Paper Report

VAECF Variational Auto-encoders for Collaborative Filtering

@ 2022/03 Chia-Jen, Yeh

Report Structure

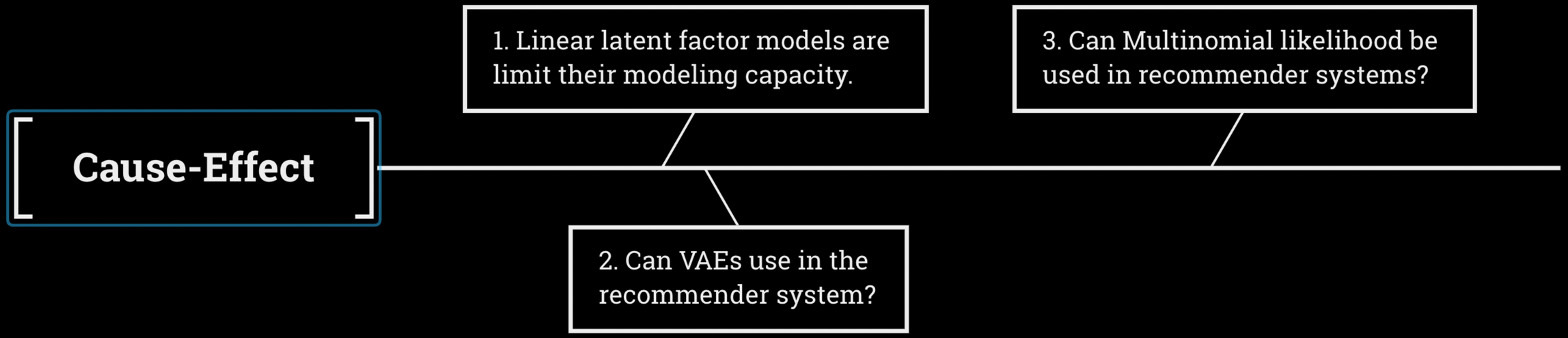
01 Motivation

02 Model Structure

03 Analysis

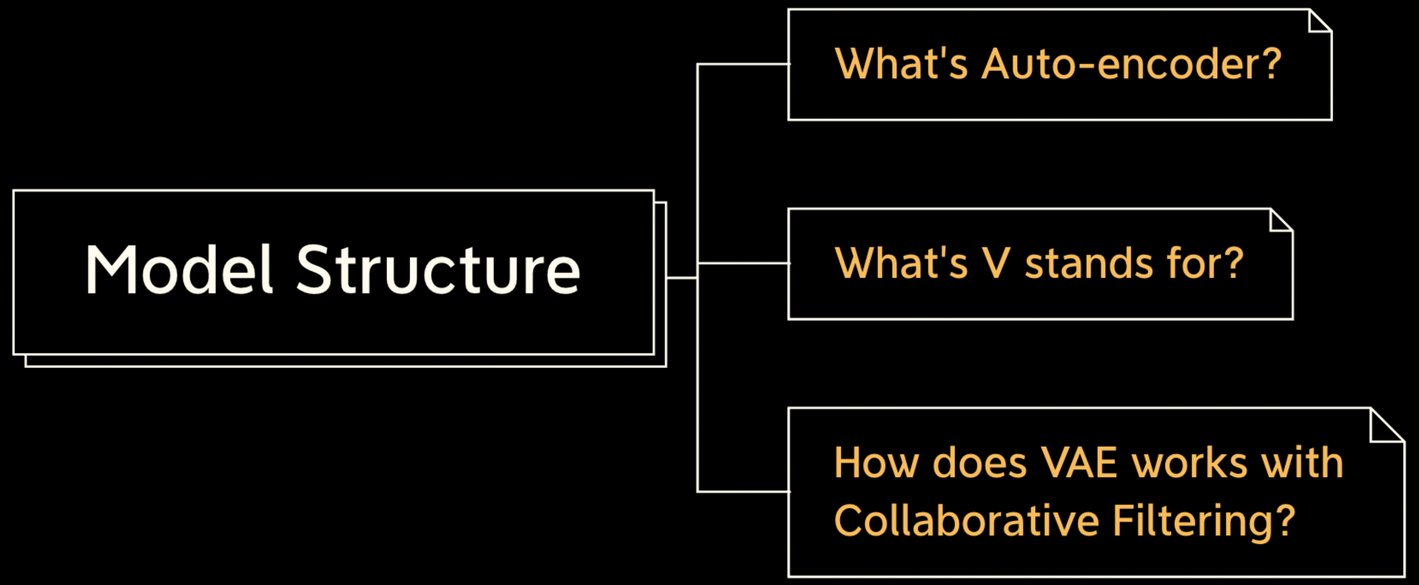
Motivation

Cause-Effect

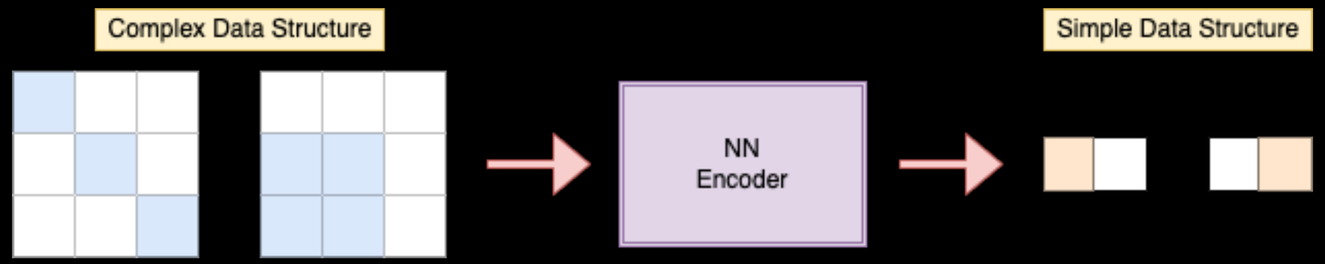


Model Structure

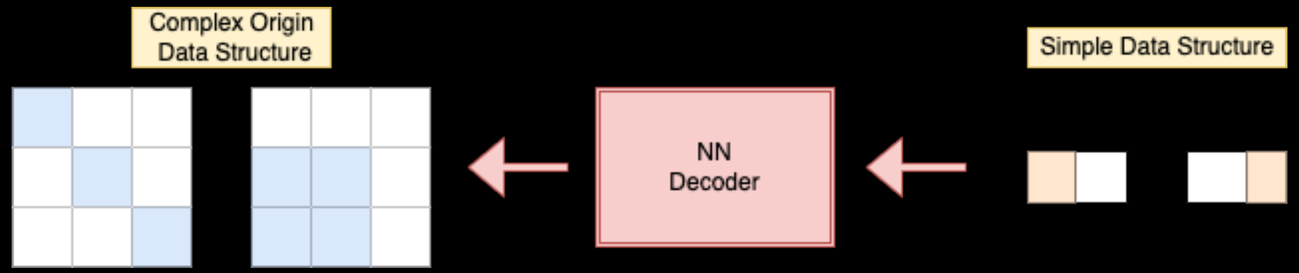
Define Questions



What is Auto-encoder?

