CSE8803/CX4803

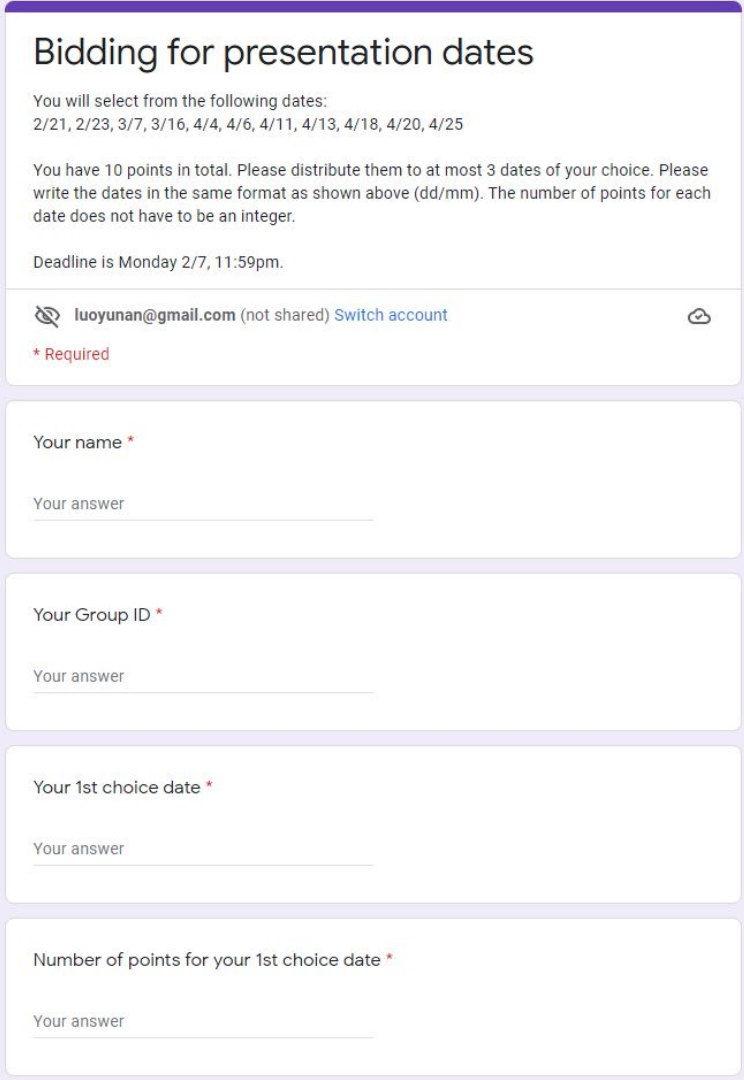
Machine Learning in Computational Biology

Lecture 7:
PCA, Autoencoder, VAE

Yunan Luo

Logistics: presentation date

- Presentation team finalized (see Canvas -> Quizzes)
- Each team to submit 3 preferred dates via a Google form
  - Link on Ed
- Deadline: Feb 7 (Mon), 11:59 PM
- Distribute 10 points into the 3 dates
- Detailed announcement on Ed
- Paper selection
  - A paper list will be released
  - More details to follow



Bidding for presentation dates

You will select from the following dates:
2/21, 2/23, 3/7, 3/16, 4/4, 4/6, 4/11, 4/13, 4/18, 4/20, 4/25

You have 10 points in total. Please distribute them to at most 3 dates of your choice. Please write the dates in the same format as shown above (dd/mm). The number of points for each date does not have to be an integer.

Deadline is Monday 2/7, 11:59pm.

luoyunan@gmail.com (not shared) Switch account

* Required

Your name *

Your answer

Your Group ID *

Your answer

Your 1st choice date *

Your answer

Number of points for your 1st choice date *

Your answer

Presenting the paper

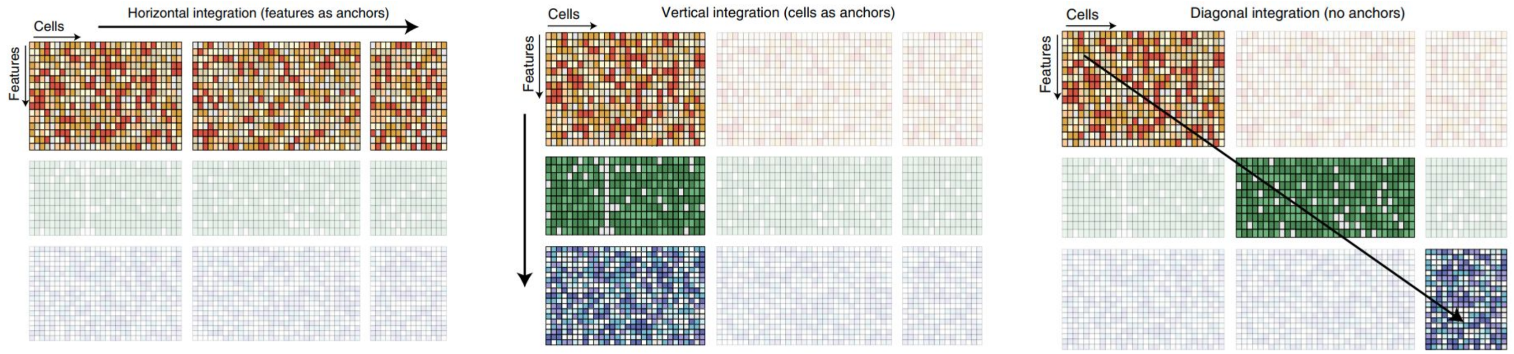
- Briefly introduce the biological problem
  - E.g., why it’s important? What’s the motivation? Why use ML/computation methods?
- Clearly state the computational problem
  - E.g., what’s are the input and output? What does the data look like?
- Present the methods and results
  - Try to identify key ideas in methods and key takeaways from results
- Optional
  - Your comments on the presented paper
  - Survey of related work
  - Follow-up research ideas or applications

Optional: share your comments on the paper

- Consider you are a reviewer of the paper, share your critical (not necessarily negative) comments/review of the paper
    - Strengths and/or weakness
    - Novelty/significance of the contribution
    - Soundness of the evaluation
    - Further improvement of the paper

Optional: survey of related work

- Positioning the paper in the context of previous and subsequent work
  - Any prior papers that substantially influenced the presented paper?
  - Any newer papers that are largely built on the presented paper
  - Make a brief comparison of them in terms of motivation, strength, limitation, methodology, etc
- Example: single-cell data integration methods


Cells
Horizontal integration (features as anchors)
Features
Cells
Vertical integration (cells as anchors)
Features
Cells
Diagonal integration (no anchors)
Features

Argelaguet, R., Cuomo, A.S.E., Stegle, O. *et al.* Computational principles and challenges in single-cell data integration. *Nat Biotechnol* 39, 1202–1215 (2021).

Optional: possible follow-up projects or applications

- Propose a follow-up research project idea
  - Any improvements of the proposed method in the paper?
  - Can you think about a solution that addresses some limitations of the proposed method?
- Propose an application based on the paper
  - Think about a new application in biology (not discussed in the paper or our class yet) where the method can be applied to.

