第8回
y=ax+bから始める
初心者向けML講座

生成モデルの基礎

最終回

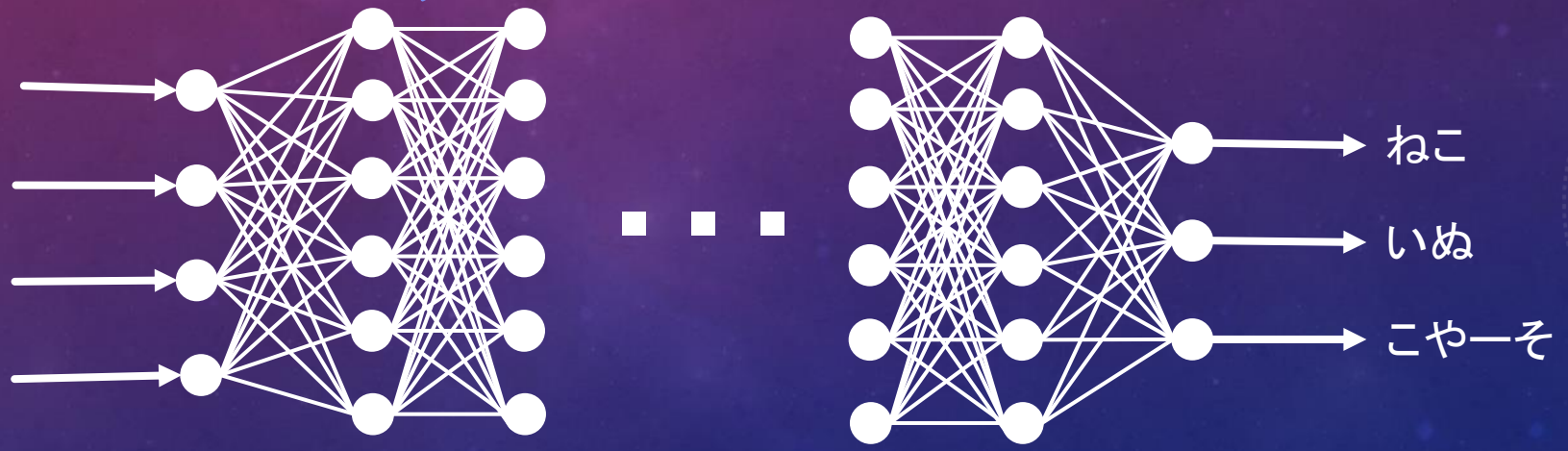
これまでのあらすじ

単純なCNNやDNNの限界 － 深いNN

- 複雑なモデルであっても、各線1本1本にある重みの更新値を求めて学習している。

線1本1本の所にある
重みwは全て偏微分値を
求めて同じ計算をしている。

$$w' = w - \lambda \frac{\partial E}{\partial w}$$



その他のアルゴリズム：Adam

- モーメンタムとRMSropの良いとこどりをしてるアルゴリズム
- s、s′：モーメンタム法　　β_1：$(0 < \beta_1 < 1)$モーメンタム法の影響度、大きいほど影響力がある
- r、r′:RMSrop　　β_2：$(0 < \beta_2 < 1)$RMSropの影響度、大きいほど影響力がある
- $\tilde{s}$、$\tilde{r}$：意味づけはしづらいが、更新回数が増えるほど小さくなりやすい
- α：全体の学習係数

$$s' = \beta_1 s + (1 - \beta_1)\frac{\partial E}{\partial w}$$

$$r' = \beta_2 r + (1 - \beta_2)\frac{\partial E}{\partial w} * \frac{\partial E}{\partial w}$$

$$\tilde{s} = \frac{s'}{1 - \beta_1^t} \quad (t\text{はそれまでの更新回数})$$

$$\tilde{r} = \frac{r'}{1 - \beta_2^t} \quad (t\text{はそれまでの更新回数})$$

$$w = w - \alpha \frac{\tilde{s}}{\sqrt{\tilde{r} + \varepsilon}}$$

回を重ねるごとに分母は
1以下の数値から1に近づくよ
つまり$\tilde{s}$、$\tilde{r}$は小さくなりやすいよ

これを考えた人は
変態だと思うよ……

Kerasの場合、結局どう使うの？

- optimizerが使うアルゴリズムの指定なのでそこを指定するだけでOK

ML Shukai

なら計算式覚える意味なくない！？

ココの指定を変えるだけでOKだよ！

```
'''optimizer定義（学習アルゴリズム）'''
model.compile(loss='categorical_crossentropy', optimizer=Adam(learning_rate=0.001), metrics=['accuracy'])

'''学習'''
history=model.fit(x_train, y_train, batch_size=batch_size, epochs=epochs, validation_data=(x_test, y_test))
```

Kerasの場合、結局どう使うの？

- optimizerが使うアルゴリズムの指定なのでそこを指定するだけでOK

重要なのは計算式ではなく
どのハイパーパラメータが
どういった影響があるかだよ！

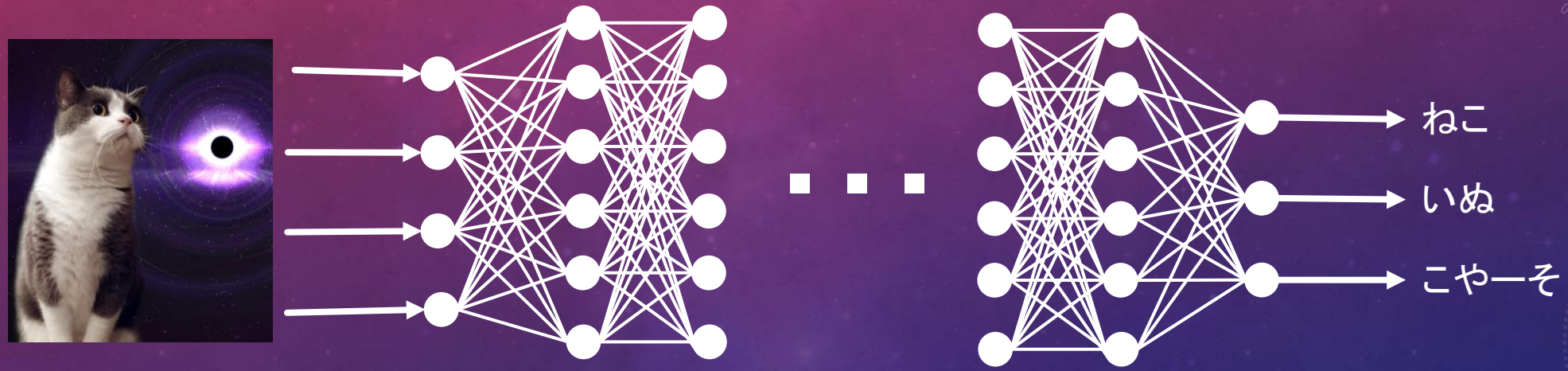
```
'''optimizer定義（学習アルゴリズム）'''
model.compile(loss='categorical_crossentropy', optimizer=Adam(learning_rate=0.001), metrics=['accuracy'])

'''学習'''
history=model.fit(x_train, y_train, batch_size=batch_size, epoch                    lidation_data=(x_test, y_test))
```

