KAIST CHEMISTRY

Variational autoencoder

Prof. Ph.D. Woo Youn Kim

Chemistry, KAIST

Goals

10주	주제	Message Passing Neural Network (MPNN)
	목표	Understanding the most general expression of graph neural network
	내용	MPNN, molecular graph representation, GGNN, supervised learning of logP and TPSA
11주	주제	Molecular generative model 1
	목표	Understanding the principle of autoencoder and unsupervised learning
	내용	Molecular autoencoder, VAE, CVAE, de novo molecular design
12주	주제	Molecular generative model 2
	목표	Understanding difference between GAN and VAE
	내용	GAN, ARAE ARAE: conditional molecular design

Review

Bayes' Rule

$$P(\theta|x) = \frac{P(x|\theta)P(\theta)}{P(x)} = \frac{P(x|\theta)P(\theta)}{\sum_{\theta} P(x|\theta)P(\theta)}$$

$$\text{Posterior probability} = \frac{(\text{likelihood}) \times (\text{prior probability})}{(\text{evidence})}$$

Prior probability (사전확률) $P(\theta)$: probability of a parameter set θ.

Posterior probability (사후확률) $P(\theta|x)$: $P(\theta)$ given an observation x.

Evidence (증거) $P(x)$: probability of the observation.

Likelihood (가능도) $L(\theta|x) = P(x|\theta)$: $P(x)$ given the parameters θ

θ = disease

x = test result (T or F)

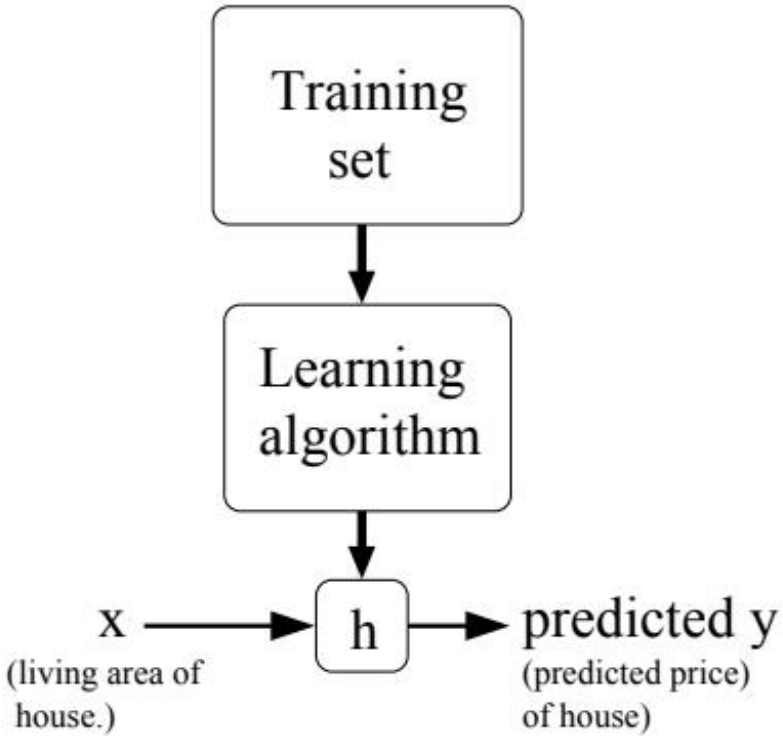
$P(x)$: probability of T and F

$P(\theta)$: probability of having a disease

$P(\theta|x)$: probability of having the disease given the test result

$P(x)$: probability of T and F.

Review



The principal of **maximum likelihood** says that we should choose θ so as to make the data as high probability as possible.

Review

$$Cost = -\sum_{i=1}^{m}\left[y^{(i)} \ln h\left(x^{(i)}\right) + \left(1 - y^{(i)}\right) \ln\left(1 - h\left(x^{(i)}\right)\right)\right]$$

$$Cost = -\sum_{i=1}^{m} p_{true}\left(x^{(i)}\right) \ln p_{pred}\left(x^{(i)}\right)$$

VS

It is also called the **cross entropy**

Gibbs entropy formula

$$S = -k_{\mathrm{B}} \sum_{i} p_i \ln p_i$$

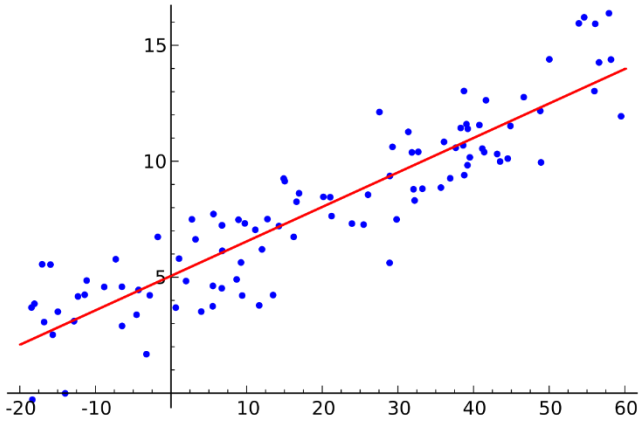
Entropy in information theory

$$S = -\sum_{i=1} p_i \ln p_i$$

Types of deep learning

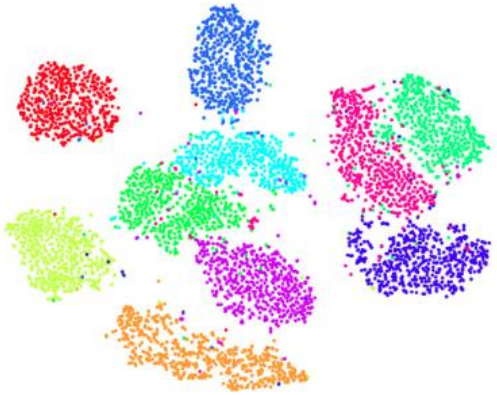
- **Supervised Learning: classification or regression**

  The network makes its guesses, then compare its answers to the known “correct” ones and make adjustments according to its errors.



- **Unsupervised Learning: clustering**

  Searching for a hidden pattern in a data set without known answers.



no labeling

- **Reinforcement Learning: game, robotics, self-driving car, etc.**

  A strategy built on observation.

  https://www.youtube.com/watch?v=JFJkpVWTQVM

Supervised vs Unsupervised Learning

Supervised Learning

Data: (x, y)
x is data, y is label

Goal: Learn a *function* to map x -> y

Examples: Classification, regression, object detection, semantic segmentation, image captioning, etc.

Unsupervised Learning

Training data is cheap

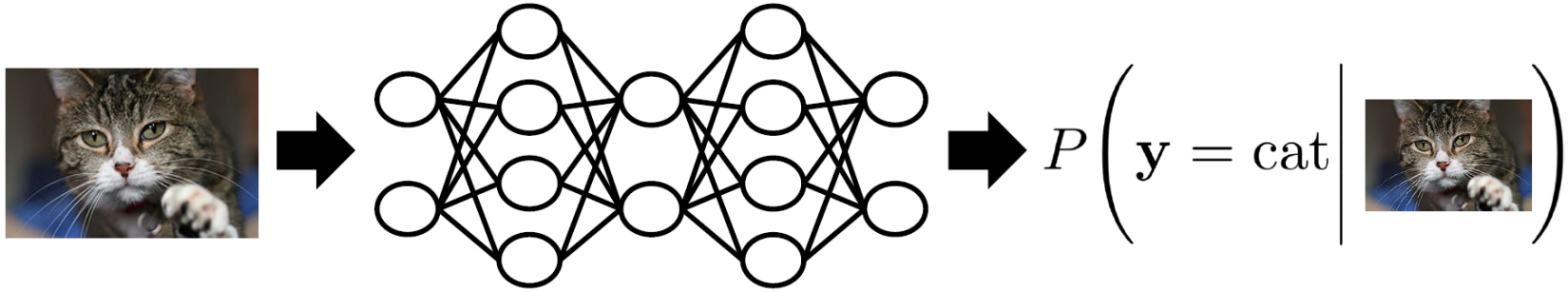
Data: x
Just data, no labels!

Goal: Learn some underlying hidden *structure* of the data

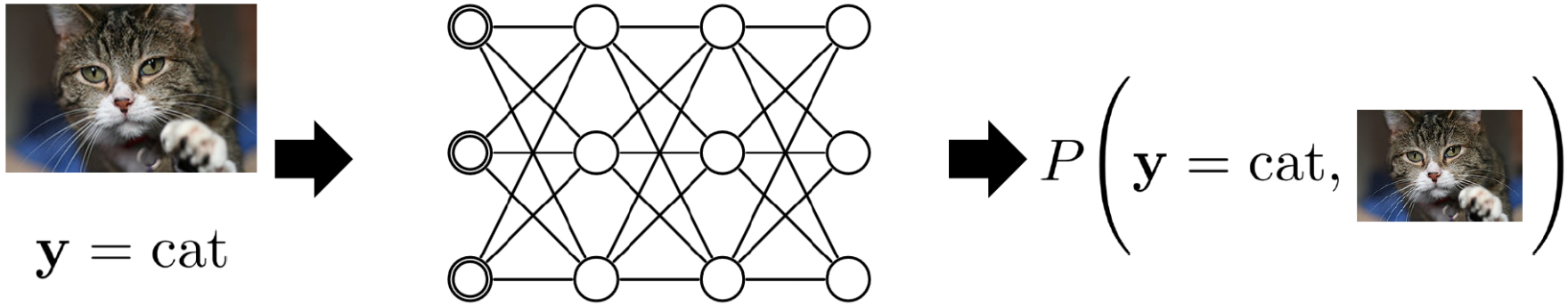
Examples: Clustering, dimensionality reduction, feature learning, density estimation, etc.

Generative model

- Given an observed variable x and a target variable y
- **Discriminative model** is a model of a conditional distribution $P(\mathbf{y}|\mathbf{x})$
  e.g., neural networks (supervised)

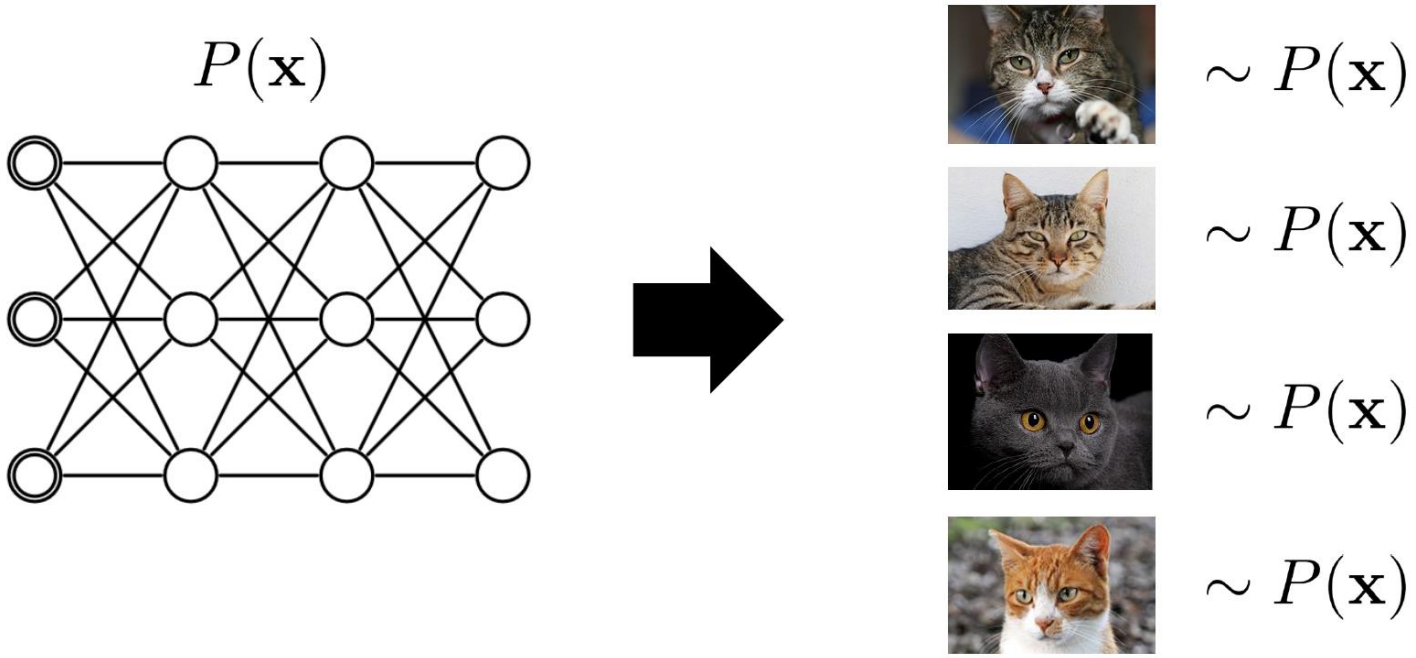


- **Generative model** is a model of a joint distribution $P(\mathbf{x}, \mathbf{y})$ (or $P(\mathbf{x})$)
  e.g., RBM, VAE, GAN (unsupervised)



Generative model

- Generative models model a full probability distribution given data
- *P*(x,y) enables us to generate new data similar to existing (training) data

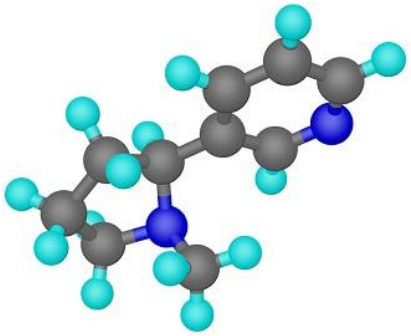


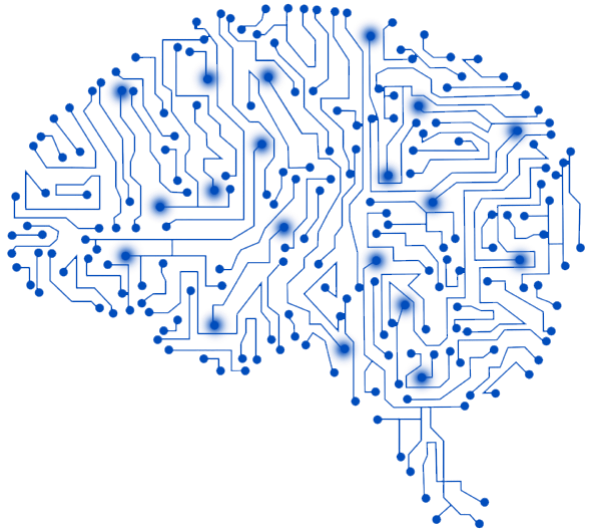
P(x) = distribution of an observed variable x (ex: pixel info [255, 255,255])

KAIST

Inverse design

- **In chemistry or materials science, it is also called the inverse design**





Solubility
Toxicity
Drug efficacy
.
.
.

