

**165B**

**Machine Learning  
Variational Auto-Encoder**

Lei Li (leili@cs)

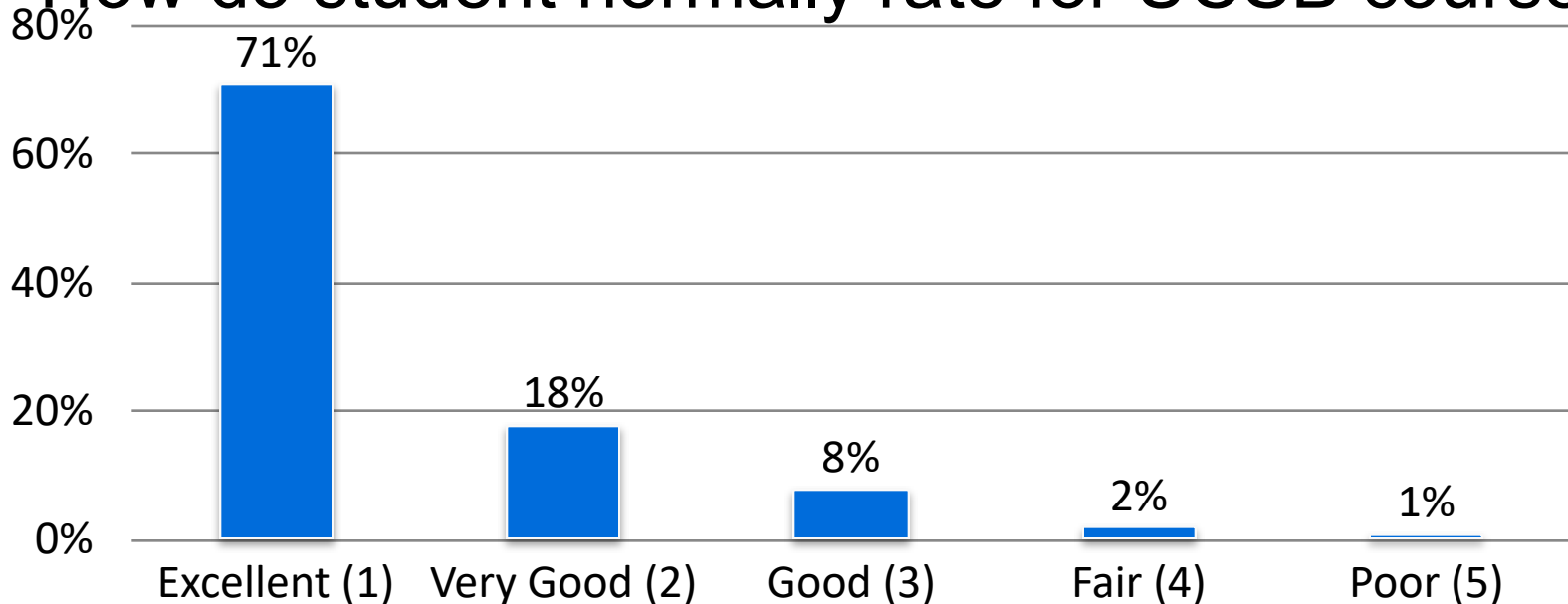
UCSB

# Course Evaluation

---

- <https://esci.id.ucsb.edu>
- Feedback is important and helpful for improving the course
- Encourage narrative comments:
  - specific aspects of the course and instruction

How do student normally rate for UCSB courses?



# Recap

---

- Graph neural network
  - message passed along graph edges
  - aggregate message/embedding by FFN
  - many variants

# **Sparse Coding**

# Representation Learning

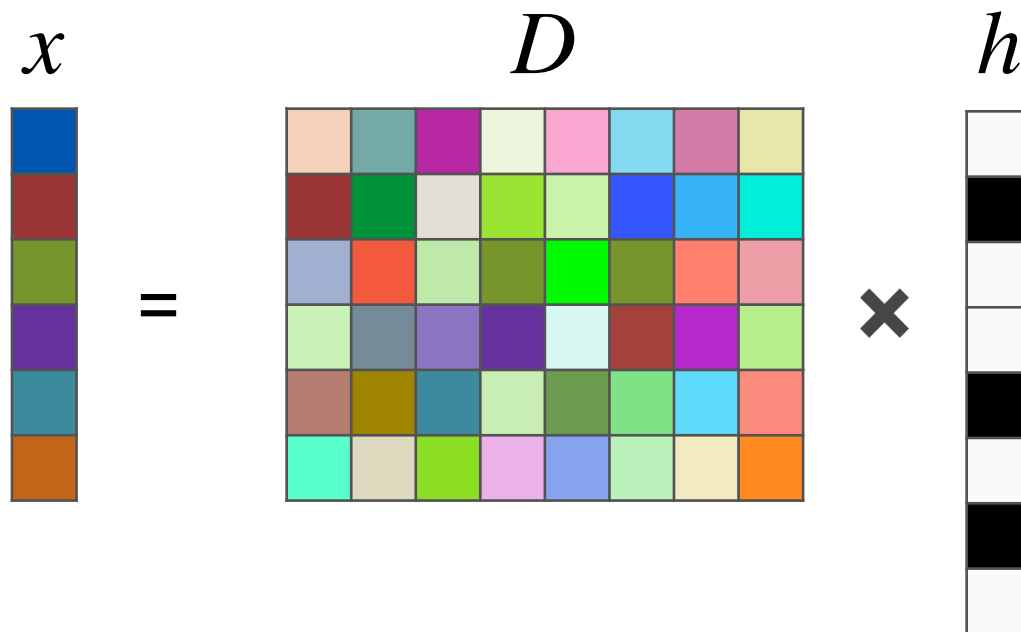
---

- Learning embeddings/features from data to perform well in downstream tasks (classification, clustering, generation)
- Examples:
  - Sparse coding: data is a sparse linear combination of learned features (dictionary)
  - Auto-encoder: learning the representation to reconstruct original data

# Sparse Coding

- Learn a dictionary of features  $D$ , such that each data  $x$  is a linear combination of these features

$$x \approx Dh \text{ s.t. } \|h\|_0 \text{ is small}$$



# Solve Sparse Coding

---

- To find  $D$  and  $h$ , direct objective

$$\min_D \frac{1}{N} \sum_{n=1}^N \min_{h^{(n)}} \frac{1}{2} \|x^{(n)} - Dh^{(n)}\|_2^2 + \lambda \|h^{(n)}\|_0$$

Reconstruction

Sparsity penalty

Cannot apply gradient descent. Non-differentiable!