引数でハイパーパラメータの指定が
出来るのでハイパーパラメータの
種類や意味が重要になる

生成モデルの紹介

今までものは全て識別モデル

- 何らかの数値予測や何であるかを識別するためのモデル



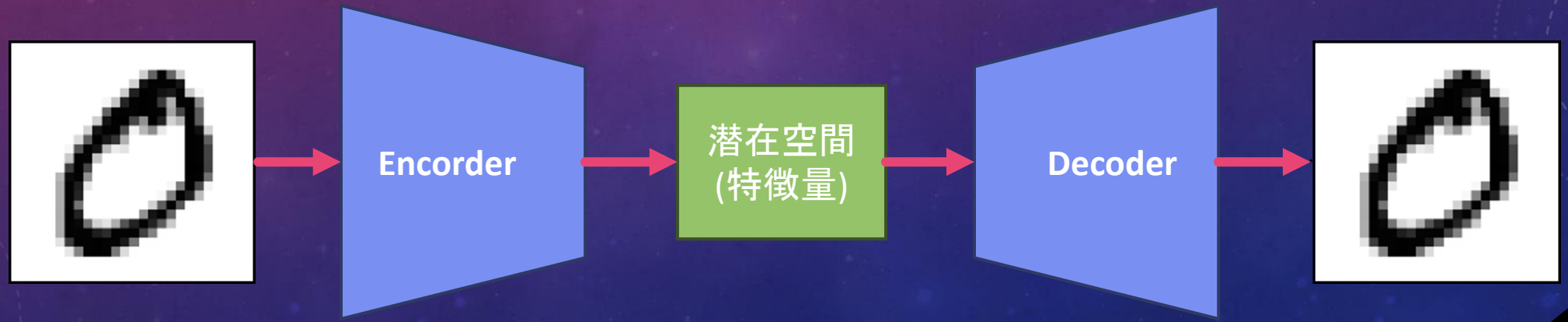
生成モデルとは？

- 訓練データを学習し、それらのデータと類似した新しいデータを生成するモデルのことを生成モデルといいます。
- 「Encoder(符号化器)」と「Decoder(複合化器)」
- 「Generator(生成器)」と「Discriminator(識別器)」

など、学習する際には複数のモデルを組み合わせて作るものが基本となります。

原始的な生成モデル「オートエンコーダ」

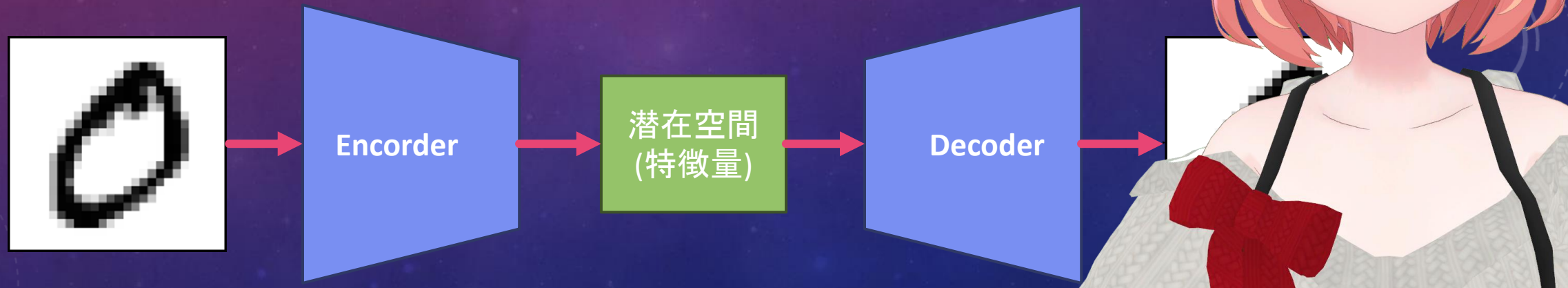
- エンコーダ(符号化器)とデコーダ(復号化器)の2つのニューラルネットワーク(NN)からなるモデル。
- エンコーダ：通常のNN(層ごとにノード数を減らす)を使い
  圧縮した潜在空間を抽出するモデル
- デコーダ：潜在空間から層ごとに徐々にノードを増やしていき
  画像を復元するモデル



