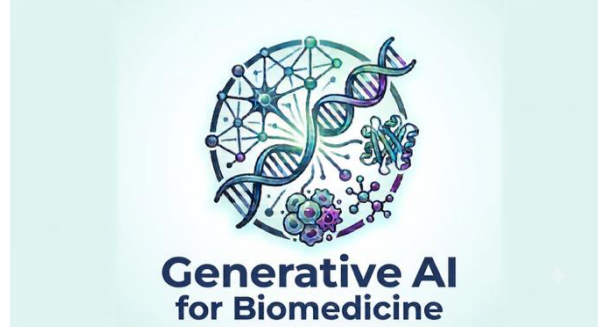




Variational Auto-Encoder

Lei Li



Language
Technologies
Institute

Carnegie Mellon University
School of Computer Science

Paper Presentation

- Please check paper presentation assignment on canvas
- Reach out to TA if not assigned

Recap

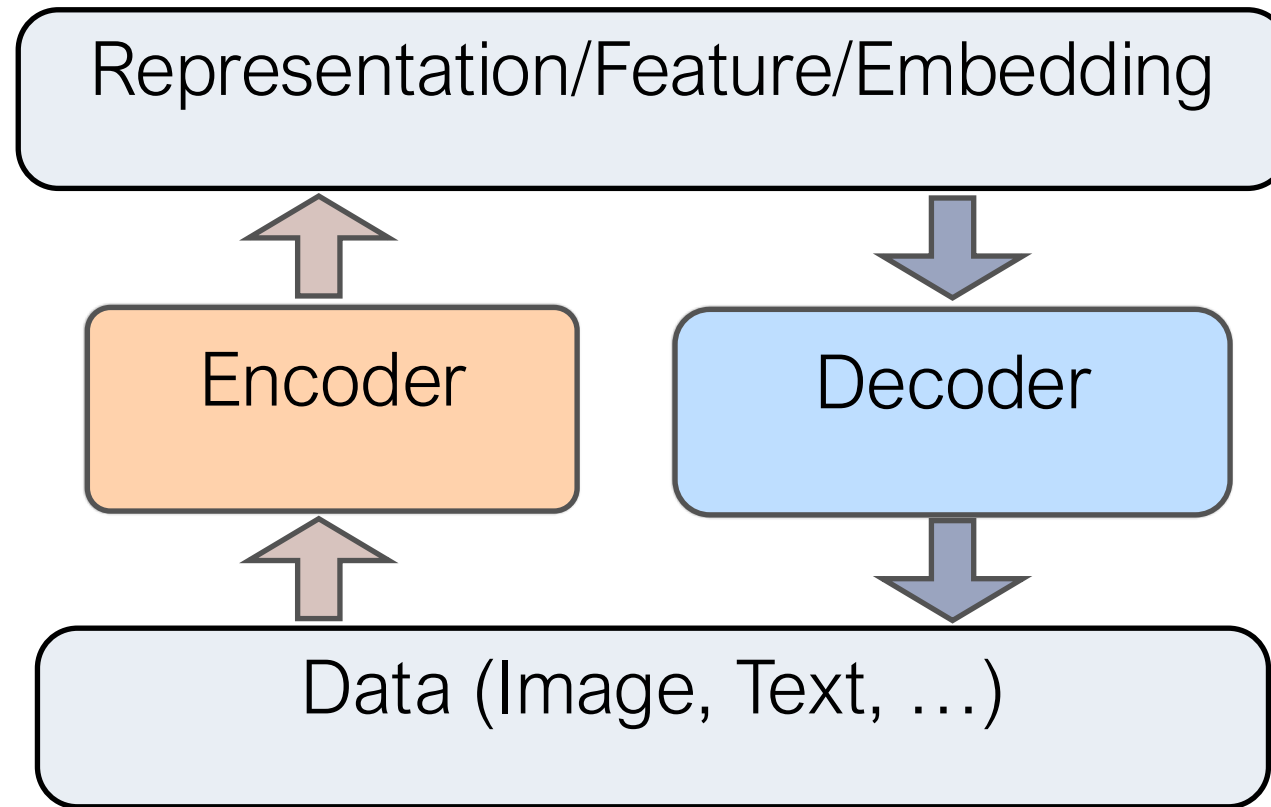
- Message Passing Neural Network
 - Nodes carry embeddings
 - message (embeddings) passed along graph edges
 - **message + aggregation + update**
 - many variants: GCN, GAT
- Equivariant Graph Neural Network
 - A special case of MPNN
 - modelling the embedding of nodes, along with 3D structures
 - rotation/translation invariant

Outline

- Auto-Encoder
- Probabilistic Graphical Model
- Variational Auto-Encoder
- VAE for Sequence data

Auto-Encoder

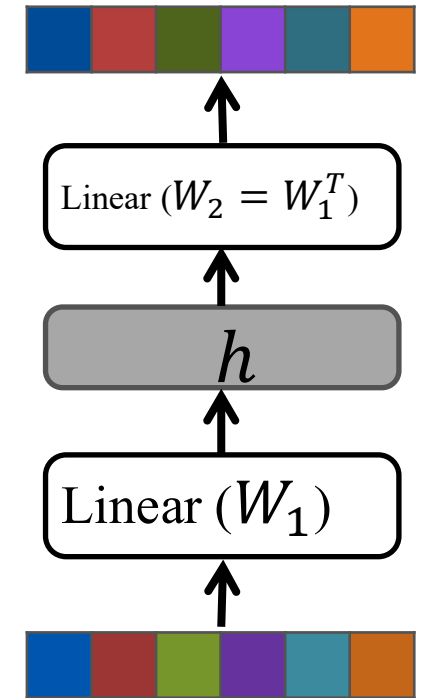
Learning representations so that data can be reconstructed!



Need to avoid learning an identity mapping

Auto-Encoder

- Simple instance:
 - encoder is linear projection
 - decoder is linear projection
 - weight tying
- Other constraints:
 - Robust construction to noise, (Denoising Auto-Encoder)
 - Hidden representations conform to a distribution (Variational Auto-Encoder)



Training Auto-Encoder

- Real-valued data (e.g. images):

- MSE loss

- $\text{MSE} = \|f(x) - x\|$

Learning linear auto-encoder

$$\text{MSE} = \|X \cdot W \cdot W^T - X\| + \lambda |W|_1$$

- Categorical data (e.g. Sentences)

- Cross-Entropy

- $CE = -\sum x_i \log f(x)_i$

Undercomplete Representation

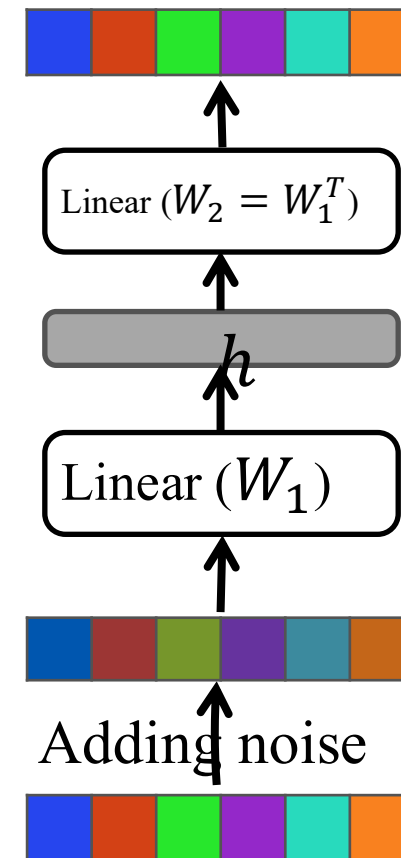
- If hidden representation is smaller than input data dimension
- Dimensionality reduction
- e.g. Principal Component Analysis (PCA)

