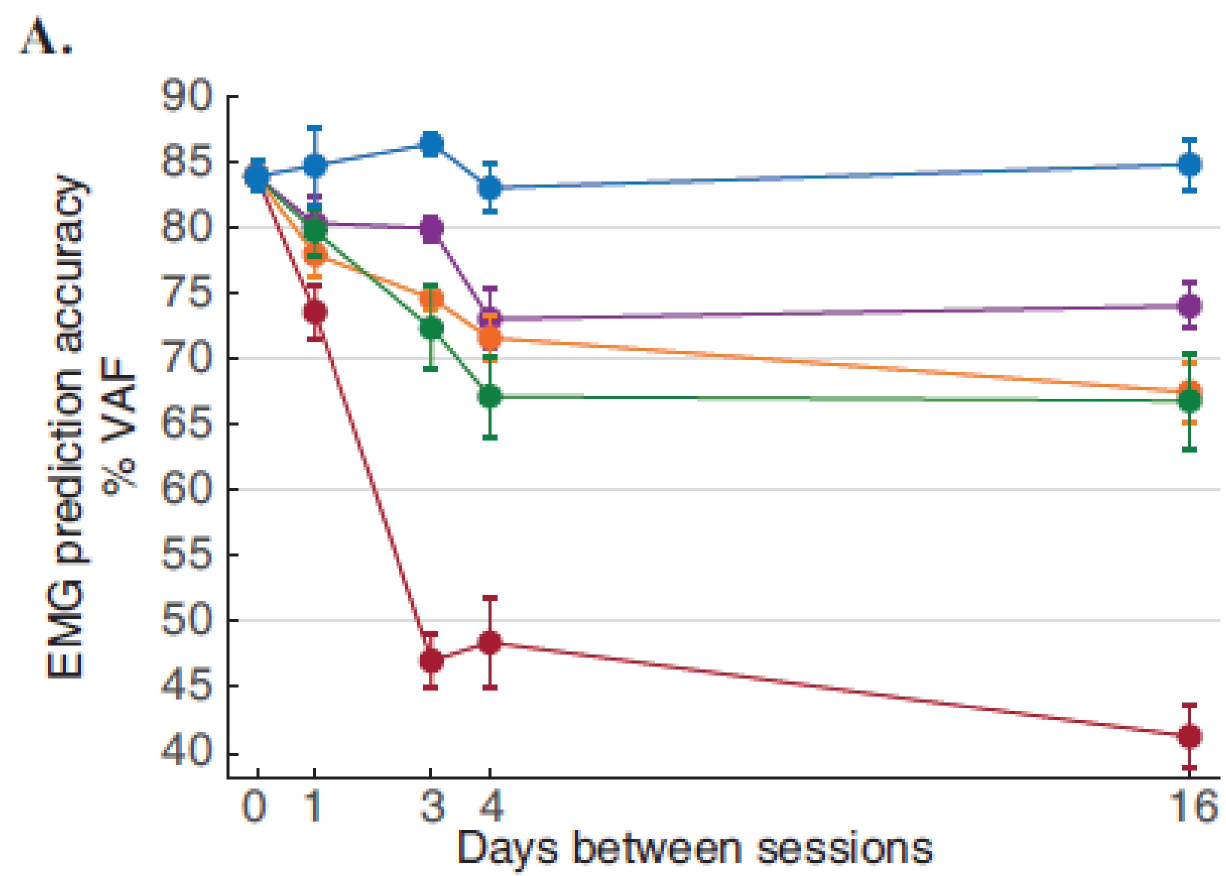


Target test accuracy

最近の研究

- 実データよりの話もちらほら
- 例えば，arXiv: 1810.00045では，脳波->動作の予測タスクに利用.
  - 脳波の日毎の違いを domain adaptation で吸収する話.