原始的な生成

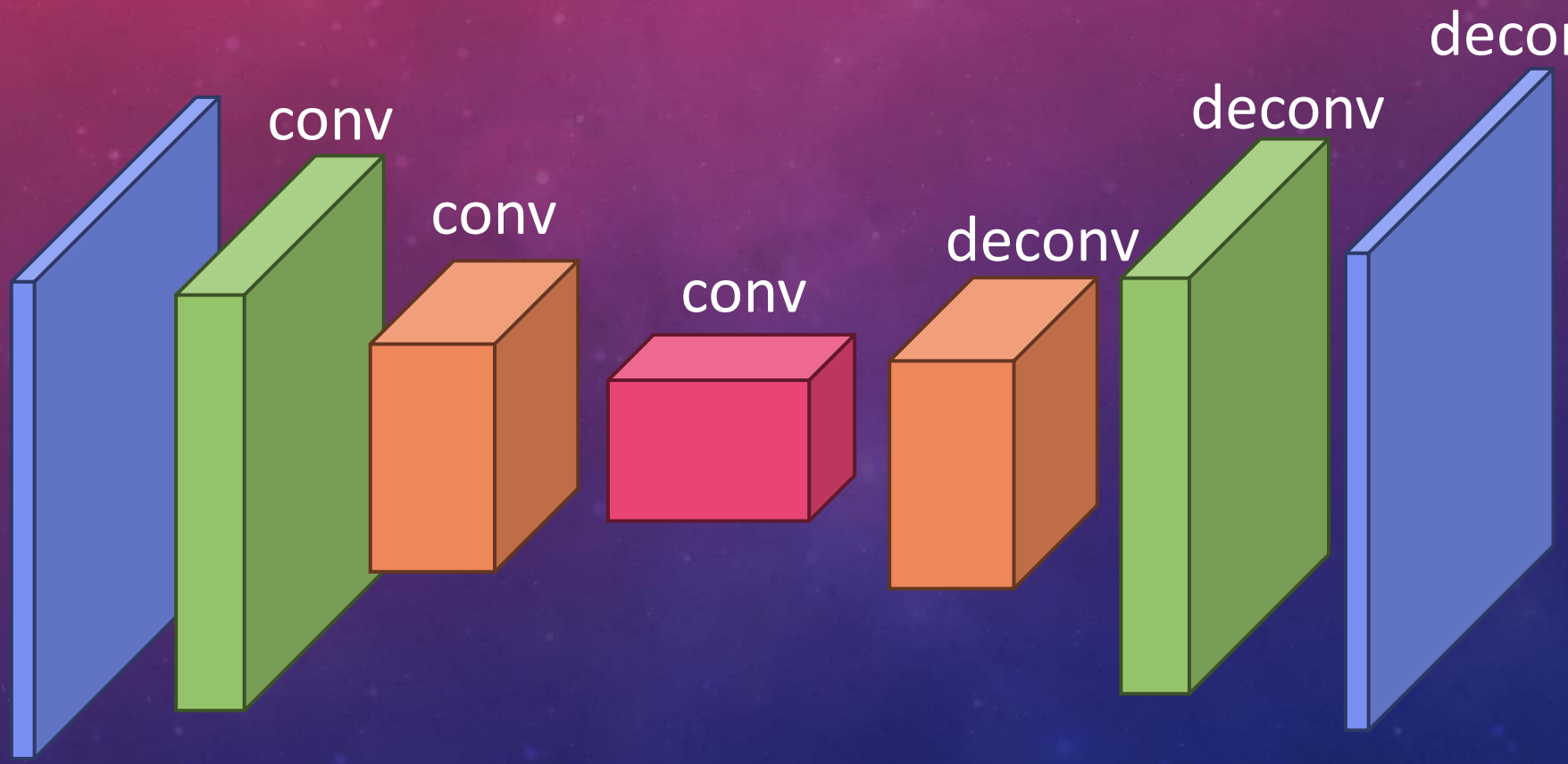
エンコーダーとデコーダーが
1層づつのものを「オートエンコーダ」、
複数層あるものを「積層オートエンコーダ」
と説明している人もいるよ

- エンコーダー(…ニューラル
  ネットワーク(N…
- エンコーダ：通常のNN(層ごとにノード数を減らす…使い
  圧縮した潜在空間を抽出するモデル
- デコーダ：潜在空間から層ごとに徐々にノードを増やしていき
  画像を復元するモデル