Input data
Structure (X)

Feature extraction (**L**)

Output data
Property (Y)

Y = f(X); f = neural networks

X' = f(X,Y); f = neural networks

Taxonomy of Generative Models

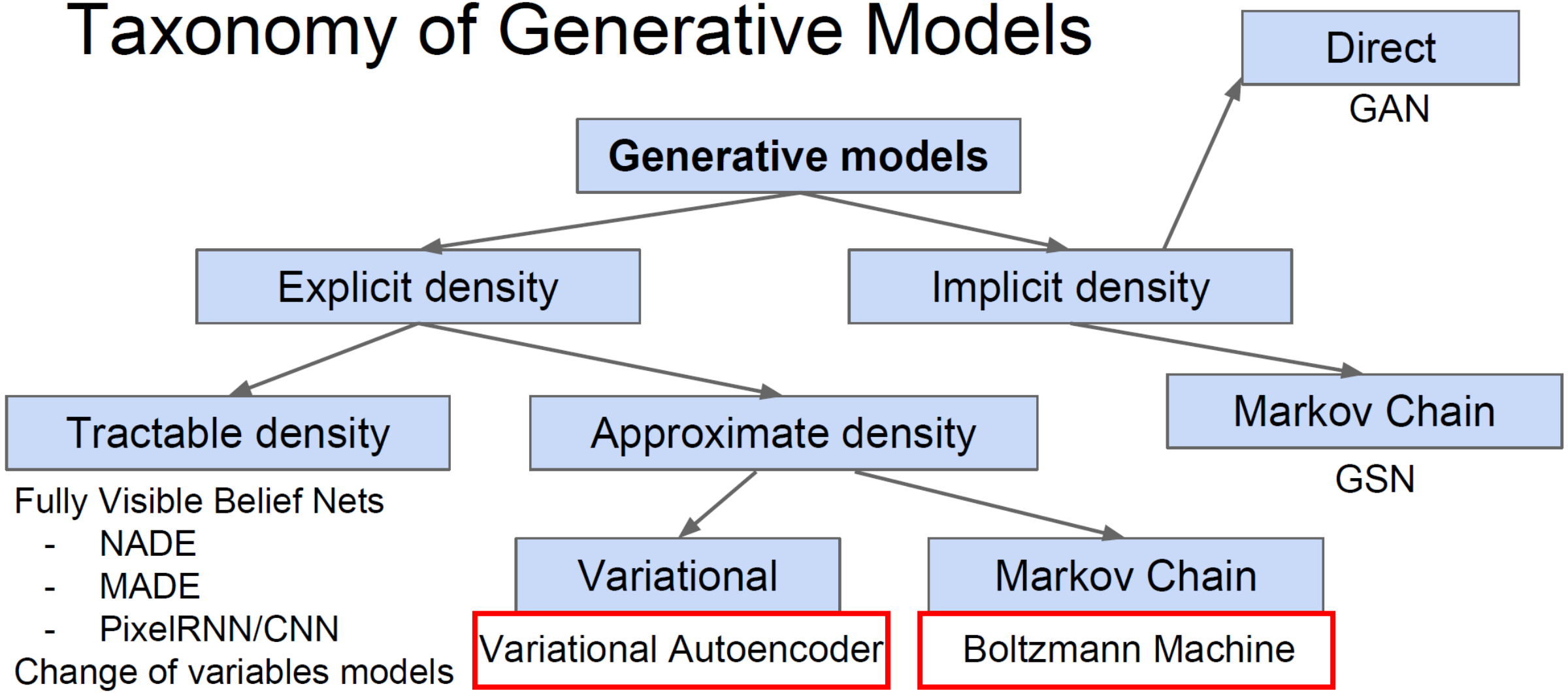


Figure copyright and adapted from Ian Goodfellow, Tutorial on Generative Adversarial Networks, 2017.

Reference paper

RESEARCH ARTICLE Open Access

Molecular generative model based on conditional variational autoencoder for de novo molecular design

Jaechang Lim[1], Seongok Ryu[1], Jin Woo Kim[1] and Woo Youn Kim[1,2*]

Abstract

We propose a molecular generative model based on the conditional variational autoencoder for de novo molecular design. It is specialized to control multiple molecular properties simultaneously by imposing them on a latent space. As a proof of concept, we demonstrate that it can be used to generate drug-like molecules with five target properties. We were also able to adjust a single property without changing the others and to manipulate it beyond the range of the dataset.

Keywords: Molecular design, Conditional variational autoencoder, Deep learning

Reference paper

ACS central science

ACS AuthorChoice

Research Article

Cite This: ACS Cent. Sci. 2018, 4, 268–276

Automatic Chemical Design Using a Data-Driven Continuous Representation of Molecules

Rafael Gómez-Bombarelli,[†,#] Jennifer N. Wei,[‡,#] David Duvenaud,[¶,#] José Miguel Hernández-Lobato,[§,#] Benjamín Sánchez-Lengeling,[‡] Dennis Sheberla,[‡] Jorge Aguilera-Iparraguirre,[†] Timothy D. Hirzel,[†] Ryan P. Adams,[▽,∥] and Alán Aspuru-Guzik*[,‡,⊥]

[†]Kyulux North America Inc., 10 Post Office Square, Suite 800, Boston, Massachusetts 02109, United States

[‡]Department of Chemistry and Chemical Biology, Harvard University, Cambridge, Massachusetts 02138, United States

[¶]Department of Computer Science, University of Toronto, 6 King's College Road, Toronto, Ontario M5S 3H5, Canada

[§]Department of Engineering, University of Cambridge, Trumpington Street, Cambridge CB2 1PZ, U.K.

[▽]Google Brain, Mountain View, California, United States

[∥]Princeton University, Princeton, New Jersey, United States

[⊥]Biologically-Inspired Solar Energy Program, Canadian Institute for Advanced Research (CIFAR), Toronto, Ontario M5S 1M1, Canada

KAIST

Contents

- Restricted Boltzmann machine
- Deep belief network & Auto-encoder
- Variational auto-encoder
- Molecular VAE
- Jointly trained VAE

Restricted Boltzmann machine

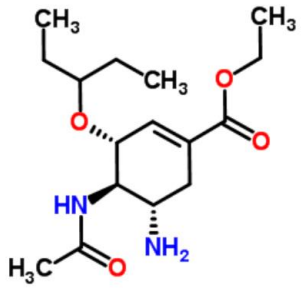
Example of generative model

- 손석희 아나운서의 목소리를 만들어내는 인공지능 모델

(박근혜 전 대통령, 문재인 대통령도 되었었는데, 현재 삭제된 상태)

: https://carpedm20.github.io/tacotron/

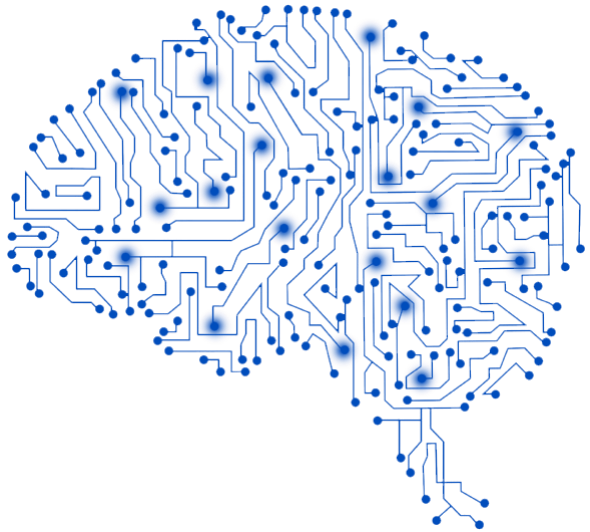
Molecular generative model



Input **Data (X)**

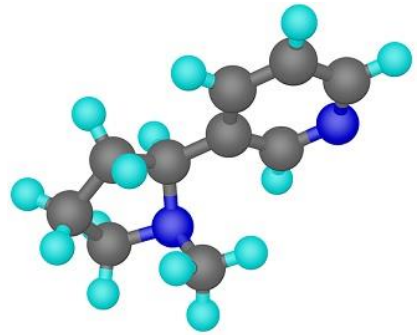
Solubility
Toxicity
Drug efficacy
.
.
.

Input data
Property (Y)



Feature extraction (**L**)

Output
Structure (X')



X' = f(X,Y); f = neural networks

P(X,Y) or P(X)

Boltzmann machine

- **Energy based model** (EBM) is an approximation of the distribution of observed variable x

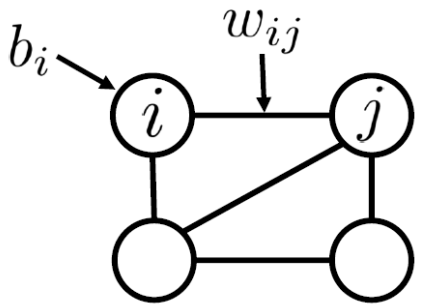
$$P(\boldsymbol{x}) = \frac{\exp\left(-E(\boldsymbol{x})\right)}{Z}$$ Statistical probability (Boltzmann factor)

where E(x) is the energy function, and Z is the partition function

- Assignments with high energy appear less likely in EBM
- Given a graph G(V,E), **Boltzmann machine** (BM) is a joint distribution on a binary vector

$$P(\mathbf{x}) \propto \exp\left(\sum_{i\in V} b_i x_i + \sum_{(i,j)\in E} w_{ij} x_i x_j\right)$$

Onsite energy correlation energy



- To generate new data using BM, we need to learn parameters of BM

Boltzmann machine

- Given training data [$x=(x_1,\ldots,x_n)$] ($P(x)$), learn the distribution $Q(x)$
  - $P(x)$: observed distribution, $Q(x)$: model distribution

- Find the optimal parameter (**b**,**w**) that minimizes the difference between $P(x)$ and $Q(x)$

- Use the Kullback–Leibler (KL) divergence as the **directed divergence** between two distributions (1951)

$$D_{KL}(P \parallel Q) = \sum_x P(\mathrm{x}) \log \frac{P(\mathrm{x})}{Q(\mathrm{x})}$$

KL divergence

$$D_{KL}(P \| Q) = \sum_x P(\mathrm{x}) \log \frac{P(\mathrm{x})}{Q(\mathrm{x})}$$

- Physical interpretation

data entropy

cross entropy

$$D_{KL}(P \| Q) = \sum_x P(\mathrm{x}) \log P(\mathrm{x}) - \sum_x P(\mathrm{x}) \log Q(\mathrm{x})$$

β=1

$$= -S_{data} - \sum_x P(\mathrm{x}) \log \frac{\exp(-\beta E(\mathrm{x}))}{Z}$$

$$= -S_{data} + \beta \sum_x E(\mathrm{x}) P(\mathrm{x}) + \sum_x P(\mathrm{x}) \log Z$$

$$= -S_P + \beta \langle E(x) \rangle_P + \log Z$$

$$= \beta \left[\langle E(x) \rangle_P - k_B T S_P \right] + \beta \left[k_B T \log Z \right]$$

$$= \beta (F_{data} - F_{model}) = -(S_{data} - S_{\mathrm{model}}(\text{cross entropy}))$$

Free energy difference between data and model distributions

KL divergence

$$D_{KL}(P \,||\, Q) = \sum_x P(\mathrm{x}) \log \frac{P(\mathrm{x})}{Q(\mathrm{x})}$$

- Property

  The KL divergence is always non-negative.

$$D_{KL}(P \,||\, Q) = -(S_{data} - S_{\text{model}}(\text{cross entropy})) \geq 0$$

  It is known as Gibbs's inequality, with zero if and only if P = Q everywhere.

- Derivative

$$-\frac{\partial D_{KL}(P \,||\, Q)}{\partial b_i} = \sum_x P(x) x_i - \sum_x Q(x) x_i = \langle x_i \rangle_{data} - \langle x_i \rangle_{\text{model}} = E_{data}[x_i] - E_{\text{model}}[x_i]$$

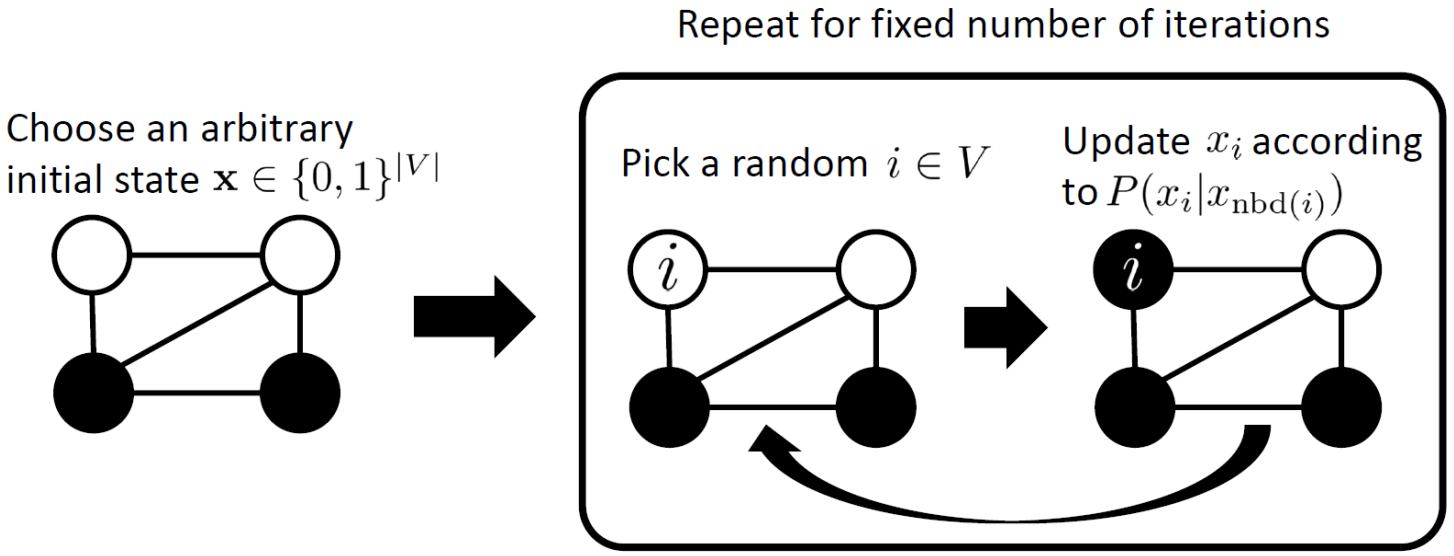
$$-\frac{\partial D_{KL}(P \,||\, Q)}{\partial w_{ij}} = \langle x_i x_j \rangle_{data} - \langle x_i x_j \rangle_{\text{model}} = E_{data}[x_i x_j] - E_{\text{model}}[x_i x_j]$$

Boltzmann machine

- Update b, w using the gradient descent

$$b_i \leftarrow b_i - \alpha \frac{\partial D_{KL}(P \,||\, Q)}{\partial b_i} \qquad w_{ij} \leftarrow w_{ij} - \alpha \frac{\partial D_{KL}(P \,||\, Q)}{\partial w_{ij}}$$