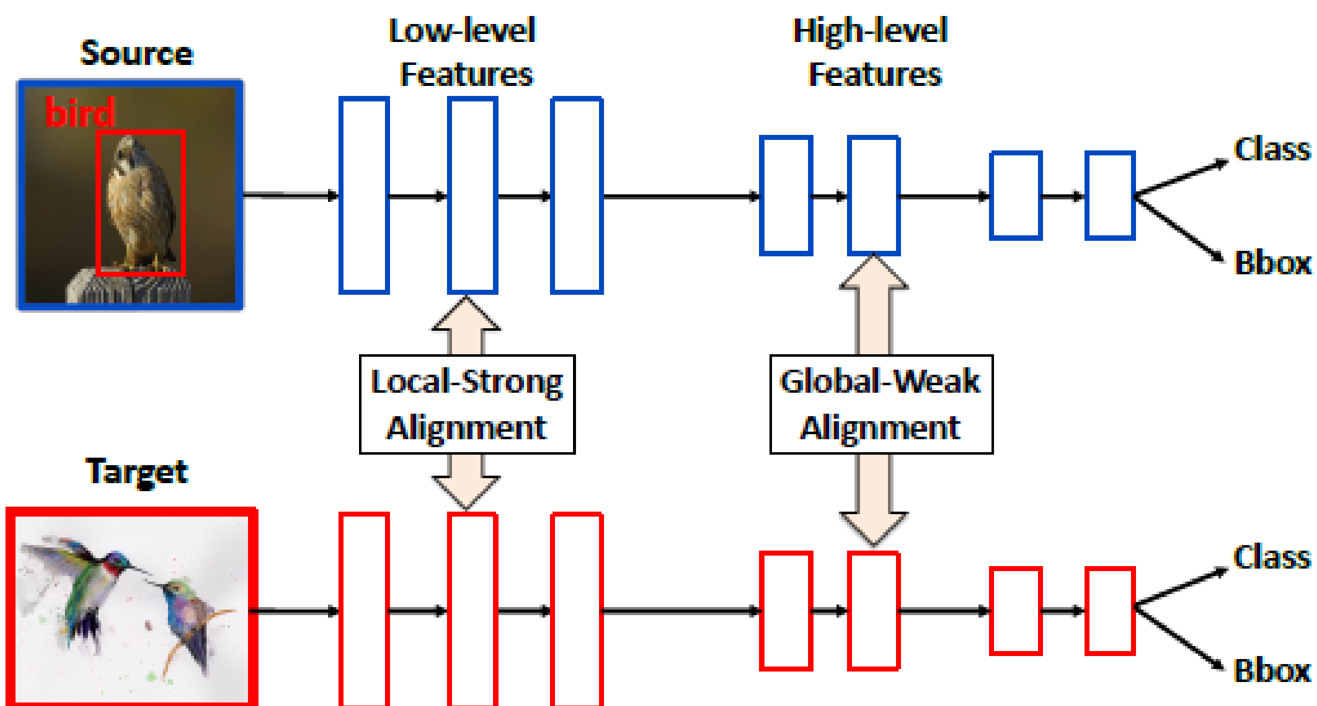




赤：adapt無し, 紫: 提案手法で adapt

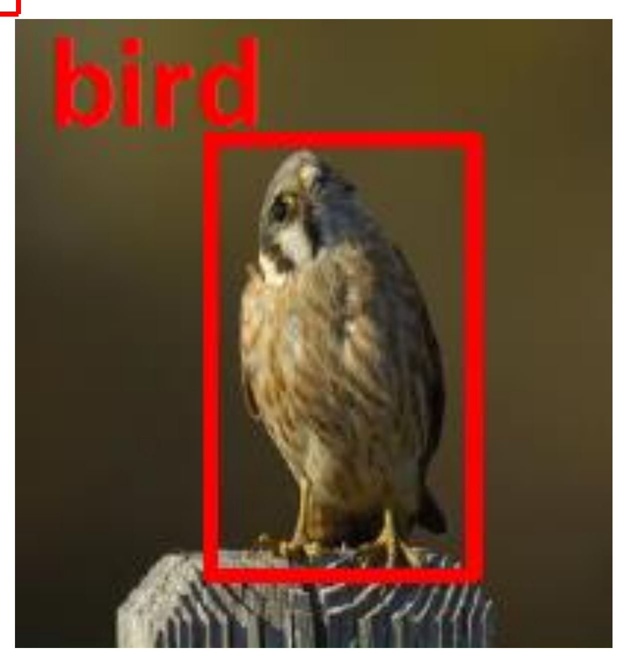
最近の研究

- Object detectionのdomain adaptation の精度向上（arXiv:1812.04798）.

