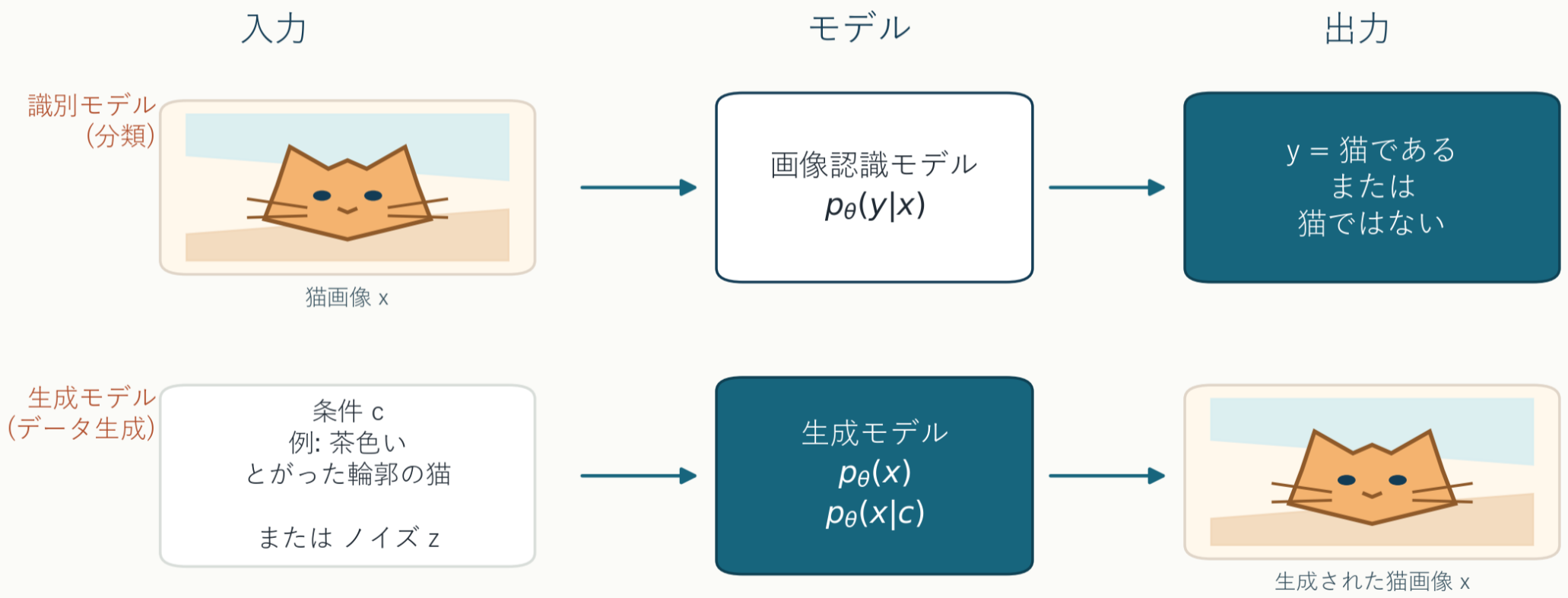
5.3 生成モデル

データの「作られ方」を学ぶモデルとして整理する

2026/05/07 三木隆斗

生成モデルとは



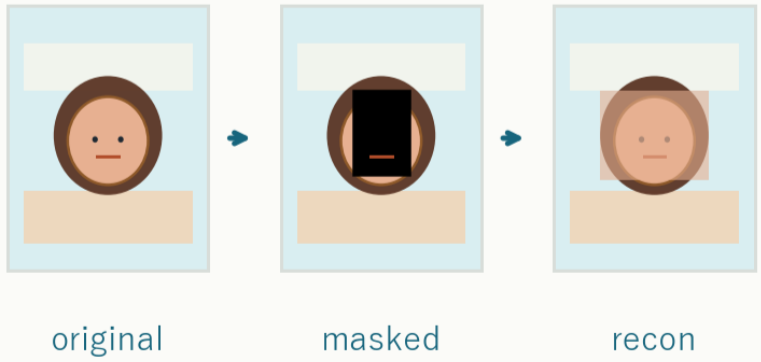
変数: x = データ, y = ラベル・予測対象, c = 条件情報, z = 生成の種, θ = 学習するパラメータ
識別モデルは「判定する」。生成モデルは「データらしい x を作る」。