# Solve Sparse Coding

---

- Relaxation:

$$\min_D \frac{1}{N} \sum_{n=1}^N \min_{h^{(n)}} \frac{1}{2} \|x^{(n)} - Dh^{(n)}\|_2^2 + \lambda \|h^{(n)}\|_1$$

Reconstruction

Sparsity penalty

With additional constraint that columns of  $D$  having norm 1

Algorithm: iteratively solve  $h$ 's and  $D$  (using the SoftThreshold from L1 regularization)

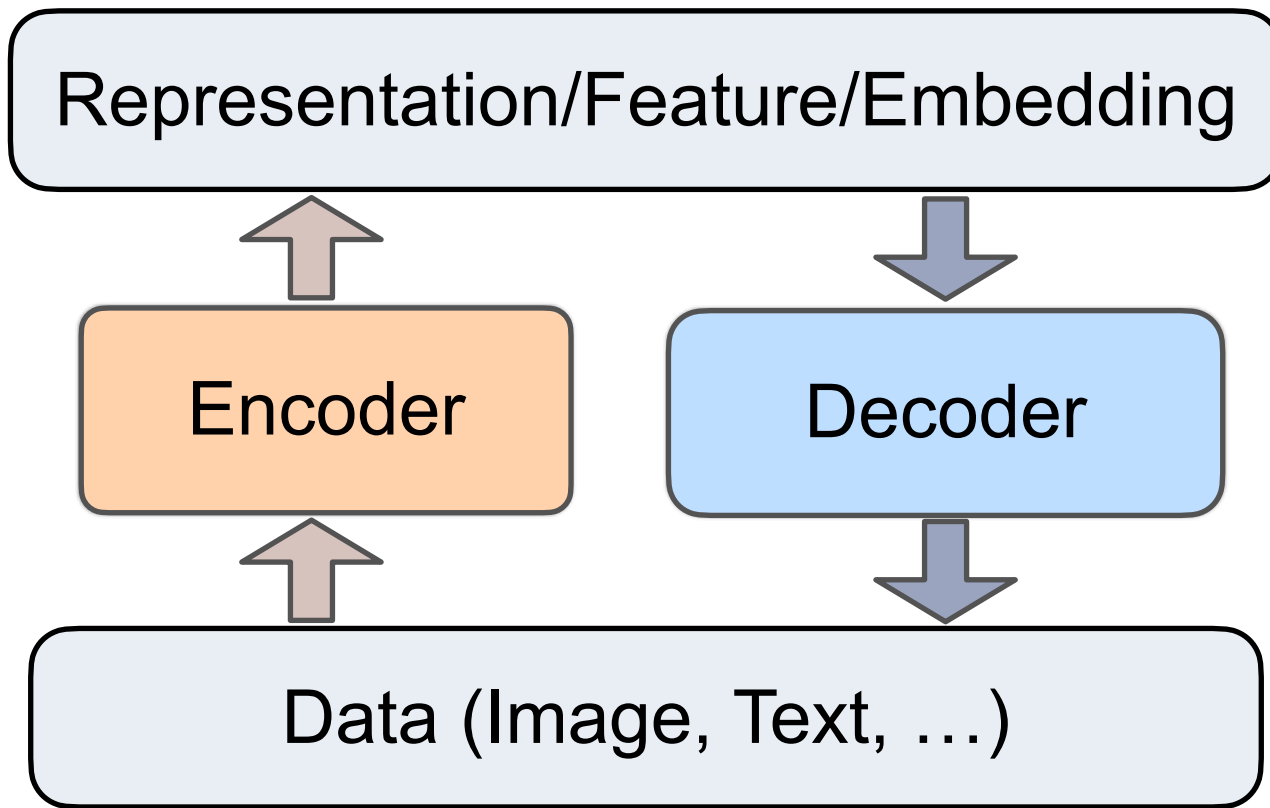


# Auto-Encoder

# Auto-Encoder

---

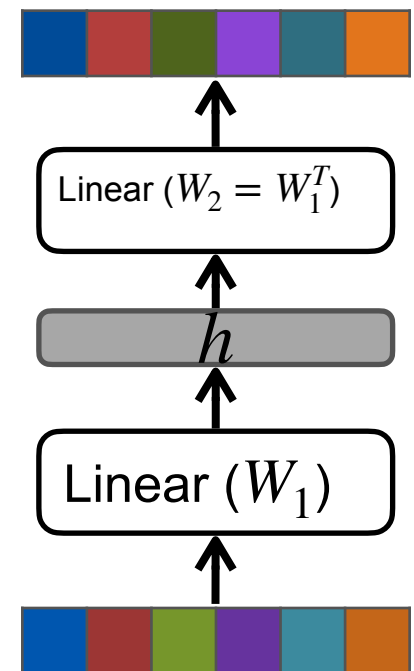
Learning representations so that data can be reconstructed from them!



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
- Categorical data (e.g. Sentences)
  - Cross-Entropy

# Undercomplete Representation

---

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

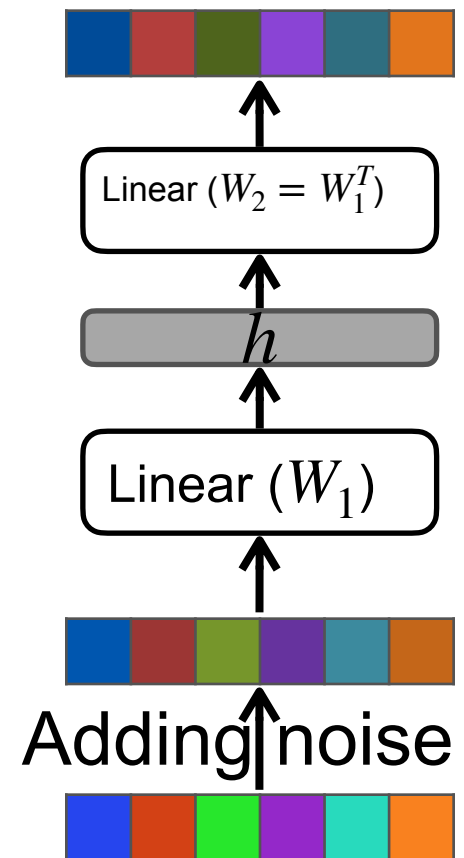
# Overcomplete Representation

---

- If hidden representation is larger than input data dimension
- Need to enforce additional constraints e.g. sparsity