畳み込みオートエンコーダ

- エンコーダに畳み込みを、
  デコーダに「逆畳み込み(転置畳み込み)を使った生成モデル

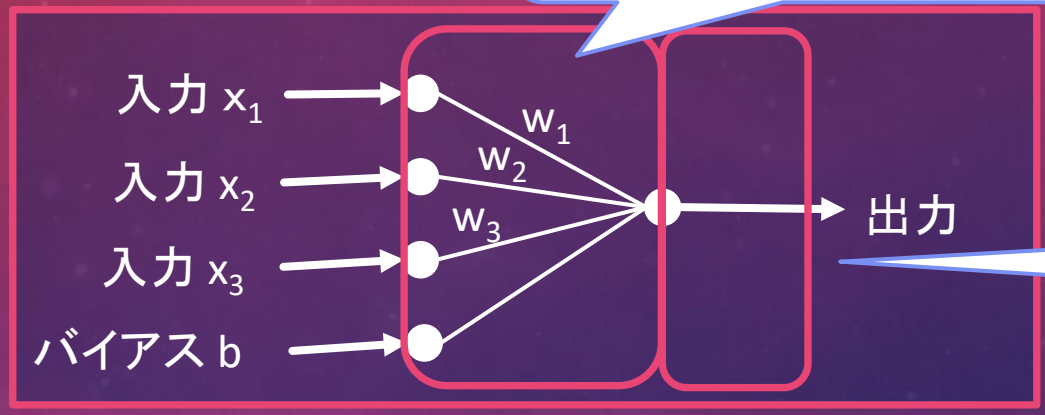


畳み込み処理の計算方法

ML Shukai

復習

パーセプトロン



入力画像にフィルターを重ねて掛け合わせ1ドットづつずらしながら計算していくのが畳み込み演算

畳み込み処理

入力X

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

×

フィルターF

1	1	1
2	0	1
−1	−1	0

＋ バイアスb 3 ＝ 出力Y

7	11
23	27

入力値×フィルター
を全て足す。
※フィルターとbが学習するパラメータ

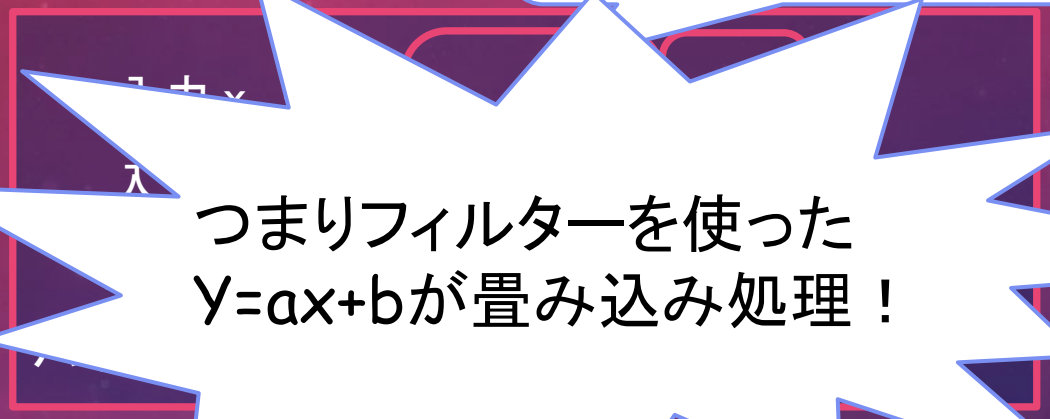
出力Yを活性化関数に通す

畳み込み処理の計算方法

ML Shukai

復習

パーセプトロン



入力値×重み
を全て足す。
※重みwとbが学習するパラメータ

つまりフィルターを使った
Y=ax+bが畳み込み処理！

活性化関数

入力画像にフィルターを重ねて掛け合わ[illegible]算していくのが畳み込み演算

畳み込み処理

入力X

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

×

フィルターF

1	1	1
2	0	1
−1	−1	0

＋ バイアスb 3 ＝ 出力Y

7	11
23	27

入力値×フィルター
を全て足す。
※フィルターとbが学習するパラメータ

出力Yを活性化関数に通す

畳み込み処理の計算方法

入力値×重み
を全て足す。
※重みwとbが学習するパラメータ

ML Shukai

復習

パーセプトロン

つまりフィルターを使った
Y=ax+bが畳み込み処理！

活性化関数

入力画像にフィルターを重ねて掛け合わ[illegible]算していくのが畳

畳み込み処理

入力x

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

× フィ[illegible]

畳み込みは第4回、第5回
の復習だよ！

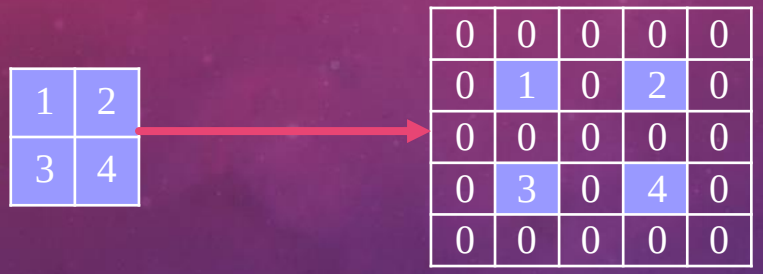
出[illegible]

入力値×フィルター
を全て足す。
※フィルターとbが学習するパラメータ

出力Yを活性[illegible]

逆畳み込み(転置畳み込み)とは？

- 入力X= $\begin{bmatrix}1&2\\3&4\end{bmatrix}$ にフィルターF= $\begin{bmatrix}1&-1\\2&1\end{bmatrix}$ で畳み込みを適用すると
- まず入力Xの各数値の間に0を増やして拡張します。



- その後、拡張したデータとフィルターで畳み込みを行います。

$$\begin{bmatrix}0&0&0&0&0\\0&1&0&2&0\\0&0&0&0&0\\0&3&0&4&0\\0&0&0&0&0\end{bmatrix} \times \begin{bmatrix}1&-1\\2&1\end{bmatrix} = \begin{bmatrix}1&2&2&4\\-1&1&-2&2\\3&6&4&8\\-3&3&-4&4\end{bmatrix}$$

逆畳み込み(転置畳み込み)とは？

- 入力X= $\begin{bmatrix}1&2\\3&4\end{bmatrix}$ にフィルターF= $\begin{bmatrix}1&-1\\2&1\end{bmatrix}$ で

畳み込みとは逆に
画像を復元するような動きが
計算できるよ！

- まず入力Xの各数値の間に0を増やして拡張します。

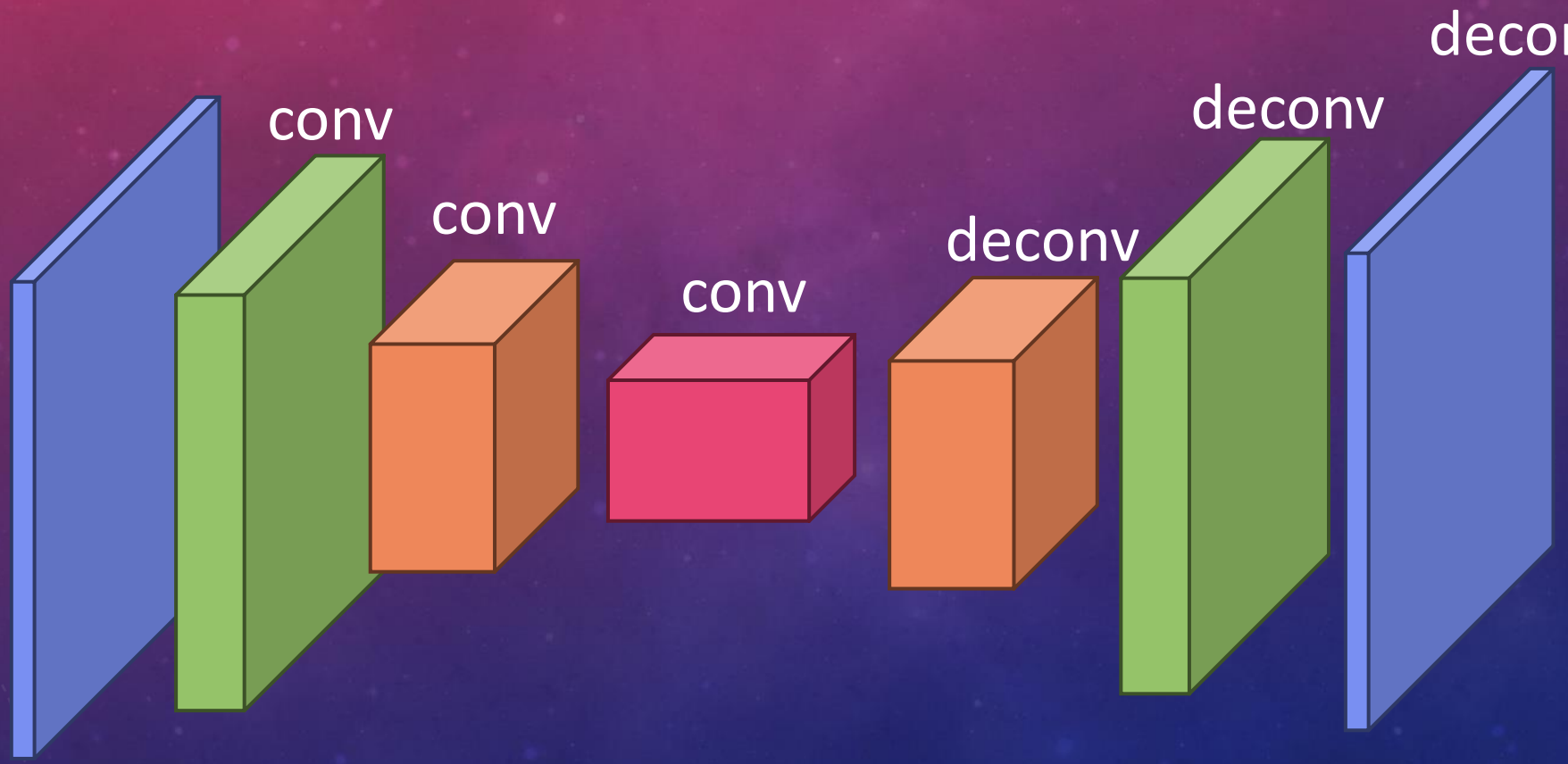
$$\begin{bmatrix}1&2\\3&4\end{bmatrix} \rightarrow \begin{bmatrix}0&0&0&0&0\\0&1&0&2&0\\0&0&0&0&0\\0&3&0&4&0\\0&0&0&0&0\end{bmatrix}$$