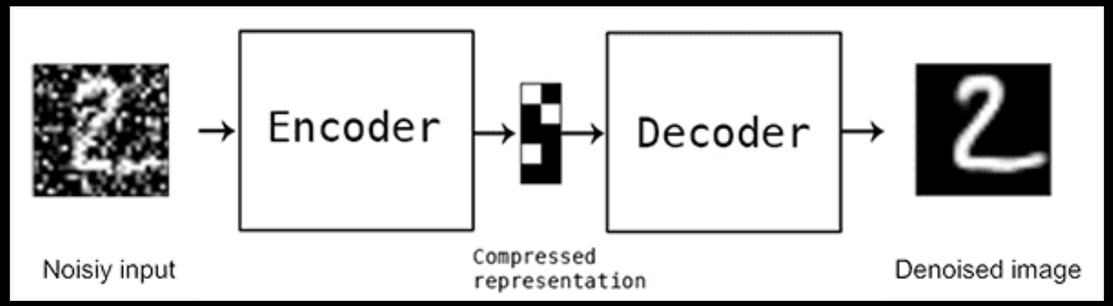


Encoder schematic



Decoder schematic

De-noising Auto-encoder (DAE)



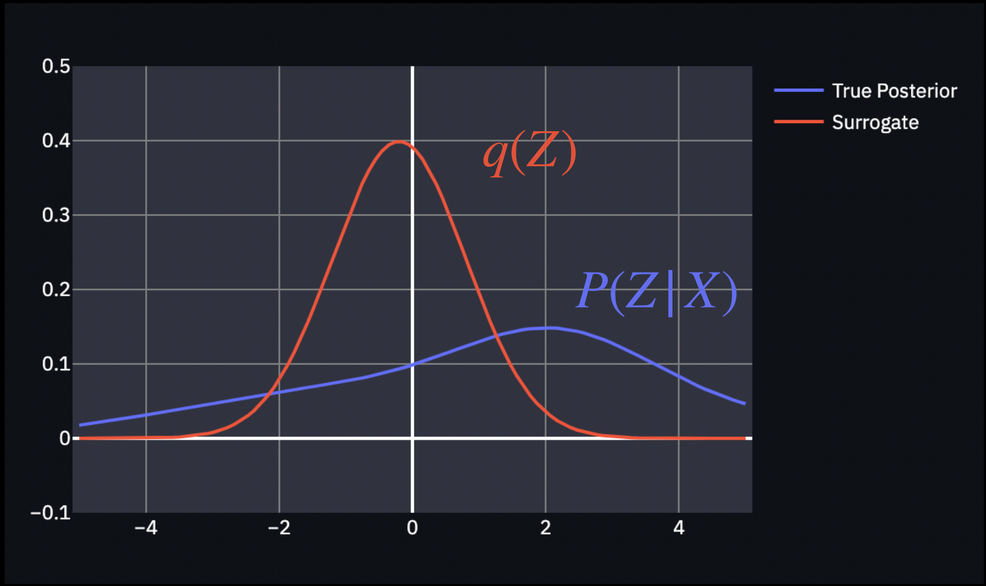
DAE schematic

- Add noises to origin data.
- Let decoder reconstruct as close as possible to origin data.
- De-noising is advocated as a train criterion for learning to extract useful features that will constitute better higher level representations of the input

What's V stand for ?

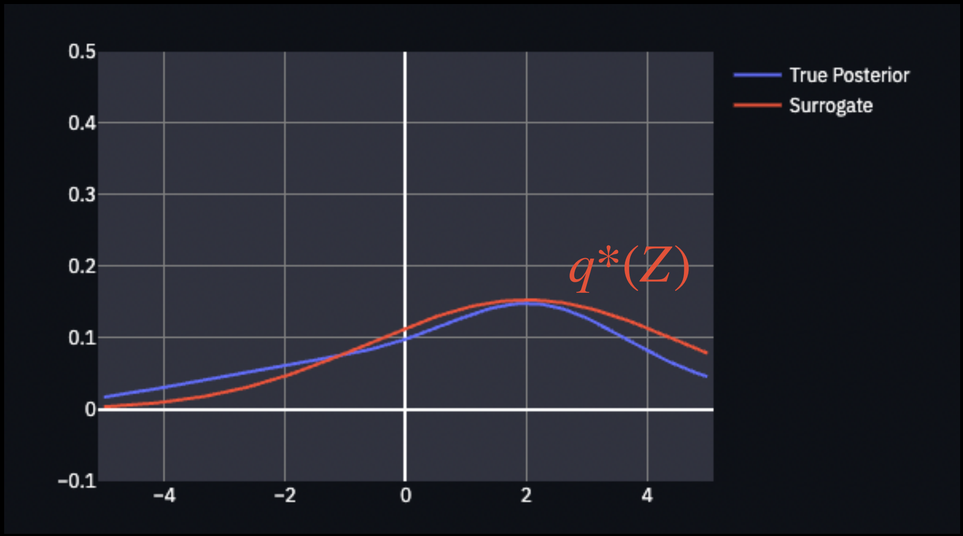
- V means Variational, which represents VAE use Variational Bayesian Inference in auto-encoder.

Variational Bayesian Inference



- Z: Latent variable.
- X: Observed data.
- $q(Z)$: Surrogate Variational Distribution(any).
- P(X): Marginal distribution of X.
- $P(Z|X)$: Posterior distribution observed from observed data X.
- Variational Inference want to find $Q(Z) \approx P(Z|X)$ as good as possible. (optimize the $Q(Z)$ to fit the given posterior distribution $P(Z|X)$)

How to measure the goodness of the fit?