生成モデルの利用例

分類や回帰だけでは、「ありそうなデータ」を作る・比べることは扱いにくい。

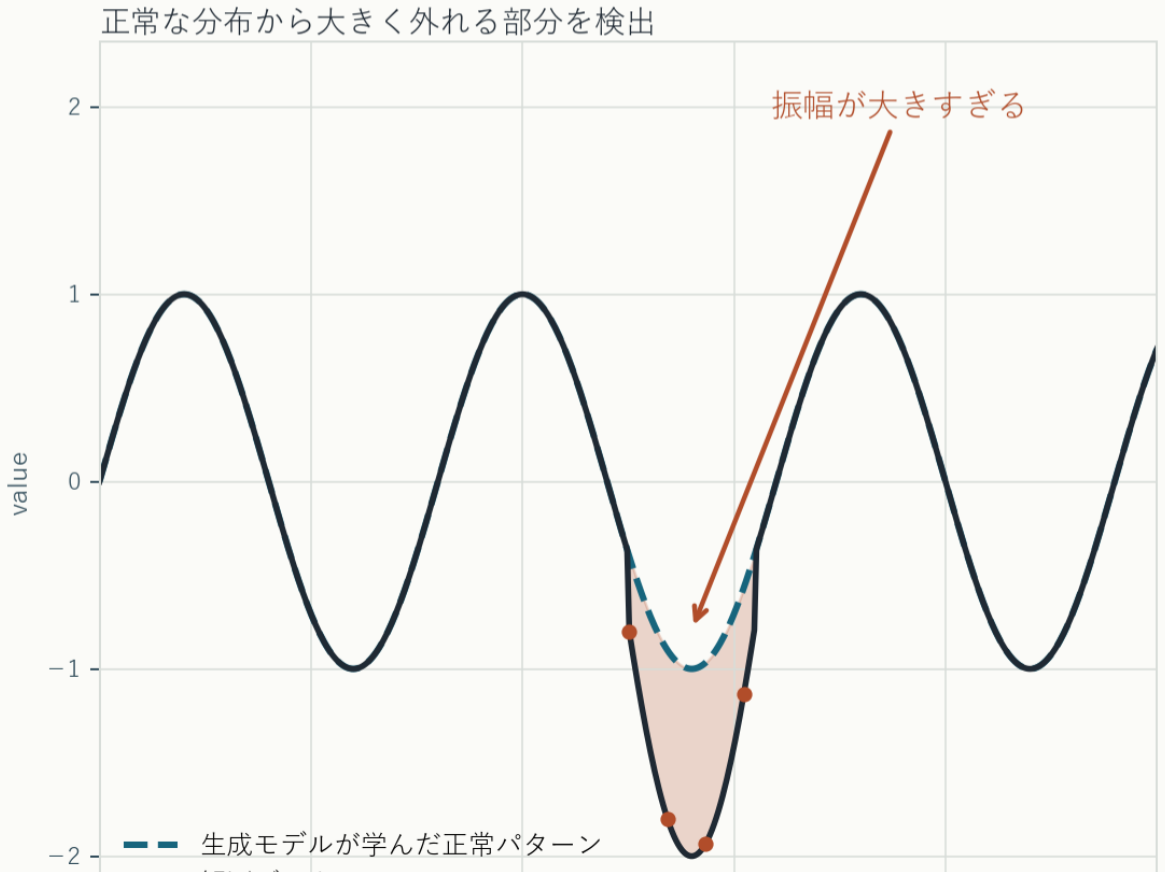
欠損補完

欠けた部分を、周囲と整合するように復元



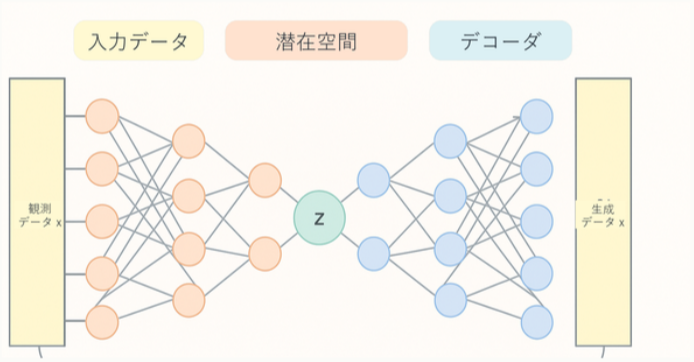
異常検知

正常な分布から大きく外れる部分を検出



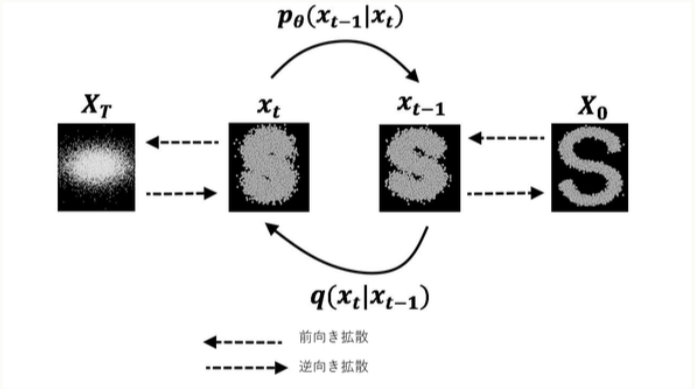
生成モデルの分類

1. 潜在変数モデル



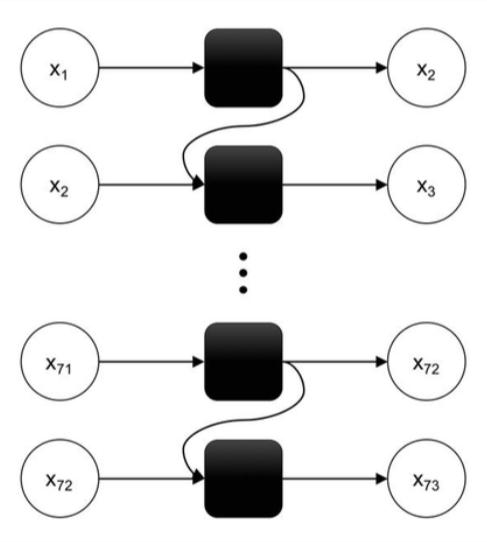
見えない原因 z を介して x を表す

2. ノイズ除去過程



ノイズを足して壊し、逆向きに復元する

3. 逐次生成



系列全体を一気に扱わず、
次の要素を予測する問題に分ける

共通点: 高次元のデータ分布 p(x) を、扱いやすい構造に分解して学習する

潜在変数モデル

観測されたデータ x の背後に、直接見えない変数 z があると考える。

観測データ x

顔写真
256 × 256 × 3 ピクセル
= 196,608 次元の数値

←

潜在変数 z

顔の向き、表情、性別、髪型など
画像を生み出す背後の要因

$$p_\theta(x) = \int p_\theta(x \mid z) p(z) dz$$

考え方: 高次元のピクセル列 x を直接扱う代わりに、
より低次元で意味のある z から x が生成されるとみなす。

潜在変数モデルの分布

記号	名前	意味	誰が決めるか
$p(z)$	事前分布	データを見る前に、`z` がどれくらい起こりやすいか	設計者が決める
$p_\theta(x \mid z)$	デコーダ / 尤度	`z` が与えられたとき、`x` がどれくらい生成されやすいか	形は設計し、θ は学習する
$p_\theta(x)$	周辺尤度	データ `x` そのものがどれくらい起こりやすいか	$p(z)$ と $p_\theta(x \mid z)$ から間接的に決まる