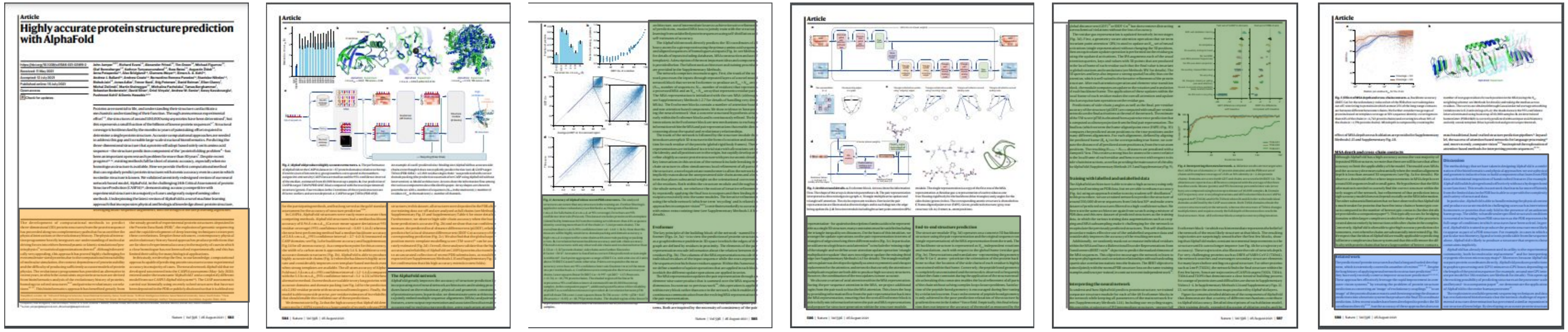
Suggestions

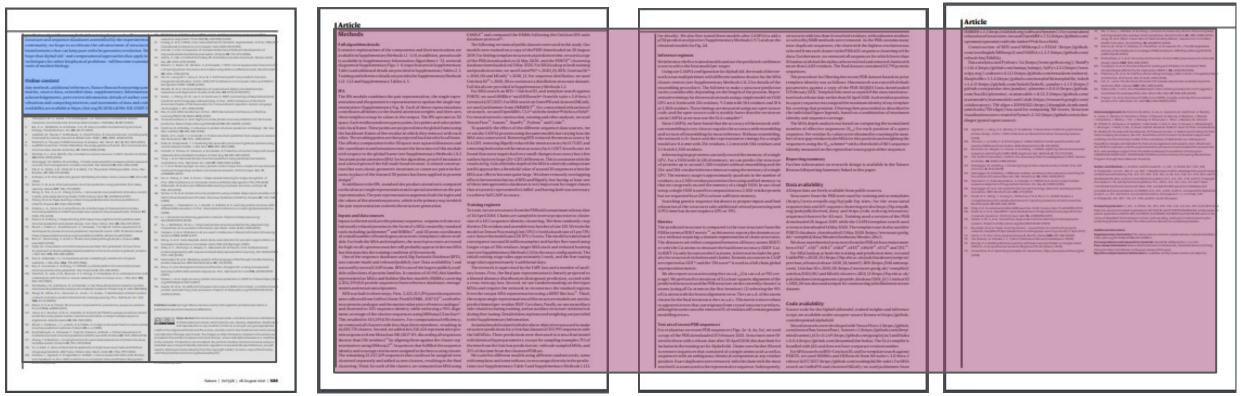
- Make sure the presentation ends within the time limit
- Don’t put to much information on a single slides
    - Avoid using long sentences or dense tables
    - Use clear, short text
    - Use illustrations/demos to show the methods/data/results
- Connect the presented paper with what we have seen in the class, if possible
- Don’t forget important information in the Supplementary Information of journal papers

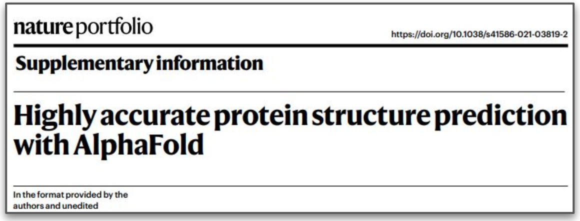
Introduction

Results
(may have a method overview)

Discussion






nature portfolio

https://doi.org/10.1038/s41586-021-03819-2

Supplementary information

Highly accurate protein structure prediction with AlphaFold

In the format provided by the authors and unedited

References

Methods

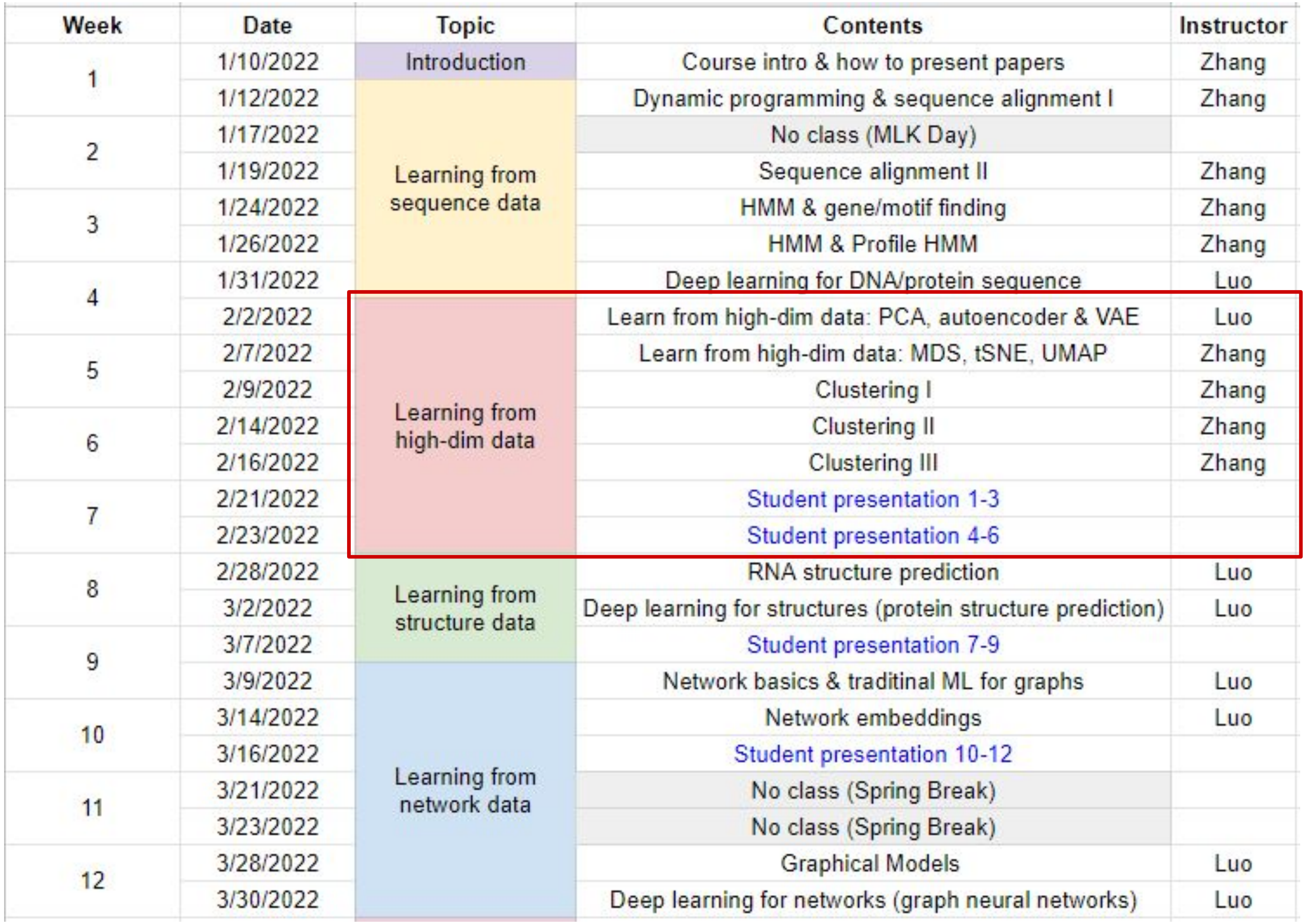
Full algorithm details

Supplementary Information (62 page)

Jumper, J., Evans, R., Pritzel, A. *et al.* Highly accurate protein structure prediction with AlphaFold. *Nature* 596, 583–589 (2021).

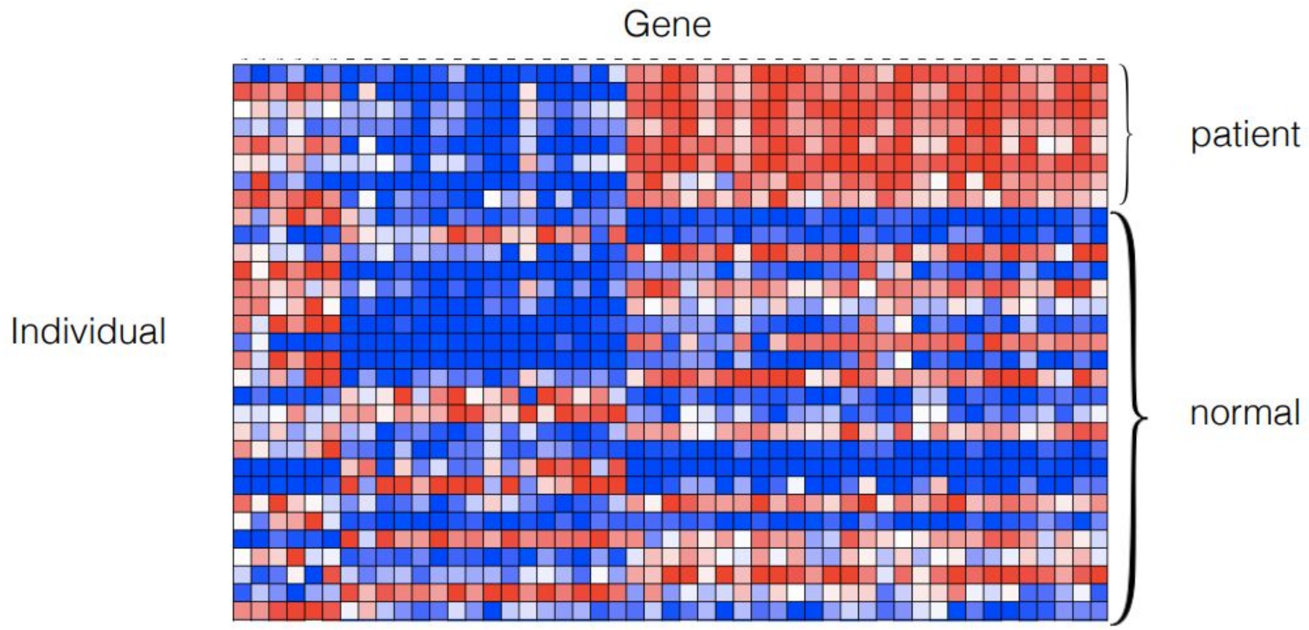
Outline

- PCA
- Autoencoder
- Variational Autoencoder (VAE)