- KL-Divergence: Finding a distance between two distributions.
- Our task is to minimize the KL-Divergence Value.

$$q^*(Z) = \underset{q(z)\in Q}{\arg\min}\, KL\big(q(Z)\,||\,P(Z|X)\big)$$

Deriving **Evidence Lower Bound (ELBO)**

$$\boxed{KL(q(Z)\,||\,P(Z|X))} = \int q(Z)\cdot\log\left(\frac{q(Z)}{p(Z|X)}\right)dZ$$

KLD

$$= \int q(Z)\cdot\log\left(\frac{q(Z)\cdot P(X)}{P(Z,X)}\right)\cdot dZ \qquad P(Z|X) = \frac{P(Z,D)}{P(D)}$$

$$= \int q(Z)\cdot\log\left(\frac{q(Z)}{P(Z,X)}\right)\cdot dZ + \int q(Z)\cdot log(P(X))\cdot dZ$$

$$= \mathbb{E}_{z\sim q(Z)}[\log\left(\frac{q(Z)}{P(Z,X)}\right)] + \mathbb{E}_{z\sim q(Z)}[logP(X)]$$

$$= -\,\mathbb{E}_{z\sim q(Z)}[\log\left(\frac{P(Z,X)}{q(Z)}\right)] + \log P(X)$$

$$\equiv -\boxed{\mathscr{L}(Q)} + \boxed{\log P(X)}$$

ELBO Evidence

Learning VAEs

$$\mathscr{L}(Q) \equiv -KL\big(q(Z)\,||\,P(Z|X)\big) + \log P(X)$$

ELBO — KLD — Evidence

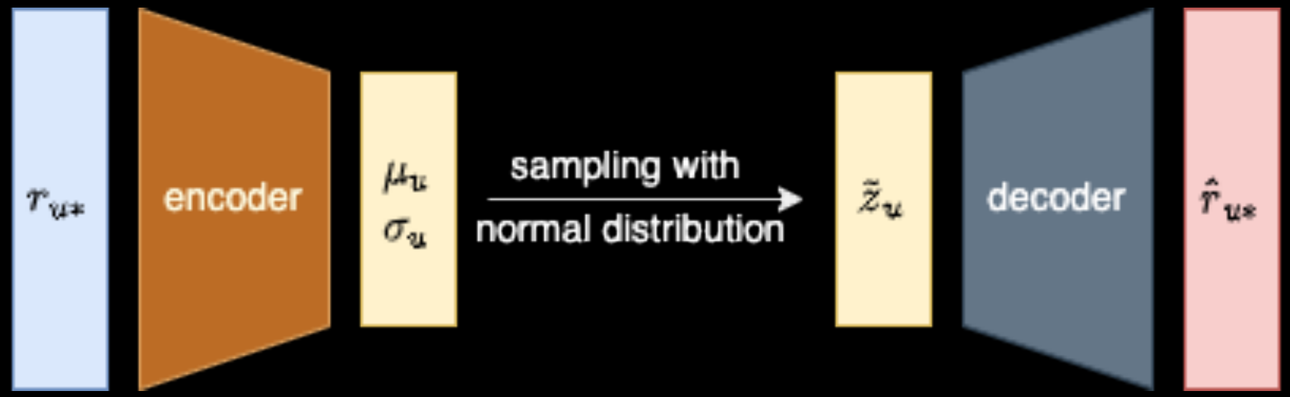
- It follows that minimizing KLD is equivalent to maximizing the ELBO.
- The quantity $\mathscr{L}(Q)$ can be taken as a learning objective in ML tasks and architectures involving distribution approximation.

$$\begin{aligned} loss &\equiv -\mathscr{L}(Q) \\ &\equiv KL\big(q(Z)\;||\;P(Z|X)\big) - \log P(X) \end{aligned}$$

Learning VAEs

We use Monte Carlo (sampling) method. In VAE, the sample num is 1.

$$\mathscr{L}(X;\theta,\phi) \equiv \mathbb{E}_{q_\phi(Z|X)}[\log p_\theta(X|Z)] - KL\big(q_\phi(Z|X) \,||\, p(Z)\big)$$
$$\leq \log p(X;\theta)$$



Reparameterize

q_ϕ Encoder: Learns to compress (reduce) the input data into an encoded representation.

p_θ Decoder: Learns to reconstruct the original data from the encoded representation to be as close to the original input as possible.

Choose likelihood

$$\mathcal{L}(X;\theta,\phi) \equiv \mathbb{E}_{q_\phi(Z|X)}[\log p_\theta(X|Z)] - KL(q_\phi(Z|X)||p(Z))$$

Multinomial log-likelihood

$$\log p_\theta(X|Z) = \int X \log \pi(Z)$$

Gaussian log-likelihood

$$\log p_\theta(X|Z) = -\int \frac{c}{2}(X-f)^2$$

Logistic log-likelihood

$$\log p_\theta(X|Z) = \int X \log \sigma(f) + (1-X)\log(1-\sigma(f))$$

We'll compare multinomial likelihood with Gaussian and logistic in Analysis.

How does VAE works with Collaborative Filtering?

	userId	itemId	rating
0	196	242	3.0
1	186	302	3.0
2	22	377	1.0
3	244	51	2.0
4	166	346	1.0
...	...	...	...
99995	880	476	3.0
99996	716	204	5.0
99997	276	1090	1.0
99998	13	225	2.0
99999	12	203	3.0

itemId	1	10	100	1000	1001	1002	1003	1004	1005	1006	...	990	991
userId													
1	5.0	3.0	5.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0
10	4.0	0.0	5.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0
100	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	3.0	0.0
101	3.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0
102	3.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
95	5.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0
96	5.0	0.0	5.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0
97	4.0	0.0	2.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0
98	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0
99	4.0	0.0	5.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0

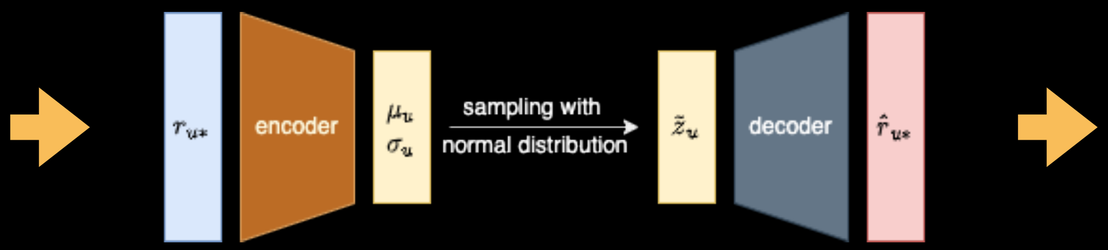
User rating history

User-Item Interaction Matrix
(Which can be seem as vector)

How does VAE works with Collaborative Filtering?

itemId	1	10	100	1000	1001	1002	1003	1004	1005	1006	...	990	991
userId													
1	5.0	3.0	5.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0
10	4.0	0.0	5.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0
100	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	3.0	0.0
101	3.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0
102	3.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
95	5.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0
96	5.0	0.0	5.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0
97	4.0	0.0	2.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0
98	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0
99	4.0	0.0	5.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0

User-Item Interaction Matrix
(Which can be seen as vectors)



Compresses & reconstructs
with **VAE structure**

	userId	itemId	rating
100000	196	302	0.003425
100001	196	377	0.000046
100002	196	51	0.000406
100003	196	346	0.000952
100004	196	474	0.002254
...	...	...	...
1586121	941	1674	0.000003
1586122	941	1640	0.000004
1586123	941	1637	0.000005
1586124	941	1630	0.000006
1586125	941	1641	0.000004

Get new rating $\hat{r}$

Work with Information Bottleneck method

$$\text{distortion} = \min_{p(t|x)} I(X,T) - \beta I(T,Y)$$

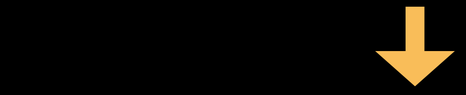
- I is the mutual information between two variable.
- X is input variable.
- Y is observed real variable.
- T is a random variable compressed representation, which can be seen as variable in hidden layer.