Overcomplete Representation

- If hidden representation is larger than input data dimension
- Need to enforce additional constraints e.g. sparsity
 - $loss = CE + \lambda |W|_1$

Denoising Auto-Encoder

- Encoder takes a noised data to produce the hidden representation
- Decoder need to recover original data from hidden representation
- make the auto-encoder robust



Variational Auto-Encoder (VAE)

- Input, intermediate and output values are no longer deterministic vectors, but are probabilistic variables
- Using Probabilistic Graphical Model (PRG) to represent dependencies

Outline

- Auto-Encoder
- ➔ • Probabilistic Graphical Model
- Variational Auto-Encoder
- VAE for Sequence data

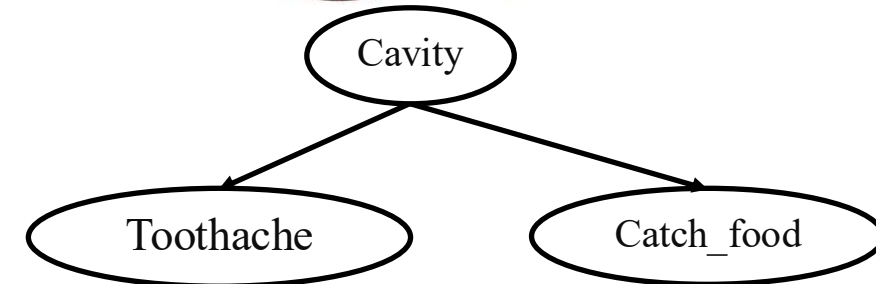
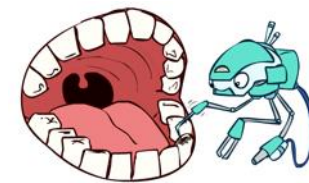
Probabilistic Graphical Models

- Models describe how the world works
- Models are always simplifications
 - May not account for every variable
 - May not account for all interactions between variables
- What do we do with probabilistic models?
 - We (or our agents) need to reason about unknown variables, given evidence
 - Example: explanation (diagnostic reasoning)
 - Example: prediction (causal reasoning)
 - Example: value of information



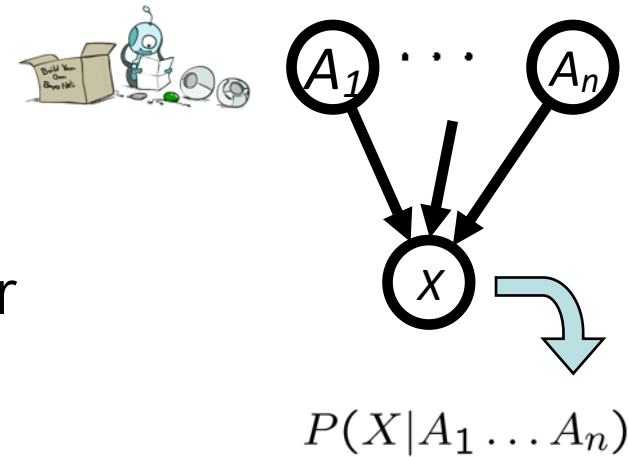
Directed PGM -- Bayesian Networks

- One type of PGM with directed graph
- Nodes: random variables (with domains)
 - Can be assigned (observed) or unassigned (unobserved)
- Arcs: interactions
 - Indicate “direct influence” between variables
 - Formally: encode conditional independence (more later)
- For now: imagine that arrows mean direct causation (in general, they don't!)



Bayesian network Semantics

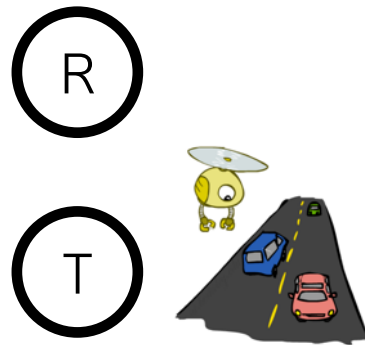
- A set of nodes, one per variable X
- A directed, acyclic graph
- A conditional distribution for each node
 - A collection of distributions over X , one for each combination of parents' values
$$P(X|a_1 \dots a_n)$$
 - CPT: conditional probability table
 - Description of a noisy “causal” process
- Directed graphical models



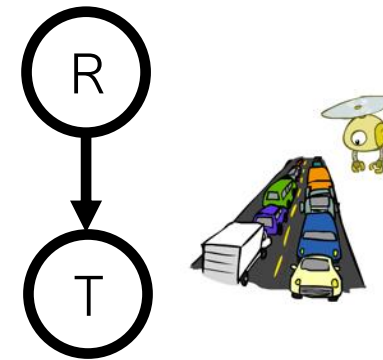
Bayesian network = Topology (graph)
+ Local Conditional Probabilities

Example: Rain and Traffic

- Variables:
 - R: It rains (=1); T: There is traffic (=1)
- Model 1: independence



Model 2: rain causes traffic



$P(R)$

+r	1/4
-r	3/4

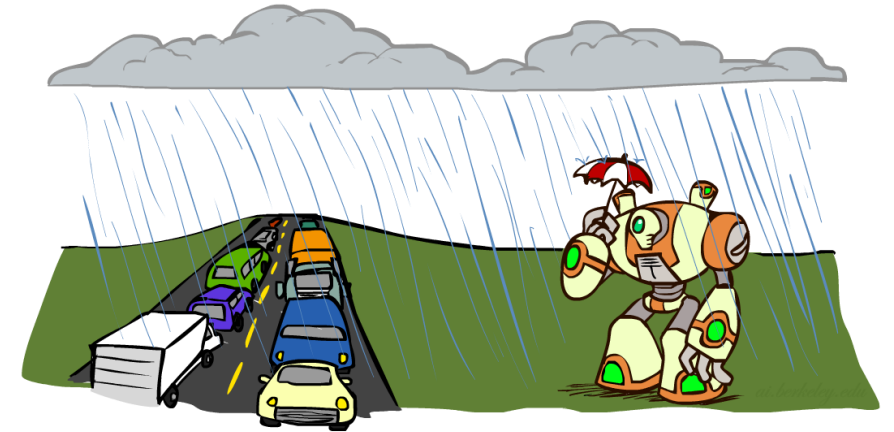
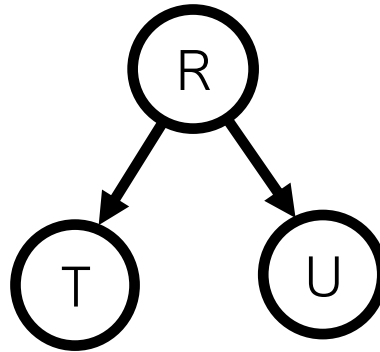
$P(T|R)$

+r	+t	3/4
+r	-t	1/4
-r	+t	1/2
-r	-t	1/2

- Why is model 2 better?