Week	Date	Topic	Contents	Instructor
1	1/10/2022	Introduction	Course intro & how to present papers	Zhang
	1/12/2022	Learning from sequence data	Dynamic programming & sequence alignment I	Zhang
2	1/17/2022		No class (MLK Day)	
	1/19/2022		Sequence alignment II	Zhang
3	1/24/2022		HMM & gene/motif finding	Zhang
	1/26/2022		HMM & Profile HMM	Zhang
4	1/31/2022		Deep learning for DNA/protein sequence	Luo
	2/2/2022	Learning from high-dim data	Learn from high-dim data: PCA, autoencoder & VAE	Luo
5	2/7/2022		Learn from high-dim data: MDS, tSNE, UMAP	Zhang
	2/9/2022		Clustering I	Zhang
6	2/14/2022		Clustering II	Zhang
	2/16/2022		Clustering III	Zhang
7	2/21/2022		Student presentation 1-3	
	2/23/2022		Student presentation 4-6	
8	2/28/2022	Learning from structure data	RNA structure prediction	Luo
	3/2/2022		Deep learning for structures (protein structure prediction)	Luo
9	3/7/2022		Student presentation 7-9	
	3/9/2022	Learning from network data	Network basics & traditinal ML for graphs	Luo
10	3/14/2022		Network embeddings	Luo
	3/16/2022		Student presentation 10-12	
11	3/21/2022		No class (Spring Break)	
	3/23/2022		No class (Spring Break)	
12	3/28/2022		Graphical Models	Luo
	3/30/2022		Deep learning for networks (graph neural networks)	Luo

Gene expression matrix



dim(features) >> num(samples)

High-dimensional data

High-dimensional data

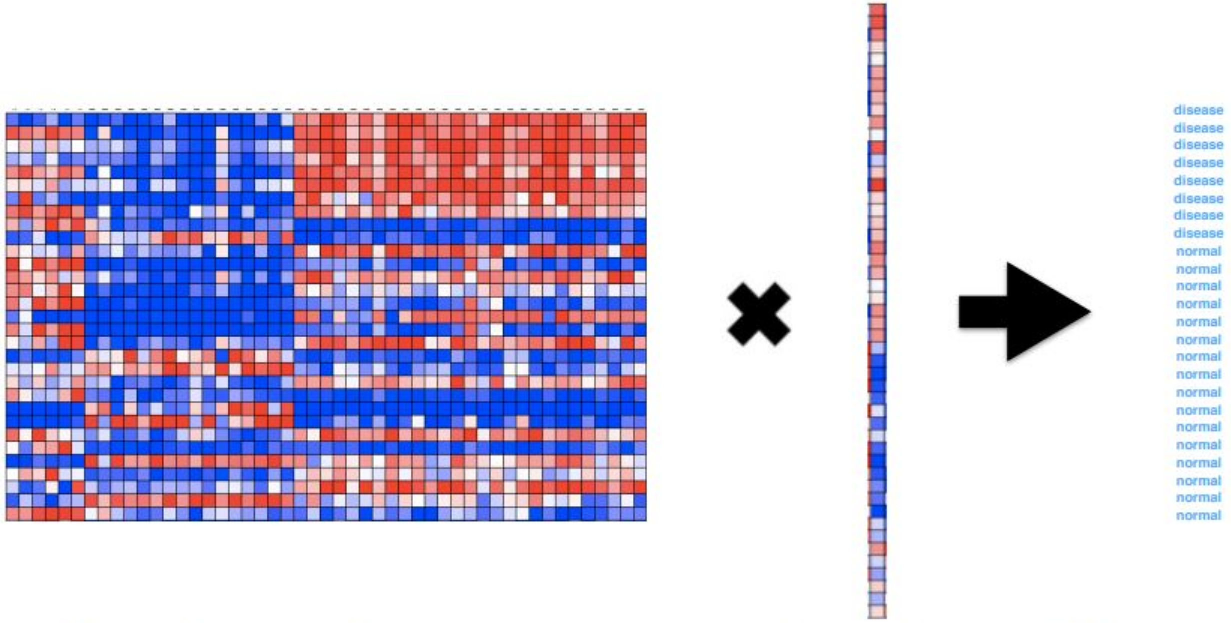
- Each sample has a large number of features/attributes

Why is high-dimension a problem? The curse of dimensionality:

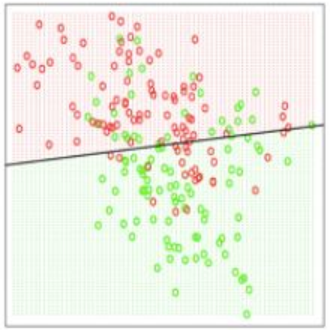
- Volume of space increases exponentially so data becomes very sparse; sparsity
- Increases the effort of searching drastically
- Makes it harder to calculate (accurate) distances between samples
- Redundancy of data
- A large number of training data samples is required to train a model for high-dim data
- Overfitting

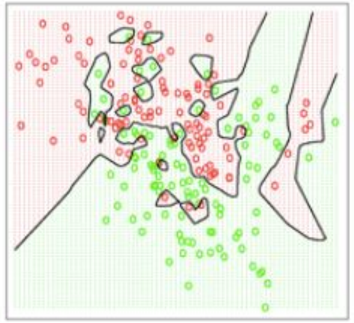
Slides credit: Xiuwei Zhang

Overfitting



$$p(\text{number of parameters}) \gg n(\text{number of data points})$$





Slides credit: CS598JP, UIUC, 2020

A solution: dimensionality reduction

Benefits:

- Reduce redundancy of data
- Identify the most relevant information (find and filter noise) & Cleaning the data
- Reduce computational complexity & Speeding up subsequent learning task
- Building simpler model later
- Visualizing, exploring and understanding the data

Slides credit: Xiuwei Zhang

Dimensionality reduction: approaches

- Linear transformation:
  - PCA
  - NMF
- Non-linear transformation
  - Autoencoder, VAE
  - MDS
  - tSNE
  - UMAP
- Different methods have different *objectives*

Slides credit: Xiuwei Zhang

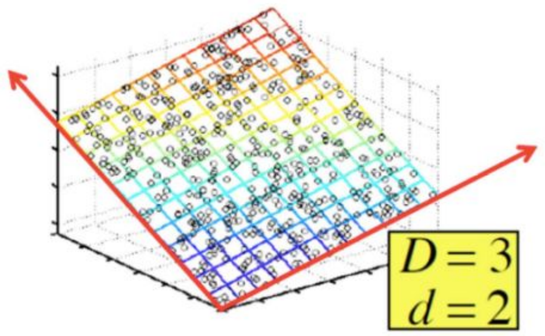
Principal Component Analysis (PCA)

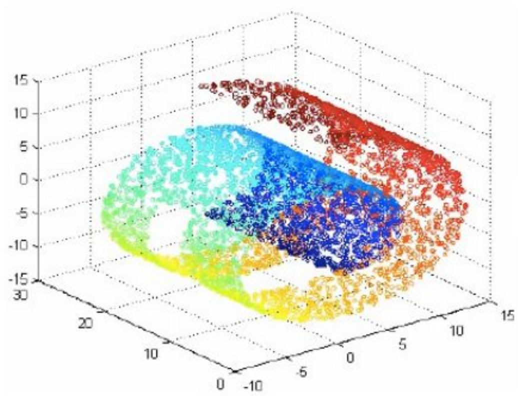
Component analysis

How to understand the main signals from the data?