Information Bottleneck is designed for finding the best tradeoff between accuracy and complexity.

- The algorithm minimizes the function value with $p(t\,|\,x)$.
- Conjectured that the training process of a DNN consists of two phases:
  1. A initial fitting phase in which $I(T, Y)$ increases. (Decoder)
  2. A compression phase in which $I(X, T)$ decreases. (Encoder)
- β is a Lagrange multiplier, which relaxing the condition to capture some fraction of the mutual information

Work with Information Bottleneck method

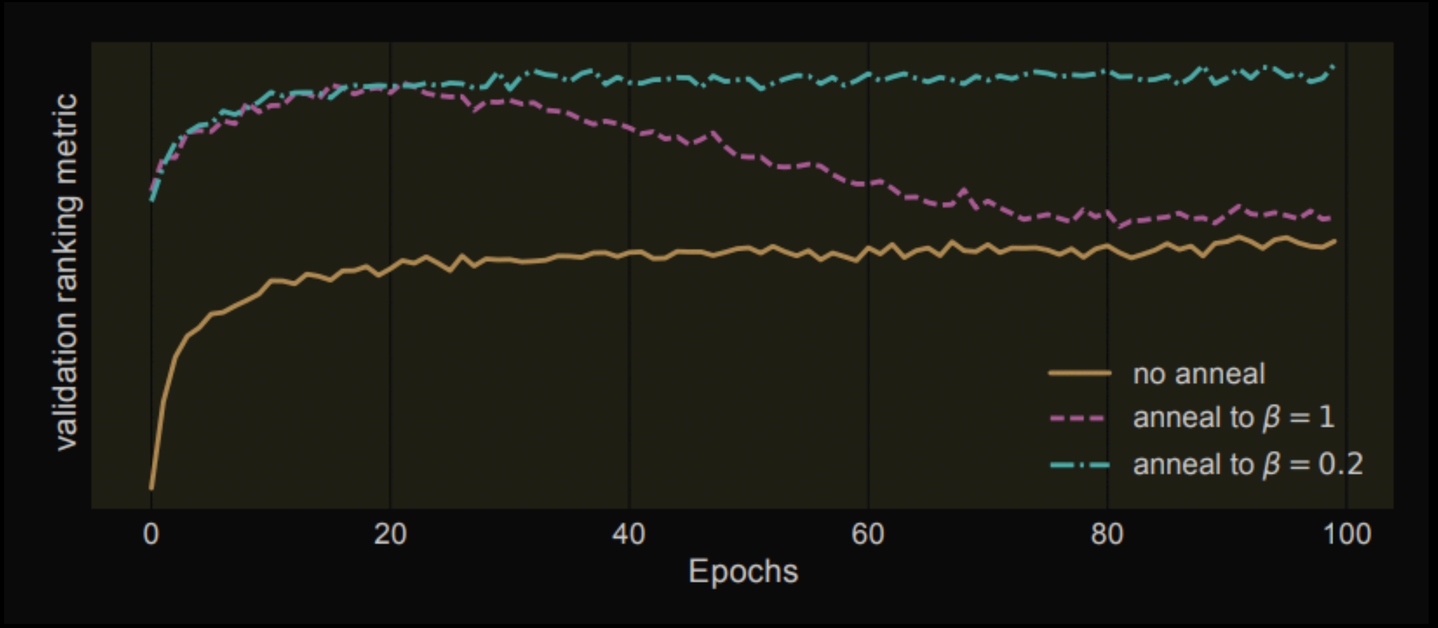
$$\mathscr{L}(X;\theta,\phi) \equiv \mathbb{E}_{q_\phi(Z|X)}[\log p_\theta(X|Z)] - KL(q_\phi(Z|X)||p(Z))$$



$$\mathscr{L}(X;\theta,\phi) \equiv \mathbb{E}_{q_\phi(Z|X)}[\log p_\theta(X|Z)] - \underline{\beta} \cdot KL(q_\phi(Z|X)||p(Z))$$

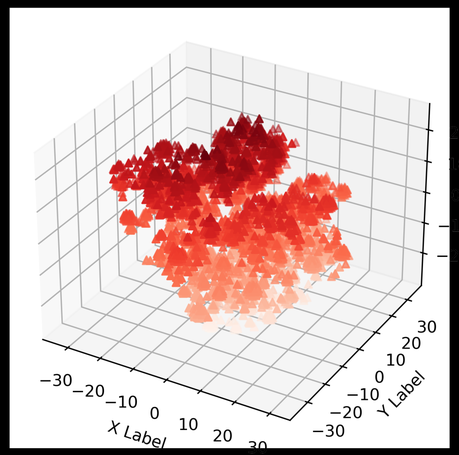
If $\beta < 1$, will weaken the influence of the prior constraint, the model is less able to generate novel user histories by ancestral sampling.

Work with Information Bottleneck method

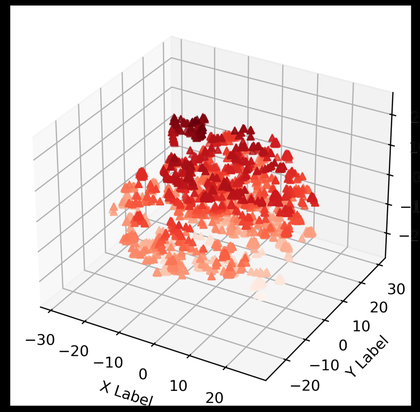


- start training with $\beta = 0$, and gradually increase β to 1. (red-dashed, anneal to $\beta = 1$).
- linearly anneal the KL term slowly over a large number of gradient updates to θ, ϕ.