Example: Traffic II

- Variables
 - T: Traffic
 - R: It rains
 - U: Umbrella



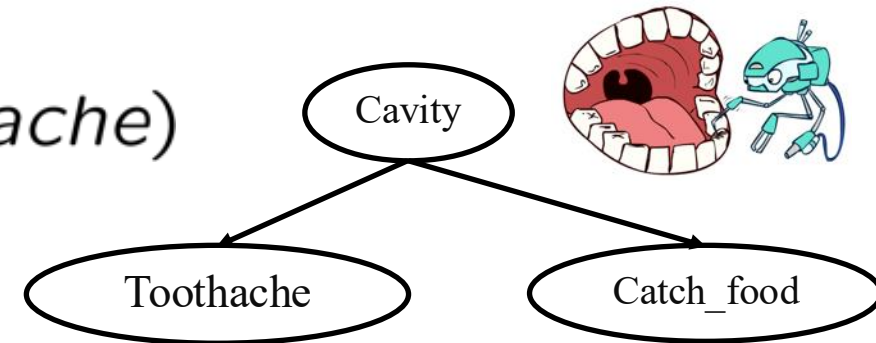
- Q: what is the prob. of rain if we observe traffic but no umbrella?
 - $P(R=1|T=1, U=0)$

Probabilities in BNs

- Bayes' nets implicitly encode the joint distribution
 - As a product of local conditional distributions

$$P(x_1, x_2, \dots, x_n) = \prod_{i=1}^n P(x_i | \text{parents}(X_i))$$

- To see what probability a BN gives to a full assignment, multiply all the relevant conditionals together
- Example: $P(+cavity, +catch, -toothache)$

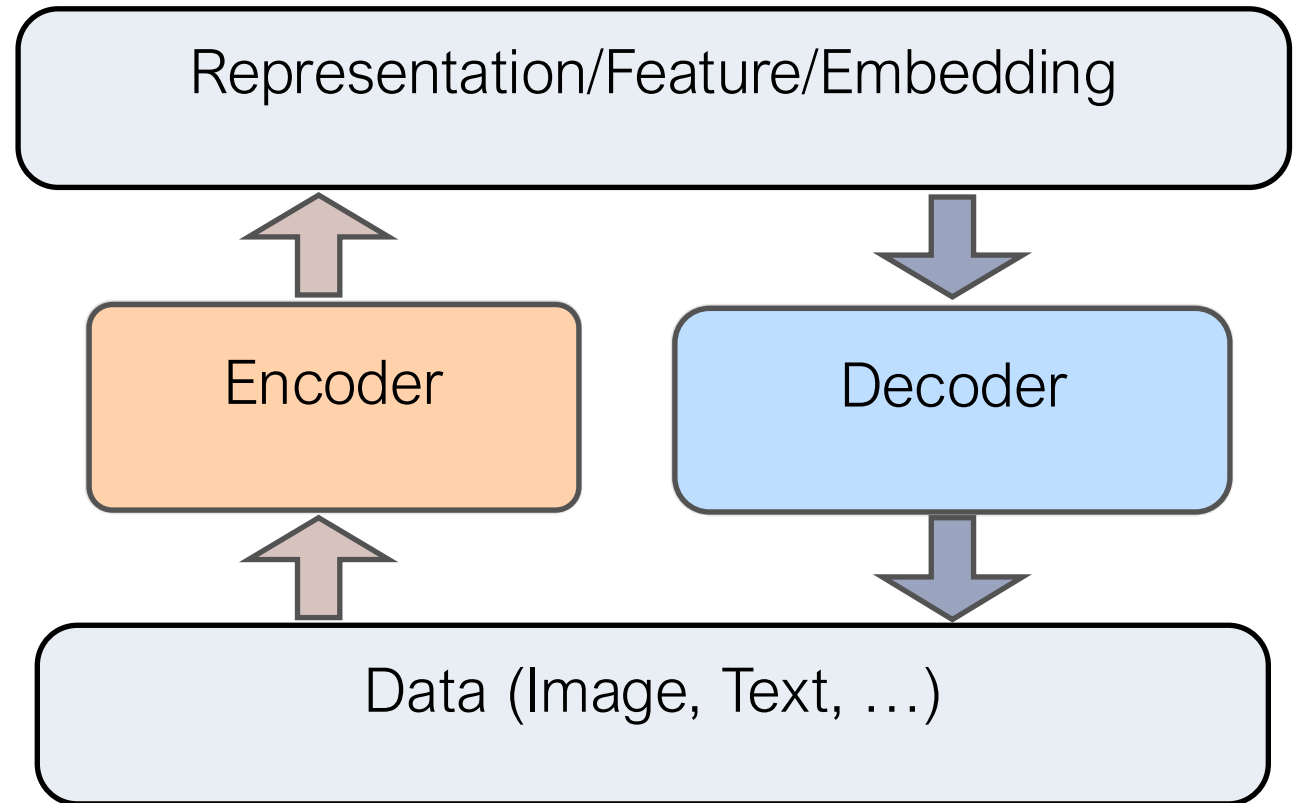


Outline

- Auto-Encoder
- Probabilistic Graphical Model
-  • Variational Auto-Encoder
- VAE for Sequence data

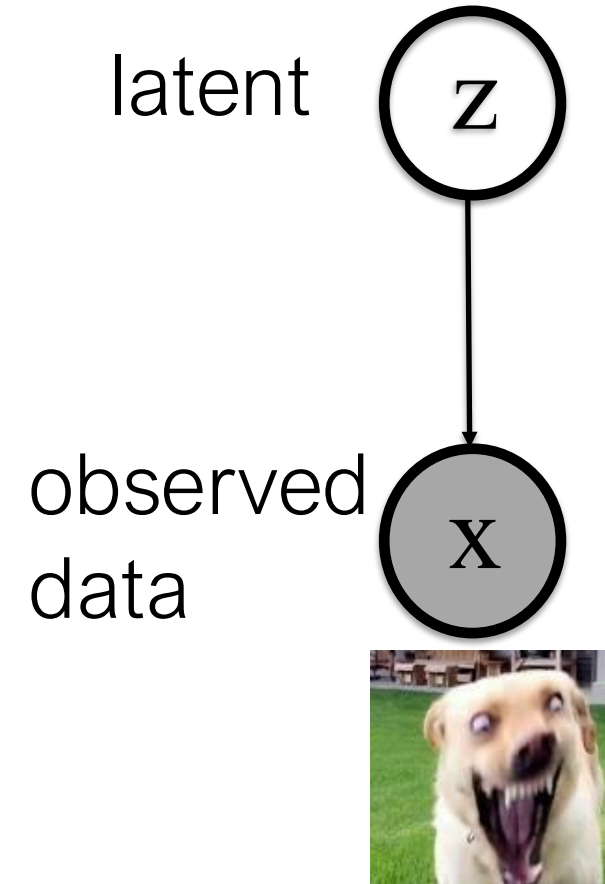
Variational Auto-Encoder

- Hidden representations follow a prior distribution
- Encoder will produce a distribution of representations (posterior distribution)



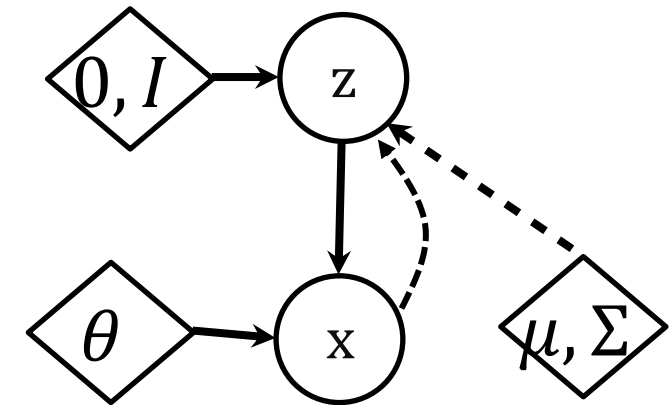
Deep Latent Model (generative model for VAE)