# 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

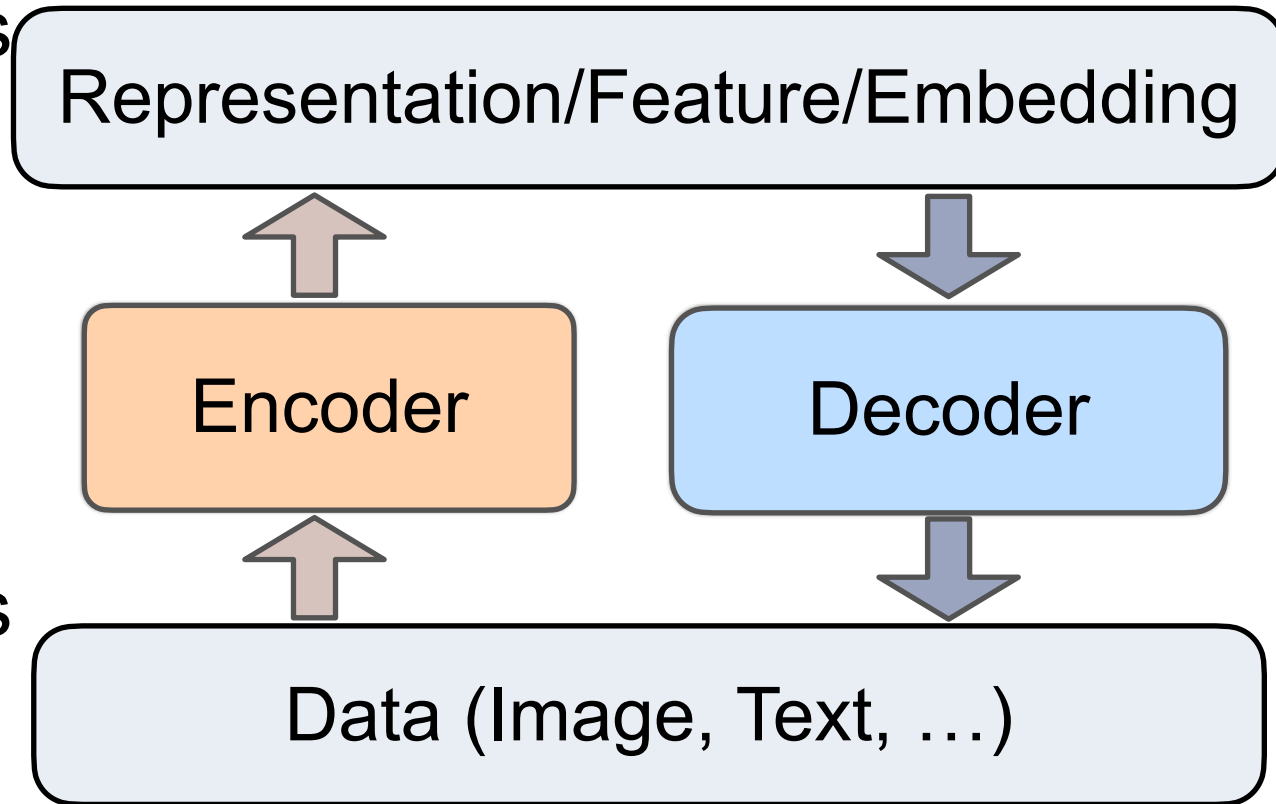


# **Variational Auto-Encoder (VAE)**



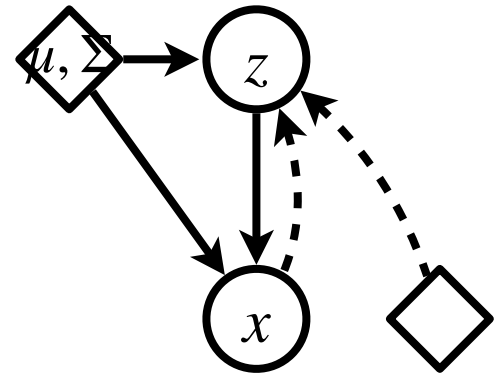
# VAE

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



# 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  $f$



- The goal is to maximize the data log-likelihood

$$\log p(X; \theta) = \log \int p(X | Z) p(Z) dZ$$

- Hard to optimize over  $\theta$ , if  $f(Z)$  is very complex such as a CNN, RNN, or Transformer.

# VAE

---

Objective: maximize the data loglikelihood

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

# VAE

---

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

- But  $\log p(x; \theta)$  is intractable.

$q(z | x; \phi)$  is the posterior  
distribution from encoder!

- 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  $\log p(x; \theta)$

# Understanding ELBO

---

$$\log p(X; \theta) \geq \mathbb{E}_q \left[ \log \frac{p(X, Z; \theta)}{q(Z | X; \phi)} \right]$$

$$\max_{\theta} \max_{\phi} \text{ELBO} = \sum_n \mathbb{E}_q \left[ \log \frac{p(x_n | z_n; \theta) p_0(z_n)}{q(z_n | x_n; \phi)} \right]$$