$$p_\theta(x) = \int p_\theta(x \mid z) p(z) dz$$

学習で最大化したいのは、この $\prod_x p_\theta(x_{data})$。

事後分布と近似事後分布

記号	名前	意味	重要な点
$p_\theta(z \mid x)$	事後分布	`x` が観測されたあと、どの `z` が原因としてありそうか	モデルから決まるが、計算が難しい
$q(z)$	補助分布	下界を作るために導入する、`z` への重み付け	EMでは事後分布に一致させる
$q_\phi(z \mid x)$	近似事後分布	計算できない事後分布を近似する分布	VAEではϕを学習する

$$p_\theta(z \mid x) = \frac{p_\theta(x \mid z)p(z)}{p_\theta(x)}$$

$p_\theta(x)$ が分母にあるため、事後分布も一般には計算しにくい。

何を最大化したいのか

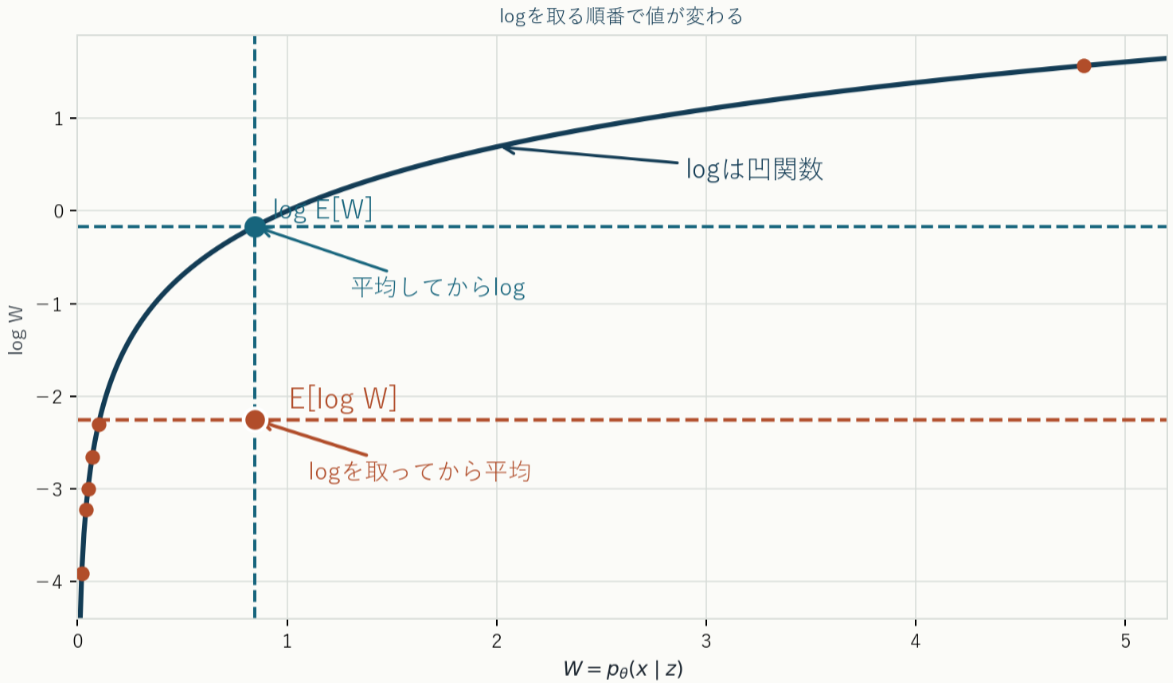
潜在変数モデルで最大化したいのは、観測データの対数尤度。

$$\log p_\theta(x) = \log \int p_\theta(x \mid z) p(z) dz = \log \mathbb{E}_{p(z)}[p_\theta(x \mid z)]$$

問題は、**logの中に潜在変数全体の積分がある** こと。

- 正確に計算するには、ありうる `z` をすべて足し合わせる必要がある
- 多くの `z` では $p_\theta(x \mid z)$ はほぼ0で、ごく一部の `z` だけが大きな値をとる
- Monte Carlo近似では、たまたま$p_\theta(x \mid z)$の大きい `z` を引けるかどうかでサンプル平均が大きくぶれる
- その平均にlogをかけるため、対数尤度や勾配の推定が不安定になりやすい

ここで $W = p_\theta(x \mid z)$ とおく。



Jensenの不等式: E[log W] <= log E[W]。logしてから平均する量は、対数尤度の下界になる。

上図のようにlogは凹関数なので、

$$\mathbb{E}[\log W] \leq \log \mathbb{E}[W]$$

つまり、**logを取ってから平均する量は、対数尤度の下界** になる。

下界とKLの式変形

ベイズの定理より、

ここで $q(z)$ は、z に関して任意に導入する補助分布。

$$\log p_\theta(x) = \log p_\theta(x, z) - \log p_\theta(z \mid x)$$

両辺で $q(z)$ の期待値を取る。左辺は z に依存しない。

$$\log p_\theta(x) = \mathbb{E}_{q(z)}[\log p_\theta(x, z)] - \mathbb{E}_{q(z)}[\log p_\theta(z \mid x)]$$