- z follows a prior distribution, e.g. Gaussian(0, I)
- $p(x|z)$ is defined by a deep neural network $f(z; \theta)$
- To learn θ , use $E_{(z|x)}[\log p(X, Z; \theta)]$



Probabilistic Graphical Model for VAE

- Assuming data X is generated from a latent variable Z
- Generation process
 - draw $Z \sim N(\mu, \Sigma)$
 - draw $X|Z \sim p(f(Z))$, defined by a neural network f with parameter θ



Variational Auto-Encoder

- Objective: maximize the data loglikelihood

$$\begin{aligned} \max_{\theta} \ell(\theta) &= \sum_n \log p(x_n; \theta) \\ &= \sum_n \log \int p(x_n | z_n; \theta) p(z_n; \theta) dz_n \end{aligned}$$

Hard to optimize over θ , if $f(Z)$ is very complex,
i.e. $p(x_n | z_n; \theta)$ is very complex
e.g. CNN, RNN, GNN, Transformer

Variational Posterior Distribution

$$\begin{aligned} \max_{\theta} \ell(\theta) &= \sum_n \log p(x_n; \theta) \\ &= \sum_n \log \int p(x_n | z_n; \theta) p(z_n; \theta) dz_n \end{aligned}$$

- But $\log p(x; \theta)$ is intractable.
- For any distribution $q(z|x, \phi)$:
$$\log p(x; \theta) \geq \mathbb{E}_{q(z|x; \phi)} \left[\log \frac{p(x, z; \theta)}{q(z|x; \phi)} \right] = \text{ELBO}$$
- Derivation via Jensen's inequality.
- Maximizing the ELBO instead of maximizing $\log p(x; \theta)$

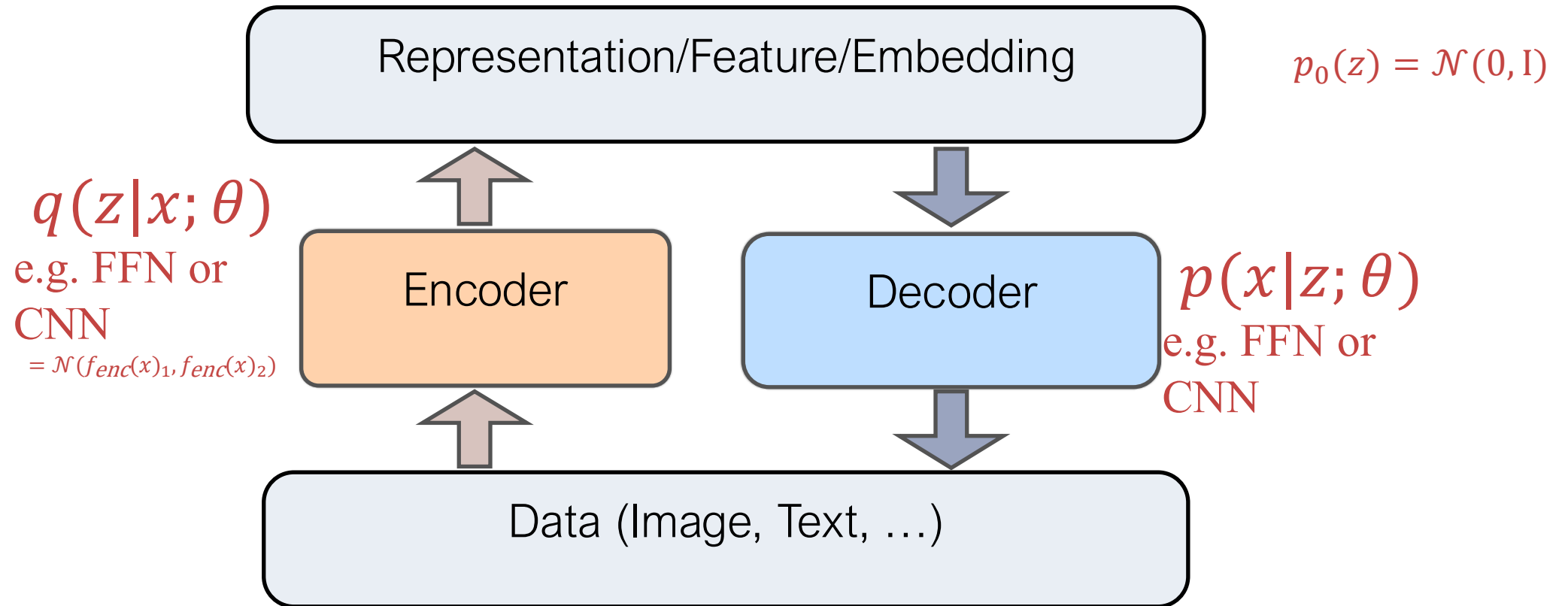
$q(z|x; \phi)$ is the posterior distribution from encoder!
Also known as variational posterior distribution, hence Variation Auto-Encoder!

Understanding ELBO

$$\begin{aligned} \max_{\theta} \max_{\phi} \text{ELBO} & \log p(X; \theta) \geq \mathbb{E}_q \left[\log \frac{p(X, Z; \theta)}{q(Z|X; \phi)} \right] \\ & = \sum_n \mathbb{E}_q \left[\log \frac{p(x_n | z_n; \theta) p_0(z_n)}{q(z_n | x_n; \phi)} \right] \\ & = \underbrace{\mathbb{E}_q [\log p(x_n | z_n; \theta)]}_{\text{Reconstruction loss}} - \underbrace{\text{KL}(q(z_n | x_n; \phi) \parallel p_0(z_n))}_{\text{Regularization}} \end{aligned}$$

Variational Auto-Encoder w/ NN

Let $q(z|x; \phi)$ and $p(x|z; \theta)$ share the same parameter θ



Training VAE

- gradient descent(ascent for max)

$$\begin{aligned} \max_{\theta} \max_{\phi} \text{ELBO} &= \sum_n \mathbb{E}_{q(z_n|x_n;\theta)} \left[\log \frac{p(x_n|z_n;\theta)p_0(z_n)}{q(z_n|x_n;\theta)} \right] \\ &= \sum_n \mathbb{E}_{q(z_n|x_n;\theta)} [r(\theta, z_n, x_n)] \\ r(\theta, z_n, x_n) &= \log \frac{p(x_n|z_n;\theta)p_0(z_n)}{q(z_n|x_n;\theta)} \end{aligned}$$

- Computing gradient:

$$\nabla_{\theta} \mathbb{E}_{q(z_n|x_n;\theta)} [r(\theta, z_n, x_n)]$$

Gradient of ELBO

$$r(\theta, z_n, x_n) = \log \frac{p(x_n | z_n; \theta) p_0(z_n)}{q(z_n | x_n; \theta)}$$

- Computing gradient:

$$\nabla_{\theta} \mathbb{E}_{q(z_n | x_n; \theta)} [r(\theta, z_n, x_n)]$$

Gradient of ELBO

$$r(\theta, z_n, x_n) = \log \frac{p(x_n | z_n; \theta) p_0(z_n)}{q(z_n | x_n; \theta)}$$

- To computing gradient:

$$\begin{aligned} & \nabla_{\theta} \mathbb{E}_{q(z_n | x_n; \theta)} [r(\theta, z_n, x_n)] \\ &= \mathbb{E}_{q(z_n | x_n; \theta)} [\nabla_{\theta} r(\theta, z_n, x_n)] + \int r(\theta, z_n, x_n) \nabla_{\theta} q(z_n | x_n; \theta) d_{z_n} \end{aligned}$$