- **Gibbs sampler** is the most popular sampling algorithm in BM



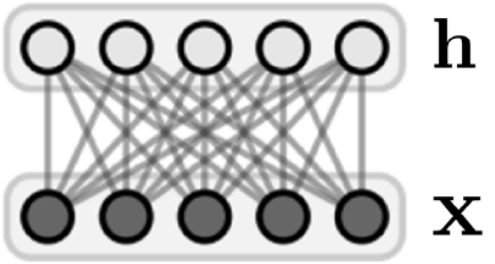
Convergence problem

Restricted Boltzmann machine

- **Restricted Boltzmann machine** (RBM) is a bipartite Boltzmann machine with visible nodes and hidden nodes

$$P(\mathbf{x}) \propto \sum_{\mathbf{h}} \exp\left(\sum_i b_i x_i + \sum_j c_j h_j + \sum_{i,j} w_{ij} x_i h_j\right)$$

Higher order potential



- Hidden nodes can be described by hidden features of visible nodes

- In RBM structure, all hidden nodes are conditionally independent given visible nodes and vise versa

no connection between visible nodes **x** and no connection between hidden nodes **h**

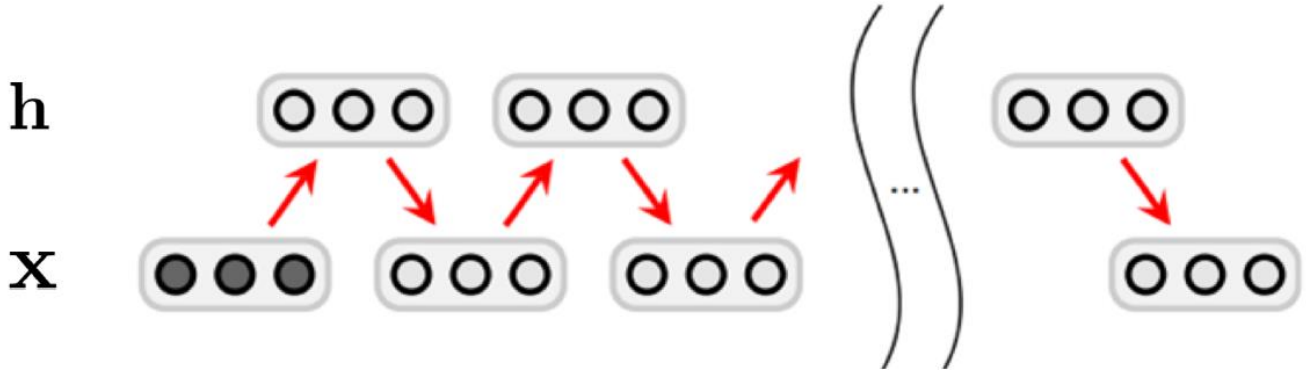
$$P(x_i = 1|\mathbf{h}) = \sigma\left(\sum_j w_{ij} h_j + b_i\right) \quad P(h_j = 1|\mathbf{x}) = \sigma\left(\sum_i w_{ij} x_i + c_j\right)$$

Restricted Boltzmann machine

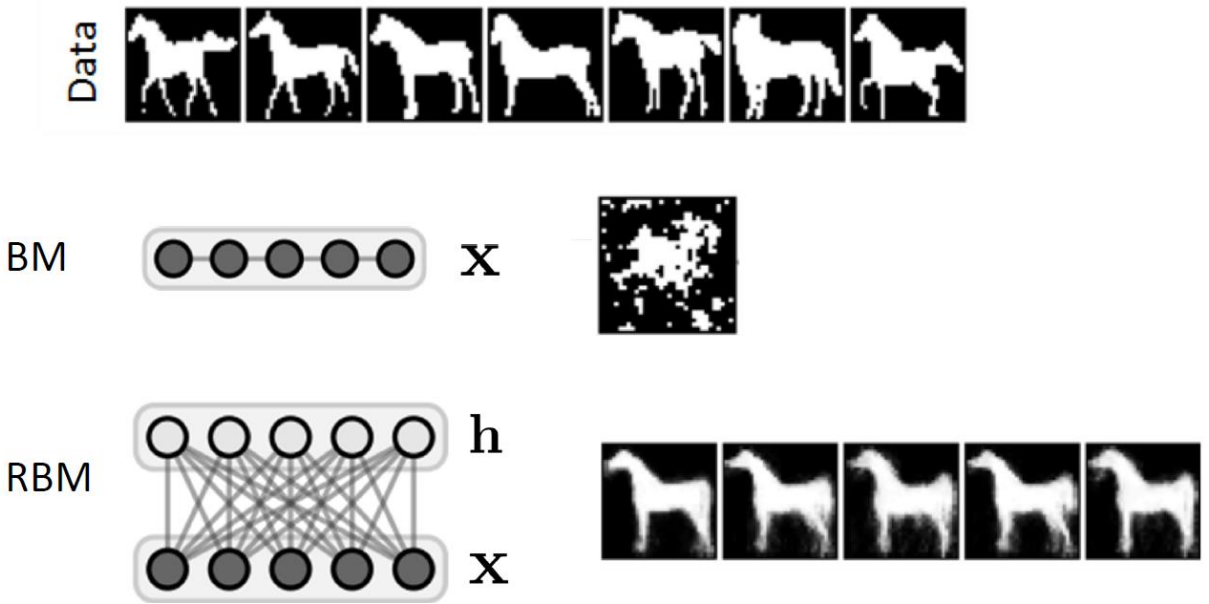
- Given training data [$x=(x_1,\ldots,x_n)$] ($P(x)$), learn the distribution $Q(x)$
  - $P(x)$: observed distribution, $Q(x)$: model distribution
- Find the optimal parameter (**b**,**w, h**) by minimizing the KL divergence
- Gradient descent converges to the global optimum with gradients

Restricted Boltzmann machine

- Due to conditional independence of RBM, block Gibbs sampling is possible



- Samples generated by BM and RBM



Deep belief network & Auto-encoder

7970 citations

Auto-enconder

Reducing the Dimensionality of Data with Neural Networks

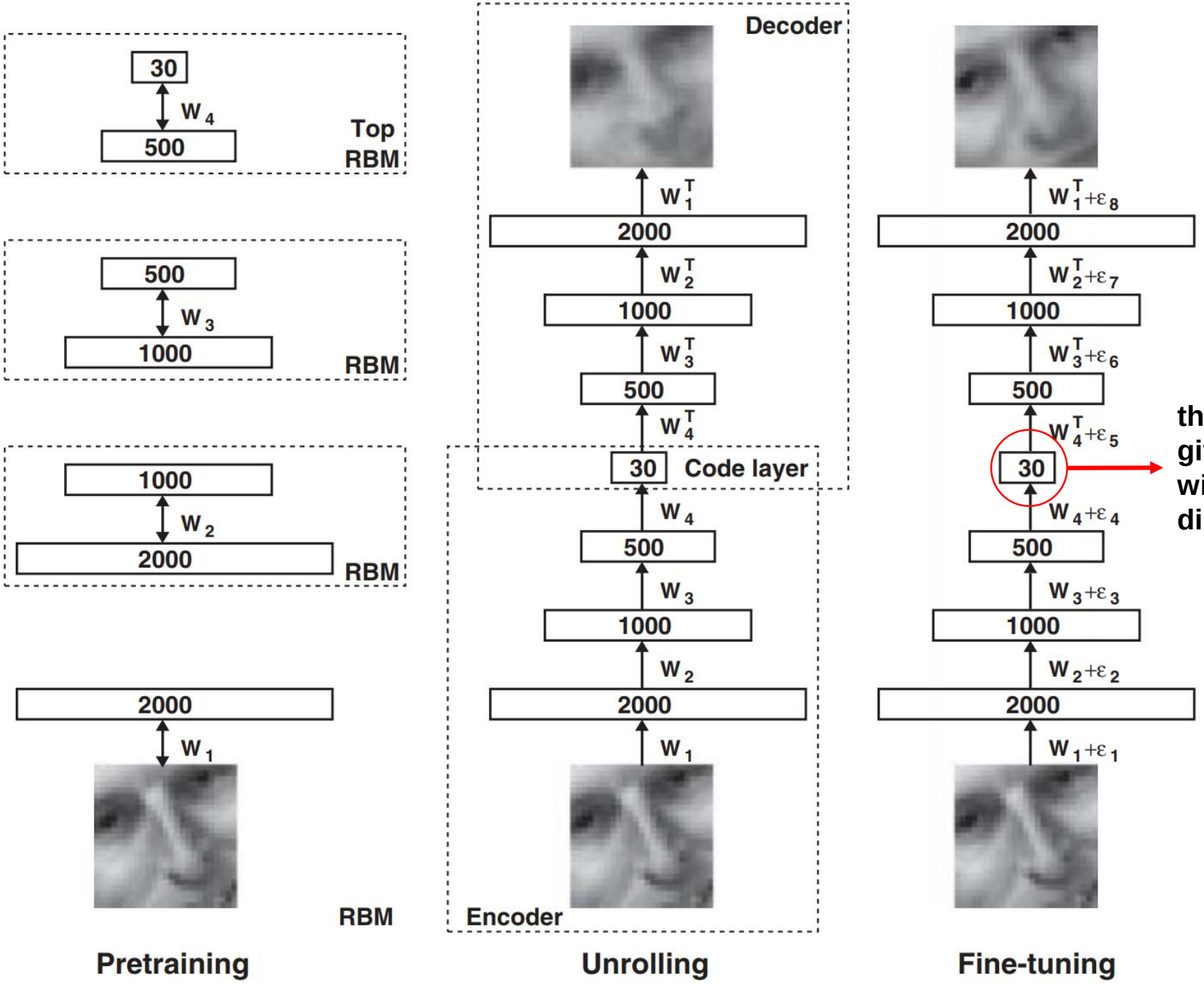
G. E. Hinton* and R. R. Salakhutdinov

High-dimensional data can be converted to low-dimensional codes by training a multilayer neural network with a small central layer to reconstruct high-dimensional input vectors. Gradient descent can be used for fine-tuning the weights in such "autoencoder" networks, but this works well only if the initial weights are close to a good solution. We describe an effective way of initializing the weights that allows deep autoencoder networks to learn low-dimensional codes that work much better than principal components analysis as a tool to reduce the dimensionality of data.

- **'Principle Component Analysis (PCA)'** has been widely used in classification problems in an unsupervised manner.
- However, PCA has limitations in very complex problems.
- This paper proposed an "auto-encoder" method to reconstruct image from extracted feature without loss of accuracy.
- It is called the **'Deep Belief Network'** which is **stacked 'Restricted Boltzmann Machine'**.

Hinton, Geoffrey E., and Ruslan R. Salakhutdinov. "Reducing the dimensionality of data with neural networks." *science* 313.5786 (2006): 504-507.

This is a stack of RBMs.

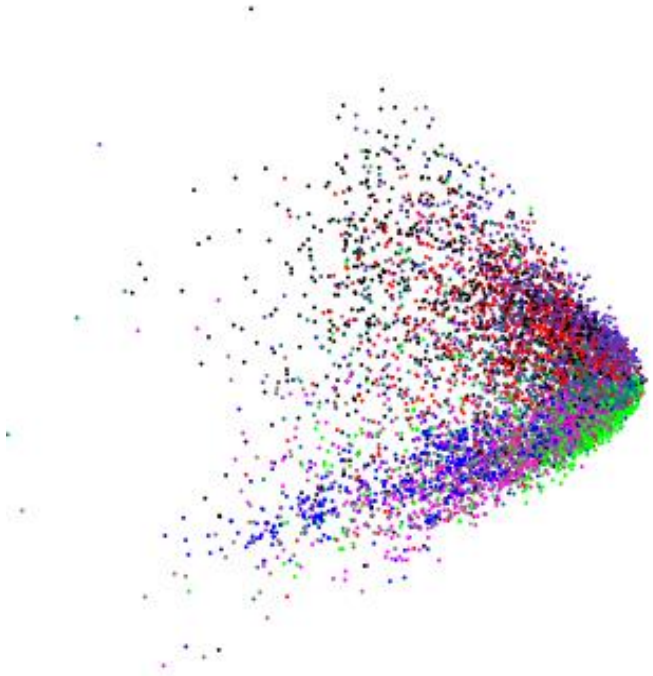


this hidden layer gives a latent vector with reduced dimension.

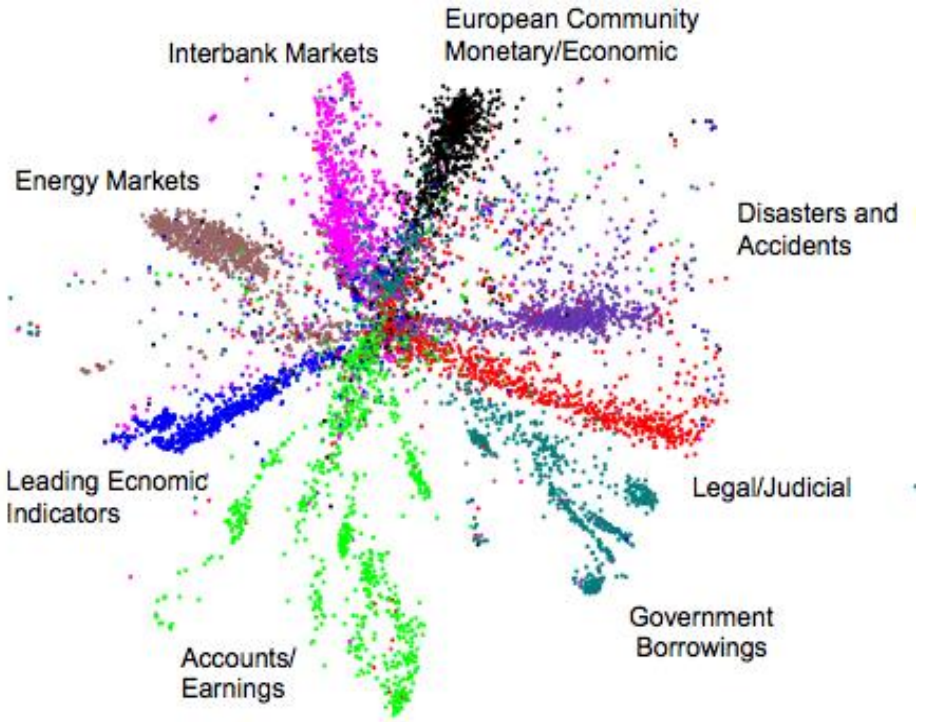
Unsupervised learning

From the raw data, the deep auto-encoder find the features through learning himself.