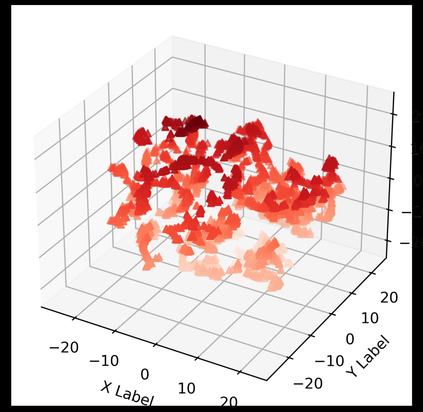
The influence of Beta



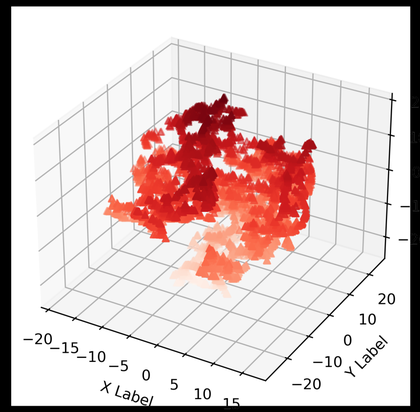
Origin Distribution



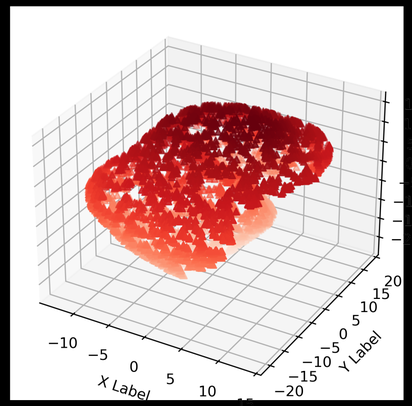
$\beta = 0$



$\beta = 0.2$

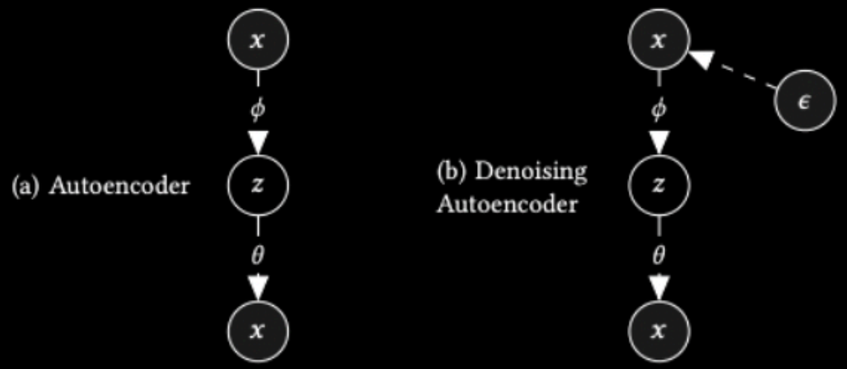


$\beta = 0.5$



$\beta = 1$

A taxonomy of auto encoders



$$\theta^{AE}, \phi^{AE} = \arg max_{\theta,\phi} \sum \log p_\theta(x \mid g_\phi(x))$$

- g_ϕ is encoder function without prior distribution.
- p_θ is decoder as VAE.

- The auto-encoder and the de-noising auto-encoder don't regularize as VAE does.
- Contrast this to the VAE, where the learning is done using a variational distribution.
- Learning auto-encoders is extremely prone to overfitting as the network learns to put all the probability mass to the non-zero entries.
- By introducing dropout at the input layer, the DAE is less prone to overfitting and we find that it also gives competitive empirical results.

Datasets

	ML-20M	Netflix	MSD
# of users	136,677	463,435	571,355
# of items	20,108	17,769	41,140
# of interactions	10.0M	56.9M	33.6M
% of interactions	0.36%	0.69%	0.14%
# of held-out users	10,000	40,000	50,000

- ML-20M : **Movielens** is a Non-commercial, personalized movie web-based recommendations system.
- Netflix : **Netflix Movies and TV Shows.**
- MSD : The **Million Song Dataset** is a freely-available collection of audio features and metadata for a million contemporary popular music tracks.

Select Likihood

	Recall@20	Recall@50	NDCG@100
Mult-VAE^{PR}	**0.395**	**0.537**	**0.426**
Gaussian-VAE^{PR}	0.383	0.523	0.415
Logistic-VAE^{PR}	0.388	0.523	0.419
Mult-DAE	**0.387**	**0.524**	**0.419**
Gaussian-DAE	0.376	0.515	0.409
Logistic-DAE	0.381	0.516	0.414

The multinomial likelihood performs better than the other likelihoods on MovieLens Dataset.

Mult-VAE Mult-DAE

Model scores

(a) ML-20M

	Recall@20	Recall@50	NDCG@100
Mult-VAE^{PR}	**0.395**	**0.537**	**0.426**
Mult-DAE	0.387	0.524	0.419
WMF	0.360	0.498	0.386
SLIM	0.370	0.495	0.401
CDAE	0.391	0.523	0.418

(b) Netflix

	Recall@20	Recall@50	NDCG@100
Mult-VAE^{PR}	**0.351**	**0.444**	**0.386**
Mult-DAE	0.344	0.438	0.380
WMF	0.316	0.404	0.351
SLIM	0.347	0.428	0.379
CDAE	0.343	0.428	0.376

(c) MSD

	Recall@20	Recall@50	NDCG@100
Mult-VAE^{PR}	**0.266**	**0.364**	**0.316**
Mult-DAE	**0.266**	**0.363**	0.313
WMF	0.211	0.312	0.257
SLIM	—	—	—
CDAE	0.188	0.283	0.237

WMF	A linear low-rank factorization model.	Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative filtering for implicit feedback datasets. In Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on. 263–272.
SLIM	A linear model which learns a sparse item-to-item similarity matrix by solving a constrained $\ell 1$-regularized optimization problem	Xia Ning and George Karypis. 2011. Slim: Sparse linear methods for top-n recommender systems. In Data Mining (ICDM), 2011 IEEE 11th International Conference on. 497–506.
CADE	Augments the standard denoising autoencoder by adding a per-user latent factor to the input.	Yao Wu, Christopher DuBois, Alice X. Zheng, and Martin Ester. 2016. Collaborative denoising auto-encoders for top-n recommender systems. In Proceedings of the Ninth ACM International Conference on Web Search and Data Mining.

Compare objects

Model scores

(a) ML-1M

	NCF	NCF (pre-train)	Mult-DAE
Recall@10	0.705	**0.730**	0.722
NDCG@10	0.426	**0.447**	0.446

(b) Pinterest

	NCF	NCF (pre-train)	Mult-DAE
Recall@10	0.872	0.880	**0.886**
NDCG@10	0.551	0.558	**0.580**

- Mult-DAE model significantly outperforms NCF without pre-training on both datasets
- And further improves on Pinterest even comparing with pre-trained NCF