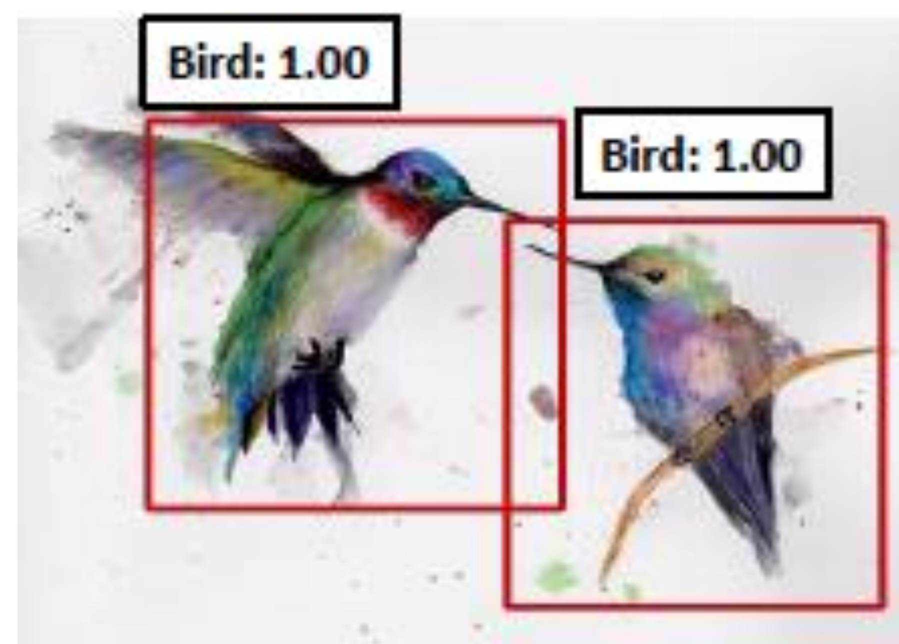


二か所で特徴量ベクトル分布の一致

・local strong alignment
・global weak alignment



Source 画像

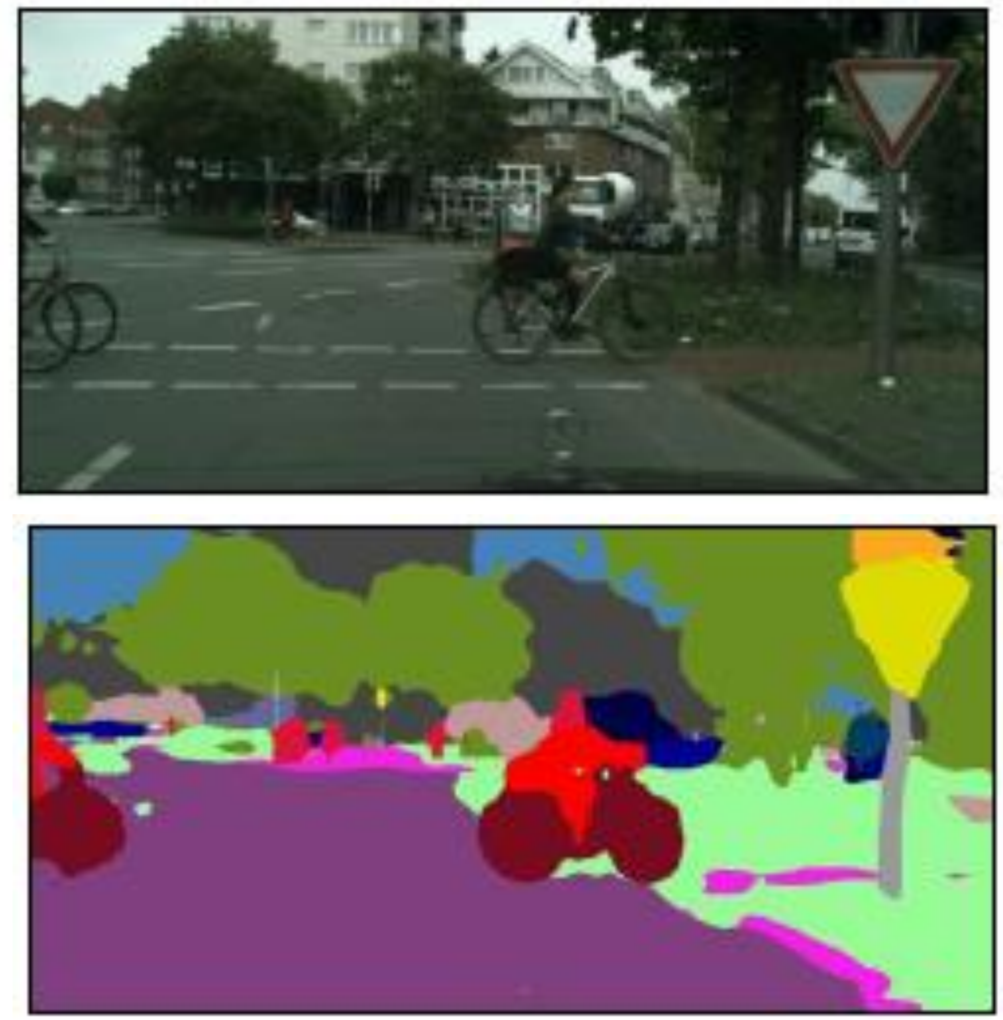


Target 画像 & adapt 結果

最近の研究

- Semantic segmentation のdomain adaptationの精度向上（arXiv:1903.04064）

source画像

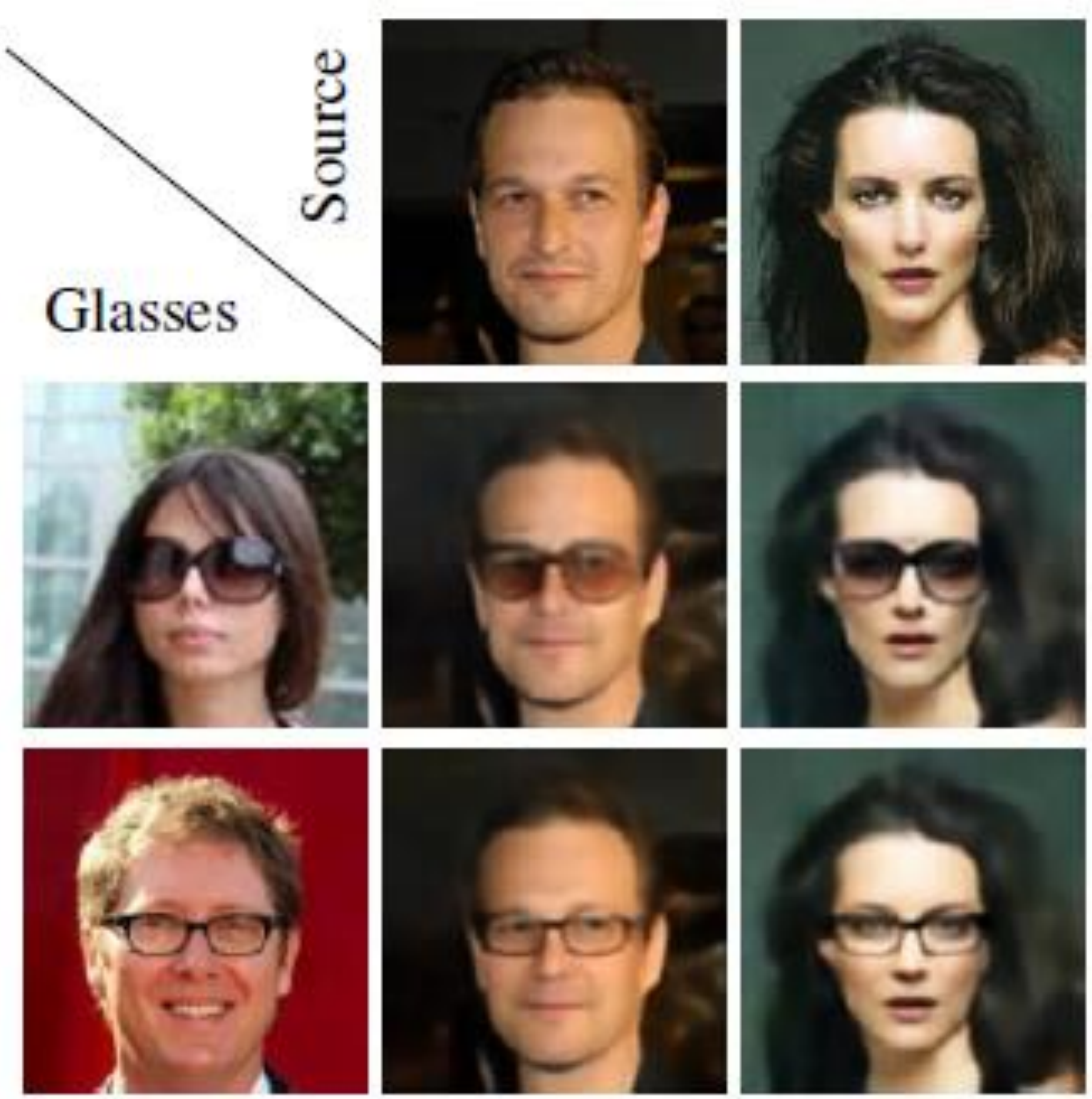


Target 画像 と adapt 結果

*「分布を合わせる」ときに「MCD で H-divergence を見積もったものを最小化」している．