Principle Component Analysis(PCA)

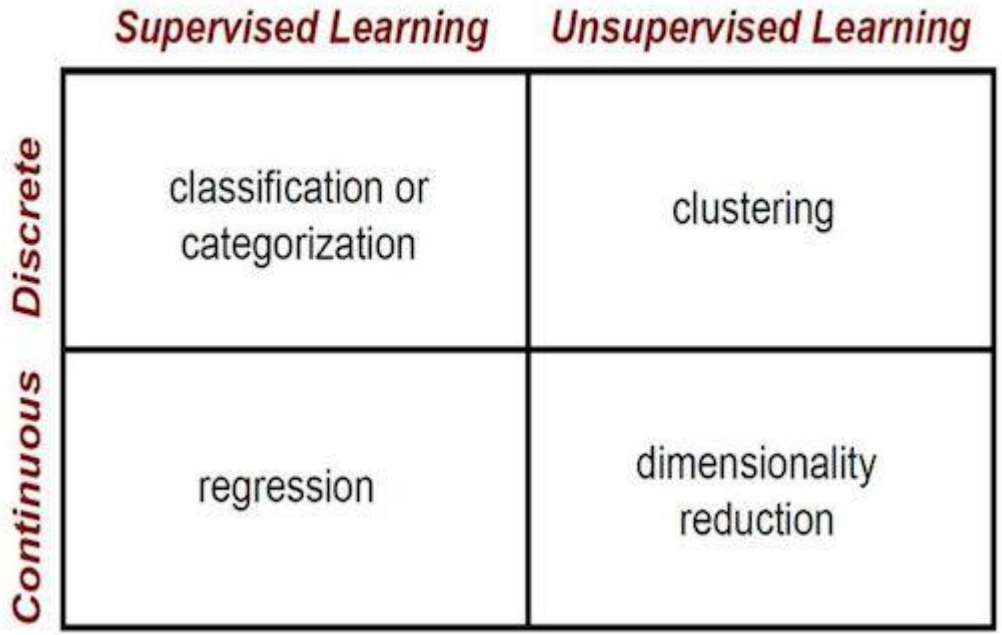


Deep Auto-encoder



Hinton, Geoffrey E., and Ruslan R. Salakhutdinov. "Reducing the dimensionality of data with neural networks." *science* 313.5786 (2006): 504-507.

Supervised vs Unsupervised



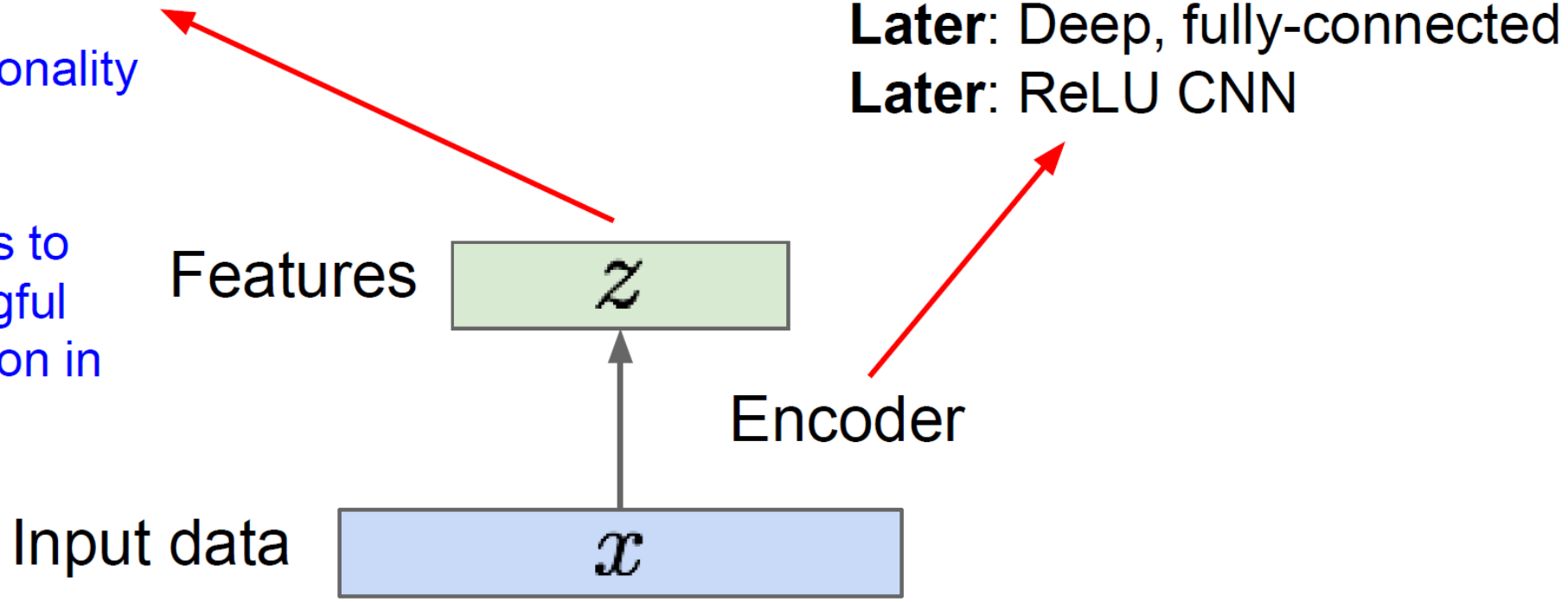
Auto-encoder

z usually smaller than **x** (dimensionality reduction)

Q: Why dimensionality reduction?

A: Want features to capture meaningful factors of variation in data

Originally: Linear + nonlinearity (sigmoid)
Later: Deep, fully-connected
Later: ReLU CNN

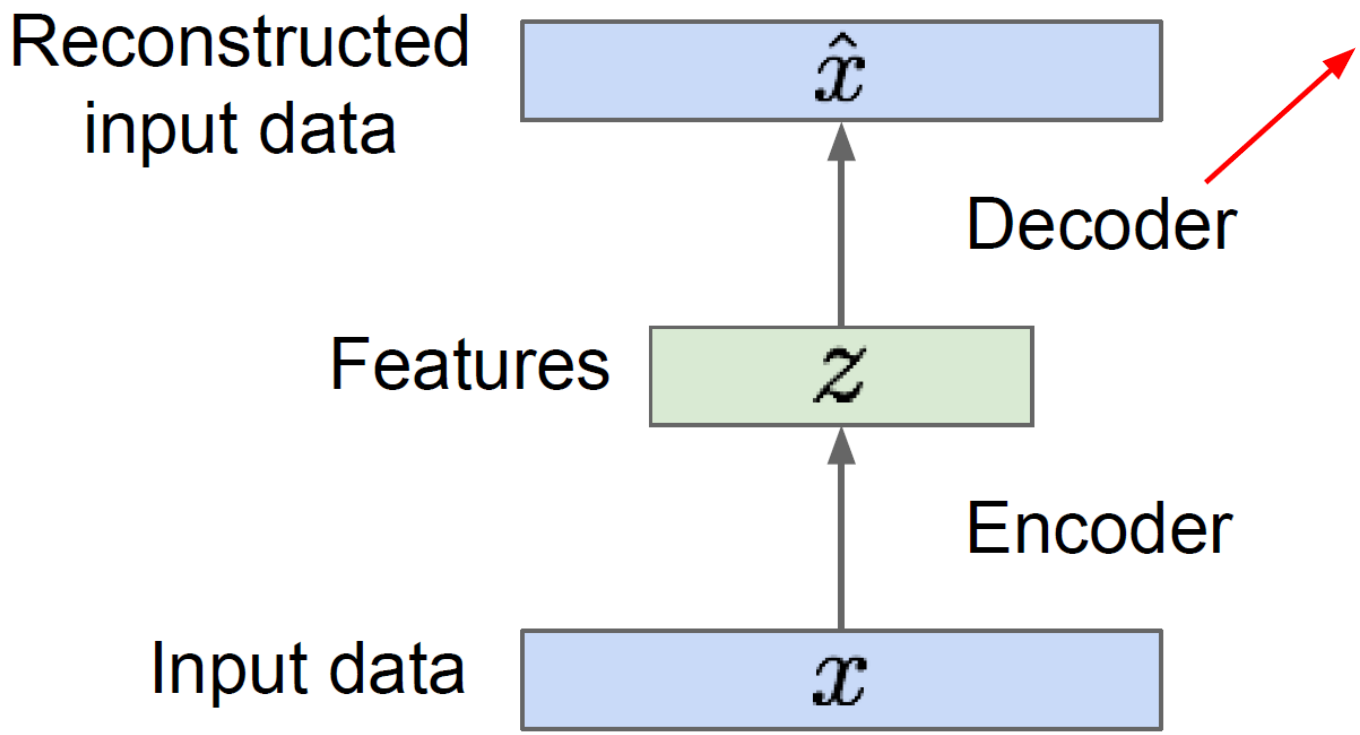


Auto-encoder

How to learn this feature representation?

Train such that features can be used to reconstruct original data

"Autoencoding" - encoding itself



Originally: Linear + nonlinearity (sigmoid)
Later: Deep, fully-connected
Later: ReLU CNN (upconv)

Auto-encoder

Train such that features can be used to reconstruct original data

L2 Loss function: $\|x - \hat{x}\|^2$

Doesn't use labels

Reconstructed input data

$\hat{x}$

Decoder

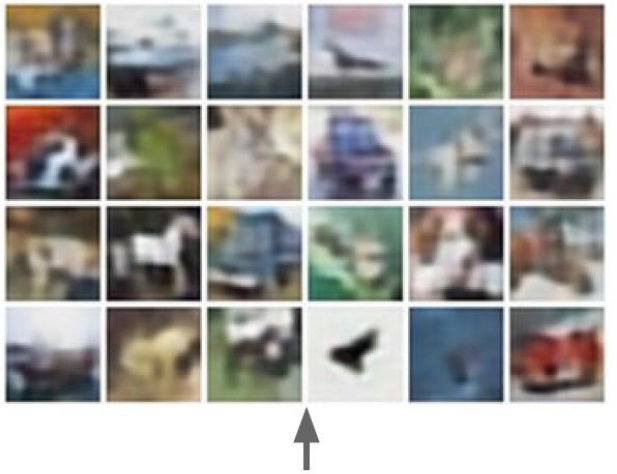
Features

z

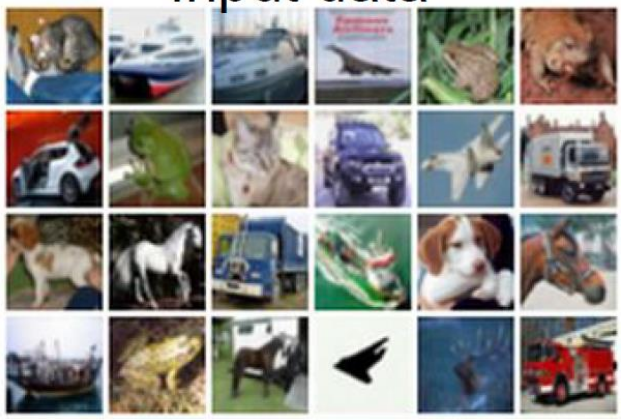
Encoder

Input data

x

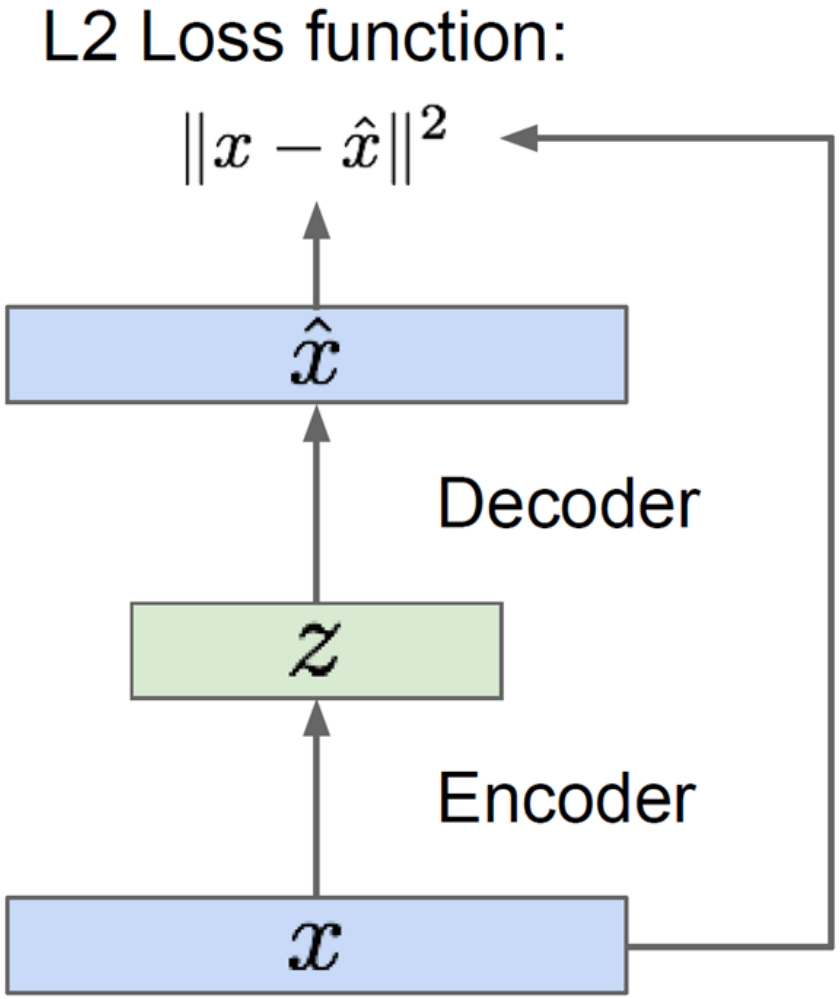


Encoder: 4-layer conv
Decoder: 4-layer upconv



Variational auto-encoder

Auto-encoder vs VAE



- Auto-encoder

  $q_\phi(z \mid x)$: NN encoder

  $p_\theta(x|z)$: NN decoder

- Variational auto-encoder

  $q_\phi(z \mid x)$: NN encoder

  $p_\theta(z)$

  $p_\theta(x|z)$: NN decoder

➔ Sampling new **z'**

$p_\theta(x' \mid z')$: NN decoder

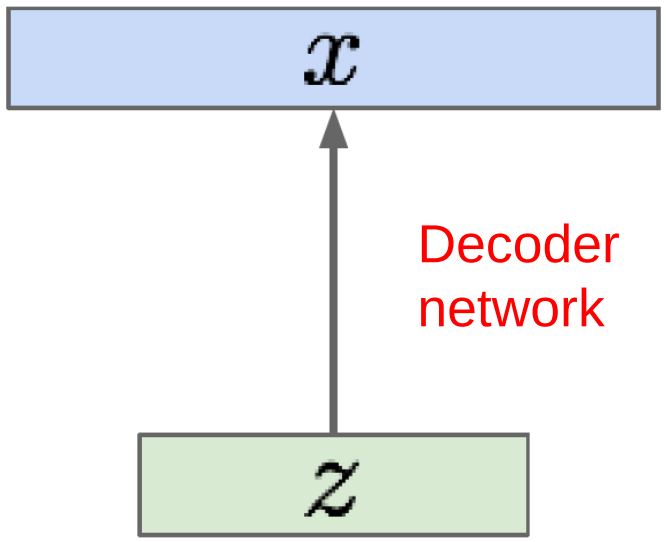
New output x' which is different from any x in the training set

VAE

- Variation of auto-encoders enables us sample from the model to generate data!
- Assume training data $\{x^{(i)}\}_{i=1}^{N}$ is generated from underlying unobserved (latent) representation **z**

Sample from true conditional

$p_{\theta^*}(x \mid z^{(i)})$



Sample from true prior

$p_{\theta^*}(z)$

- We want to estimate the true parameters θ^* of this generative model.
- No information on the true prior $p_{\theta^*}(z)$

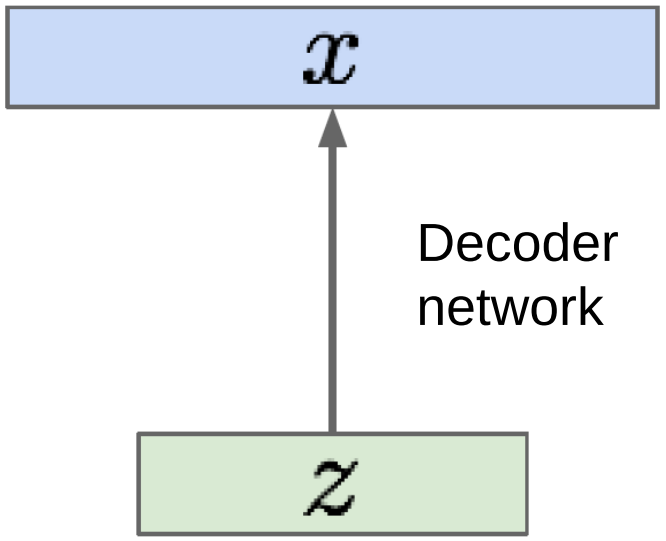
  ➔ choose prior p(z) to be simple, e.g. Gaussian.
- Conditional p(x|z) is complex (generates image) ➔ represent with neural network

VAE

- We want to estimate the true parameters θ^* of this generative model.