- その後、拡張したデータとフィルターで畳み込みを行いま

$$\begin{bmatrix}0&0&0&0&0\\0&1&0&2&0\\0&0&0&0&0\\0&3&0&4&0\\0&0&0&0&0\end{bmatrix} \times \begin{bmatrix}1&-1\\2&1\end{bmatrix} = \begin{bmatrix}1&2&2&4\\-1&1&-2&2\\3&6&4&8\\-3&3&-4&4\end{bmatrix}$$

畳み込みオートエンコーダ

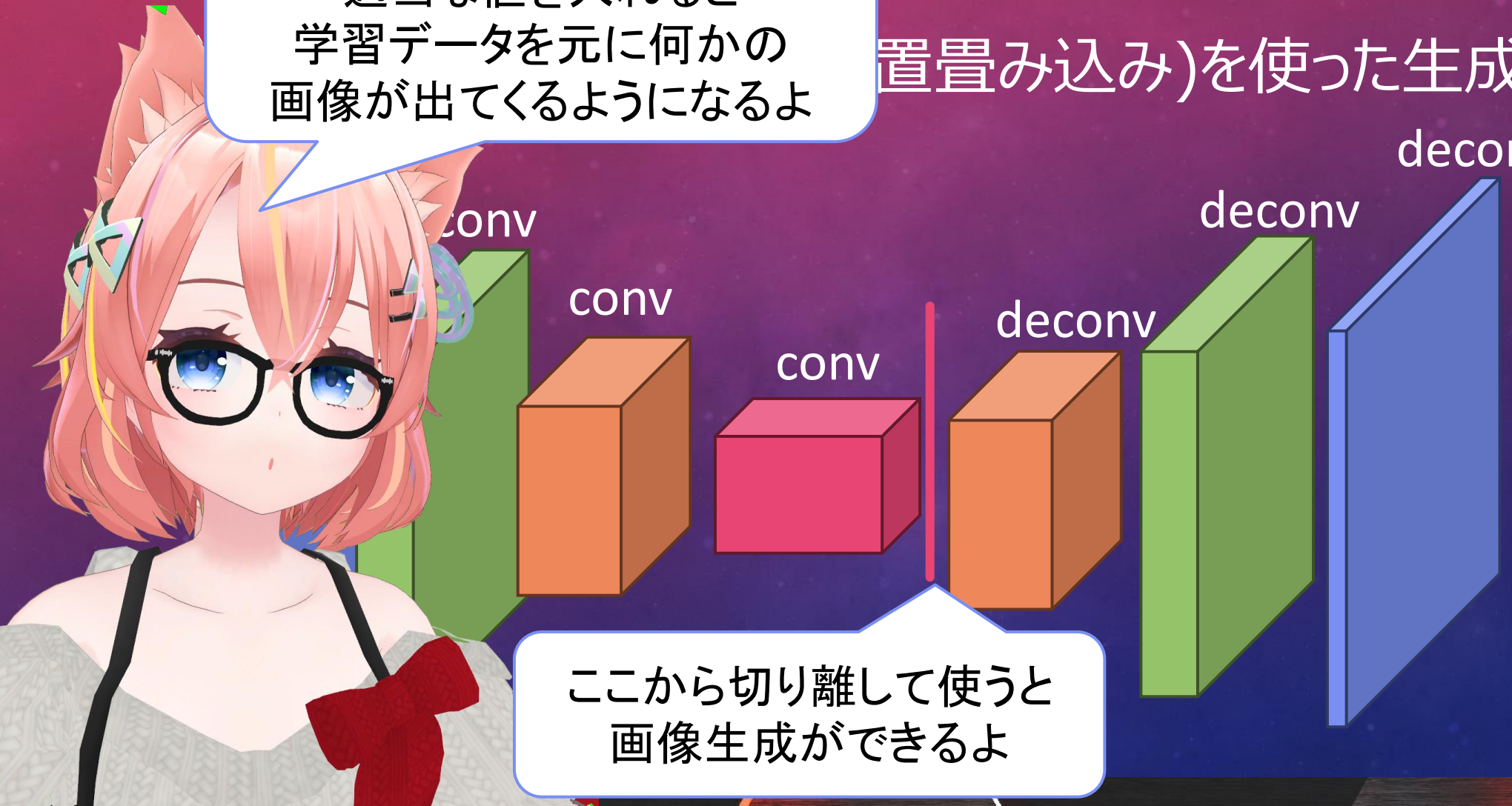
- エンコーダに畳み込みを、
  デコーダに「逆畳み込み(転置畳み込み)を使った生成モデル