1. sample $z_n \sim q(z_n | x_n; \theta) = \mathcal{N}(f(x_n)_1, f(x_n)_2)$, then compute average of $\nabla_{\theta} r(\theta, z_n, x_n)$

Gradient of ELBO

$$r(\theta, z_n, x_n) = \log \frac{p(x_n | z_n; \theta) p_0(z_n)}{q(z_n | x_n; \theta)}$$

Computing gradient:

$$\begin{aligned} & \nabla_{\theta} \mathbb{E}_{q(z_n | x_n; \theta)} [r(\theta, z_n, x_n)] \\ &= \mathbb{E}_{q(z_n | x_n; \theta)} [\nabla_{\theta} r(\theta, z_n, x_n)] + \int r(\theta, z_n, x_n) \nabla_{\theta} q(z_n | x_n; \theta) d_{z_n} \end{aligned}$$

2. rewrite as

$$\int r(\theta, z_n, x_n) \nabla_{\theta} q(z_n | x_n; \theta) d_{z_n} = \mathbb{E}_{q(z_n | x_n; \theta)} [r(\theta, z_n, x_n) \nabla_{\theta} \log q(z_n | x_n; \theta)]$$

- then sample $z_n \sim q(z_n | x_n; \theta) = \mathcal{N}(f(x_n)_1, f(x_n)_2)$
- compute average of $r(\theta, z_n, x_n) \nabla_{\theta} \log q(z_n | x_n; \theta)$

Problem — high variance!

Reparameterization Trick

$$q(z_n|x_n; \theta) = \mathcal{N}(f(x_n)_1, f(x_n)_2) = \mathcal{N}(\mu_\theta(x_n), \Sigma_\theta(x_n))$$

Treating $\epsilon \sim N(0,1)$, standard Gaussian distribution, then

$$\mathbb{E}_{q(z_n|x_n;\theta)}[r(\theta, z_n, x_n)] = \mathbb{E}_{\epsilon \sim N(0,1)}[r(\theta, z_n, x_n)]$$

where $z_n = \Sigma_\theta^{\frac{1}{2}}(x_n)\epsilon + \mu_\theta(x_n)$

Taking gradient does not depend on the distribution

Reparameterization Trick

$$\begin{aligned} & \nabla_{\theta} \mathbb{E}_{q(z_n|x_n; \theta)} \left[\frac{\log(p(x_n|z_n; \theta)p_0(z_n))}{q(z_n|x_n; \theta)} \right] \\ &= \nabla_{\theta} \mathbb{E}_{q(z_n|x_n; \theta)} [\log p(x_n|z_n; \theta)] - \text{KL}(q(z_n|x_n; \theta) \parallel p_0(z_n)) \end{aligned}$$

$$\text{Let } z_n = \Sigma_{\theta}^{\frac{1}{2}}(x_n)\epsilon + \mu_{\theta}(x_n)$$

$$\begin{aligned} &= \nabla_{\theta} \mathbb{E}_{\epsilon \sim N(0,1)} [\log p(x_n|z_n; \theta)] - \text{KL}(\mathcal{N}(\mu_{\theta}(x_n), \Sigma_{\theta}(x_n)) \parallel \mathcal{N}(0,1)) \\ &= \nabla_{\theta} \mathbb{E}_{\epsilon \sim N(0,1)} [\log p(x_n|z_n; \theta)] - \frac{1}{2} \left(\mu_{\theta}(x_n)^T \mu_{\theta}(x_n) + \text{tr}(\Sigma_{\theta}(x_n)) - M - \log \text{Det}(\Sigma_{\theta}(x_n)) \right) \\ &= \mathbb{E}_{\epsilon \sim N(0,1)} [\nabla_{\theta} \log p(x_n|z_n; \theta)] - \nabla_{\theta} \frac{1}{2} \left(\mu_{\theta}(x_n)^T \mu_{\theta}(x_n) + \text{tr}(\Sigma_{\theta}(x_n)) - M - \log \text{Det}(\Sigma_{\theta}(x_n)) \right) \end{aligned}$$

Compute Gradient using Reparameterization Trick

For each data point x_n , current parameter θ

Step 1: sample $\epsilon \sim N(0,1)$

Step 2: using encoder forward to compute $\mu, \Sigma = f_{\text{enc}}(x_n; \theta)$

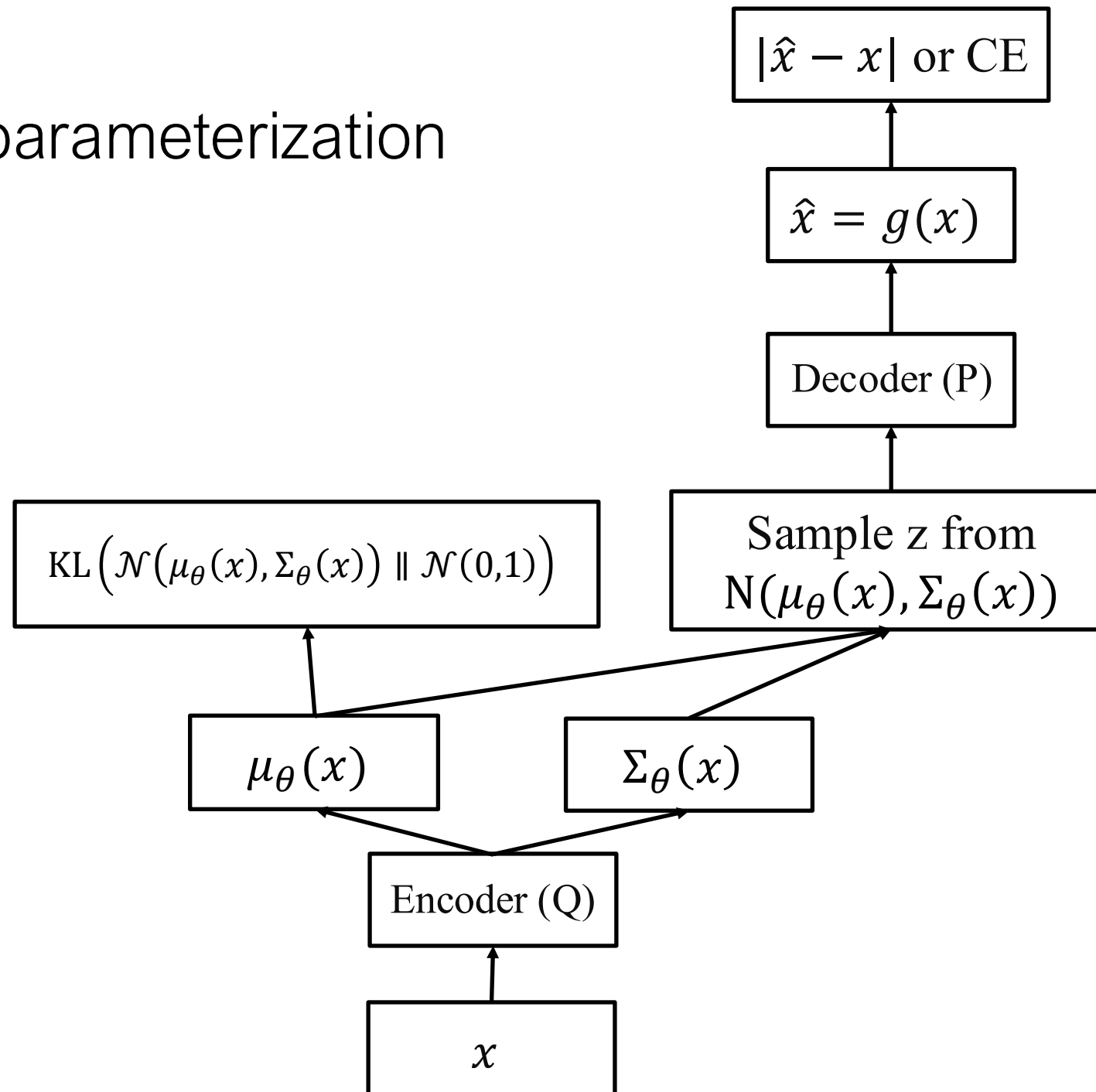
Step 3: $z(\theta) = \Sigma^{\frac{1}{2}}\epsilon + \mu$