ここに $-\mathbb{E}_{q(z)}[\log q(z)] + \mathbb{E}_{q(z)}[\log q(z)]$ を足す。

$$\log p_\theta(x) = \underbrace{\mathbb{E}_{q(z)}[\log p_\theta(x, z)] - \mathbb{E}_{q(z)}[\log q(z)]}_{F(q,\theta)} + \underbrace{\mathbb{E}_{q(z)}\left[\log \frac{q(z)}{p_\theta(z \mid x)}\right]}_{\mathrm{KL}(q(z) \,\|\, p_\theta(z|x))}$$

$H(q) = -\mathbb{E}_{q(z)}[\log q(z)]$ なので、
$F(q, \theta) = \mathbb{E}_{q(z)}[\log p_\theta(x, z)] + H(q)$。

下界とKL

前の式変形から、対数尤度は次の2つに分けられる。

$$\log p_\theta(x) = F(q, \theta) + \mathrm{KL}(q(z) \,\|\, p_\theta(z \mid x))$$

ここで $F(q, \theta)$ は、直接最大化したい $\log p_\theta(x)$ の代わりに扱う量。

$$F(q, \theta) = \mathbb{E}_{q(z)}[\log p_\theta(x, z)] + H(q)$$

KLは二つの分布のずれを測る量で、常に0以上。

$$\log p_\theta(x) \geq F(q, \theta)$$

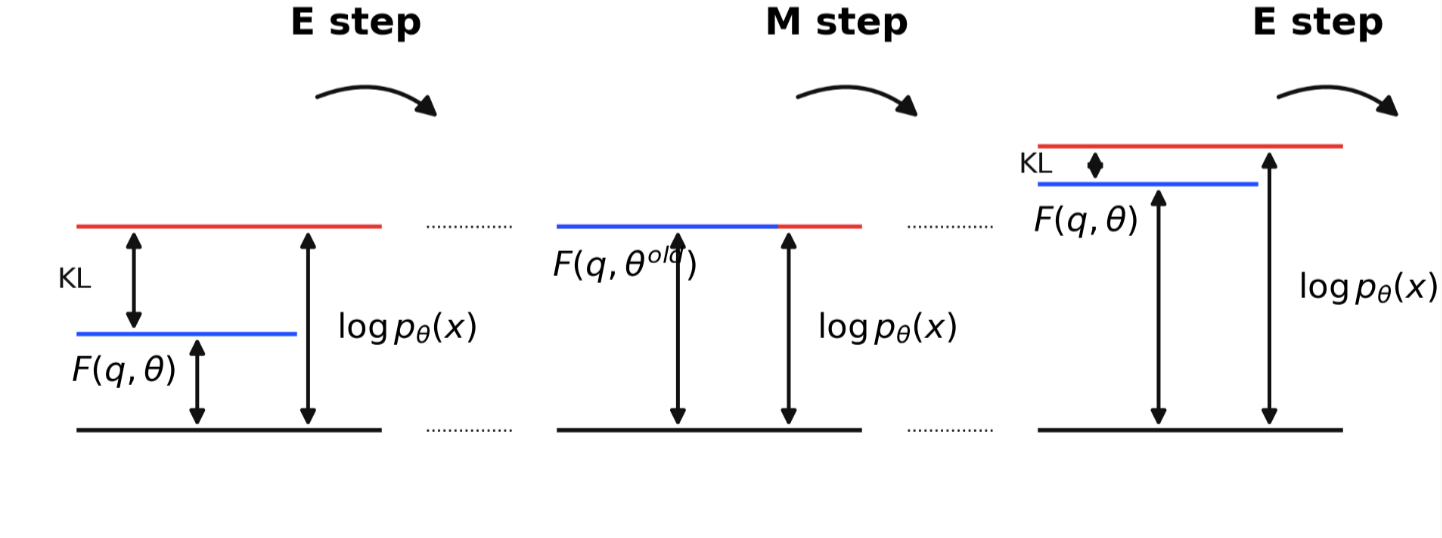
したがって $F(q, \theta)$ は、対数尤度の**下界**になる。

EMアルゴリズム（expectation-maximization）

EMは、下界 $F(q, \theta)$ を使って対数尤度を単調に改善する。

$$\log p_\theta(x) = F(q, \theta) + \mathrm{KL}(q(z) \,\|\, p_\theta(z \mid x))$$

E-step: $q(z) = p_\theta(z \mid x)$ としてKLを0にする。
M-step: $q(z)$ を固定して $\theta \leftarrow \arg\max_\theta F(q, \theta)$ を解く。



EMからVAEへ

EMは強力だが、深層生成モデルでは困る。

- 真の事後分布 $p_\theta(z \mid x)$ が計算しにくい
- ニューラルネットワークを入れると閉形式で解けない
- E-stepを厳密に実行できない

そこでVAEでは、真の事後分布の代わりに **近似事後分布** を学習する。

$$q_\phi(z \mid x)$$

VAEとは(Variational Autoencodeer)

VAEは、確率的なオートエンコーダ。

入力 x

↓ エンコーダ $q_\phi(z \mid x)$

潜在変数 z の確率分布

↓ デコーダ $p_\theta(x \mid z)$

再構成された x

目的は、**よく再構成すること** と **潜在空間を整えること** の両立。

VAEのELBOの導出

任意の近似事後分布 $q_\phi(z \mid x)$ に対して、次が成り立つ。

$$\log p_\theta(x) = \mathbb{E}_{q_\phi(z|x)}\left[\log \frac{p_\theta(x,z)}{q_\phi(z \mid x)}\right] + \mathrm{KL}(q_\phi(z \mid x) \,\|\, p_\theta(z \mid x))$$

右辺第2項は非負なので、これを落とすと下界になる。

$$\log p_\theta(x) \geq \mathbb{E}_{q_\phi(z|x)}\left[\log \frac{p_\theta(x,z)}{q_\phi(z \mid x)}\right]$$