Sample from true conditional

$p_{\theta^*}(x \mid z^{(i)})$



Sample from true prior

$p_{\theta^*}(z)$

How to train the model?

Learn model parameters to maximize likelihood of training data

$$p_\theta(x) = \int p_\theta(z) p_\theta(x|z) dz$$

Now with latent z

$p_\theta(z)$: Gaussian prior

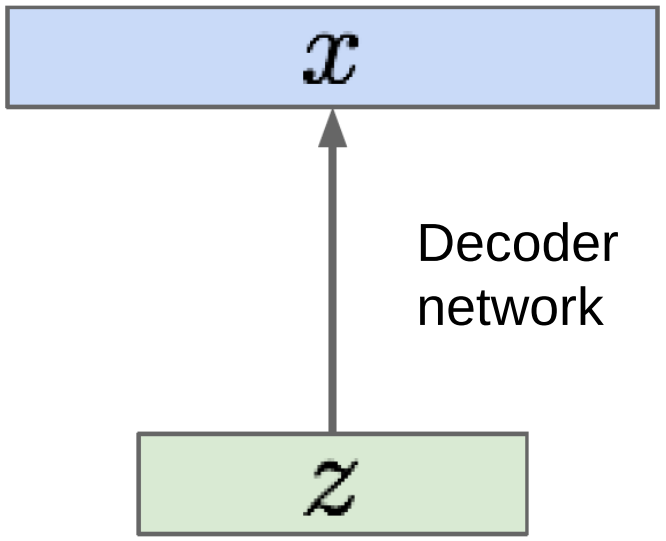
$p_\theta(x|z)$: NN decoder

Problem

- We want to estimate the true parameters of this generative model.

Sample from true conditional

$p_{\theta^*}(x \mid z^{(i)})$



Sample from true prior

$p_{\theta^*}(z)$

$$p_\theta(x) = \int p_\theta(z) p_\theta(x|z) dz$$

Now with latent z

$p_\theta(z)$: Gaussian prior

$p_\theta(x|z)$: NN decoder

Intractable integral over all possible **z**

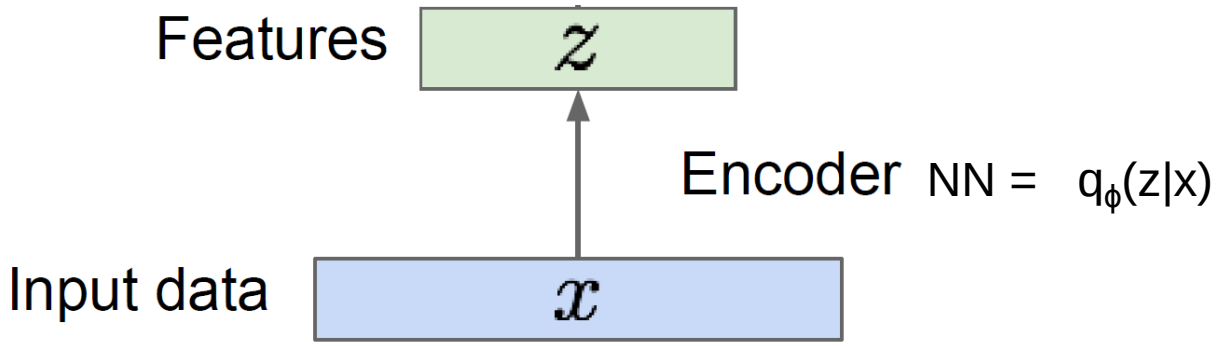
Posterior density is also intractable

$$p_\theta(z|x) = p_\theta(x|z) p_\theta(z) / p_\theta(x)$$

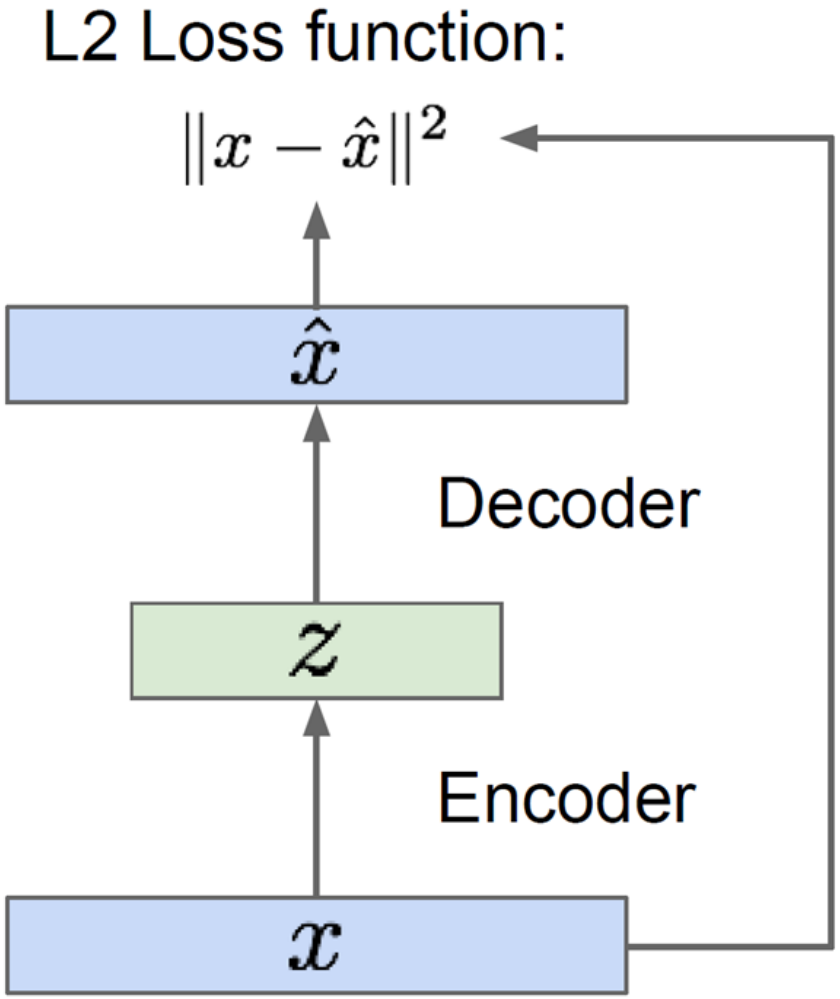
Intractable data likelihood

Solution

In addition to decoder network modeling $p_\theta(x|z)$, define additional encoder network $q_\phi(z|x)$ that approximates $p_\theta(z|x)$ to obtain **z**



Auto-encoder vs VAE



- Auto-encoder

  $q_\phi(z \mid x)$: NN encoder

  $p_\theta(x|z)$: NN decoder

- Variational auto-encoder

  $q_\phi(z \mid x)$: NN encoder

  $p_\theta(z)$: Gaussian

  $p_\theta(x|z)$: NN decoder

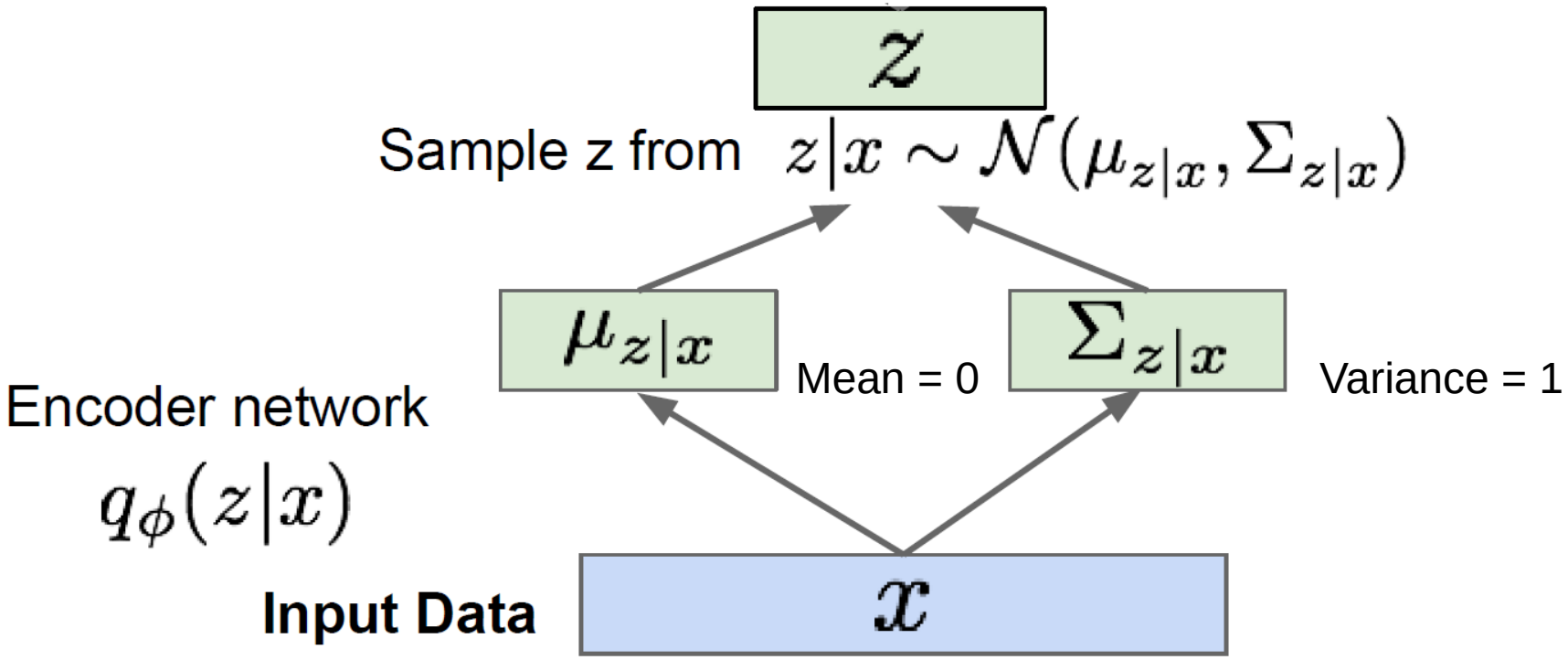
➔ Sampling new **z'**

$p_\theta(x' \mid z')$: NN decoder

New output x' which is different from any x in the training set

Latent vector with Gaussian prior

$p_\theta(z)$: Gaussian prior



Objective function of VAE

With our encoder and decoder networks, let's work out the (log) data likelihood:

$$\log p_\theta(x^{(i)}) = \mathbf{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] \quad (p_\theta(x^{(i)}) \text{ Does not depend on } z)$$

$$= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\right] \quad \text{(Bayes' Rule)}$$

$$= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\frac{q_\phi(z \mid x^{(i)})}{q_\phi(z \mid x^{(i)})}\right] \quad \text{(Multiply by constant)}$$

$$= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z)}\right] + \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z \mid x^{(i)})}\right] \quad \text{(Logarithms)}$$

$$= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z)) + D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z \mid x^{(i)}))$$

The expectation wrt. z (using encoder network) let us write nice KL terms

Objective function of VAE

With our encoder and decoder networks, let's work out the (log) data likelihood:

$$\log p_\theta(x^{(i)}) = \mathbf{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] \quad (p_\theta(x^{(i)}) \text{ Does not depend on } z)$$

$$= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\right] \quad \text{(Bayes' Rule)}$$

$$= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\frac{q_\phi(z \mid x^{(i)})}{q_\phi(z \mid x^{(i)})}\right] \quad \text{(Multiply by constant)}$$

$$= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z)}\right] + \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z \mid x^{(i)})}\right] \quad \text{(Logarithms)}$$

$$= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z)) + D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z \mid x^{(i)}))$$

Decoder network gives $p_\theta(x|z)$, can compute estimate of this term through sampling. (Sampling differentiable through reparam. trick, see paper.)

This KL term (between Gaussians for encoder and z prior) has nice closed-form solution!

$p_\theta(z|x)$ intractable (saw earlier), can't compute this KL term :(But we know KL divergence always >= 0.