=  
=

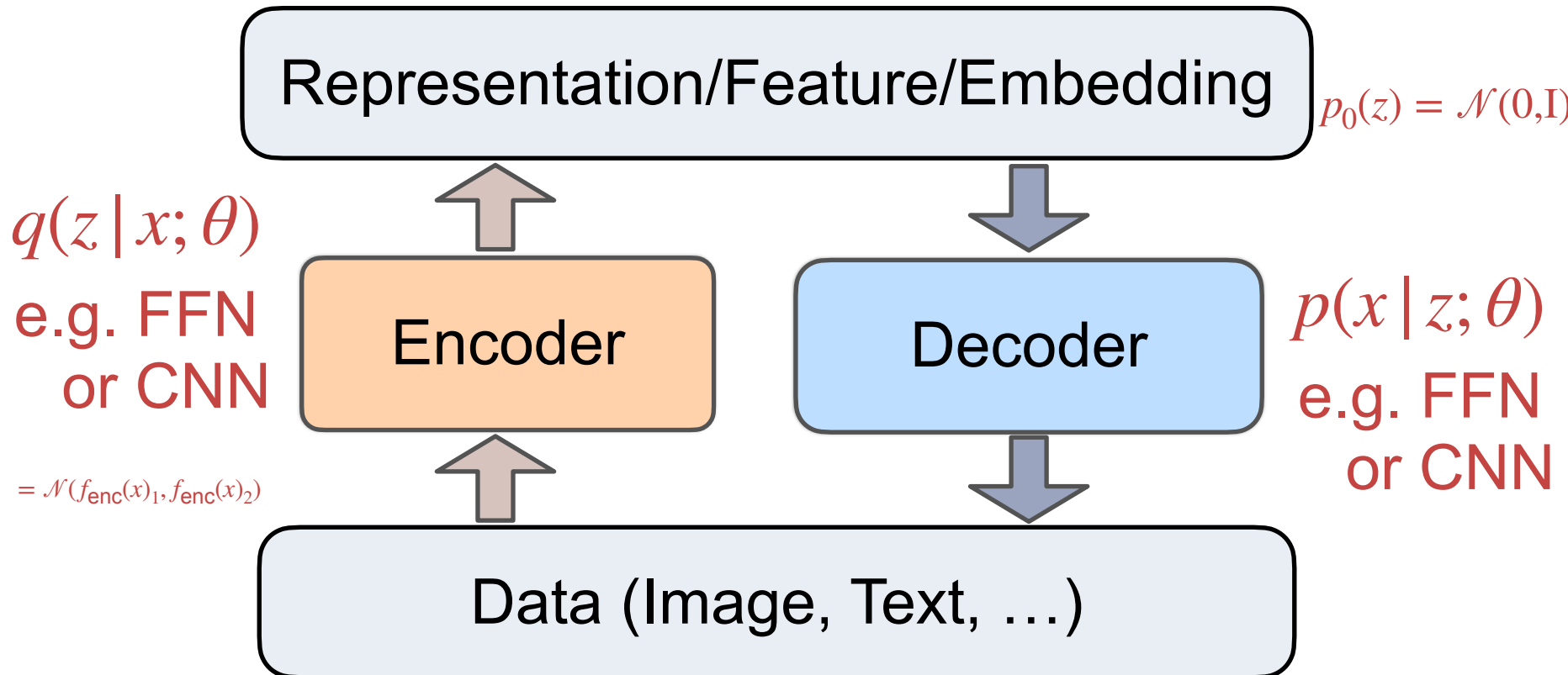
$$= \mathbb{E}_q \left[ \log p(x_n | z_n; \theta) \right] - \text{KL} \left( q(z_n | x_n; \phi) \| p_0(z_n) \right)$$

Reconstruction loss

Regularization

# VAE

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



# Training VAE

---

gradient descent(ascent for max)

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

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

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

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:

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



# 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:

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

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:

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

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  $r(\theta, z_n, x_n) \nabla_{\theta} q(z_n | x_n; \theta)$

**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

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

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

Taking gradient does not depend on the distribution

# Reparameterization Trick

---

$$\begin{aligned} & \nabla_{\theta} \mathbb{E}_{q(z_n|x_n;\theta)} \left[ \log \frac{p(x_n | z_n; \theta) p_0(z_n)}{q(z_n | x_n; \theta)} \right] \\ &= \nabla_{\theta} \mathbb{E}_{q(z_n|x_n;\theta)} [\log p(x_n | z_n; \theta)] - \text{KL} (q(z_n | x_n; \theta) \| p_0(z_n)) \\ &= \nabla_{\theta} \mathbb{E}_{\epsilon \sim \mathcal{N}(0,1)} [\log p(x_n | z_n; \theta)] - \text{KL} (\mathcal{N}(\mu_{\theta}(x_n), \Sigma_{\theta}(x_n)) \| \mathcal{N}(0,1)) \\ &= \nabla_{\theta} \mathbb{E}_{\epsilon \sim \mathcal{N}(0,1)} [\log p(x_n | z_n; \theta)] - \frac{1}{2} (\mu_{\theta}(x_n)^T \mu_{\theta}(x_n) + \text{tr}(\Sigma_{\theta}(x_n)) - M - \log \text{Det}(\Sigma_{\theta}(x_n))) \\ &= \mathbb{E}_{\epsilon \sim \mathcal{N}(0,1)} [\nabla_{\theta} \log p(x_n | z_n; \theta)] - \nabla_{\theta} \frac{1}{2} (\mu_{\theta}(x_n)^T \mu_{\theta}(x_n) + \text{tr}(\Sigma_{\theta}(x_n)) - M - \log \text{Det}(\Sigma_{\theta}(x_n))) \end{aligned}$$