Key assumptions

1. Low-rank assumption: High-dimensional data lies on a lower dimensional space (a.k.a, manifold)
2. Projections in the lower-dimensional space describes major properties of the data





Principal Component Analysis (PCA)

Goal: Find a projection of the data onto directions that maximize variance of the original data

- **Intuition**: those are directions in which most information is encoded

Definition: Principal Components (PC) are orthogonal directions that capture most of the variance in the data

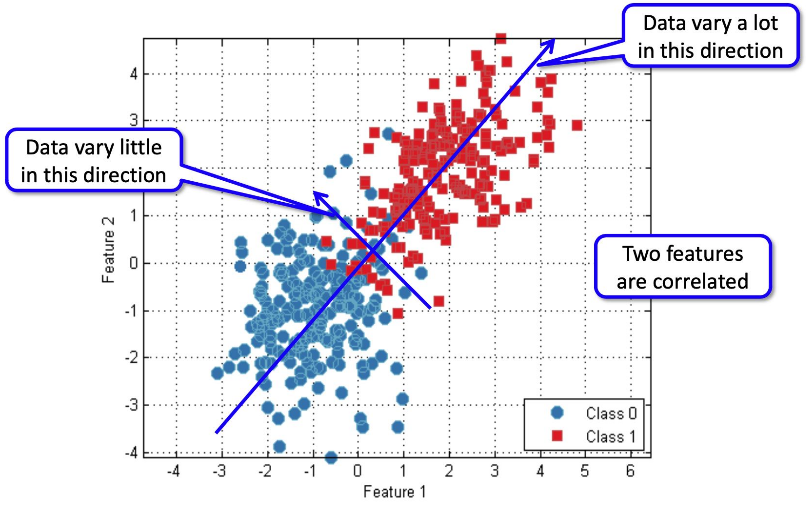


Figure credit: Le Song

Adapted from CMSC422-0101

PCA: Finding principal components

- 1st PC:
  - Projection of data points along 1st PC discriminates data most along any one direction
- 2nd PC:
  - Next orthogonal direction of greatest variability
- 3rd PC…

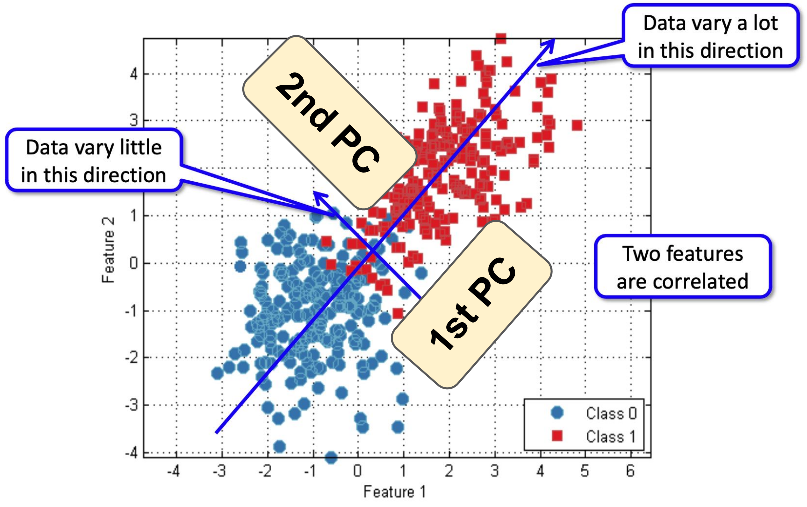


Figure credit: Le Song

PCA notation

- Input data points: matrix $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N]$ of size $\boldsymbol{D} \times \boldsymbol{N}$
- $\boldsymbol{x}_i$ is the $\boldsymbol{i}$-th column, i.e., the $\boldsymbol{i}$-th example
- $\boldsymbol{x}_{ij}$ is the $\boldsymbol{j}$-th feature of example $\boldsymbol{i}$
- We assume the data is centered, i.e., $\frac{1}{N}\sum_{i=1}^{N} \boldsymbol{x}_i = \vec{0}$
    - If not centered, replace $\boldsymbol{x}_i$ by $\boldsymbol{x}_i - \boldsymbol{\mu}$, where $\boldsymbol{\mu} = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{x}_i$

Finding the 1st PC

Given *N* data points, $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N]_{D\times N}$, $\boldsymbol{x}_i \in \mathbb{R}^D$, find a direction *w* where $\|\boldsymbol{w}\| = 1$, such that the variation of the data along direction *w* is maximized.

- The sample variance on the projected on vector *w* is $\sum_{i=1}^{N}(\boldsymbol{w}^T\boldsymbol{x}_i)^2 = \boldsymbol{w}^T\boldsymbol{X}\boldsymbol{X}^T\boldsymbol{w}$

Find the 1st PC by solving the following optimization problem

$$\max_{\boldsymbol{w}} \boldsymbol{w}^T\boldsymbol{X}\boldsymbol{X}^T\boldsymbol{w}$$

such that: $\|\boldsymbol{w}\| = 1$

Finding the 1st PC

$$\boxed{\begin{array}{c}\max_{\boldsymbol{w}} \boldsymbol{w}^T \boldsymbol{X} \boldsymbol{X}^T \boldsymbol{w} \\ \text{such that:} \quad \|\boldsymbol{w}\| = 1\end{array}}$$

- Construct Lagrange multiplier

$$\max_{\boldsymbol{w}} \boldsymbol{w}^T \boldsymbol{X} \boldsymbol{X}^T \boldsymbol{w} - \lambda(\|\boldsymbol{w}\| - 1)$$

- Take the derivative with respect to $\boldsymbol{w}$ and set it to 0 => solutions are vectors $\boldsymbol{w}$ such that

$$\boldsymbol{X} \boldsymbol{X}^T \boldsymbol{w} = \lambda \boldsymbol{w}$$

Eigenvector of $\boldsymbol{X}\boldsymbol{X}^T$!

$$\boldsymbol{X}\boldsymbol{X}^T\boldsymbol{w} = \lambda\boldsymbol{w}$$

The eigenvalue problem

- For a given matrix ***A***

$$\boldsymbol{A}\boldsymbol{w} = \boldsymbol{\lambda}\boldsymbol{w}$$

w is the eigenvector and ***λ*** is the eigenvalue

- There are multiple solutions $\boldsymbol{w}_1$, $\boldsymbol{w}_2$, …,
  with different (or same) eigenvalues λ_1, λ_2, …

- The eigenvectors are ortho-normal (symmetric, positive semi-definite)
  - $w_i^T w_j = 0, w_i^T w_i = 1$

- Let $\boldsymbol{A}=\boldsymbol{X}\boldsymbol{X}^T$ and find the eigenvectors and eigenvalues of ***A***

PCA formally

- If we rank eigenvalues from large to small
  - The 1st PC is the eigenvector of $\boldsymbol{XX^T}$ associated with the largest eigenvalue
  - The 2nd PC is the eigenvector of $\boldsymbol{XX^T}$ associated with the 2nd largest eigenvalue
  - …
- The eigenvalue $\lambda_i / \sum \lambda_i$ denotes the percentage of variance accounted for by the i-th PC $\boldsymbol{w_i}$

Q1: how to find eigenvalues/eigenvectors?