畳み込みオートエンコーダー

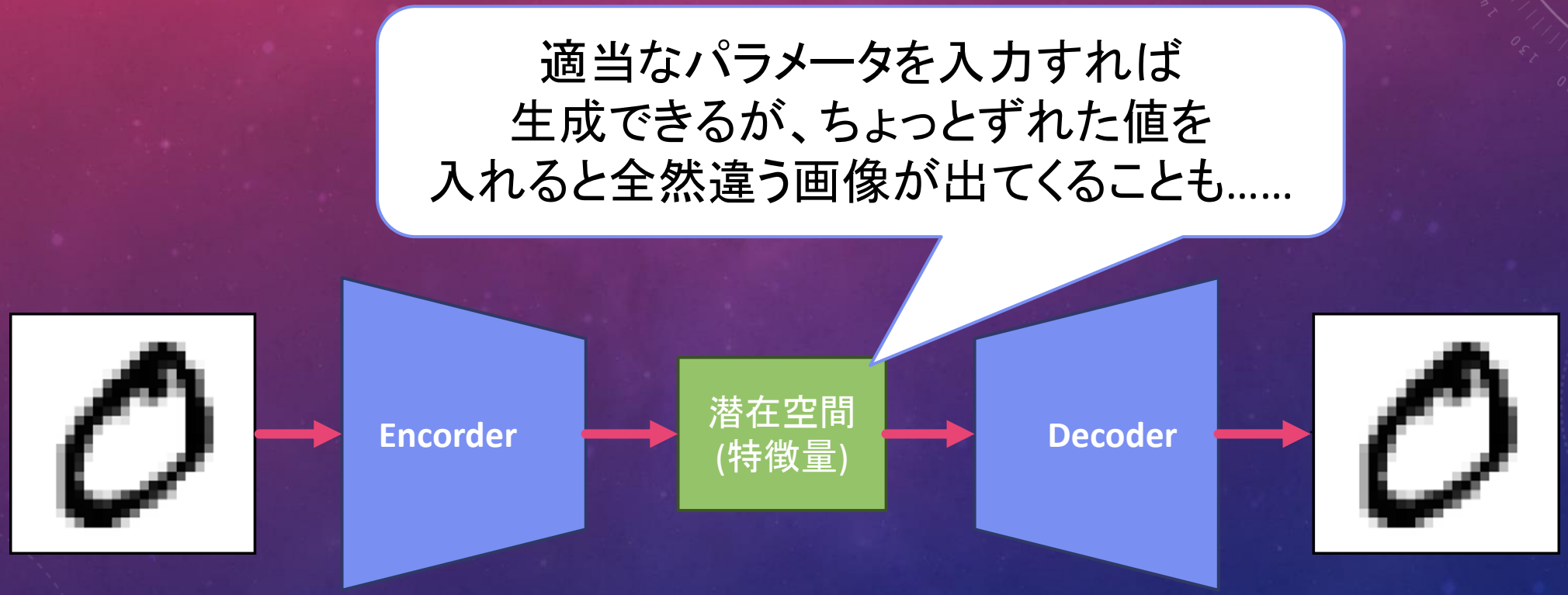
- エ

置畳み込み)を使った生成モデル



オートエンコーダの問題

- 潜在空間のパラメータのうちどの値をとればどういう画像が生成できるのか全く分からない

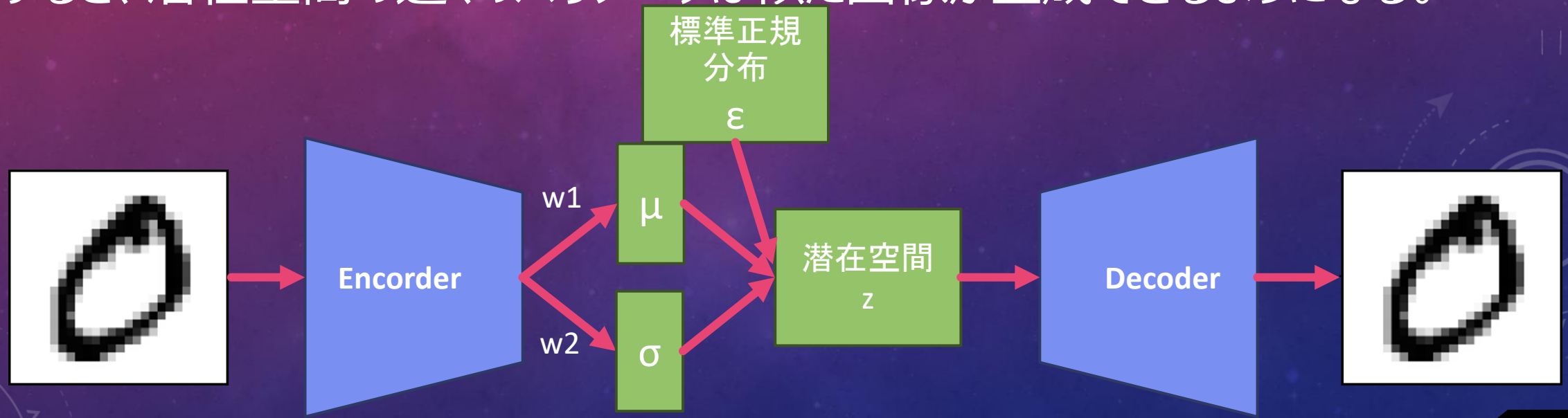


VAE(Variational Autoencoder)