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

# Compute Gradient using Reparameterization Trick

---

For each data point  $x_n$ , current parameter  $\theta$

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

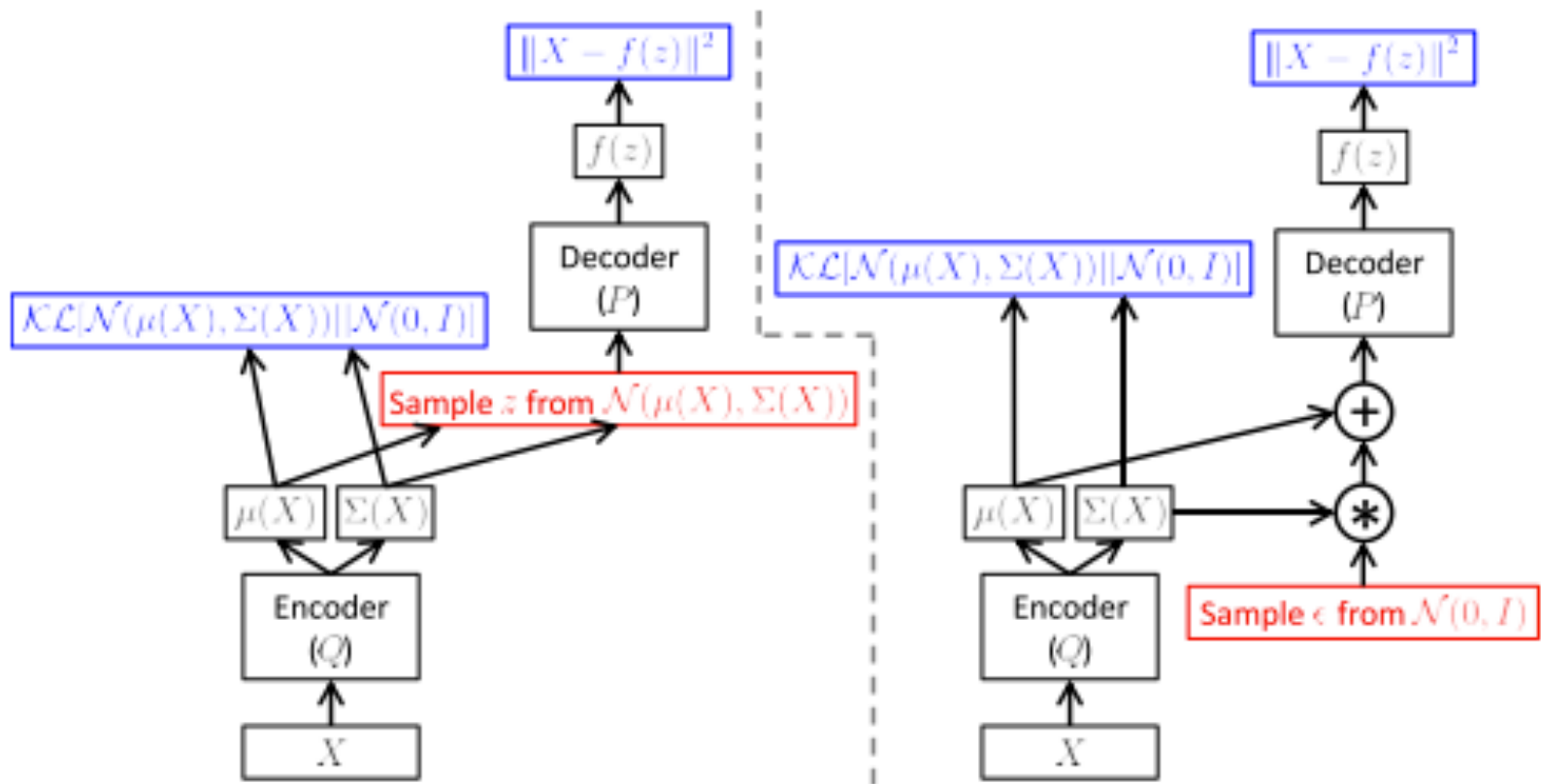
$\text{err} = \log p(x_n | z_n; \theta) - \beta \cdot \text{KL} (q(z | x_n; \theta) || p_0(z))$  , then

using back-propagation to compute gradient for  $\theta$

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


# Training VAE

- Reparameterization trick

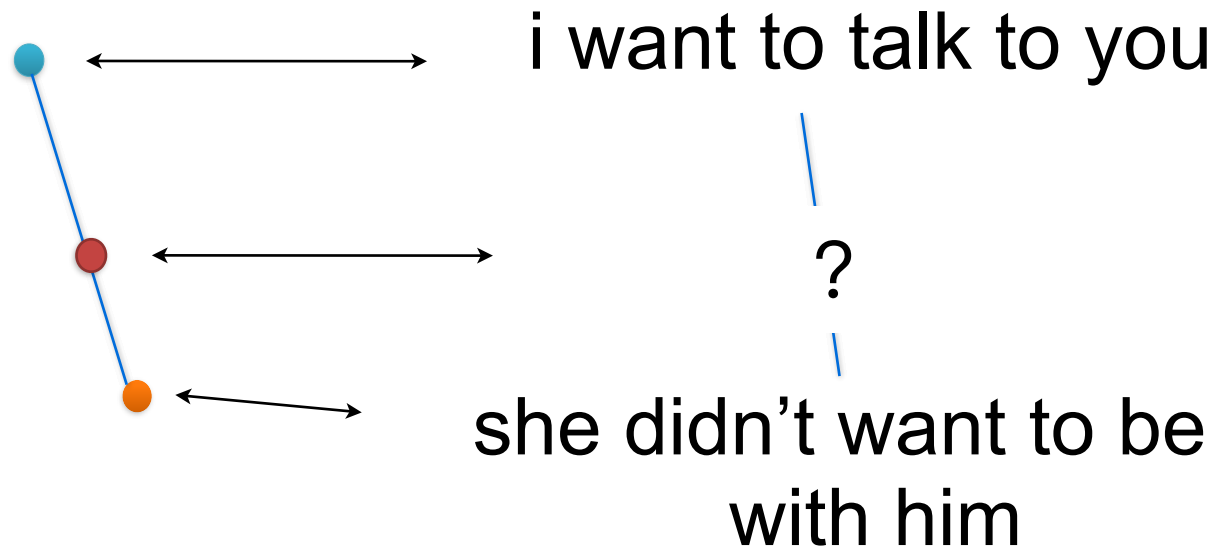


# Sentence VAE

# Generating Sentence from Continuous vectors

---

- Key challenge: Interpolation in continuous space should yield reasonable sentences





# 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 RNN (or LSTM, transformer), CRNN model:

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

$$p(x_t | x_{<t}, z; \theta) = \text{softmax}(W \cdot h_t)$$

$$h_t = \text{RNN}(h_{t-1}, [x_{t-1}, z], \theta)$$

# VAE for Sentence Generation

Decoding:

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

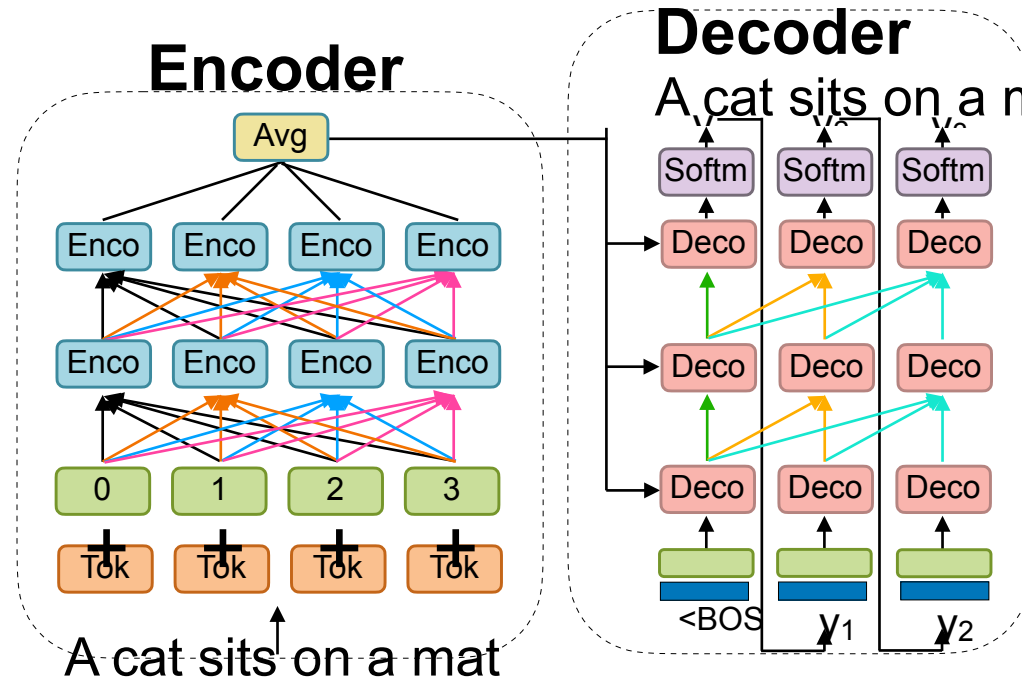
generate  $x$  from  
Transformer( $z$ ) or  
LSTM( $z$ )