Singular value decomposition (SVD)

The SVD is a factorization of a $m \times n$ matrix into

$$\boldsymbol{A} = \boldsymbol{U}\,\boldsymbol{\Sigma}\,\boldsymbol{V}^T$$

where $\boldsymbol{U}$ is a $m \times m$ orthogonal matrix, $\boldsymbol{V}^T$ is a $n \times n$ orthogonal matrix and $\boldsymbol{\Sigma}$ is a $m \times n$ diagonal matrix.

$$\boldsymbol{A} = \begin{pmatrix} \vdots & \cdots & \vdots \\ \boldsymbol{u}_1 & \cdots & \boldsymbol{u}_n \\ \vdots & \cdots & \vdots \end{pmatrix} \begin{pmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_n \end{pmatrix} \begin{pmatrix} \cdots & \mathbf{v}_1^T & \cdots \\ \vdots & \vdots & \vdots \\ \cdots & \mathbf{v}_n^T & \cdots \end{pmatrix}$$

SVD in Python:

```
from scipy import linalg
U, s, Vh = linalg.svd(A)
```

Singular value decomposition (SVD)

Theorem: if a square matrix S is a real and symmetric matrix, then its SVD can be represented as

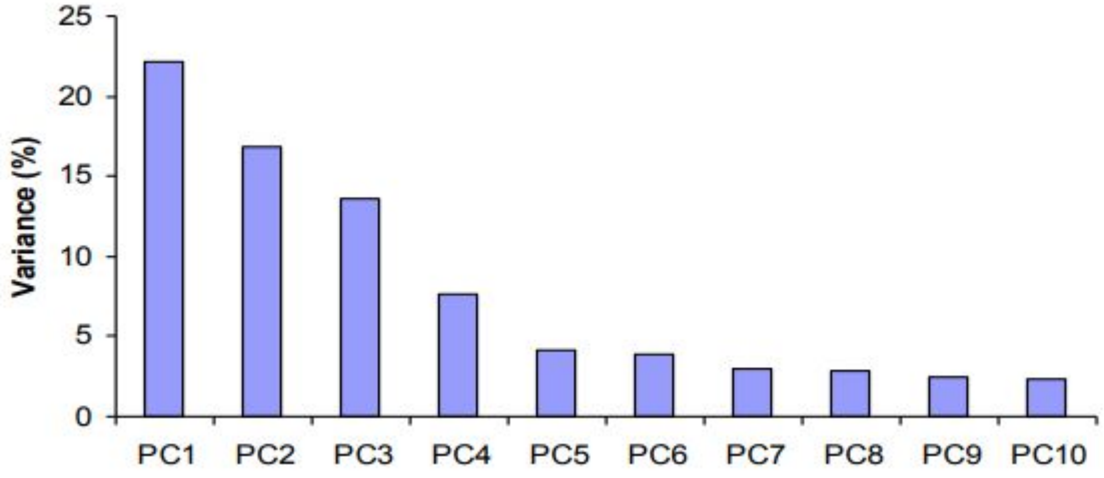
$$\mathbf{S} = \mathbf{V}\boldsymbol{\Lambda}\mathbf{V}^{\mathsf{T}}$$

where columns of **V** are eigenvectors of **S** and diagonal elements of **Λ** are eigenvalues of **S**

$$\boldsymbol{\Lambda} = \mathrm{diag}(\lambda_1, \ldots, \lambda_m), \quad \lambda_i \geq \lambda_{i+1}$$

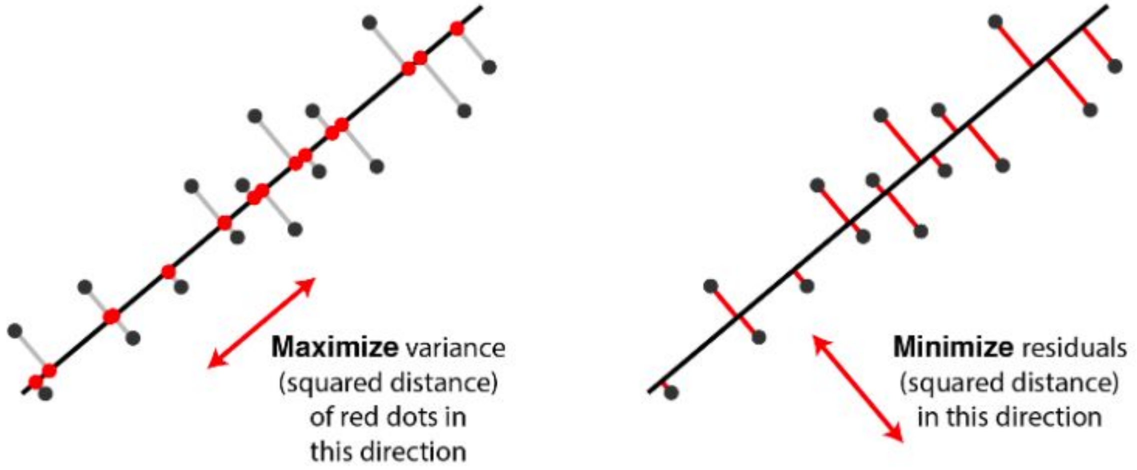
Q2: how many PCs

- The eigenvalue λ_i denotes the amount of variability captured along dimension $\boldsymbol{w_i}$
- Can ignore the components of lower variance (less significant)



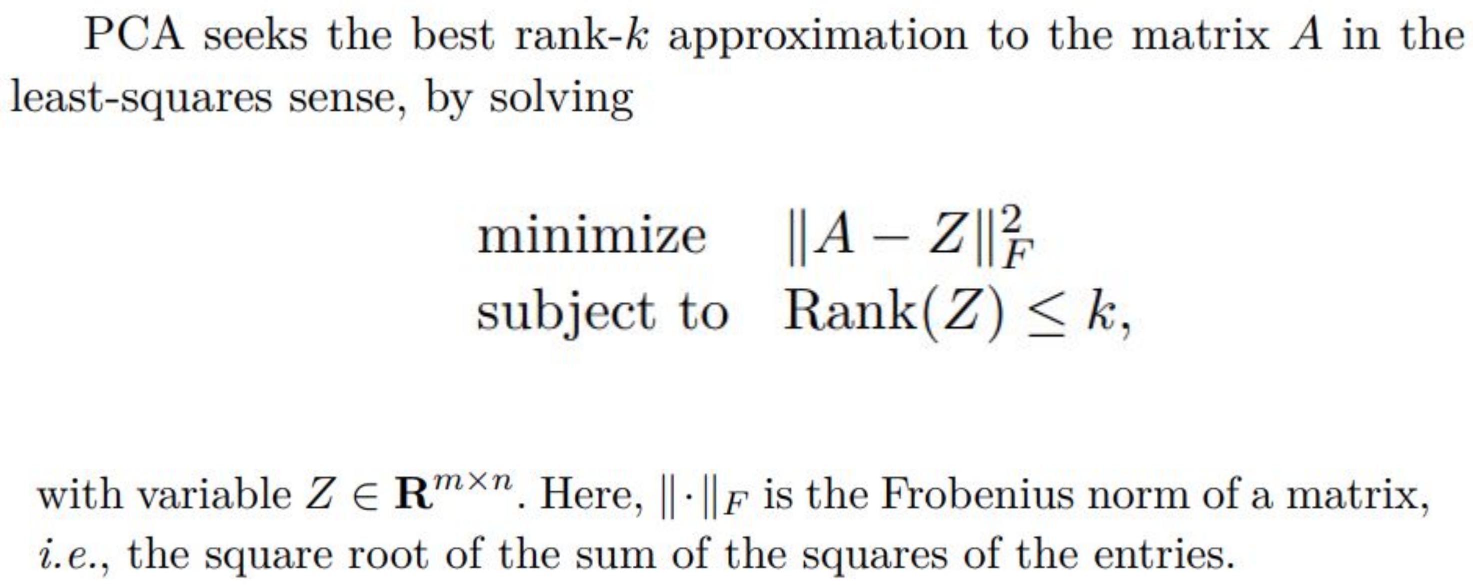
Alternative interpretation 1: residual minimization

PCA finds vectors ***v*** such that projection on to these vectors minimizes reconstruction error


Maximize variance (squared distance) of red dots in this direction
Minimize residuals (squared distance) in this direction

(image source)

Alternative interpretation 2: low-rank approaximation



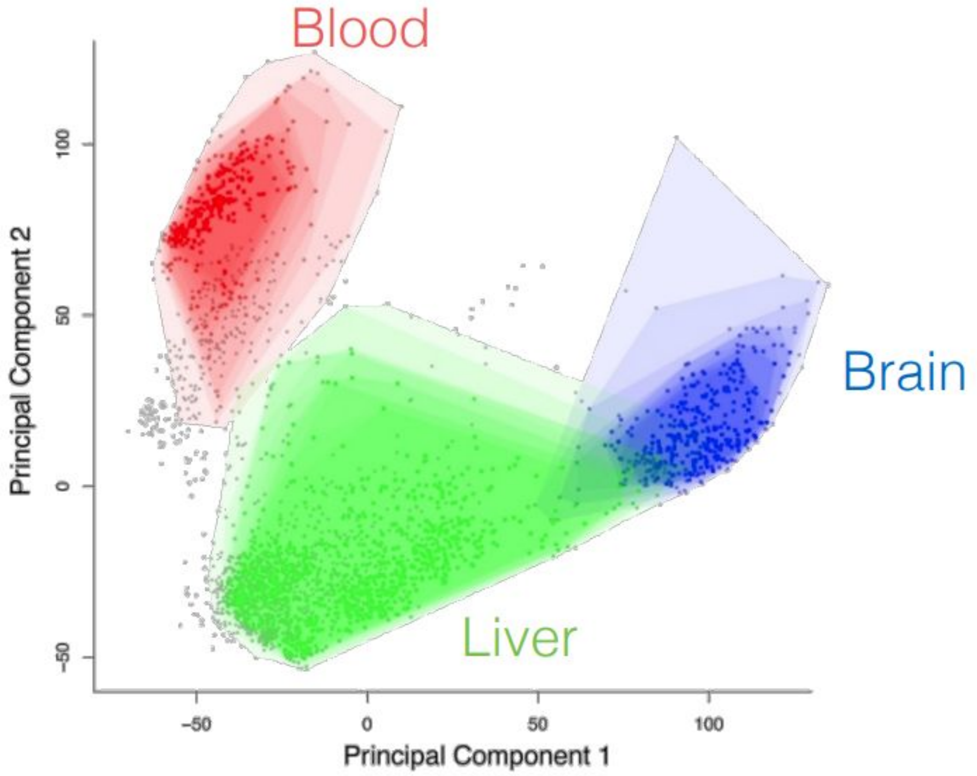
PCA seeks the best rank-k approximation to the matrix A in the least-squares sense, by solving