Objective function of VAE

With our encoder and decoder networks, let's work out the (log) data likelihood:

$$\log p_\theta(x^{(i)}) = \mathbf{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] \quad (p_\theta(x^{(i)}) \text{ Does not depend on } z)$$

$$= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\right] \quad \text{(Bayes' Rule)}$$

$$= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\frac{q_\phi(z \mid x^{(i)})}{q_\phi(z \mid x^{(i)})}\right] \quad \text{(Multiply by constant)}$$

$$= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z)}\right] + \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z \mid x^{(i)})}\right] \quad \text{(Logarithms)}$$

$$= \underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)}, \theta, \phi)} + \underbrace{D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z \mid x^{(i)}))}_{\geq 0}$$

Tractable lower bound which we can take gradient of and optimize! ($p_\theta(x|z)$ differentiable, KL term differentiable)

Objective function of VAE

With our encoder and decoder networks, let's work out the (log) data likelihood:

$$\log p_\theta(x^{(i)}) = \mathbf{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] \quad (p_\theta(x^{(i)}) \text{ Does not depend on } z)$$

$$= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\right] \quad \text{(Bayes' Rule)}$$

$$= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\frac{q_\phi(z \mid x^{(i)})}{q_\phi(z \mid x^{(i)})}\right] \quad \text{(Multiply by constant)}$$

$$= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z)}\right] + \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z \mid x^{(i)})}\right] \quad \text{(Logarithms)}$$

$$= \underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)}, \theta, \phi)} + \underbrace{D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z \mid x^{(i)}))}_{>0}$$

$$\log p_\theta(x^{(i)}) \geq \mathcal{L}(x^{(i)}, \theta, \phi)$$

Variational lower bound ("ELBO")

$$\theta^*, \phi^* = \arg\max_{\theta,\phi} \sum_{i=1}^{N} \mathcal{L}(x^{(i)}, \theta, \phi)$$

Training: Maximize lower bound

Objective function of VAE

With our encoder and decoder networks, let's work out the (log) data likelihood:

$$\log p_\theta(x^{(i)}) = \mathbf{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log p_\theta(x^{(i)})\right] \quad (p_\theta(x^{(i)}) \text{ Does not depend on } z)$$

$$= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\right] \quad \text{(Bayes' Rule)}$$

$$= \mathbf{E}_z\left[\log \frac{p_\theta(x^{(i)} \mid z)p_\theta(z)}{p_\theta(z \mid x^{(i)})}\frac{q_\phi(z \mid x^{(i)})}{q_\phi(z \mid x^{(i)})}\right] \quad \text{(Multiply by constant)}$$

$$= \mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z)}\right] + \mathbf{E}_z\left[\log \frac{q_\phi(z \mid x^{(i)})}{p_\theta(z \mid x^{(i)})}\right] \quad \text{(Logarithms)}$$

$$= \underbrace{\mathbf{E}_z\left[\log p_\theta(x^{(i)} \mid z)\right] - D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z))}_{\mathcal{L}(x^{(i)}, \theta, \phi)} + \underbrace{D_{KL}(q_\phi(z \mid x^{(i)}) \,||\, p_\theta(z \mid x^{(i)}))}_{\geq 0}$$

Reconstruct the input data

L2 loss
Cross entropy
Softmax

Make approximate posterior distribution close to prior

$$\log p_\theta(x^{(i)}) \geq \mathcal{L}(x^{(i)}, \theta, \phi)$$

Variational lower bound ("ELBO")

$$\theta^*, \phi^* = \arg\max_{\theta,\phi} \sum_{i=1}^{N} \mathcal{L}(x^{(i)}, \theta, \phi)$$

Training: Maximize lower bound

Molecular VAE

Reference paper

RESEARCH ARTICLE Open Access

Molecular generative model based on conditional variational autoencoder for de novo molecular design

Jaechang Lim[1], Seongok Ryu[1], Jin Woo Kim[1] and Woo Youn Kim[1,2*]

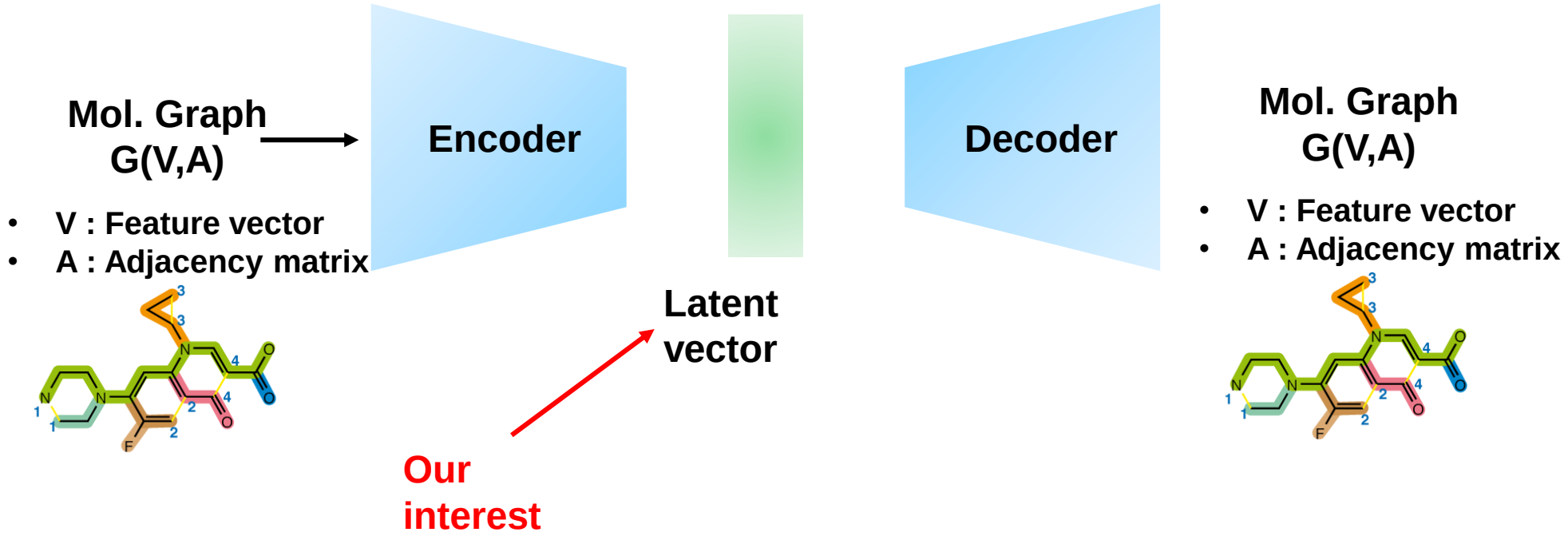
Abstract

We propose a molecular generative model based on the conditional variational autoencoder for de novo molecular design. It is specialized to control multiple molecular properties simultaneously by imposing them on a latent space. As a proof of concept, we demonstrate that it can be used to generate drug-like molecules with five target properties. We were also able to adjust a single property without changing the others and to manipulate it beyond the range of the dataset.

Keywords: Molecular design, Conditional variational autoencoder, Deep learning

Molecular autoencoder

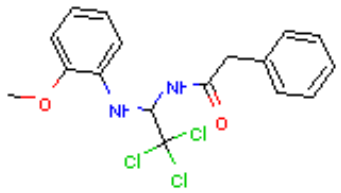
Molecular Structure Embedding via Autoencoder

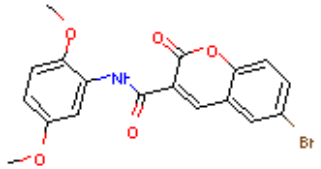


KAIST

Variational autoencoder (VAE)

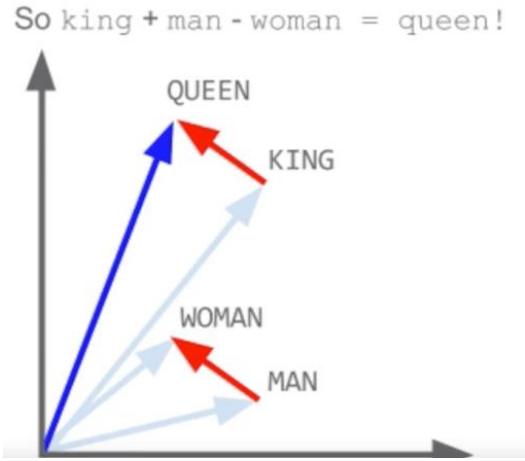
✓ **Embedding vector operation**





Encoder

Encoder

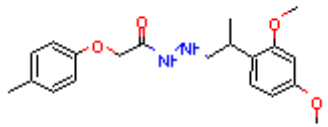


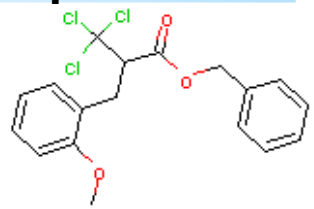
0.9 x Latent vector 1 ✚ 0.1 x Latent vector 2 ⟶ Latent vector 3

0.1 x Latent vector 1 ✚ 0.9 x Latent vector 2 → Latent vector 4

Decoder

Decoder





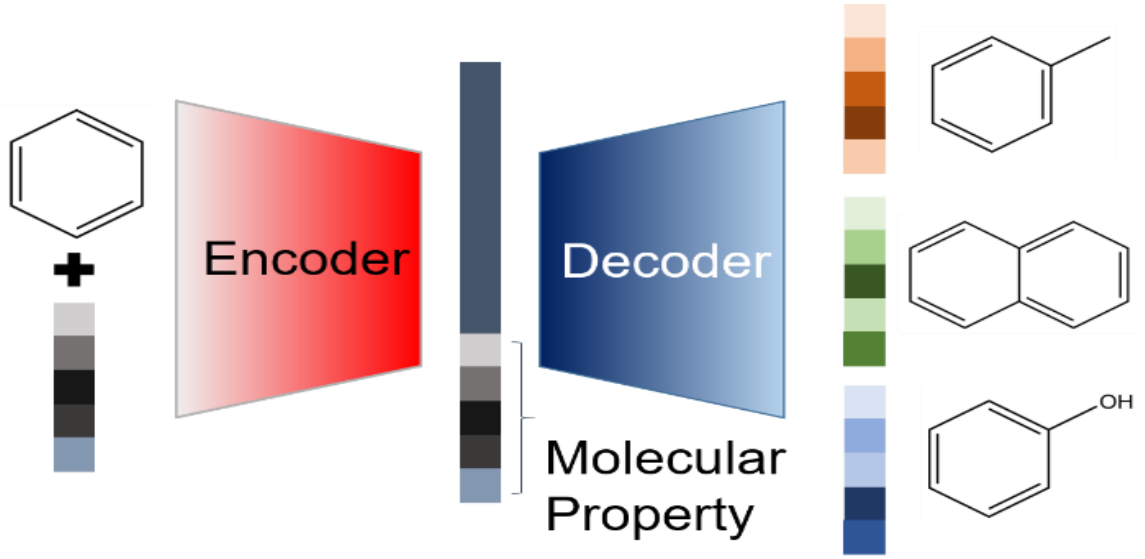
→ Represent molecules through vectors in a latent space.

→ Discontinuous graph/string representation to continuous vector representation

KAIST

Conditional variational autoencoder (CVAE)

it “generates molecules with desired properties”.



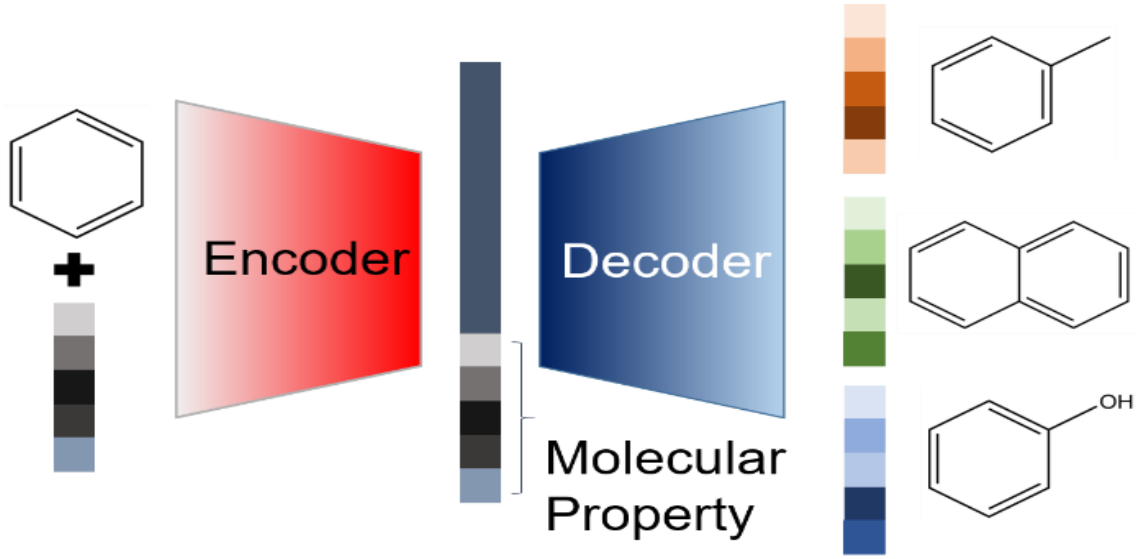
- ✓ controlling several properties simultaneously
- ✓ embedding molecular properties into the condition vector
- ✓ Ex) Molecular weight, partition coefficient (LogP), HBD, HBA, TPSA
- ✓ 500,000 molecules randomly selected from the ZINC data set
- ✓ 400,000 for training and 100,000 for test

KAIST

J. Lim, S. Ryu, J. W. Kim, W. Y. Kim, “Molecular generative model based on conditional variational autoencoder for de novo molecular design”, *J. Cheminform*. 10:31 (2018)

Conditional variational autoencoder (CVAE)

it “generates molecules with desired properties”.