さらに同時分布を $p_\theta(x,z) = p_\theta(x \mid z)p(z)$ と分解すると、

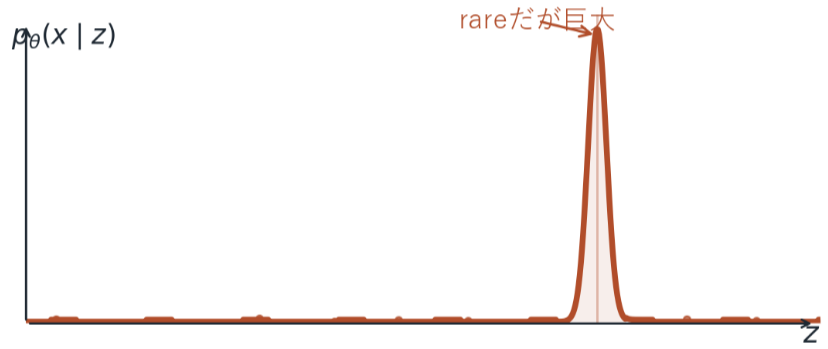
$$\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x \mid z)] - \mathrm{KL}(q_\phi(z \mid x) \,\|\, p(z))$$

これがVAEで最大化するELBO。
第1項は **再構成**、第2項は **潜在空間を事前分布に近づける正則化**。

VAEの強みと弱み：ELBOで何が変わるか

真の対数尤度

$$\log \mathbb{E}_{p(z)}[\, p_\theta(x \mid z)\,]$$

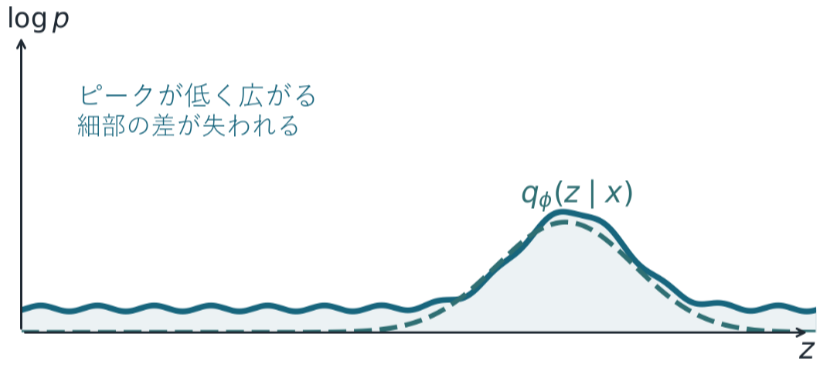


- rareな z が支配
- Monte Carlo分散大
- 学習不安定
- sharpな生成が可能

→ **高品質だが難しい**

ELBO

$$\mathcal{L}_{\text{ELBO}} = \mathbb{E}_{q_\phi(z \mid x)}[\log p_\theta(x \mid z)] - \mathrm{KL}(q_\phi(z \mid x) \| p(z))$$



- logが巨大値を圧縮
- qϕ(z|x) が有望な z に集中
- 分散減少・学習安定
- ただし平均化されやすい

→ **安定だがぼやけやすい**

ELBOは「扱いやすい下界」にする代わりに、ピークを平均化しやすい

KLが強すぎると qϕ(z|x) ≈ p(z) となり、zを使わない（事後崩壊）

拡散モデルの発想

拡散モデルは、生成を **ノイズ除去** として考える。

きれいなデータ

↓ 少しずつノイズを足す

ほぼ完全なノイズ

↓ 学習した逆過程で戻す

新しいデータ

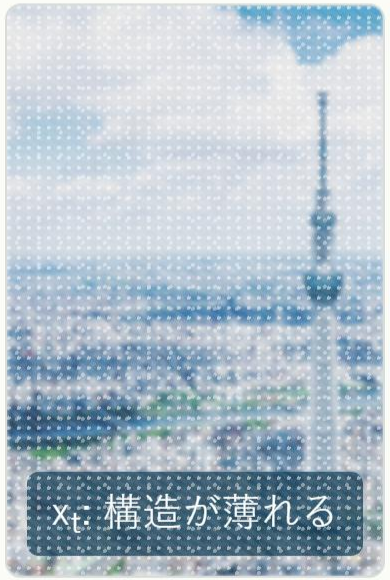
一気に作るのではなく、**壊す過程** と **直す過程** に分ける。

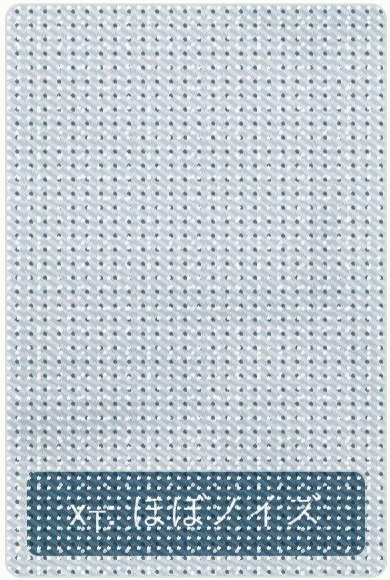
写真で見る拡散過程

前向き過程 q

$x_0 \rightarrow x_1 \rightarrow \dots \rightarrow x_T$

少しずつノイズを足して、構造を壊す





逆過程 p_θ

$x_T \rightarrow \dots \rightarrow x_1 \rightarrow x_0$

学習したモデルで、少しずつ構造を戻す

前向き過程

前向き過程では、データに少しずつガウスノイズを加える。

$$q(x_t \mid x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I)$$

$q(x_t \mid x_{t-1})$は一個前の状態 x_{t-1}を入力として、
少しだけノイズを増やした次状態x_tを生成する確率分布

時刻が進むほど、元の情報は失われる。

十分大きな `T` では、ほぼ標準正規ノイズになる。
（β_tは(0,1)の時刻tのノイズ分散を表すスケジュール。変化量の大きさを表す。
Iは単位行列、$\mathcal{N}$は多変量正規分布）