そのため，正確には GAN ではないが，学習操作としてはかなり類似している．

最近の研究

Disentangle の文脈だと「特徴量ベクトル分布の一致」を利用して「domain 間で共通した特徴量」の抽出を行うことがある (O.Press, et al., ICLR2019)



Source domain の特徴量= 「顔の特徴量」

Glasses domain の特徴量
= 「顔の特徴量」と「galsses の特徴量」が
ごちゃごちゃに混ざっている.

共通特徴量をうまく抽出することで,
「顔特徴量」と「glasses 特徴量」を分離.

結果, 「Source domain の顔特徴量 +
Glasses domain のgalass 特徴量」を基に
画像を reconstruct, なんてこともできている.

DA まとめ

- GAN を使うことでちょっとした domain の違いは吸収できることを示唆.
- 最近では，「実データ」とか「classification 以外」にも適用されつつある.
  - 特徴量ベクトルを一致させる，は他の文脈でも利用されている.
- target データにラベル付けは不要なので，とりあえず試すのが良いかも.

5. まとめ

まとめ

- GAN は，二つの分布を一致させるような学習.
  - そのために綺麗な絵を作れたりする.
- 安定化のための手法として SN を中心に紹介.
  - GAN の training がうまく行かないときに試して頂ければ！
- p_z の選択で結構結果が変わる.　Future work 的な話.
- 画像生成以外のタスクへの応用もちょっとだけなされている.
  - ここでは domain adaptation の話を紹介.　最近，発展が著しい印象.

ご清聴ありがとうございました