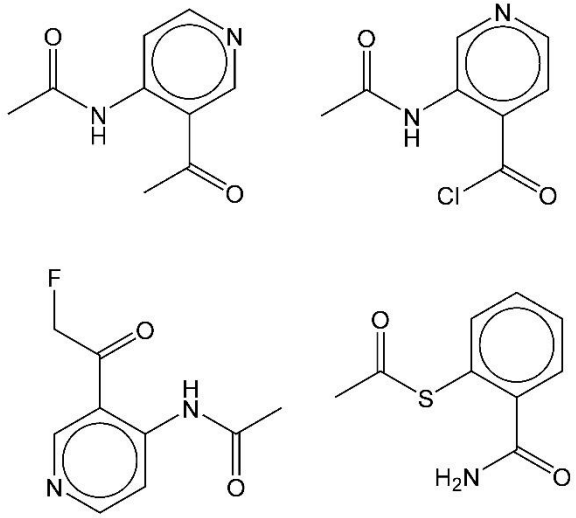
$$E[\log P(X|z)] - D_{KL}[Q(z|X) \parallel P(z)] \Rightarrow E[\log P(X|z,c)] - D_{KL}[Q(z|X,c) \parallel P(z|c)]$$

J. Lim, S. Ryu, J. W. Kim, W. Y. Kim, “Molecular generative model based on conditional variational autoencoder for de novo molecular design”, *J. Cheminform.* 10:31 (2018)

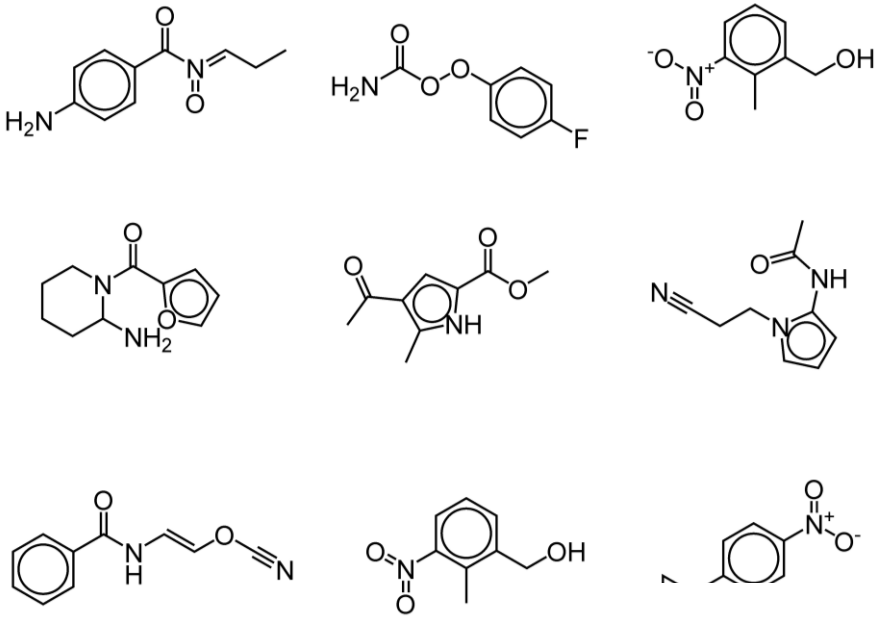
CVAE – Example 1

✓ **5 fixed target properties**

A little change in molecular structure

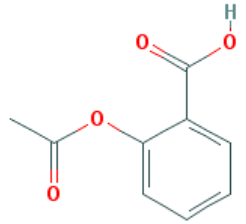


A drastic change in molecular structure



Molecular weight	Log P	HBD	HBA	TPSA
180.04	1.31	1	3	63.6

Aspirin



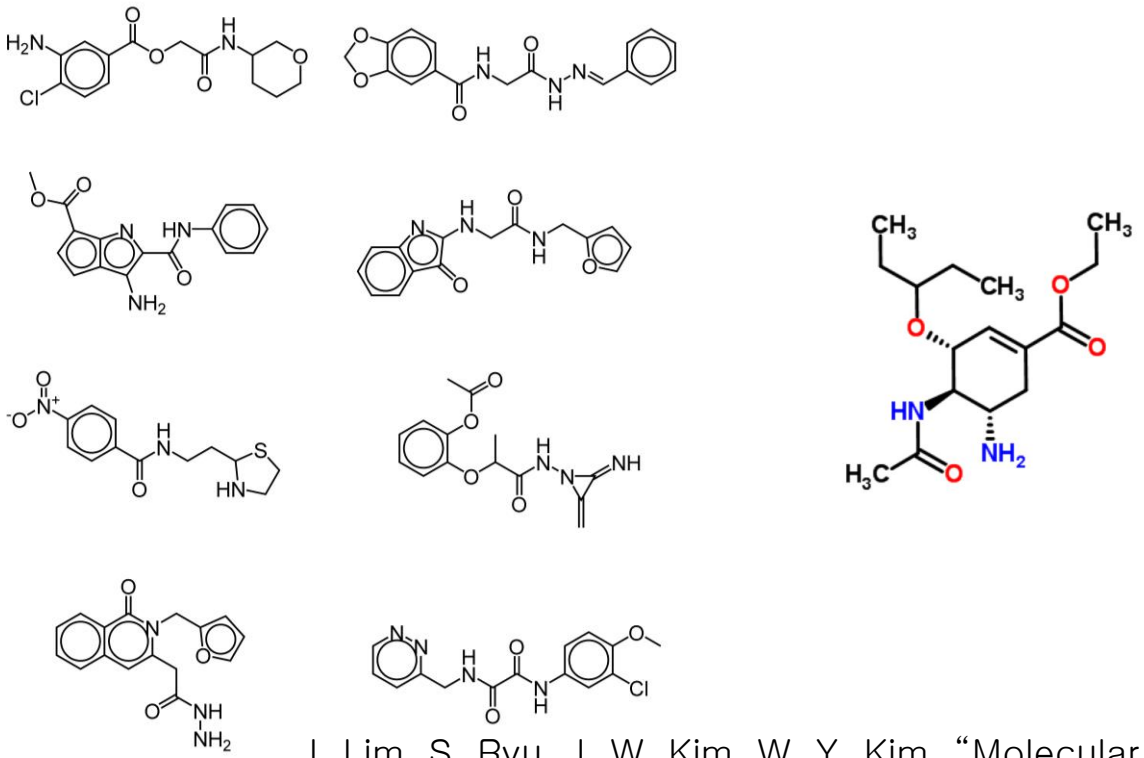
KAIST

CVAE – Example 2

✓ **5 fixed target properties**

Property of Tamiflu

Molecular weight	Log P	HBD	HBA	TPSA
312.2	1.285	2	5	90.64



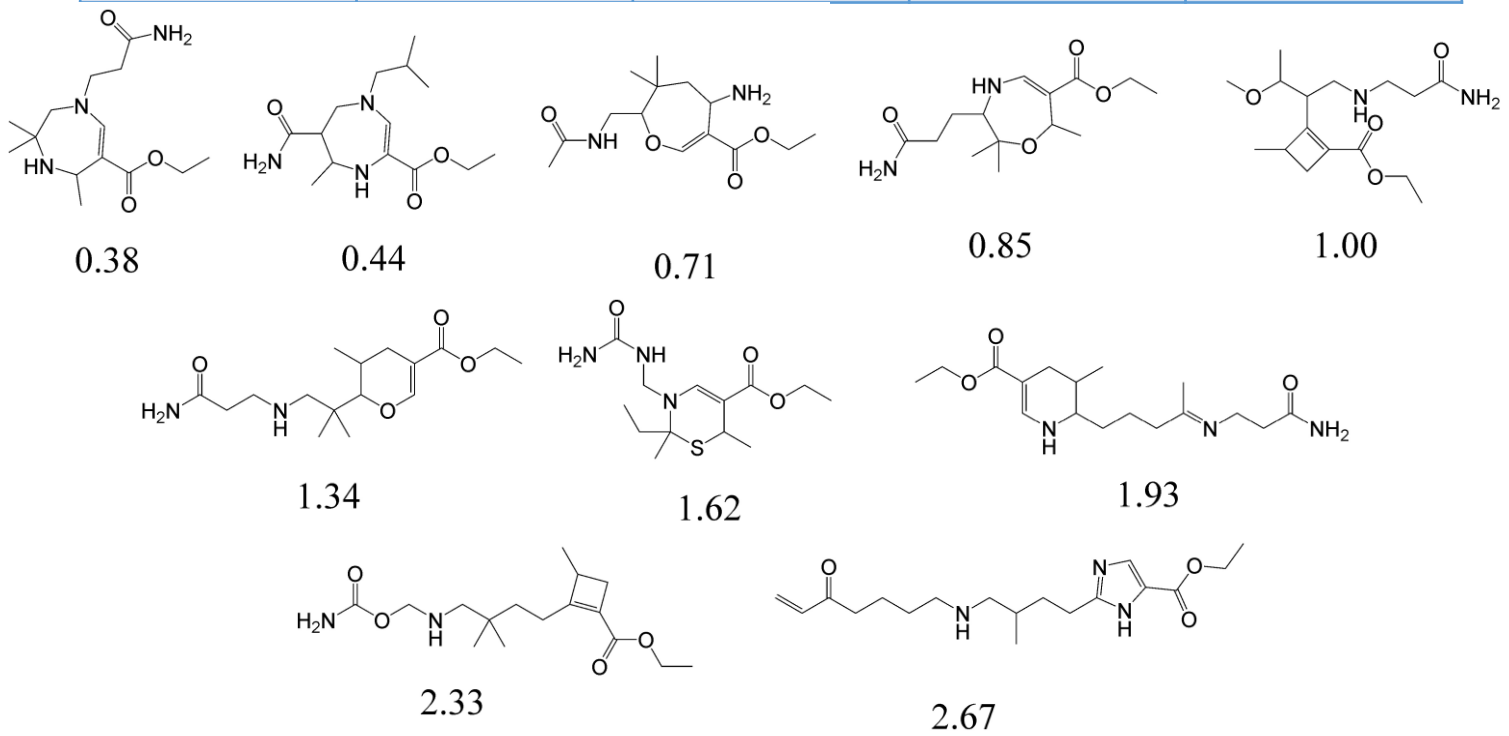
J. Lim, S. Ryu, J. W. Kim, W. Y. Kim, "Molecular generative model based on conditional variational autoencoder for de novo molecular design", *J. Cheminform*. 10:31 (2018)

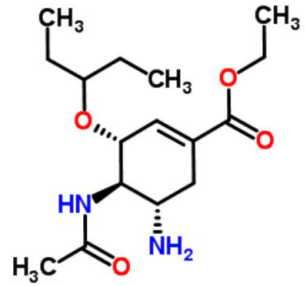
CVAE – Example 3

✓ Tuning one property while fixing the other four properties

Gradual change of LogP from 0.3 to 3.0 while keeping all other properties of Tamiflu fixed

Molecular weight	Log P	HBD	HBA	TPSA
312.2	***1.285***	2	5	90.64





J. Lim, S. Ryu, J. W. Kim, W. Y. Kim, "Molecular generative model based on conditional variational autoencoder for de novo molecular design", *J. Cheminform.* 10:31 (2018)

Jointly trained VAE

Reference paper

ACS central science

ACS AuthorChoice

Research Article

Cite This: ACS Cent. Sci. 2018, 4, 268–276

Automatic Chemical Design Using a Data-Driven Continuous Representation of Molecules

Rafael Gómez-Bombarelli,[†,#] Jennifer N. Wei,[‡,#] David Duvenaud,[¶,#] José Miguel Hernández-Lobato,[§,#] Benjamín Sánchez-Lengeling,[‡] Dennis Sheberla,[‡] Jorge Aguilera-Iparraguirre,[†] Timothy D. Hirzel,[†] Ryan P. Adams,[▽,∥] and Alán Aspuru-Guzik*[,‡,⊥]

[†]Kyulux North America Inc., 10 Post Office Square, Suite 800, Boston, Massachusetts 02109, United States

[‡]Department of Chemistry and Chemical Biology, Harvard University, Cambridge, Massachusetts 02138, United States

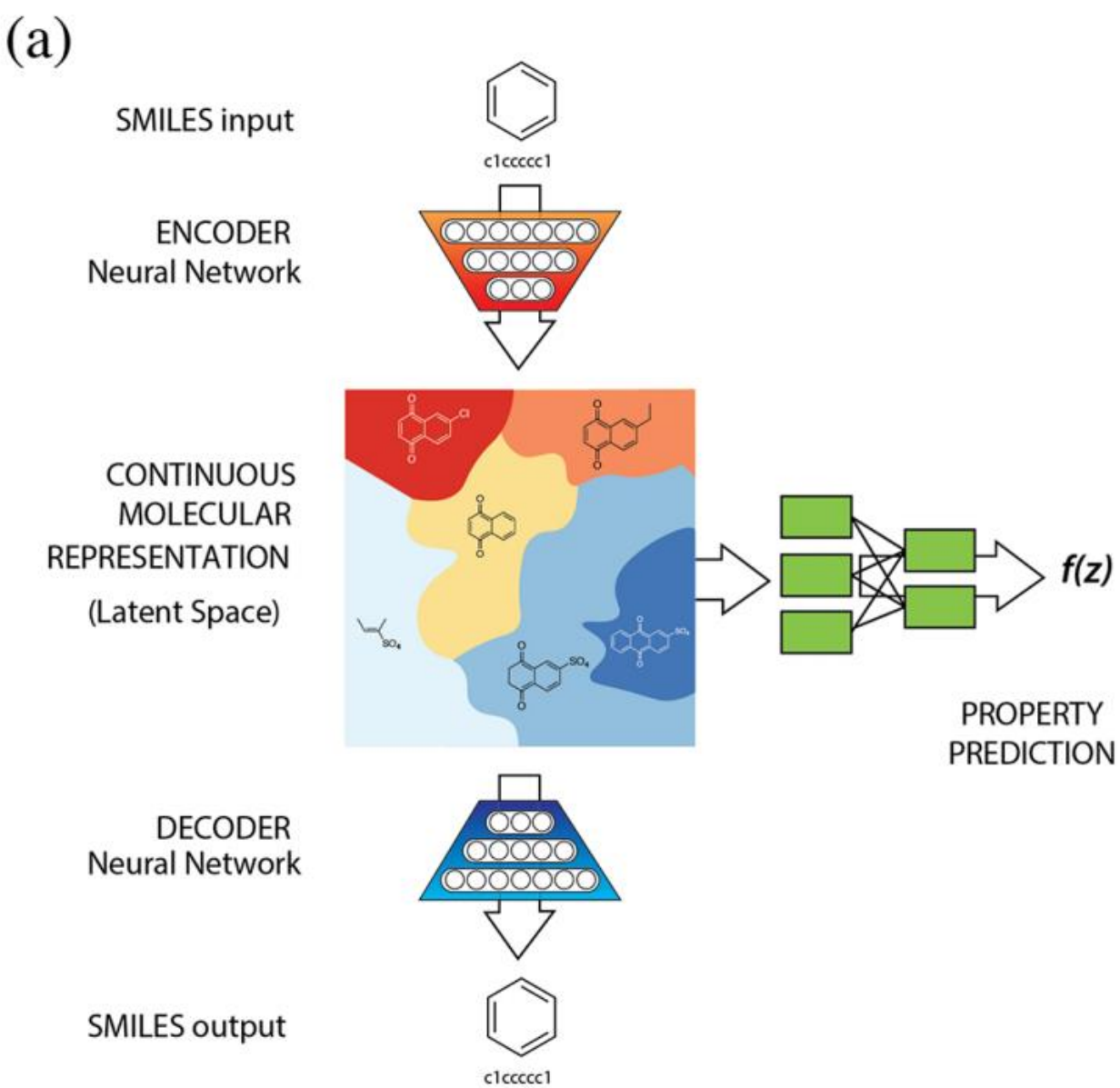
[¶]Department of Computer Science, University of Toronto, 6 King's College Road, Toronto, Ontario M5S 3H5, Canada

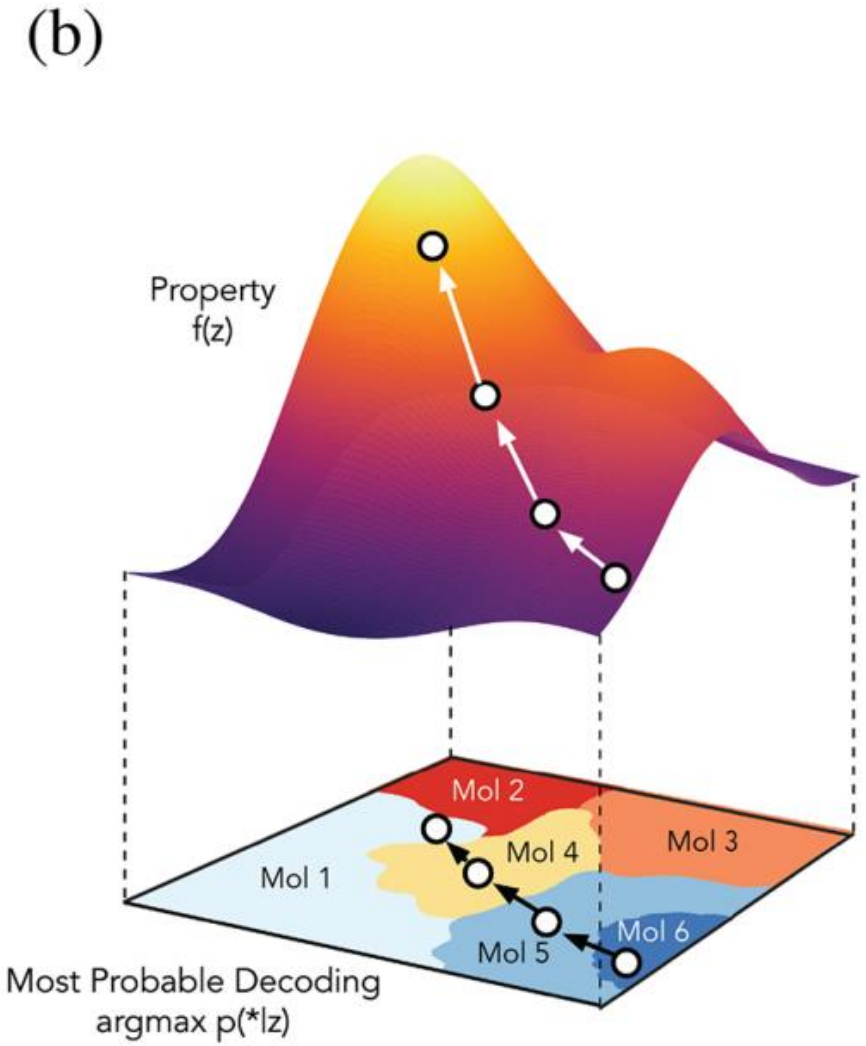
[§]Department of Engineering, University of Cambridge, Trumpington Street, Cambridge CB2 1PZ, U.K.