Step 4: using decoder forward to compute $p(x_n|z(\theta); \theta)$

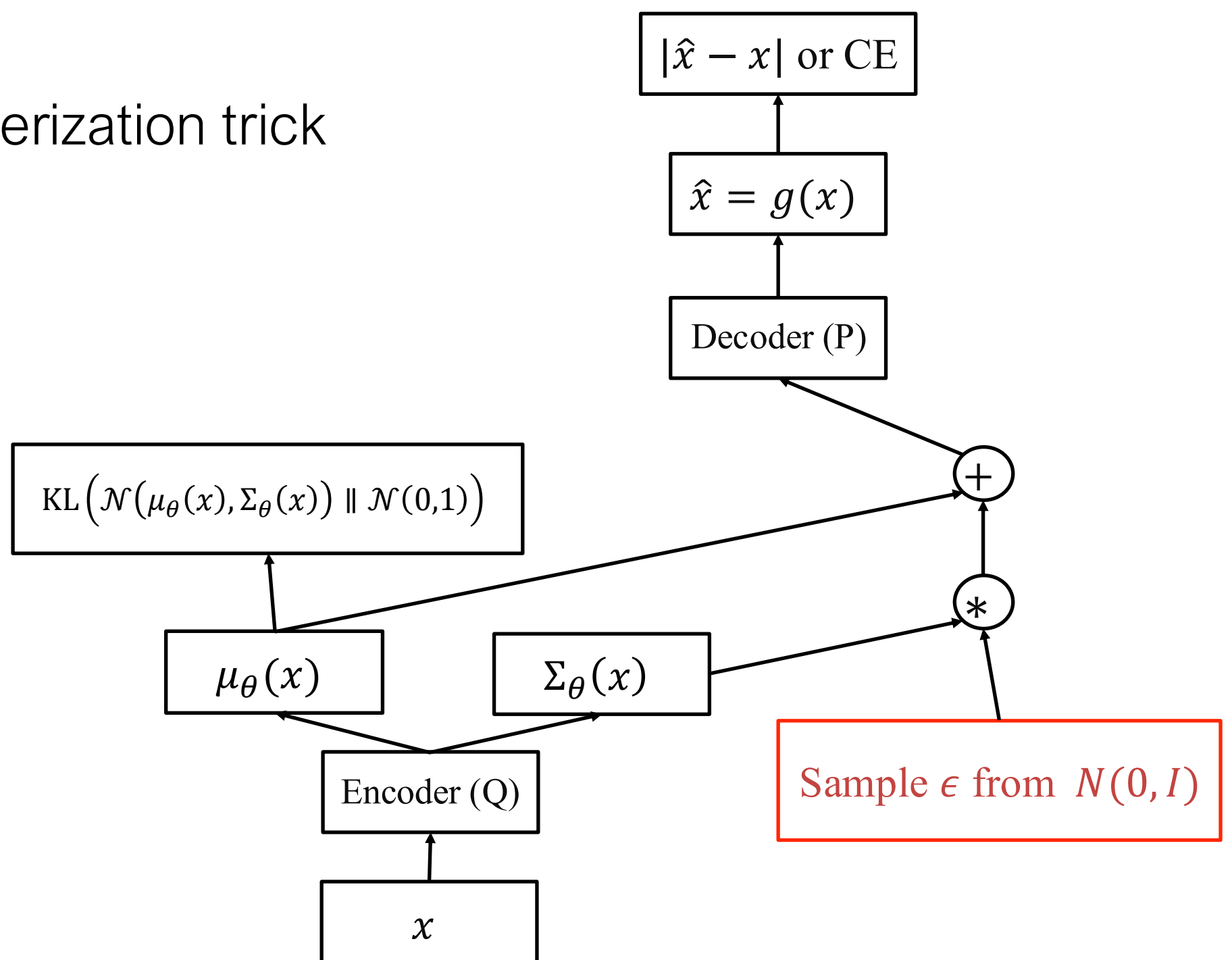
Step 5: define $\text{err} = \log p(x_n|z_n; \theta) - \beta \cdot \text{KL}(q(z|x_n; \theta) \parallel p_0(z))$, then using **back-propagation** to compute gradient for θ


$$\frac{1}{2}(\mu_{\theta}(x_n)^T \mu_{\theta}(x_n) + \text{tr}(\Sigma_{\theta})(x_n) - M - \log \text{Det}(\Sigma_{\theta}(x_n)))$$

Without reparameterization



Reparameterization trick

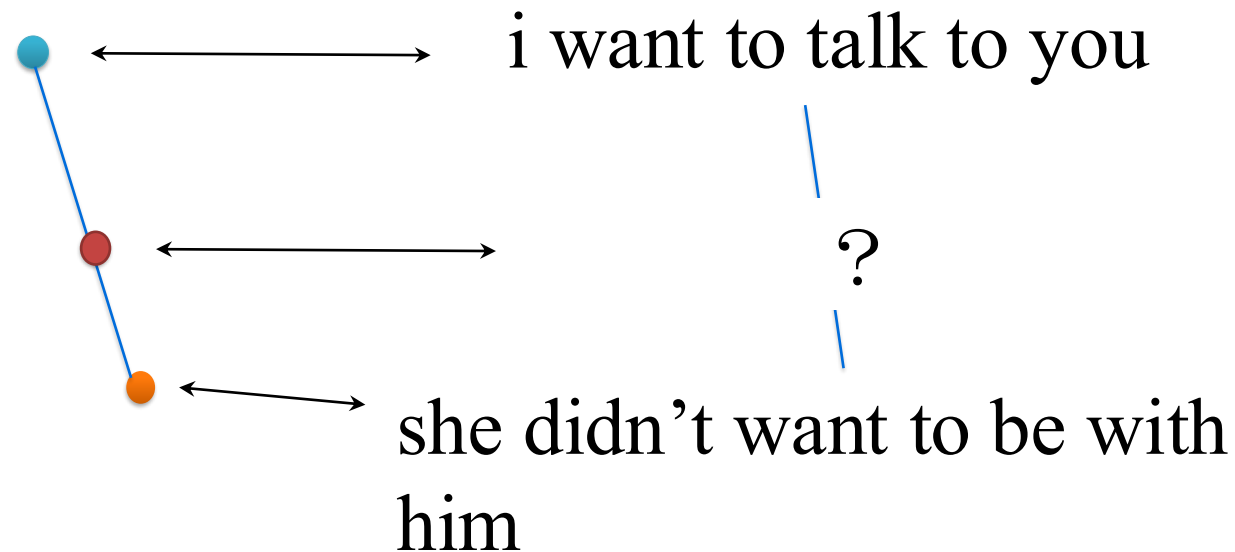


Outline

- Auto-Encoder
- Probabilistic Graphical Model
- Variational Auto-Encoder
- • VAE for Sequence data

Generating Sequence from Continuous Vectors

- Key challenge: Interpolation in continuous space should yield reasonable sequence (e.g. a sentence)