Encoding:

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

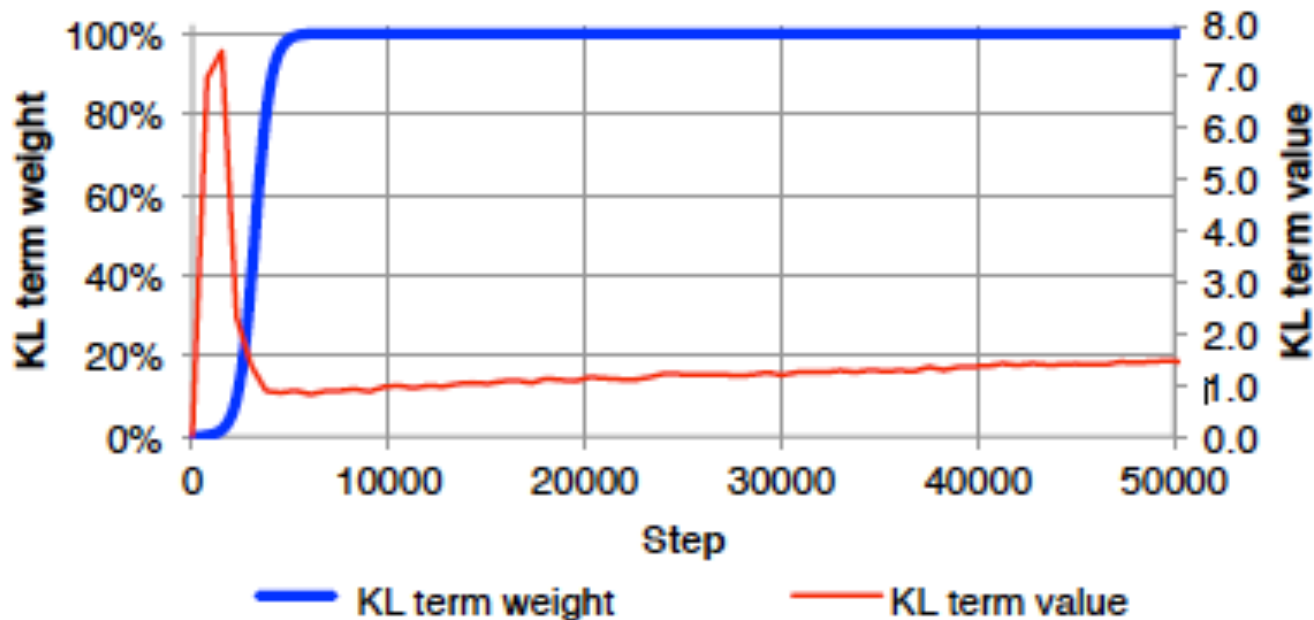
$$\mu = W_1 \cdot h_t, \sigma^2 = \exp W_2 \cdot h_t$$