- エンコーダー出力のμを平均、σ^2を分散とみなして
- 「平均０、分散１のランダムなパラメータε」とμ、σを使い

$$z=\mu+\varepsilon\sigma$$

とすると、潜在空間の近くのパラメータは似た画像が生成できるようになる。



※w1,w2は学習パラメータ

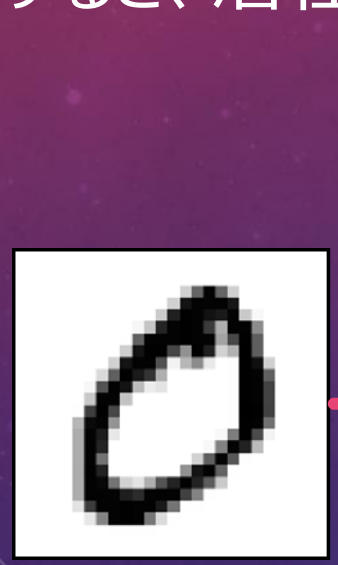
VAE(Variational A

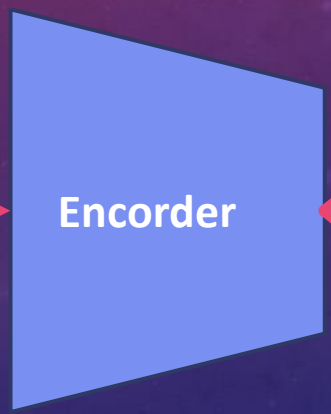
- エンコーダー出力のμ
- 「平均０、分散１のラ

何故かはここに書くには
時間も余白も足りないので省略！

$$z=\mu+\varepsilon\sigma$$

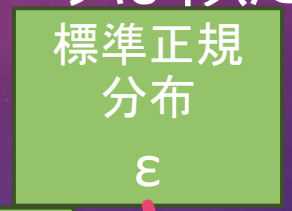
とすると、潜在空間の近くのパラメータは似た画像が生成できる

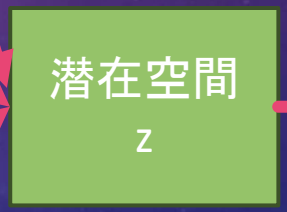




w1

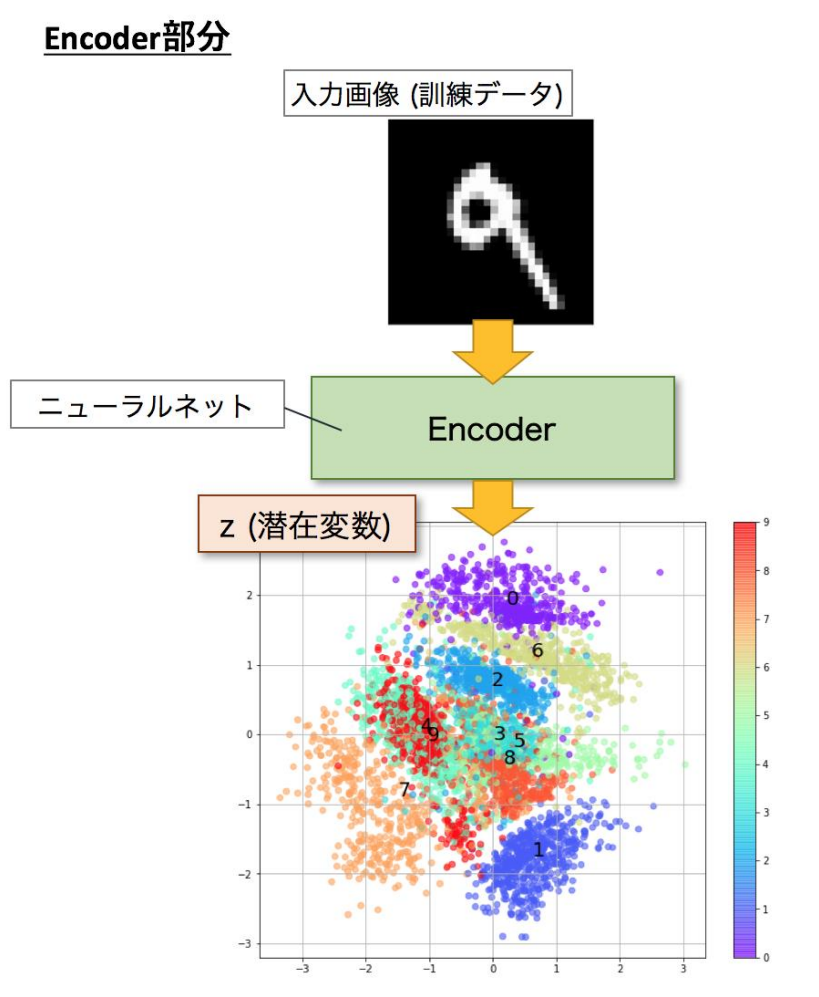
w2





※w1,w2は学習パラメータ

VAEの潜在空間



Variational Autoencoder徹底解説より引用
https://qiita.com/kenmatsu4/items/b029d697e9995d93aa24