Conditional Sequence Generation

- Given a latent variable z , a sequence of text tokens $x = (x_1, x_2, \dots, x_t)$ can be generated with Transformer (or RNN, LSTM, CNN) :

$$p(x|z; \theta) = \prod_t p(x_t|x_{<t}, z; \theta)$$

$$p(x_t|x_{<t}, z; \theta) = \text{softmax}(W \cdot h_t)$$
$$h_t = \text{Transformer_decoder}([x_{0:t-1}, z], \theta)$$

VAE for Sequence Generation

- Decoding:

- $z \sim N(0, I)$

 - generate x from $\text{Transformer}(z)$ or $\text{LSTM}(z)$

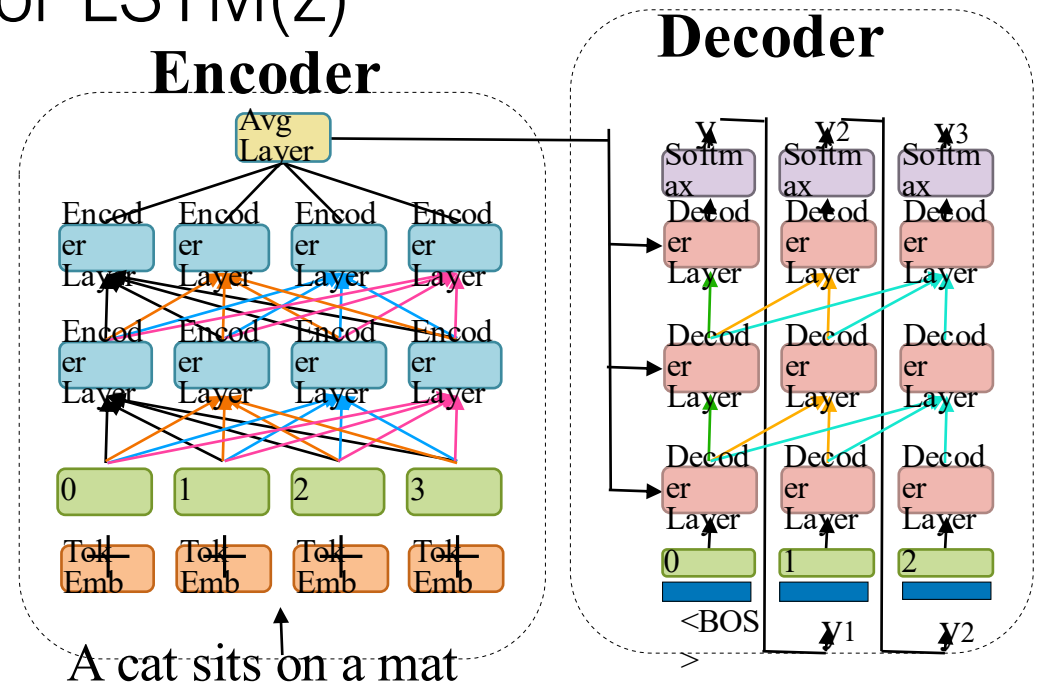
A cat sits on a mat

- Encoding:

- $q(z|x) = N(\mu, \sigma^2)$

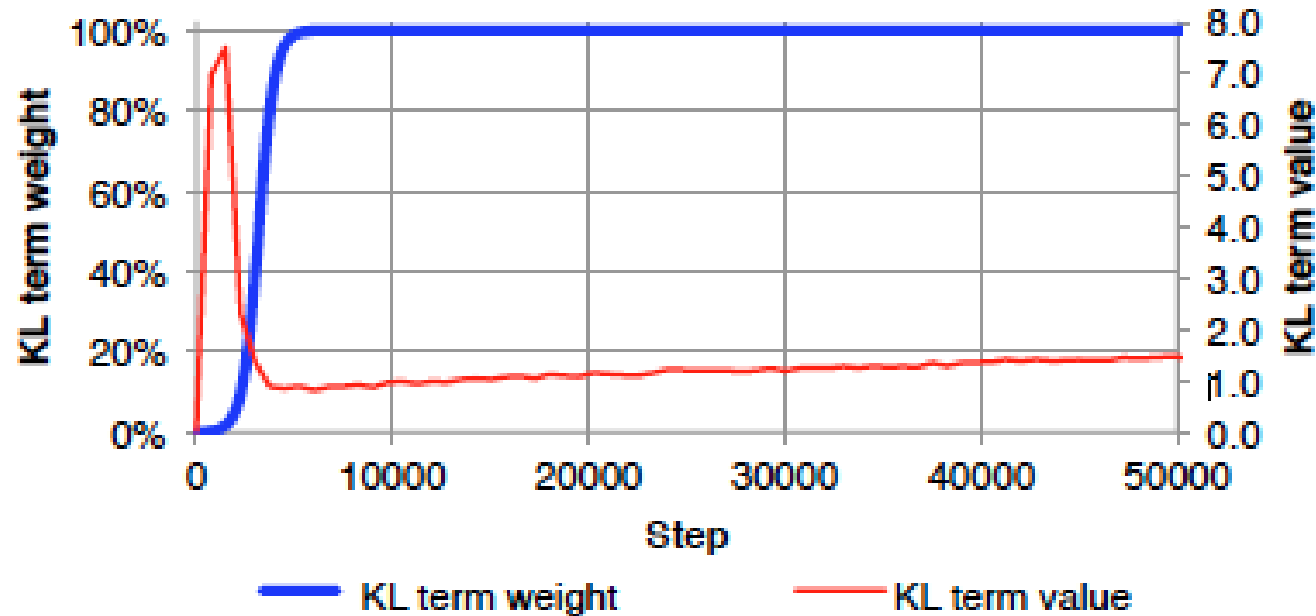
- $\mu = W_1 \cdot h_t, \sigma^2 = \exp W_2 \cdot h_t$

- $h_t = \text{Transformer}(x; \theta)$



Training VAE: Posterior Collapse

- KL term in ELBO collapses to zero and latent variable encodes little information.
- Solution: KL annealing & word dropout



Examples on Sentence Interpolation

“ i want to talk to you . ”

“i want to be with you . ”

“i do n’t want to be with you . ”

i do n’t want to be with you .

she did n’t want to be with him .

he was silent for a long moment .

he was silent for a moment .

it was quiet for a moment .