$$\begin{array}{ll} \text{minimize} & \|A - Z\|_F^2 \\ \text{subject to} & \operatorname{Rank}(Z) \leq k, \end{array}$$

with variable $Z \in \mathbf{R}^{m \times n}$. Here, $\|\cdot\|_F$ is the Frobenius norm of a matrix, *i.e.*, the square root of the sum of the squares of the entries.

https://stanford.edu/~boyd/papers/pdf/glrm.pdf

Example: Tissue-specific gene expression



Slides credit: CS598JP, UIUC, 2020

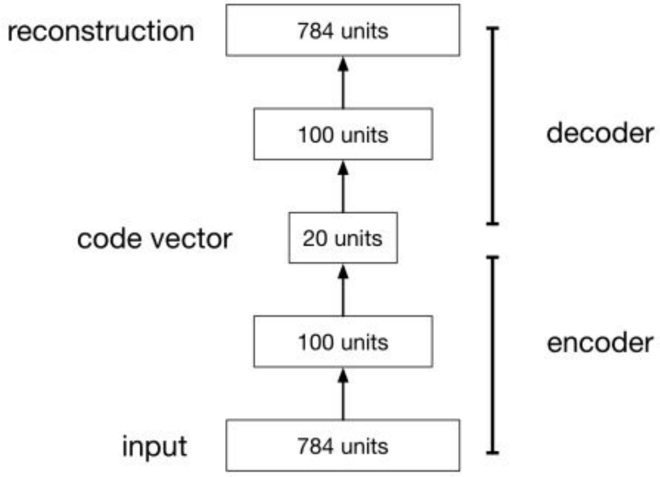
Summary: PCA

- What you should know:
  - Goal: Find a projection of the data onto directions that maximize variance of the original data
  - Optimization objective & algorithm
- Pros
  - Eigenvector method
  - No tuning of parameters
  - No local optima
- Cons
  - Only based on covariance (2nd order statistics)
  - Limited to linear projections
- Next: Nonlinear dimensionality reduction

Autoencoder

Autoencoders

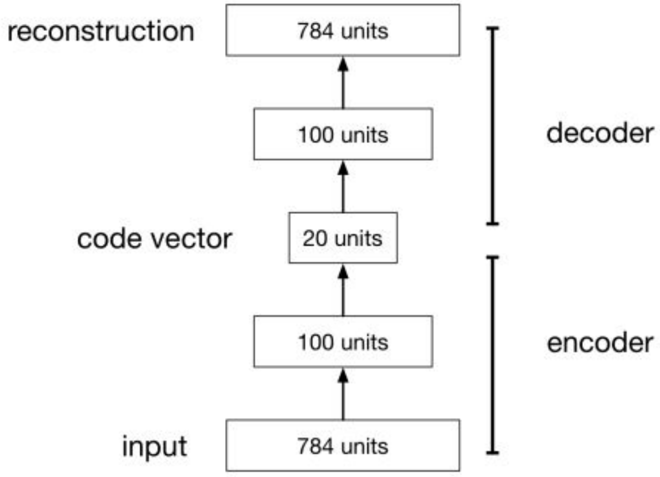
- A neural network to find latent space representation of the original data
  - Unsupervised method (with no labeled training data)
- To make this non-trivial, we add a bottleneck layer whose dimension is much smaller than the input



Adapted from slides of Roger Grosse and Jimmy Ba

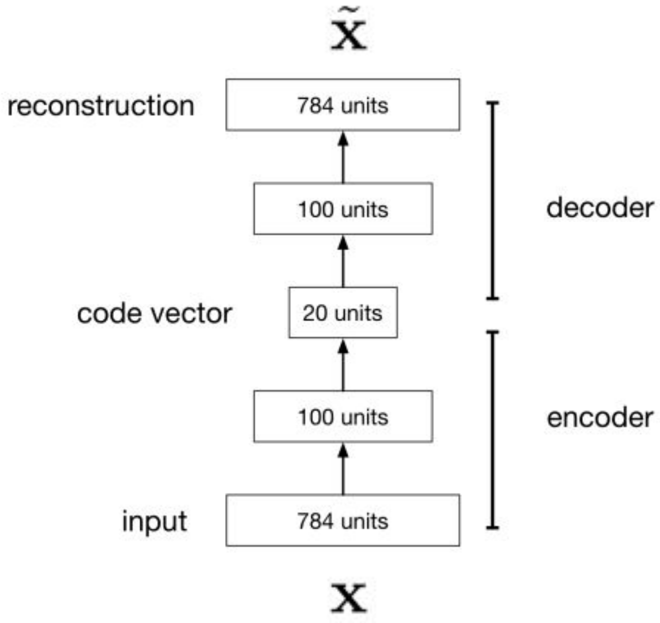
Why autoencoders?

- Map high-dimensional data to 2D for visualization
- Compression
- Learn abstract features in an unsupervised way so you can apply them to a supervised task
- Learn a semantically meaningful representation where you can, e.g., interpolate between different images.



Autoencoders: approach

- **Goal**: Find the latent space representation that *best* represent the important information in the original data
    - Recall PCA: maximize the variance
- **Approach**: bottleneck layer
    - Forces the network to create a compressed representation of the input data (dimensionality reduction)
    - Forces the network to remove redundancy and noise
- **Objective**: reconstruction error $\mathcal{L}(\mathbf{x}, \tilde{\mathbf{x}}) = \|\mathbf{x} - \tilde{\mathbf{x}}\|^2$
    - Can add regularization term to avoid overfitting (identity mapping)



Autoencoders: connection to PCA

Loss function: $\mathcal{L}(x, \widetilde{x})$ (reconstruction error)

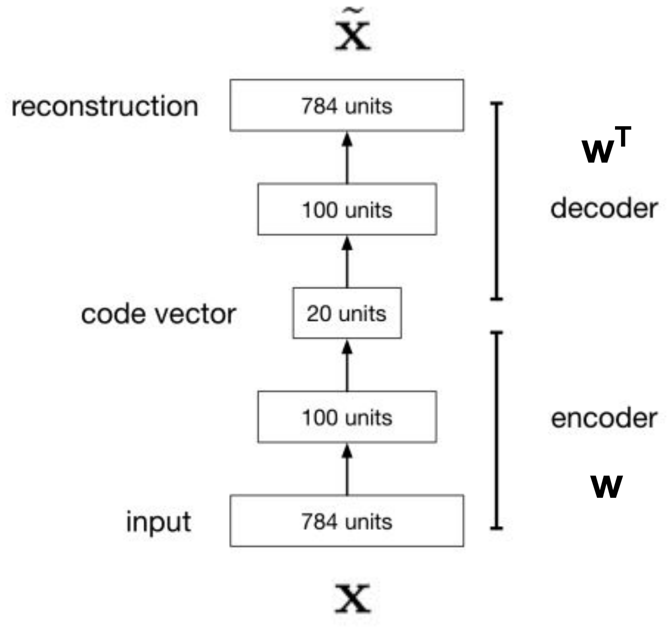
Mean square error (MSE):

$$\frac{1}{D}\sum_i ||x_i - \widetilde{x}_i||^2$$

What if we remove non-linearity in NN?