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



# 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 .*

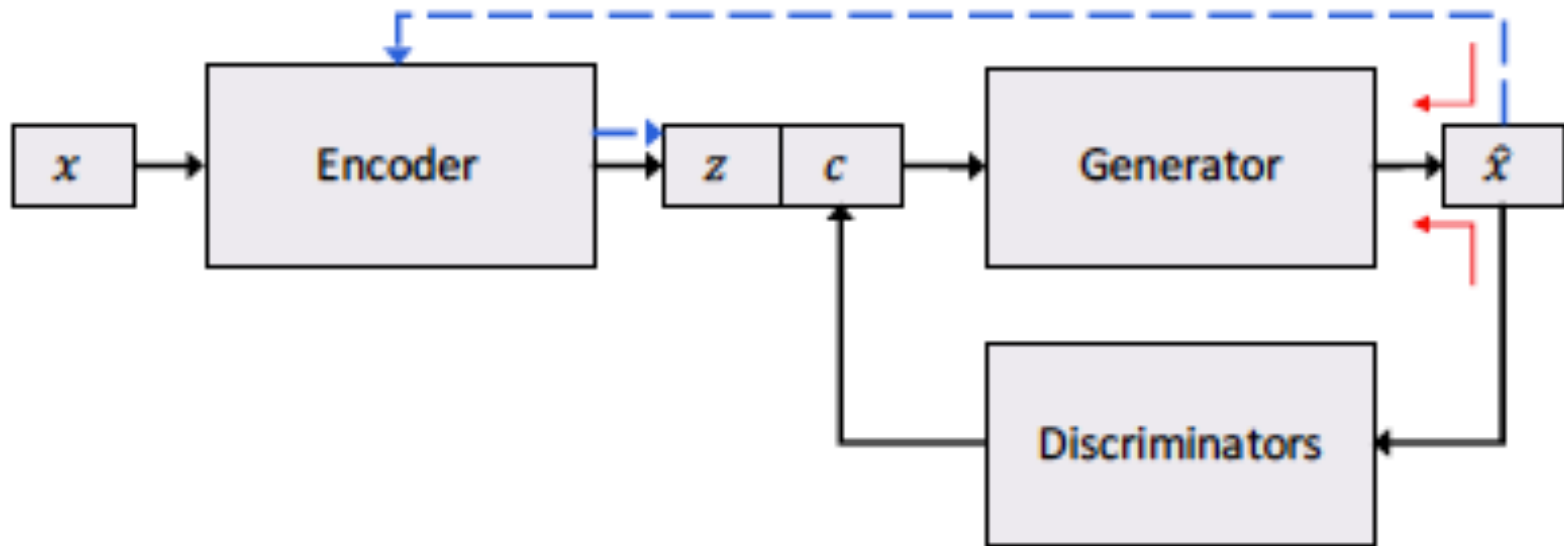
*there was a pause .*

**it was my turn .**

---

# Variants

- Controllable sentence generation with both continuous and discrete labels



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

# 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)

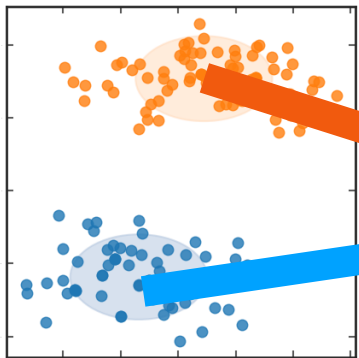
# Deep Latent Variable Models for Text

---

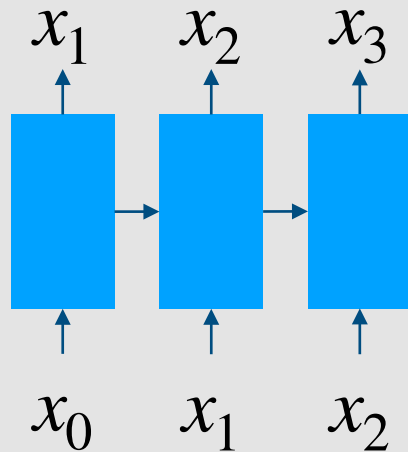
- Interpretable Deep Latent Representation from Raw Text
  - Learning Exponential Family Mixture VAE [ICML 20]
- Disentangled Representation Learning for Text Generation
  - Data to Generation: VTM [ICLR 20b]
  - Learning syntax-semantic representation [ACL 19c]
- One model to acquire 4 language skills
  - Mirror Generative NMT [ICLR 20a]

# 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]

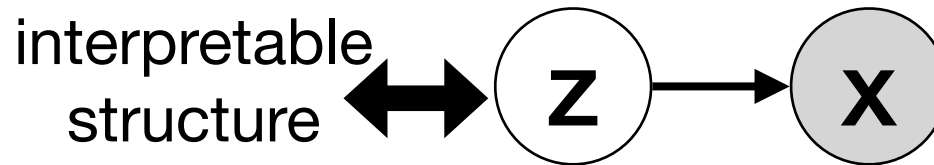
.....

Generate Sentences with  
interpretable factors



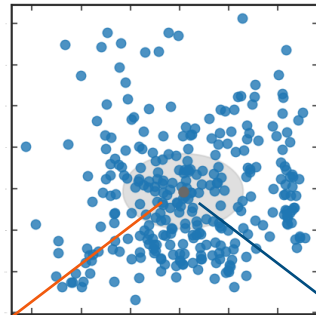
# How to Interpret Latent Variables in VAEs?

## Variational Auto-encoder (VAE)



(Kingma & Welling, 2013)

$z$ :  
continuous latent variables

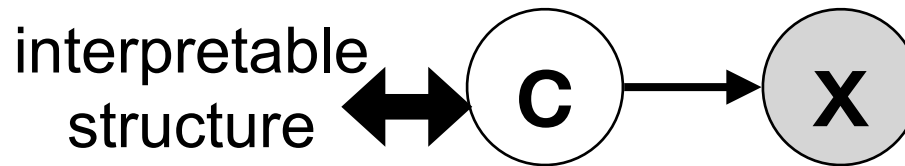


Will it be humid in New York today?  
Remind me about my meeting.

difficult to interpret discrete factors

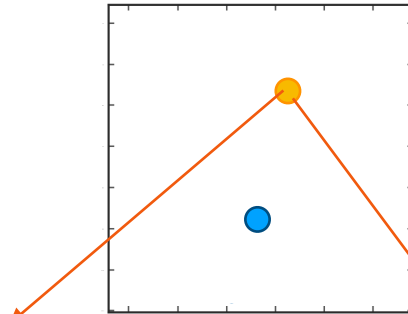
# VAEs Introduce Latent Variables

## Variational Auto-encoder (VAE)



(Zhao et al, 2018b)

$c$ : **discrete**  
latent  
variables



Remind me about my  
meeting.

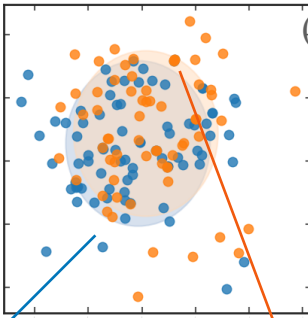
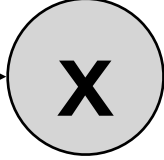
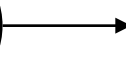
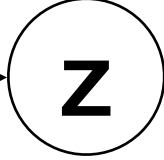
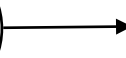
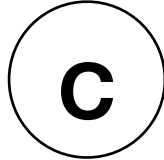
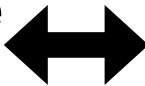
Remind me about the  
football game.

expressiveness  
is limited.

# Discrete Variables Could Enhance Interpretability - but one has to do it right!

## Gaussian Mixture Variational Auto-encoder (GM-VAE)

interpretable structure



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

$c$ : discrete component

$z$ : continuous latent variable

Will it be overcast tomorrow?

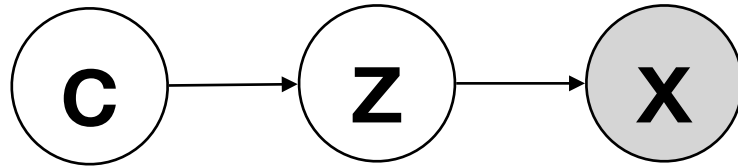
Remind me about the football game.