逆過程

生成では、ノイズから出発して逆向きに戻す。

$$p_\theta(x_{t-1} \mid x_t)$$

$p_\theta(x_{t-1} \mid x_t)$ は、
ノイズを含んだ状態 x_t を入力として、
少しだけノイズを除去した状態 x_{t-1} を生成する確率分布

ノイズ予測として学ぶ

実用上は、混ざっているノイズ ϵ をニューラルネットワークに予測させる。
前向き過程でx_0からx_tを生成する際に用いたノイズϵをネットワーク$\epsilon_\theta(x_t, t)$に予測させて

$$L_{\text{simple}} = \mathbb{E}\left[\|\epsilon - \epsilon_\theta(x_t, t)\|^2\right]$$

を最小化する
生成問題を、**ノイズを当てる回帰問題** に変える。

自己回帰モデル

自己回帰モデルは、系列全体の分布を **一気に直接学ぶ** のではなく、
条件付き分布の積に分解して扱う。

$$p(x_1, \ldots, x_T) = \prod_{t=1}^{T} p(x_t \mid x_{<t})$$

高次元の系列全体 $p(x)$ をそのまま求めるのは難しい。

そこで、

> これまでの文脈 $x_{<t}$ から、次の要素 x_t を予測する

という小さな問題に分ける。

次トークン予測として学ぶ

文章なら、各時点で次のトークンの分布を予測する。

$$p(\text{next token} \mid \text{context})$$

学習では、正解の系列を見ながら、

$$\sum_{t=1}^{T} \log p(x_t \mid x_{<t})$$

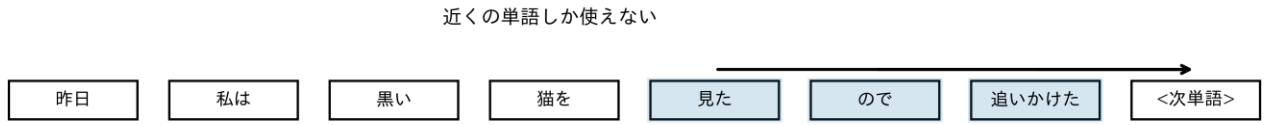
を大きくする。

生成では、出力したトークンを文脈に戻し、また次を予測する。

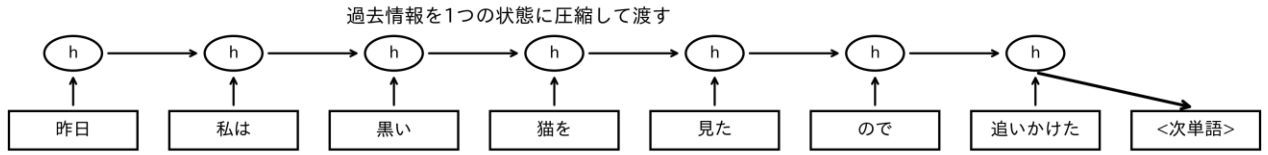
この反復により、系列全体が少しずつ作られる。

文脈の持ち方の違い

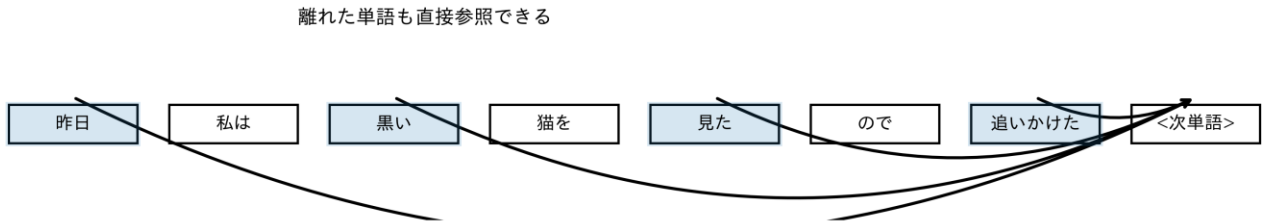
n-gram：直前の単語だけを見る



RNN：過去情報を隠れ状態に圧縮する



Transformer：重要な位置を直接参照する



自己回帰モデルの違いは、
主に **過去の情報をどう持つか** にある。

方法	使う文脈	参照方法
n-gram	直近 $n-1$ 個	近くのみ
RNN	隠れ状態	圧縮して保持
Transformer	attention	直接参照

LLMは自己回帰Transformerを
大規模データで学習したもの。

まとめ

生成モデルは、**データがどのように生じるかを確率的に記述するモデル**。

見方	役割
潜在変数モデル	見えない構造を内部表現する
ノイズ除去過程	ノイズからデータらしい構造を逐次的に回復する
自己回帰モデル	系列全体を条件付き分布に分解する

共通する核心は、多次元で扱いにくい分布を分解して簡単にすること。

Persona駆動型LLMによる
災害後居住地選択モデル

津波避難後の再定住過程を

persona bank + loading function + LLM choice で記述する

発表の流れ

1. 背景と問題意識
2. LLMの役割
3. データと予測タスク
4. 実験・比較するモデル
5. Persona 推論
6. 実験結果
7. 考察
8. 今後の課題