[▽]Google Brain, Mountain View, California, United States

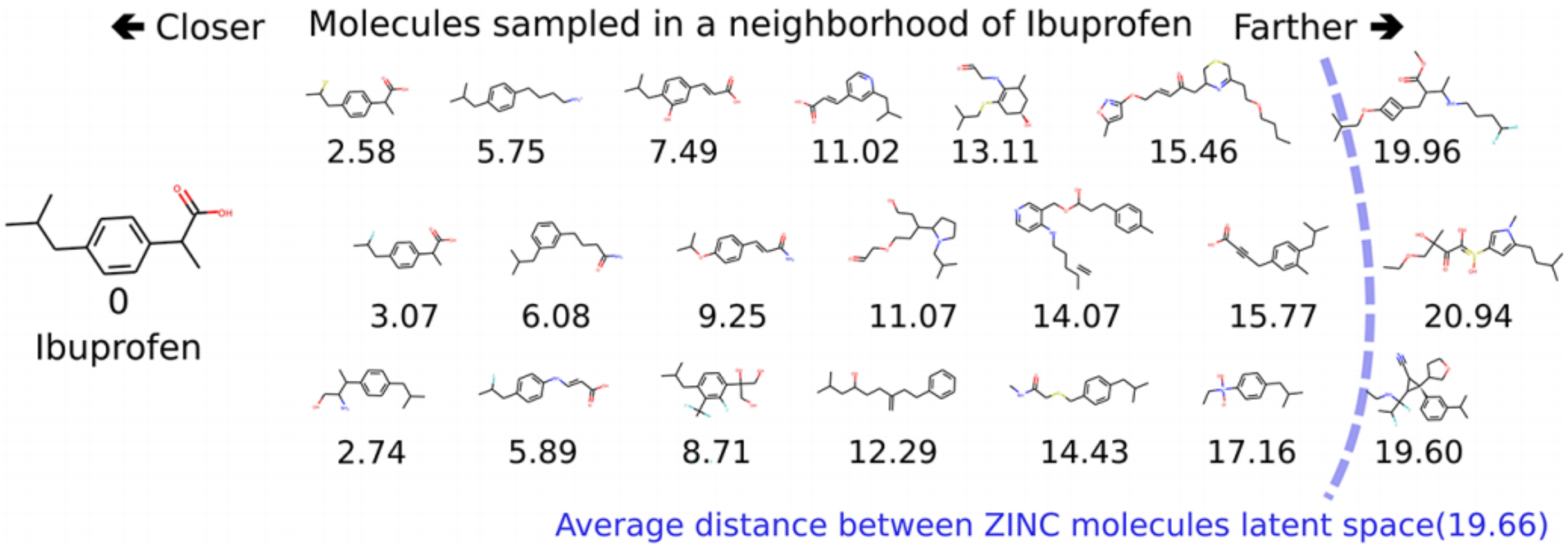
[∥]Princeton University, Princeton, New Jersey, United States

[⊥]Biologically-Inspired Solar Energy Program, Canadian Institute for Advanced Research (CIFAR), Toronto, Ontario M5S 1M1, Canada

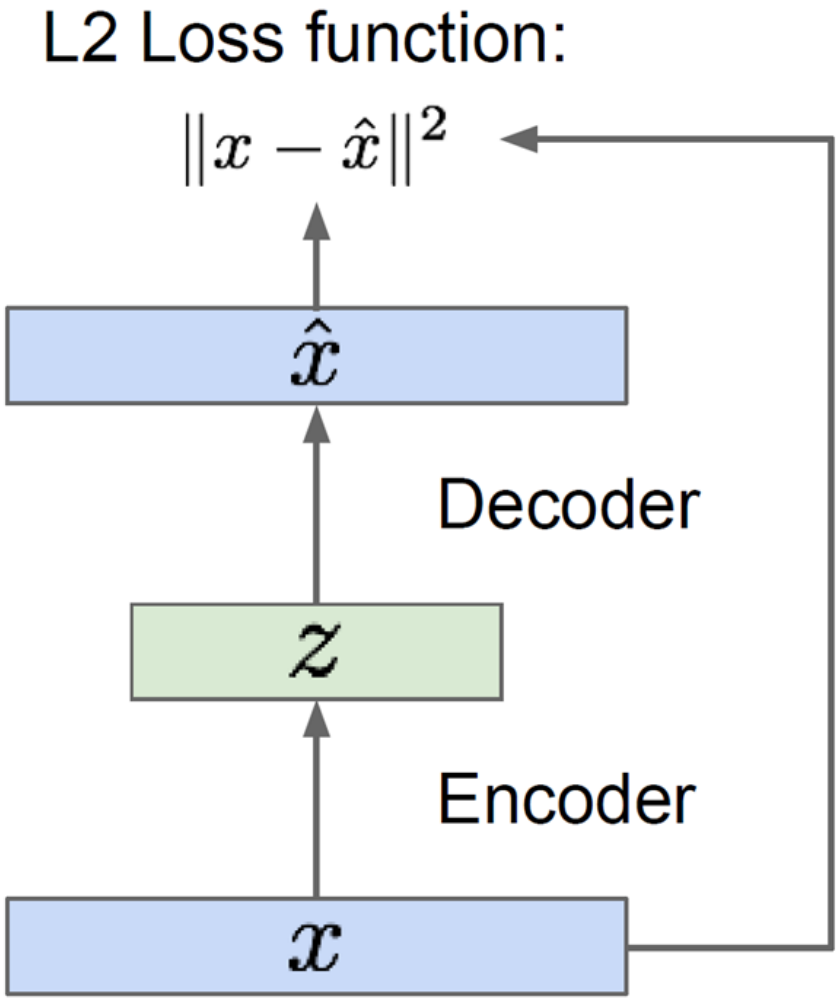

(a)
SMILES input
c1ccccc1
ENCODER
Neural Network
CONTINUOUS
MOLECULAR
REPRESENTATION
(Latent Space)
f(z)
PROPERTY
PREDICTION
DECODER
Neural Network
SMILES output
c1ccccc1


(b)
Property
f(z)
Mol 1
Mol 2
Mol 3
Mol 4
Mol 5
Mol 6
Most Probable Decoding
argmax p(*|z)

Sampling in latent space



Summary



- Auto-encoder

  $q_\phi(z \mid x)$: NN encoder

  $p_\theta(x|z)$: NN decoder

- Variational auto-encoder

  $q_\phi(z \mid x)$: NN encoder

  $p_\theta(z)$: Gaussian

  $p_\theta(x|z)$: NN decoder

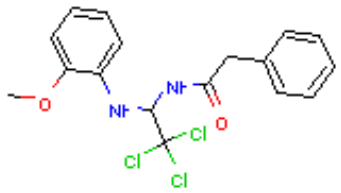
➔ Sampling new **z'**

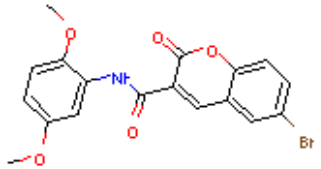
$p_\theta(x' \mid z')$: NN decoder

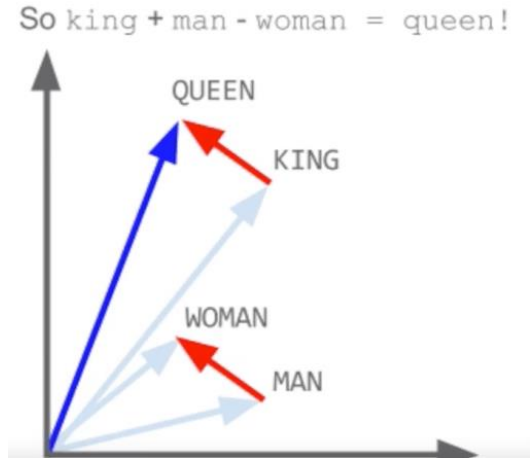
New output x' which is different from any x in the training set

Summary

✓ **Embedding vector operation**







Encoder

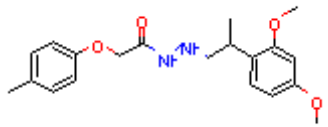
Encoder

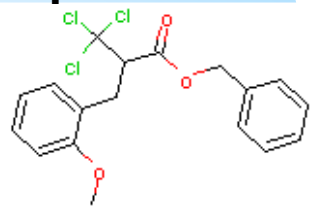
0.9 x Latent vector 1 ✚ 0.1 x Latent vector 2 ⟶ Latent vector 3

0.1 x Latent vector 1 ✚ 0.9 x Latent vector 2 → Latent vector 4

Decoder

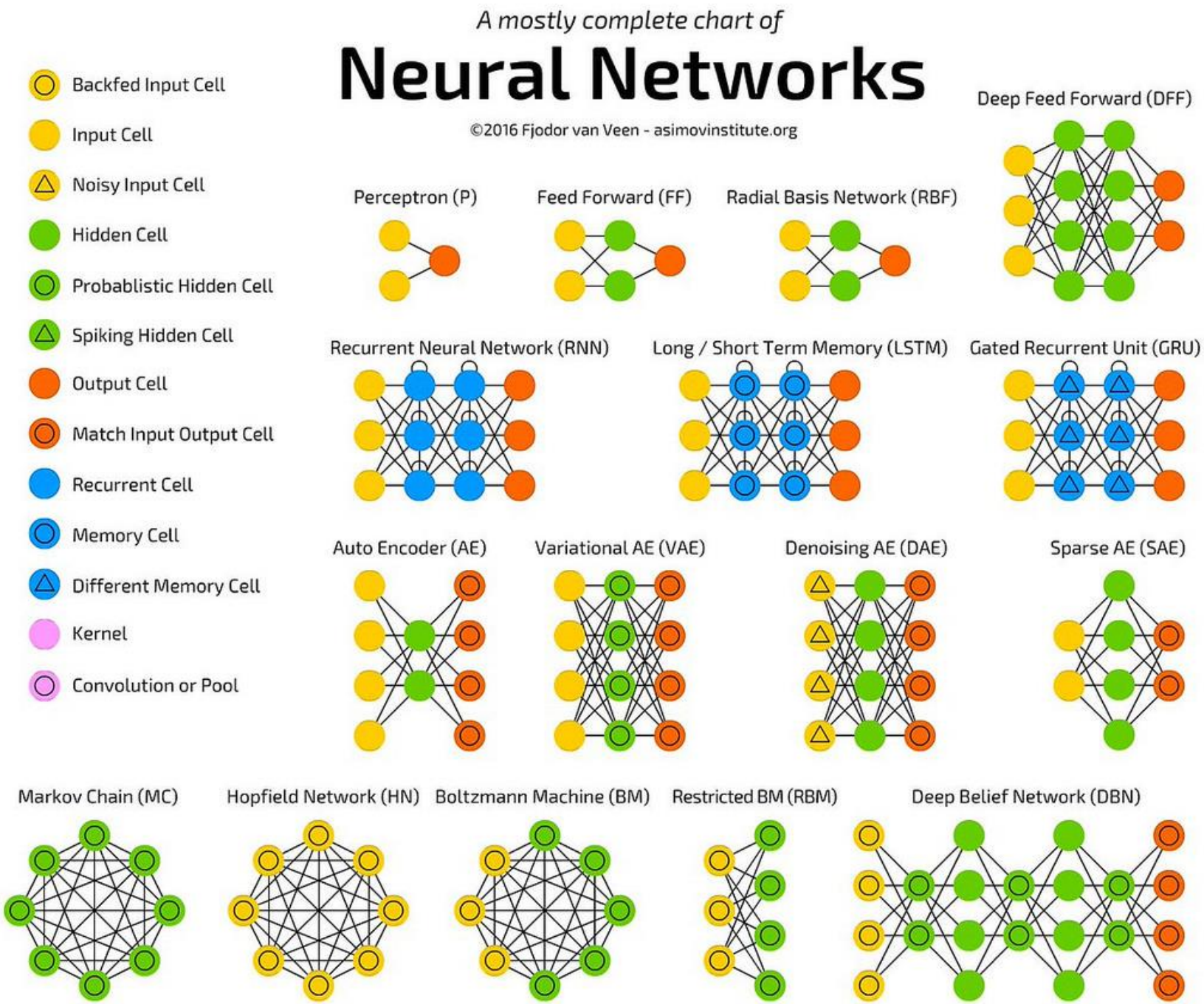
Decoder





→ **Represent molecules through vectors in a latent space.**

→ **Discontinuous graph/string representation to continuous vector representation**



http://www.asimovinstitute.org/neural-network-zoo/

New terms

- Boltzmann machine
- Restricted Boltzmann machine
- Gibbs sampling
- Kullback–Leibler (KL) divergence
- Deep belief network
- Auto-encoder
- Variational auto-encoder (VAE)
- Conditional variational auto-encoder (CVAE)
- Jointly trained VAE