Model scores - Experiments on Amazon Clothing Dataset

VaeCF

	NDCG@50	Recall@50	Train (s)	Test (s)
VAECF	0.0525	0.1541	28.0891	1.1412

NCF

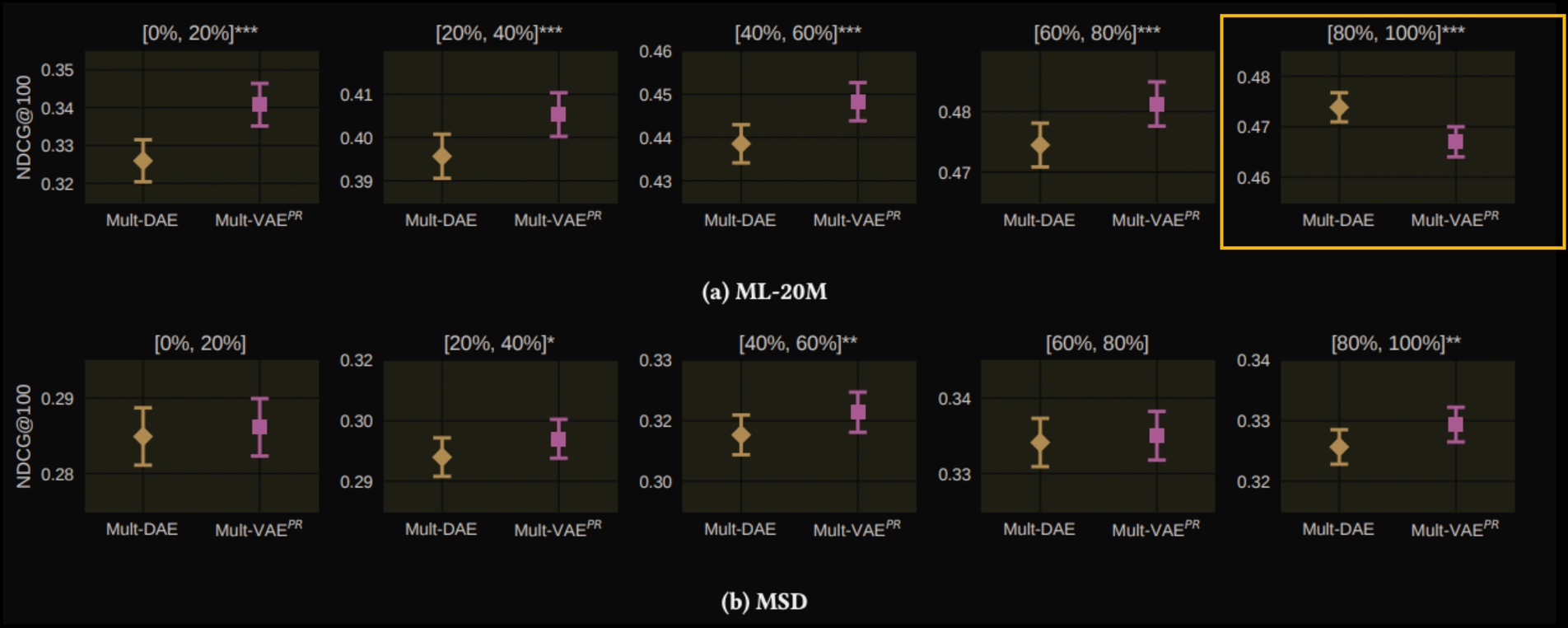
	NDCG@50	Recall@50	Train (s)	Test (s)
GMF	0.0395	0.1142	81.4996	1.1415
MLP	0.0393	0.1201	84.0161	2.3517
NeuMF	0.0393	0.1196	83.1110	2.6415
NeuMF_pretrained	0.0410	0.1213	84.9208	2.6554

Compare

	NDCG@50	Recall@50	Train (s)	Test (s)
VAECF	0.0525	0.1541	28.0891	1.1412
NeuMF	0.0410	0.1213	84.9208	2.6554

NCF vs Mult-VAECF (personal experiment)

When does Mult-VAE perform better/worse than Mult-DAE?



This indicates the stronger prior assumption could potentially hurt the performance when a lot of data is available for a user.

SWOT

Strengths • Variational inference can have better performance in the case of fewer data and the case of data generation. • The concept of compressing and extracting information, which can produce better results depending on the situation. • In addition, through the Beta value, the model doesn't have to rely entirely on prior knowledge. • Compared to GAN, VAEs requires fewer data and is easier to train.	Weaknesses • More similar with original data is not better for data generation. • When the amount of data is large enough, the effect of VAE may not be better than DAE.
Opportunities • Any situation that can combined with Variational Inference. • May be used for Cold-start Problems. • May be used for Unsupervised Task.	Threats • …

Additional

VAECF 模型訓練資料

	userId	itemId	rating
0	196	242	3.0
1	186	302	3.0
2	22	377	1.0
3	244	51	2.0
4	166	346	1.0
...	...	...	...
99995	880	476	3.0
99996	716	204	5.0
99997	276	1090	1.0
99998	13	225	2.0
99999	12	203	3.0

圖三、用戶(userId)對基金(itemId)的評分(rating)資料

- 與 Matrix Factorization 相同，VAECF所需要的訓練資料來自各別用戶對於商品的評分
- 在目前基金推薦系統中，針對用戶的交易資料作了以下處理：
  - 1. 將身分證字號轉換為 User Index
  - 2. 將商品編號轉換為 Item Index
  - 3. 將購買次數設定為 rating 值
- 所得用戶商品評分資料如左圖所示

VAECF 運行流程(1)

將用戶購買紀錄展開成用戶與商品的關係矩陣(商品數量 × 用戶數量)

	userId	itemId	rating
0	196	242	3.0
1	186	302	3.0
2	22	377	1.0
3	244	51	2.0
4	166	346	1.0
...	...	...	...
99995	880	476	3.0
99996	716	204	5.0
99997	276	1090	1.0
99998	13	225	2.0
99999	12	203	3.0

商品總數

用戶數量

itemId	1	10	100	1000	1001	1002	1003	1004	1005	1006	...	990	991
userId													
1	5.0	3.0	5.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0
10	4.0	0.0	5.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0
100	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	3.0	0.0
101	3.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0
102	3.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
95	5.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0
96	5.0	0.0	5.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0
97	4.0	0.0	2.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0
98	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0
99	4.0	0.0	5.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0

User-Item Interaction Matrix

VAECF 運行流程(2)

每一次 forward 時，取出個別用戶的ROW ，作為模型的輸入資料，如下圖所示：

itemId	1	10	100	1000	1001	1002	1003	1004	1005	1006	...	990	991
userId													
1	5.0	3.0	5.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0
10	4.0	0.0	5.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0
100	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	3.0	0.0
101	3.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0
102	3.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
95	5.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0
96	5.0	0.0	5.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0
97	4.0	0.0	2.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0
98	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0
99	4.0	0.0	5.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0