背景：災害後に発生する「移動」

戦争・紛争・巨大災害の発生後には、被災者の避難や移住といった住人の「移動」が発生する。

とくに人口減少傾向にある地域では、災害後の居住地選択が次に影響する。

- 地域の人口流出
- 復興事業の進み方
- 長期的な地域再編

これらのマネジメントに居住者の移動の理解は不可欠である

既存理論で扱いにくい点

既存の交通計画理論は、避難や通勤のような比較的短期の移動を扱うことが多い。

一方で、災害後の移住・定住は数十年単位の長期現象である。

そのため、以下が難しい。

- 長期的な定量データを得にくい
- 世帯ごとの履歴が不完全になりやすい
- 復興制度・地域条件・家族事情が絡む
- 観測属性だけでは選好や判断基準を捉えにくい

LLMを使う動機

大規模言語モデル（LLM）は、人間の判断過程を文脈依存的に表現しうる。

従来の表形式変数中心の離散選択モデルでは扱いにくい、半構造的な意思決定文脈を入力できる。

例：

- 世帯属性
- 居住履歴の文章要約
- 災害後の地域文脈
- 仮住まい・帰還・自力再建の候補説明
- 行動類型としての persona

元データの概要

使用データは2026年3月に実施された被災者の個人属性、居住地選択履歴などのアンケートデータ。
災害時居住地域が 宮城県・岩手県・福島県の3県のいずれかであった世帯のデータ

項目	件数
世帯数	5,002
各世帯の状態数	11,662
居住地の移動のサンプル数	6,660

学習・評価データ分割

分割は各世帯ごとに行った(同一の世帯の情報は同じ分類に分けられている)

分割	移動サンプル数	世帯サンプル数
訓練データ	4,691	3,542
バリデーション	982	709
テスト	987	751
LLM 評価	200	151

LLM評価は計算コストのため、`test_df.head(200)` を使用した。

目的変数の分布

全移動サンプル の分布。　各世帯の移動を6類型に分類した。

選択	サンプル数	比率
現在地にとどまる	519	7.8%
被災地域内の仮住まいへ移る	241	3.6%
被災地域外の仮住まいへ移る	1,676	25.2%
災害公営住宅に移る	340	5.1%
自力再建先に移る	3,286	49.3%
帰還する	598	9.0%

'自力再建先へ移る' が最多で、少数クラスの識別が難しい。
LLM評価データではstay current areaが含まれていなかった

予測タスク

各世帯の時点 t までの情報から、次時点 $t+1$ の居住状態変化を予測する。

$$x_{i,t} = \phi(s_{i,0:t}), \qquad y_{i,t} = g(s_{i,t}, s_{i,t+1}) \in \mathcal{Y}$$

ここで、

- $x_{i,t}$: 時点 t までに観測可能な属性・履歴
- $y_{i,t}$: 次状態への遷移を6類型へ写像したラベル
- $s_{i,t}$: 地域、住宅類型、段階、時点を含む状態

特徴量

時点 t までに分かる情報だけから特徴量を作る。

- 元居住地に関する情報
- 世帯属性
- 時点tでの現在情報
- 時点tまでの履歴情報
- 現在地と元居住地との関係

将来情報など予測時点 t で本来分からない情報は入力から除外する。
推論時の特徴量数は31(encoding前)

Personaを使った推論

世帯属性・居住履歴(教師データ) を使ってLLMで persona を作成

↓

persona bank(学習データから作成したペルソナの辞書) に保存

↓

個人属性などから予測対象世帯に近い personaを採用(loading)

↓

現在状態・履歴・persona・候補説明を推論LLMへのprompt に入れる

↓

LLM が6択から選択

Liu et al.,(2024)より引用

persona loading(コサイン類似度)

属性・履歴要約から作った表形式の類似度でpersona を選ぶ。
mはpersonaの1つ、ψで属性などの情報をベクトル化すると

$$m^*(x) = \arg\max_m \mathrm{sim}\left(\psi(x), \psi(p_m)\right)$$

simが二つの類似度を表す
Cosine 類似度の場合：

$$\mathrm{sim}(x, p_m) = \frac{\psi(x)^\top \psi(p_m)}{\|\psi(x)\| \|\psi(p_m)\|}$$

Learned loading

各個人に「どの persona を対応させるべきか」は観測されない。
教科書パートの潜在変数zがペルソナと対応する。
そこで persona 割当を潜在変数とみなし、EM風に loading 関数を学習する。

$\mathrm{sim}(x, p_m)$を更新しながら学習する

E-step 的な部分

現在の類似度関数に従って、各 personaとxの類似度 を計算する。

M-step 的な部分

それぞれの類似度の時の選択肢との適合率から類似度関数を更新する。

*Liu et al.,(2024)でこの学習がモデルの要点であったので実装しましたが、理解しきれませんでした。

評価指標

- **Accuracy**
  $\mathrm{Acc} = \frac{1}{N}\sum_i \mathbf{1}(\hat{y}_i = y_i)$
  個票単位の正解率。直感的だが、多数派クラスに影響されやすい。

- **Macro F1**
  $\mathrm{MacroF1} = \frac{1}{|\mathcal{Y}|}\sum_{c\in\mathcal{Y}} \mathrm{F1}_c$
  各選択肢を同じ重みで評価。少数クラスを拾えているかを見る。

- **Share MAE**
  $\mathrm{ShareMAE} = \frac{1}{|\mathcal{Y}|}\sum_{c\in\mathcal{Y}} |\pi_c - \hat{\pi}_c|$
  観測シェアと予測シェアの平均ずれ。人口移動分布の再現性を見る。

- **Share JSD**
  $\mathrm{JSD} = \frac{1}{2}D_{\mathrm{KL}}(\pi \,\|\, m) + \frac{1}{2}D_{\mathrm{KL}}(\hat{\pi} \,\|\, m)$,
  $m = \frac{1}{2}(\pi + \hat{\pi})$。分布全体の形のずれを見る。