mode-collapse

# Do it right for VAE w/ hierarchical priors - Dispersed Exponential-family Mixture VAE

Exponential-family Mixture VAE



↓ adding dispersion term in training

**Dispersed EM-VAE**

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

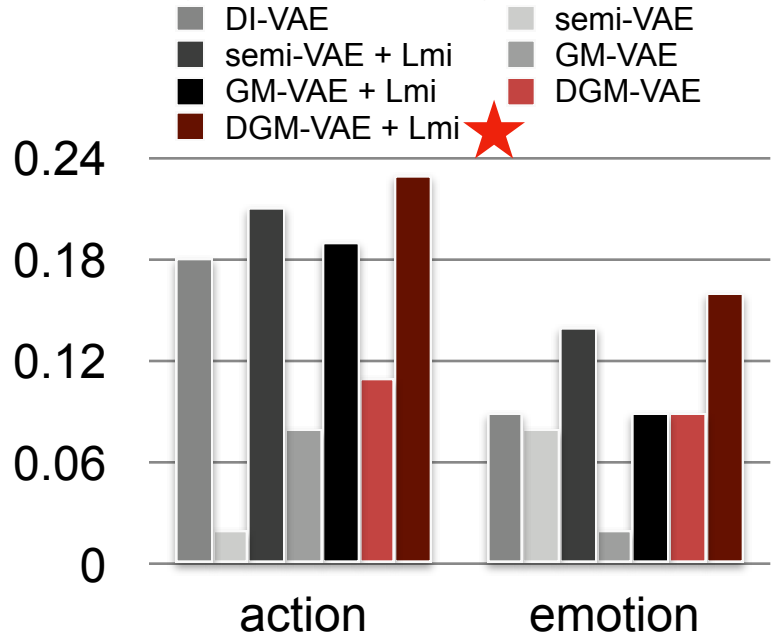
dispersion term

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

# Generation Quality and Interpretability

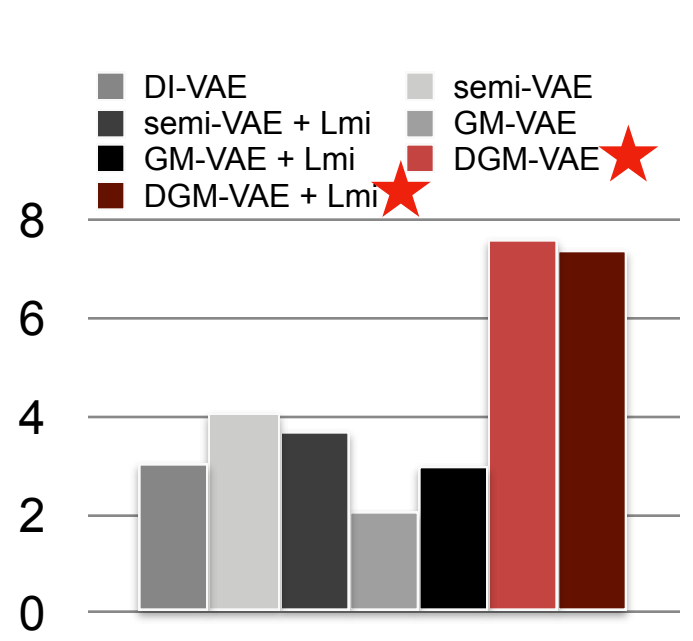
DGM-VAE obtains the best performance in interpretability and reconstruction

Homogeneity with golden label in DD



Best interpretability

BLEU of reconstruction in DD



Best reconstruction

# Latent Variables Learned by DEM-VAE are Semantically Meaningful

Example actions and corresponding utterances (classified by  $q_{\phi}(c | x)$ )

## Inferred action=Inform-route/address

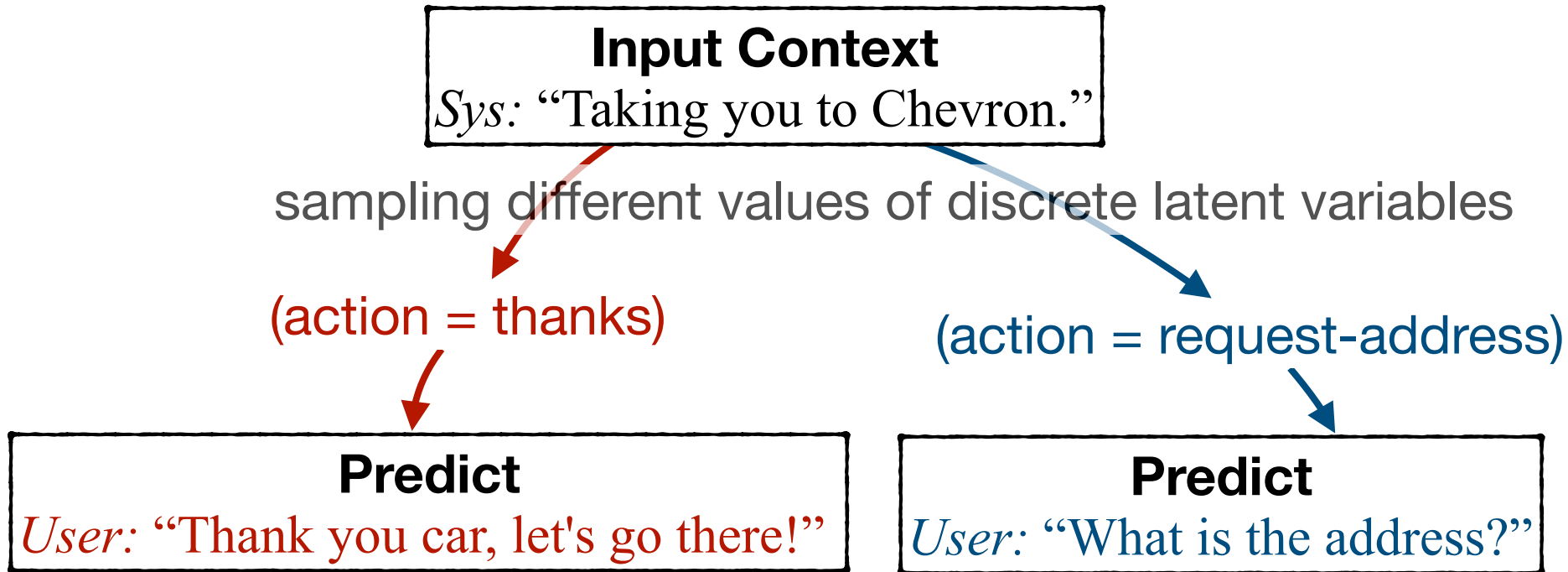
“There is a Safeway 4 miles away.”  
“There are no hospitals within 2 miles.”  
“There is Jing Jing and PF Changs.”  
...

## Inferred action =Request-weather

“What is the weather today?”  
“What is the weather like in the city?”  
“What's the weather forecast in New York?”  
...

Utterances of the same actions could be assigned with the same discrete latent variable  $c$ .

# Generate Sensible Dialog Response with DEM-VAE



Responses with different actions are generated by sampling different values of discrete latent variables.

# Summary

---

- Auto-Encoder: learning representation by reconstruction
- Variational Auto-Encoder: put prior on latent representation and use variational method to train



# Next Up

---

- Generative Adversarial Network