MNISTをVAEで学習すると
似た数字が潜在変数の
近くにいることがわかるよ
不思議だね！

ML Shukai

オートエンコーダーの活用

オートエンコーダはデータの生成以外にも、

- クラスタリング
- 異常検知
- ノイズ除去

などにも活用されている。

オートエンコーダーの活用

オートエンコーダはデータの生成以

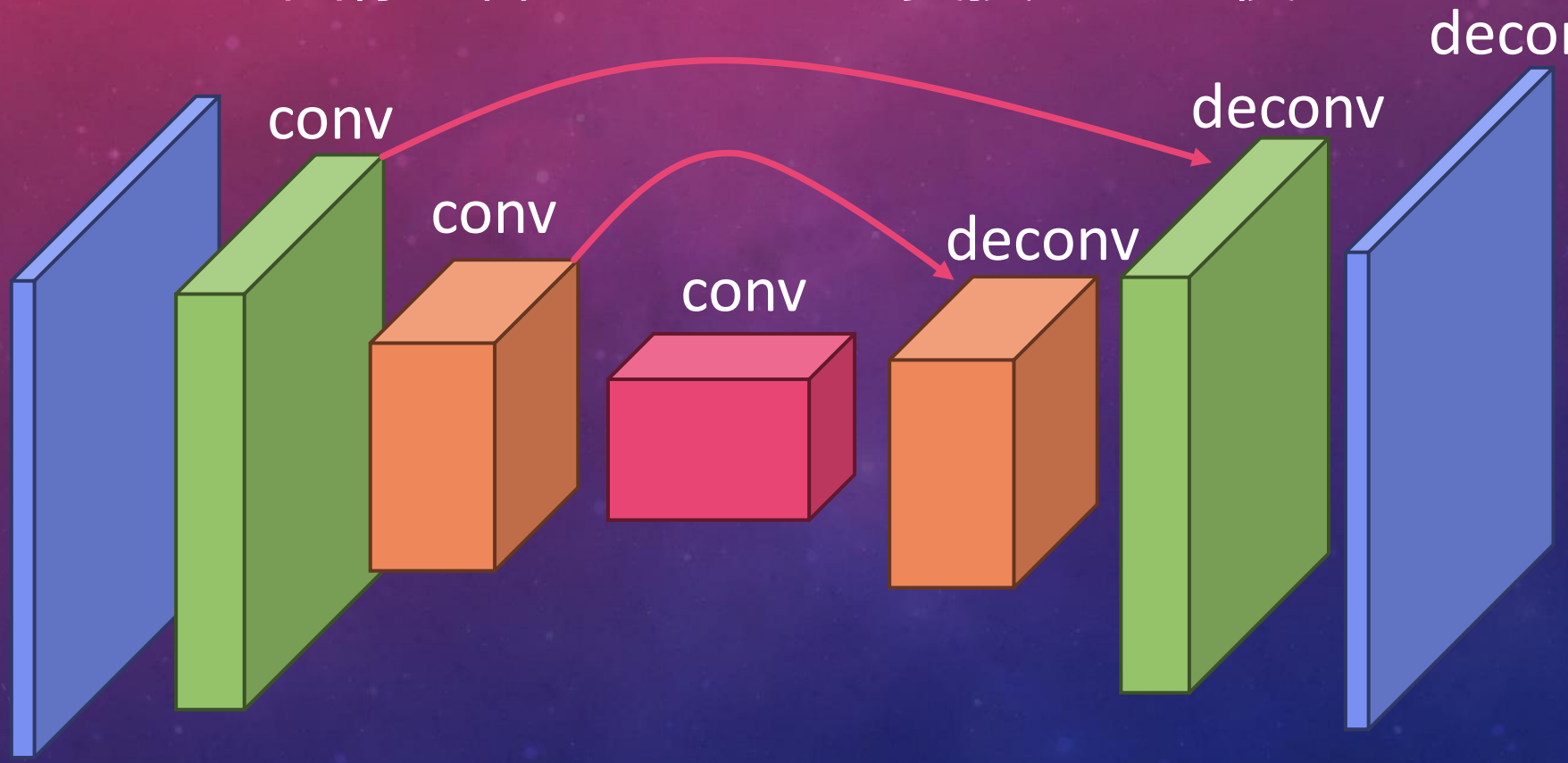
- クラスタリング
- 異常検知
- ノイズ除去

などにも活用されている。

stable diffusionのノイズ除去は
この辺の応用っぽいよ！

U-Net

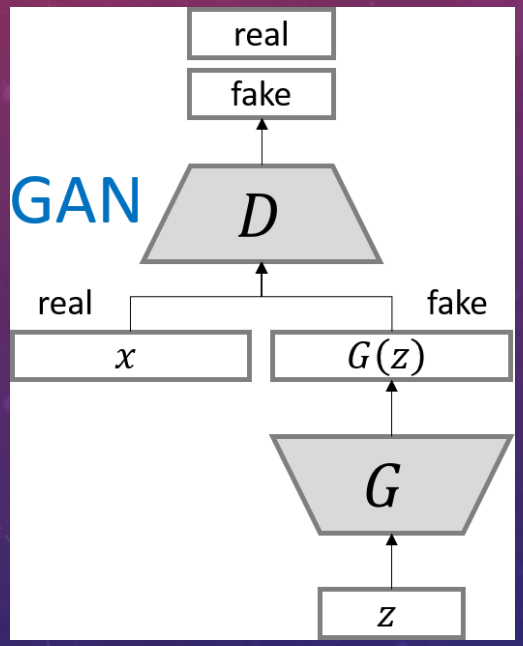
- 畳み込みオートエンコーダーにResNetでやったスキップ接続を加えるとU-Netという仕組みになり、画素１ドットごとに分類するセマンティック・セグメンテーションや画像の着色などスタイル変換タスクに使われるなどしています。



GAN

- Generative Adversarial Networks(GAN、敵対的生成ネットワーク)
- これは乱数zからGenerator(生成器)を通して画像を生成し、その画像と本物の画像を混ぜてDiscriminator(識別器)に判別させる手法で

生成器は識別器を騙すように学習し、識別器は画像が本物か生成器で作られた偽物かを見破るように学習していきます。

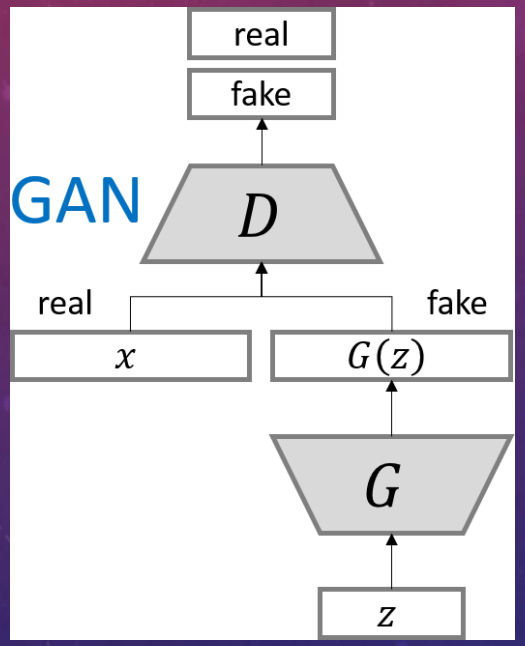


引用:https://github.com/hwalsuklee/tensorflow-generative-model-collections

GAN

- Generative Adversarial Networks(GAN、敵対的生成ネットワーク)
- これは乱数zからGenerator(生成器)を通して画像を生成し、その画像と本物の画像を混ぜてDiscriminator(識別器)に判別させる手法で

生成器は識別器を騙すように学習し、識別器は画像が本物か生成器を見破るように学習していきます。



生成器と識別器のどちらかが
強くなりすぎると学習が進まなくなる
ので交互にちょっとづつ学習させる
のが基本だよ

引用:https://github.com/hwalsuklee/tensorflow-gene

複数のモデルを組み合わせるという手法

- オートエンコーダーやGANといった複数のモデルを組み合わせるという考え方はAlphaGoやStable Deffusion、強化学習のActor-cliticなど現在の様々なAIの基本的な考え方になっています。

- 最新のAI技術を学ぶ際も今までやってきたものに限らず様々な基礎的な機構の何と何が組み合わせって出来ているか？などはAIを学ぶにあたって必須となる考え方です。

最後に

- このy=ax+bから始めるML講座が皆さんの機械学習を学ぶきっかけや一助になれば本当に幸いです。

- ８か月という長いようで短い期間でしたが、

ご視聴いただきありがとうございました。

y=ax+bから始めるML講座

おわり

ありがとうございました

少し間を開けたら中級編もやりたいですね！

y=ax+bから始めるML講座

おわり

ありがとうございました

少し間を開けたら中級編もやりたいですね！