> 記号：N は評価サンプル数、i は個票、y_i は正解ラベル、$\hat{y}_i$ は予測ラベル、$\mathbf{1}(\cdot)$ は条件が真なら1の関数。
> $\mathcal{Y}$ は6つの選択肢集合、c は選択肢、$\mathrm{F1}_c$ は選択肢 c のF1(precisionとrecallの調和平均)。
> π_c は観測シェア、$\hat{\pi}_c$ は予測シェア、$\pi, \hat{\pi}$ はそれぞれ全選択肢の分布、m は両者の平均分布、D_{KL} はKLダイバージェンス。

比較モデル

モデル	特徴
Most Frequent	最頻クラス baseline
Current Area Majority	現在地ごとの多数派
Rolling Average	直近期の遷移シェア
Domain Heuristic	ルールベース
MNL	多項ロジット
NN	多層パーセプトロン
7B Cosine Persona	cosine 類似度で persona を選ぶ LLM
7B Learned Loading	学習した loading 関数で persona を選ぶ LLM

LLMはQwen/Qwen2.5-3B-InstructとQwen/Qwen2.5-7B-Instruct(一部推論)を使用

LLM評価と同じ200件でのbaseline

model	accuracy	macro F1	balanced acc	share MAE
Current Area Majority	0.785	0.426	0.461	0.048
MNL	0.790	0.311	0.392	0.050
NN	0.725	0.317	0.388	0.050
Most Frequent	0.590	0.148	0.200	0.164
Domain Heuristic	0.165	0.080	0.247	0.292

3B時点の結果

3Bでは、personaを使っても choice が十分に多様化しなかった。

model	acc	macro F1	bal. acc	share MAE	主な傾向
No-Persona LLM	0.620	0.153	0.194	0.136	self rebuilt 98%
Cosine Persona LLM	0.480	0.137	0.150	0.136	self rebuilt 76%, within 24%
Learned Persona LLM	0.440	0.128	0.138	0.136	self rebuilt 74%, within 26%
Top3 average	0.620	0.129	0.194	0.113	self rebuilt 96%

最大比率を占める自力再建に集中して6個の選択肢を考えさせることが出来なかった。
また選択されたペルソナが同一の場合同一の選択肢しか選ばれず、
LLMによる推論ではなくペルソナあてはめによる推論となっていた

7Bとの比較

モデルを大きくしてより複雑なパターンを捉える事を考えた。

model	accuracy	macro F1	balanced acc	share MAE	JSD
7B Learned persona Loading	0.755	**0.442**	0.443	0.024	**0.005**
7B Cosine-Only Persona	0.735	0.316	0.385	**0.022**	0.008
Current Area Majority	0.785	0.426	**0.461**	0.048	0.044
MNL	**0.790**	0.311	0.392	0.050	0.045
NN	0.725	0.317	0.388	0.050	0.047

ペルソナモデルのどちらも他のモデルと同等程度の精度を示している

特にlearned loadingは分布の正確性という点で優れている

Choice share の比較

choice	observed	learned	cosine
public housing	0.095	0.045	0.035
self rebuilt	0.590	0.635	0.640
temporary outside	0.235	0.250	0.245
temporary within	0.025	0.025	0.030
return origin	0.055	0.045	0.050

stayがtestデータになかった為5つだが、小さい分布も拾えている事が分かる

Persona loading の集中度

同じ200件に対して、loading方法ごとの persona 使用を比較する。

loading	使用persona数	top persona	top2 share	傾向
Cosine-only	4	347: 150件 (75.0%)	96.0%	少数personaに強く集中
Learned loading	5	268: 114件 (57.0%)	81.5%	集中は残るが、cosineより分散

cosine類似度は属性・履歴の近さに寄りやすい。
learned loading は観測choiceへの適合も使うため、persona使用がやや分散した。
ただどちらも十分に多様なペルソナを選んでいるとは言い難い。

Personaごとのchoice分布

最新 learned loading では、同一personaでも choice が固定されない。

persona	public	self	outside	within	return
268	0.018	0.614	0.325	0.026	0.018
232	0.082	0.755	0.082	0.000	0.082
672	0.091	0.591	0.227	0.091	0.000
4	0.000	0.636	0.364	0.000	0.000
160	0.250	0.000	0.000	0.000	0.750

3Bの時とは対照的にPersona は choice を固定するラベルではなく、
選択傾向の事前分布として働いている。

考察

観点	結果	示唆
LLMの効果	7Bでは3Bよりも choice が多様化し、外部一時避難・帰還も出るようになった	モデルの大きさは考える問題の複雑性に応じて大きくする必要がある
prompt設計	実験の途中でプロンプトを少し変えただけで結果が大きく変わった。	promptを強くしすぎると persona 推論ではなく domain heuristic に近づく
Cosine loading	分布再現は良いが、persona選択が少数に集中し macro F1 が低い	情報を均等に扱うと選択行動に効く personaの選択が難しい
Learned loading	macro F1: 0.316→0.442、 balanced acc: 0.385→0.443 (cosineからの変化)	観測 choice を使って loading を学習することに有効性がある

限界と今後の課題

- prompt による結果変動が大きい
- learned loading はまだ少数personaに集中している
- LLM がどのような理由で選択したかの検証(解釈性の検討)
- 政策シナリオ入力により、復興制度や住宅供給が選択に与える影響を検討する

まとめ

- 津波避難後の居住地選択を、6分類の逐次的選択問題として定式化した
- MNL / NN / baseline と同等な精度を出すLLMモデルが作られた
- Persona という潜在表現を使って、LLMに文脈依存的な選択を行わせた
- 潜在表現の学習にトライした

LLMの利用について

- chatgptとcodexを使用しました
- 教科書の理解の為の壁打ちや論文理解にchatgptを使用して
  コードの理解と編集にcodexを使用しました
  早すぎて過程は全然追えてなかったです
- スライド作成・図の作成にもcodexを活用しました
  スライド作成に関してはご助言もあり、自分で文章のところはかなり作りました
  図の作成は頼りきりになってしまいました