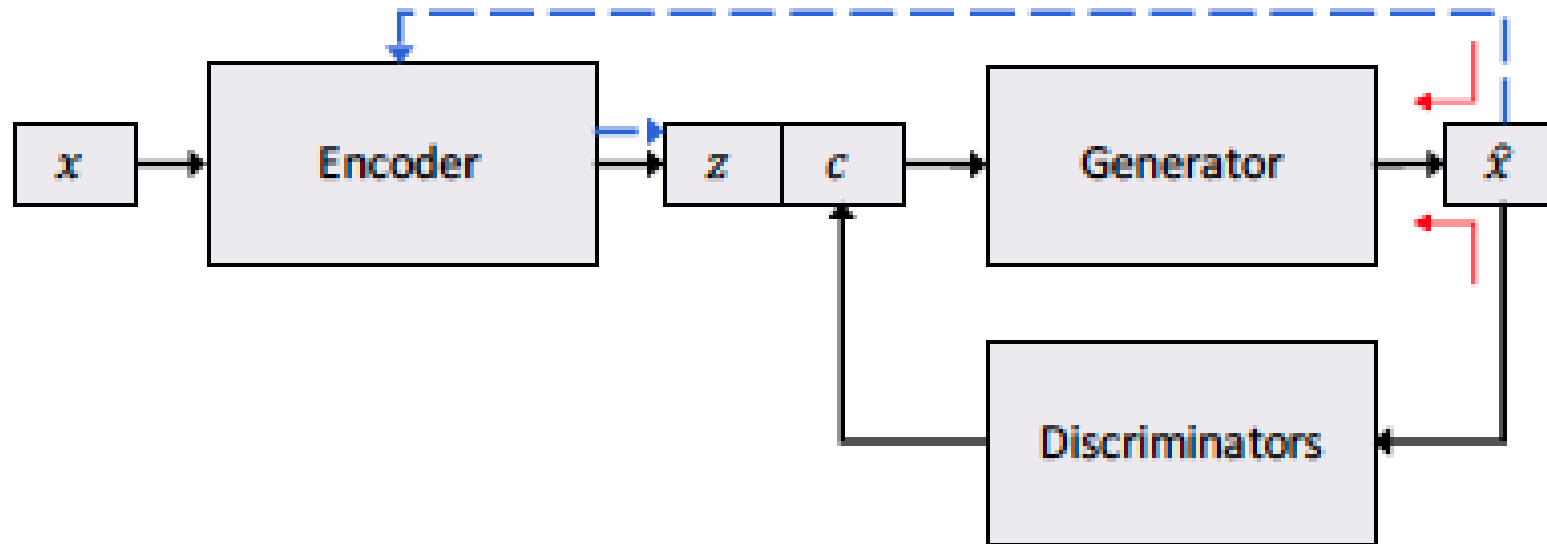
it was dark and cold .

there was a pause .

it was my turn .

Variants - CVAE

- Controllable sentence generation with both continuous and discrete labels



Generating with Varying Semantic Label

the film is strictly routine !
the film is full of imagination .

after watching this movie , i felt that disappointed .
after seeing this film , i 'm a fan .

the acting is uniformly bad either .
the performances are uniformly good .

this is just awful .
this is pure genius .

the acting is bad .
the movie is so much fun .

none of this is very original .
highly recommended viewing for its courage , and ideas .

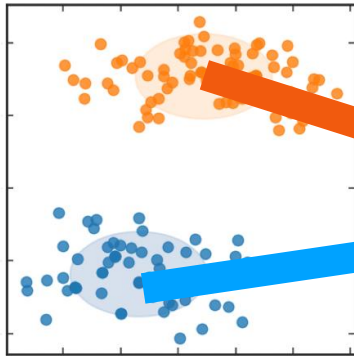
too bland
highly watchable

i can analyze this movie without more than three words .
i highly recommend this film to anyone who appreciates music .

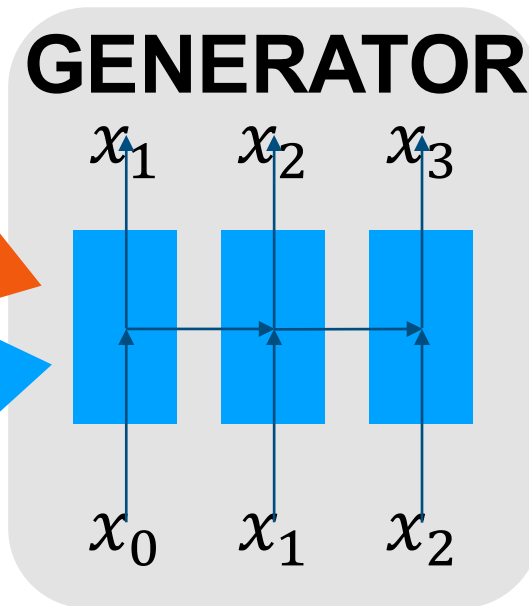
Toward Controlled Generation of Text, (Hu et. al. ICML 2017)

Variant: Learning Interpretable Latent Representation

Latent structure
dialog actions



GENERATOR



Sampling

“Remind me about the
football game.”

[action=remind]

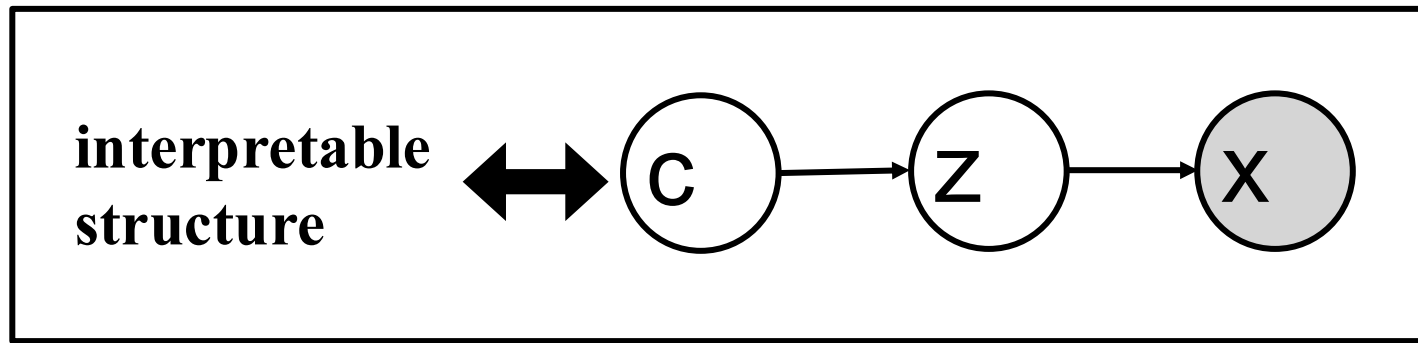
“Will it be overcast
tomorrow?”

[action=request]

.....

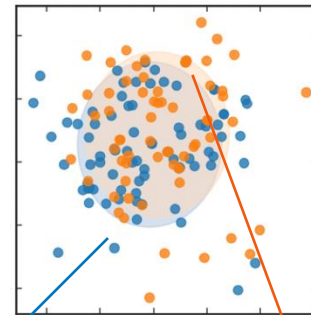
Generate Sentences with interpretable factors

Gaussian Mixture Variational Auto-encoder (GM-VAE)



C : **discrete**
component

Z : **continuous**
latent variable

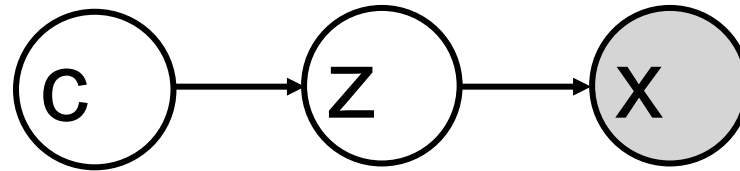


(Dilokthanakul et al., 2016; Jiang et al., 2017)

Will it be overcast tomorrow?
Remind me about the football game.

Dispersed Exponential-family Mixture VAE

Exponential-family Mixture VAE



↓ adding dispersion term in training

Dispersed EM-VAE

$$L(\theta; x) = \text{ELBO} + \beta \cdot L_d,$$

dispersion term

$$L_d = \mathbb{E}_{q_\phi(c|x)} A(\boldsymbol{\eta}_c) - A(\mathbb{E}_{q_\phi(c|x)} \boldsymbol{\eta}_c).$$

Summary

- Auto-Encoder
 - learning representation by reconstruction
- Variational Auto-Encoder
 - Introduce uncertainty for variables
 - put prior on latent representation and use variational method to train
 - Generation (decoder) → generate data from sampled latent
 - Posterior (Encoder) → infer latent representation from data
 - Reparameterization trick