When we remove the non-linearity term in neural network (activation function), (and force encoder and decoder to have the same weights) autoencoder is equivalent to PCA :

$$\hat{w} = \arg\min_w \mathbb{E}[||x - \underbrace{\boxed{w^T w x}}_{\boxed{\widetilde{x}}}||^2]$$



PCA

$\mathbf{Z}_{(rxN)} = \mathbf{W}^T{}_{(rxD)}\,\mathbf{X}_{(DxN)}$

A different objective function:

$\min || X - WZ ||^2$

$= \min || X - WW^TX ||^2$

Autoencoders: connection to PCA

Autoencoders learn to project the data, not onto a subspace, but onto a nonlinear manifold

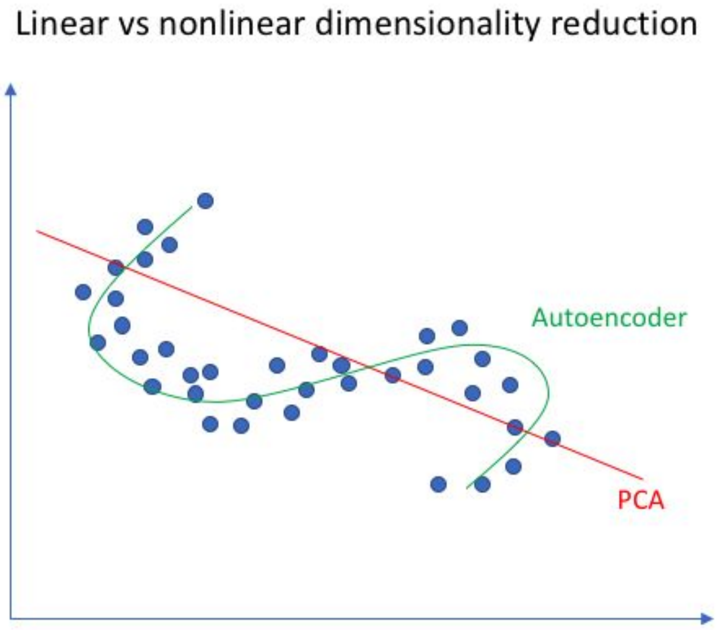


Image source: https://www.jeremyjordan.me/autoencoders/

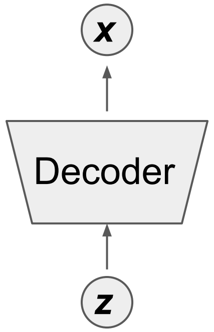
Adapted from slides of Roger Grosse and Jimmy Ba

Variational autoencoder (VAE)

Variational Autoencoders

Generative models: allow us to sample from the models to generate new data.

Assume training data ***x*** is generated from underlying unobserved (latent) representation **z**



Sample ***x*** from conditional ***p(x|z)***

A complex function, use a neural network

Sample ***z*** from prior ***p(z)***

Use prior knowledge/assumption: Assume it's a Gaussian distribution

Intuition: consider ***x*** as an image, ***z*** as latent factors to generate ***x***: attributes, orientation, etc

Adapted from CS231N, Stanford

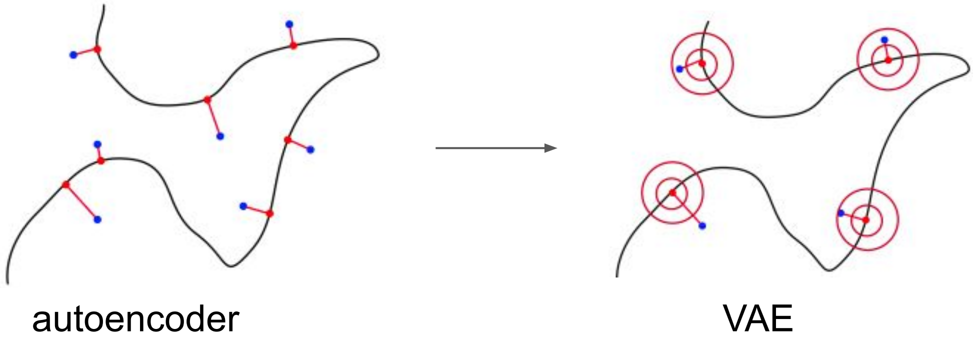
Generative models

- Maximum likelihood

$$p(\mathbf{x}) = \int p(\mathbf{z})p(\mathbf{x} \mid \mathbf{z})\, \mathrm{d}\mathbf{z}$$

- Problem of autoencoder
  - If ***z*** is low-dimensional and the decoder is deterministic, then ***p(x)*** = 0 almost everywhere

- Idea of VAE: instead of encoding an input as a single point, we encode it as a distribution over the latent space.



Adapted from slides of Roger Grosse and Jimmy Ba

Maximum likelihood

$$p(\mathbf{x}) = \boxed{\int} \boxed{p(\mathbf{z})} \boxed{p(\mathbf{x} \mid \mathbf{z})} \, \mathrm{d}\mathbf{z}$$

Intractable to compute for every z

✗

Gaussian prior

✓

Neural network

✓

Maximizing a lower bound

$$\begin{aligned}\log p(\mathbf{x}) &= \log \int p(\mathbf{z})\, p(\mathbf{x}|\mathbf{z})\, \mathrm{d}\mathbf{z} \\ &= \log \int q(\mathbf{z})\, \frac{p(\mathbf{z})}{q(\mathbf{z})} p(\mathbf{x}|\mathbf{z})\, \mathrm{d}\mathbf{z}\end{aligned}$$

Auxiliary distribution **q(z)**

Maximizing a lower bound

$$\log p(\mathbf{x}) = \log \int p(\mathbf{z})\, p(\mathbf{x}|\mathbf{z})\, \mathrm{d}\mathbf{z}$$

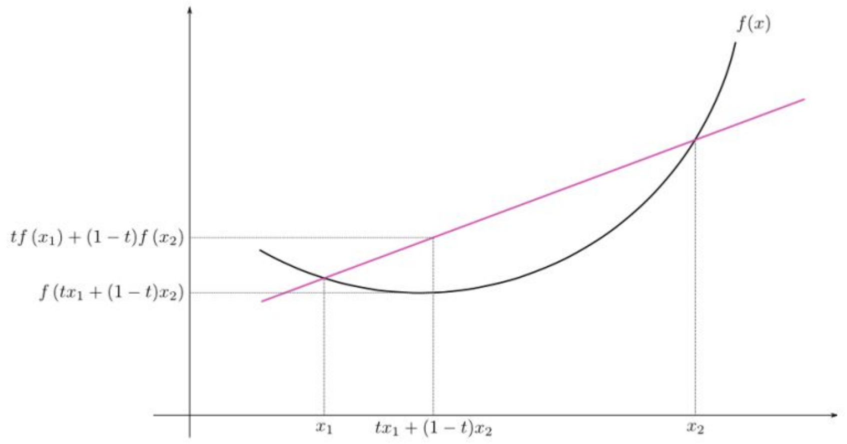
$$= \log \int q(\mathbf{z})\, \frac{p(\mathbf{z})}{q(\mathbf{z})} p(\mathbf{x}|\mathbf{z})\, \mathrm{d}\mathbf{z}$$

$$\geq \int q(\mathbf{z}) \log \left[\frac{p(\mathbf{z})}{q(\mathbf{z})}\, p(\mathbf{x}|\mathbf{z}) \right] \mathrm{d}\mathbf{z}$$

Auxiliary distribution **q(z)**

Jensen's inequality

Convex functions



$$tf(x_1) + (1-t)f(x_2) \geq f(tx_1 + (1-t)x_2)$$

Generalized version $$\int_x f(x)p(x) \geq f(\int_x xp(x))$$

Slides credit: CS598JP, UIUC, 2020

Maximizing a lower bound

$$\log p(\mathbf{x}) = \log \int p(\mathbf{z})\, p(\mathbf{x}|\mathbf{z})\, \mathrm{d}\mathbf{z}$$

$$= \log \int q(\mathbf{z})\, \frac{p(\mathbf{z})}{q(\mathbf{z})} p(\mathbf{x}|\mathbf{z})\, \mathrm{d}\mathbf{z}$$

Auxiliary distribution **q(z)**

$$\geq \int q(\mathbf{z}) \log \left[\frac{p(\mathbf{z})}{q(\mathbf{z})}\, p(\mathbf{x}|\mathbf{z}) \right] \mathrm{d}\mathbf{z}$$

Jensen’s inequality

Maximizing a lower bound

$$\log p(\mathbf{x}) = \log \int p(\mathbf{z})\, p(\mathbf{x}|\mathbf{z})\, \mathrm{d}\mathbf{z}$$

$$= \log \int q(\mathbf{z})\, \frac{p(\mathbf{z})}{q(\mathbf{z})} p(\mathbf{x}|\mathbf{z})\, \mathrm{d}\mathbf{z}$$