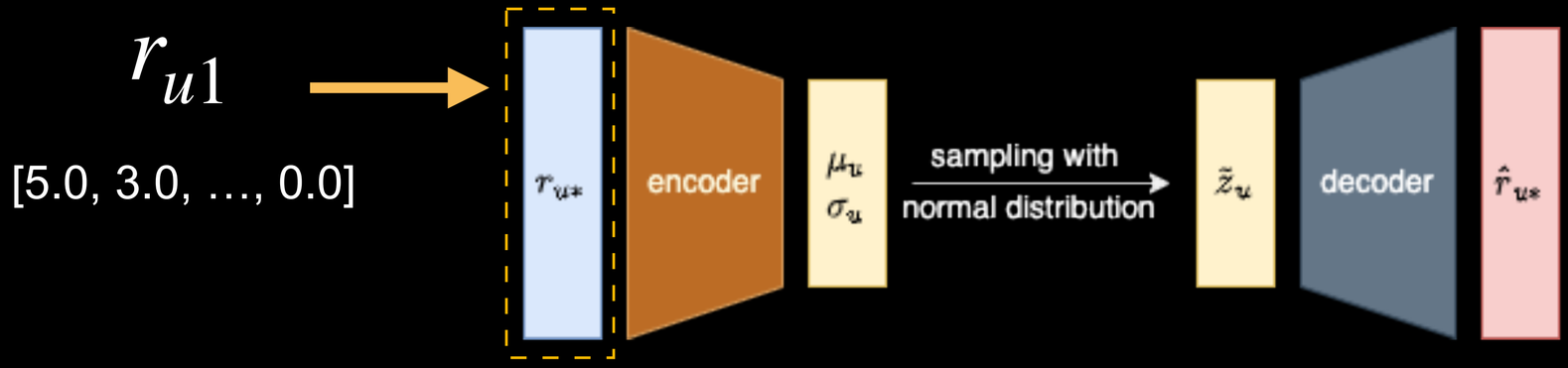
5.0	3.0	5.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0

shape 為 1 × 商品數量，
向量的每個位置分別代表該用戶對不同商品的評分

為方便表示將此範例輸入 r_{u1} 簡化為 [5.0, 3.0, …, 0.0]

VAECF 運行流程(Training)

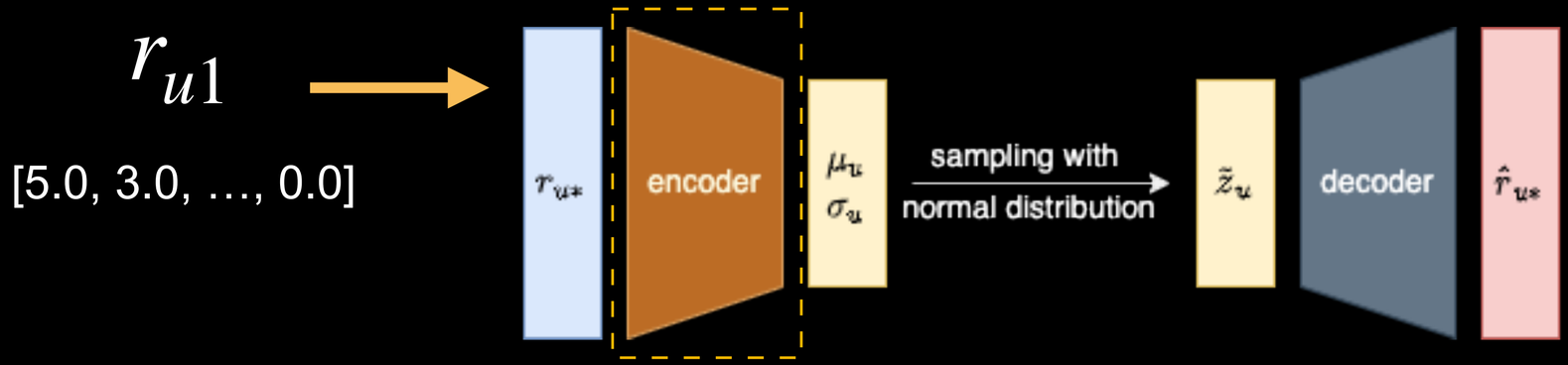
在訓練過程中，每一個epoch會遍歷所有user，輸入 r_{u*} 經VAE模型輸出 $\hat{r}_{u*}$



Step 1. 輸入 r_{u1}

VAECF 運行流程(Training)

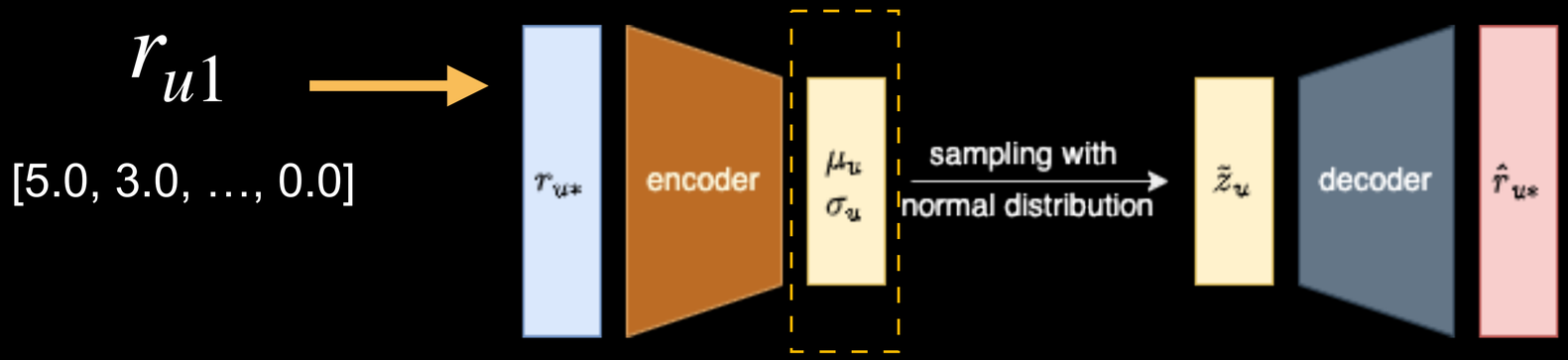
在訓練過程中，每一個epoch會遍歷所有user，輸入 r_{u*} 經VAE模型輸出 $\hat{r}_{u*}$



Step 2. Encoder 輸出 hidden factor h

VAECF 運行流程(Training)

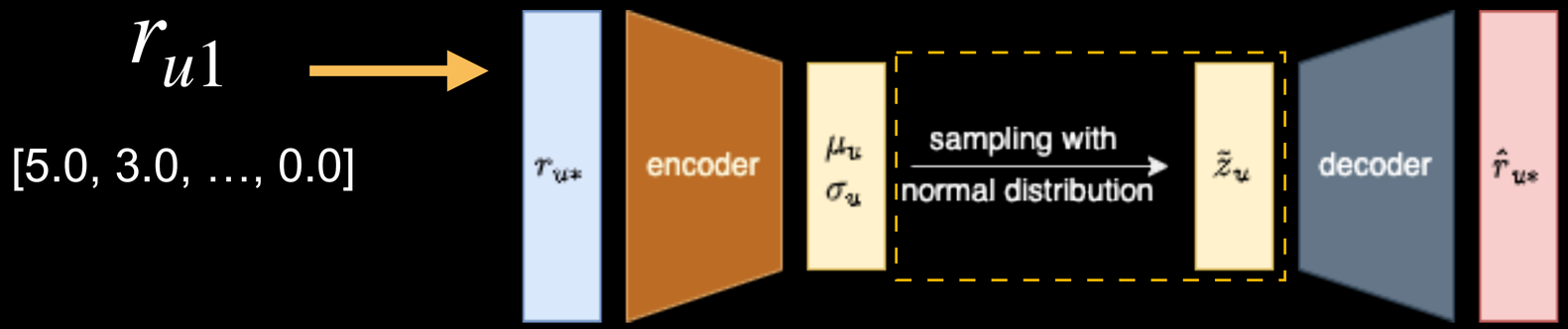
在訓練過程中，每一個epoch會遍歷所有user，輸入 r_{u*} 經VAE模型輸出 $\hat{r}_{u*}$



Step 3. 模型由 h 中取出中位數μ_u與變異數σ_u

VAECF 運行流程(Training)

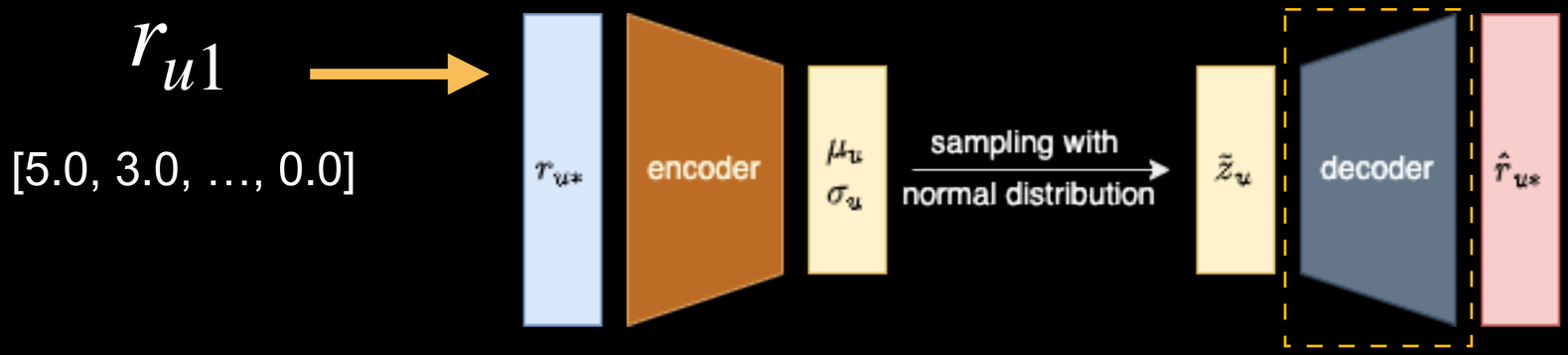
在訓練過程中，每一個epoch會遍歷所有user，輸入 r_{u*} 經VAE模型輸出 $\hat{r}_{u*}$



Step 4. 透過中位數μ_u與變異數σ_u 產生服從常態分佈的機率分佈 $\tilde{z}_u$
(此動作稱作reparameterize)

VAECF 運行流程(Training)

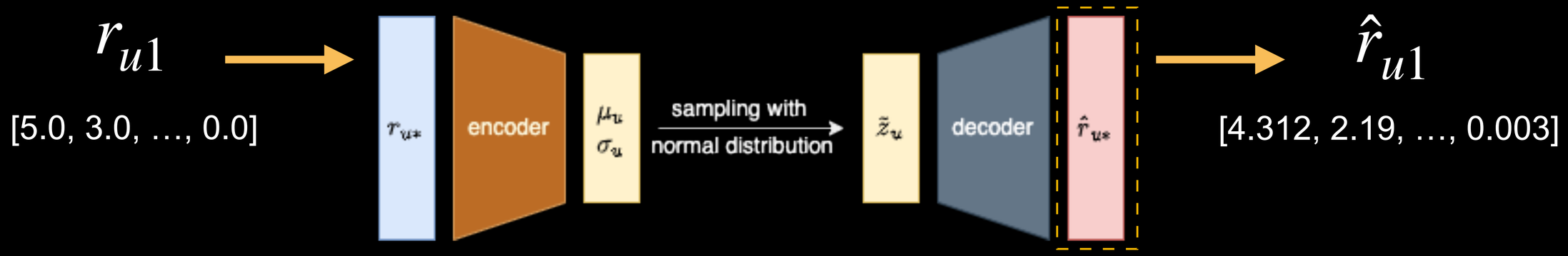
在訓練過程中，每一個epoch會遍歷所有user，輸入 r_{u*} 經VAE模型輸出 $\hat{r}_{u*}$



Step 5. 訓練 Decoder 盡可能將分佈 $\tilde{z}_u$ 還原回原始 Input

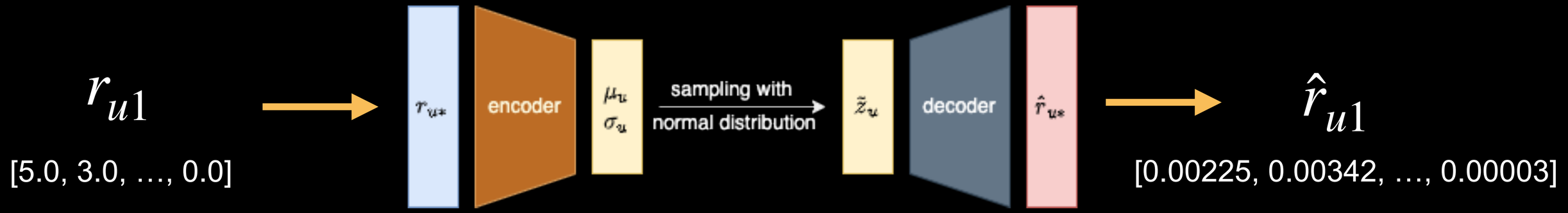
VAECF 運行流程(Training)

在訓練過程中，每一個epoch會遍歷所有user，輸入 r_{u*} 經VAE模型輸出 $\hat{r}_{u*}$



Step 6. Decoder 輸出結果 $\hat{r}_u$

VAECF 運行流程(Inference)



訓練完成後，將用戶的 Input 放到模型中產生出來的 Ouput 即 VAE 模型預測該用戶的新評分

VAECF 運行流程(Inference)

$\hat{r}_{u1}$

[0.00225, 0.00342, …, 0.00003]

	userId	itemId	rating
100000	196	302	0.003425
100001	196	377	0.000046
100002	196	51	0.000406
100003	196	346	0.000952
100004	196	474	0.002254
...	...	...	...

將取得的結果$\hat{r}_{u1}$ 轉換回用戶對基金的評分資料，依據分數的高低進行排序後推薦給用戶