Auxiliary distribution **q(z)**

$$\geq \int q(\mathbf{z}) \log \left[\frac{p(\mathbf{z})}{q(\mathbf{z})}\, p(\mathbf{x}|\mathbf{z}) \right] \mathrm{d}\mathbf{z}$$

Jensen’s inequality

$$= \mathbb{E}_q \left[\log \frac{p(\mathbf{z})}{q(\mathbf{z})} \right] + \mathbb{E}_q \left[\log p(\mathbf{x}|\mathbf{z}) \right]$$

Maximizing a lower bound

$$\log p(\mathbf{x}) \geq \mathbb{E}_q \left[\log \frac{p(\mathbf{z})}{q(\mathbf{z})} \right] + \mathbb{E}_q \left[\log p(\mathbf{x}|\mathbf{z}) \right]$$

- Reconstruction term
  - Encourages the model to reconstruct the input

- If we parameterize p(x|z) as Gaussian

$$\begin{aligned} \log p(\mathbf{x}|\mathbf{z}) &= \log \mathcal{N}(\mathbf{x}; G_\theta(\mathbf{z}), \eta \mathbf{I}) \\ &= \log \left[\frac{1}{(2\pi\eta)^{D/2}} \exp \left(-\frac{1}{2\eta} \|\mathbf{x} - G_\theta(\mathbf{z})\|^2 \right) \right] \\ &= -\frac{1}{2\eta} \|\mathbf{x} - G_\theta(\mathbf{z})\|^2 + \text{const} \end{aligned}$$

Maximizing a lower bound

$$\log p(\mathbf{x}) \geq \mathbb{E}_q\left[\log \frac{p(\mathbf{z})}{q(\mathbf{z})}\right] + \mathbb{E}_q\left[\log p(\mathbf{x}|\mathbf{z})\right]$$

- Can be written as $-\mathrm{D}_{\mathrm{KL}}(q(\mathbf{z})\|p(\mathbf{z}))$. KL term.
- DKL is the Kullback-Leibler (KL) divergence $\mathrm{D}_{\mathrm{KL}}(q(\mathbf{z})\|p(\mathbf{z})) \triangleq \mathbb{E}_q\left[\log \frac{q(\mathbf{z})}{p(\mathbf{z})}\right]$
  - Widely used to measure the distance between two probability distributions
- Typically, $p(\mathbf{z}) = N(\mathbf{0}, \mathbf{1})$
  - The KL term encourages q(z) to be close to the standard normal distribution $N(\mathbf{0}, \mathbf{1})$.

Maximizing a lower bound

Variational lower bound

$$\log p(\mathbf{x}) \geq \mathbb{E}_q \left[\log p(\mathbf{x}|\mathbf{z})\right] - \mathrm{D}_{\mathrm{KL}}(q \| p)$$

The role of each of the two terms:

The reconstruction term

$$\mathbb{E}_q[\log p(\mathbf{x}|\mathbf{z})] = -\frac{1}{2\sigma^2}\mathbb{E}_q[\|\mathbf{x} - G_\theta(\mathbf{z})\|^2] + \mathrm{const}$$

is minimized when q is a point mass on

$$\mathbf{z}_* = \arg\min_{\mathbf{z}} \|\mathbf{x} - G_\theta(\mathbf{z})\|^2.$$

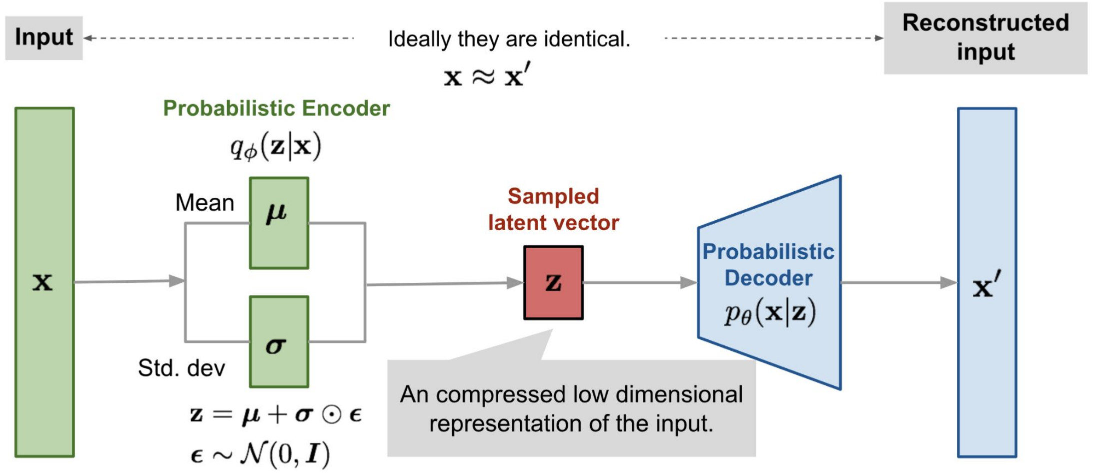
But a point mass would have infinite KL divergence.

Adapted from slides of Roger Grosse and Jimmy Ba

VAE

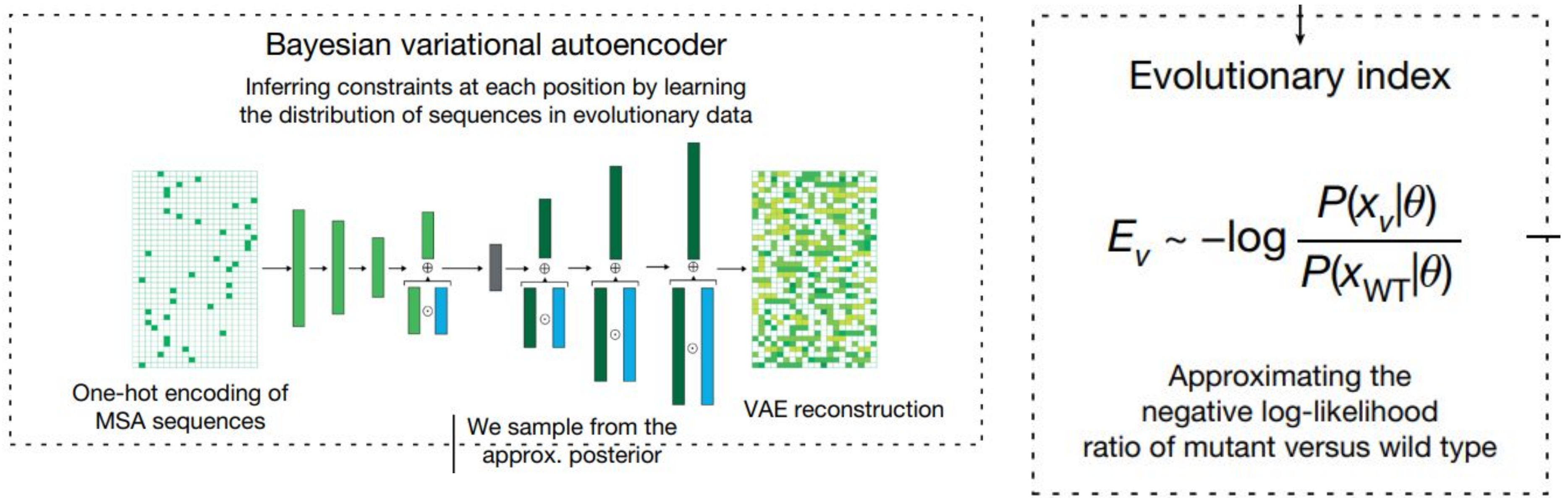
$$\log p(\mathbf{x}) \geq \mathbb{E}_q [\log p(\mathbf{x}|\mathbf{z})] - \mathrm{D}_{\mathrm{KL}}(q||p)$$

Further reading: “reparameterization trick”
(Kigma & Welling, 2013)


Input
Ideally they are identical.
$\mathbf{x} \approx \mathbf{x}'$
Reconstructed input
Probabilistic Encoder
$q_\phi(\mathbf{z}|\mathbf{x})$
$\mathbf{x}$
Mean
$\boldsymbol{\mu}$
Std. dev
$\boldsymbol{\sigma}$
$\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}$
$\boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I})$
Sampled latent vector
$\mathbf{z}$
An compressed low dimensional representation of the input.
Probabilistic Decoder
$p_\theta(\mathbf{x}|\mathbf{z})$
$\mathbf{x}'$

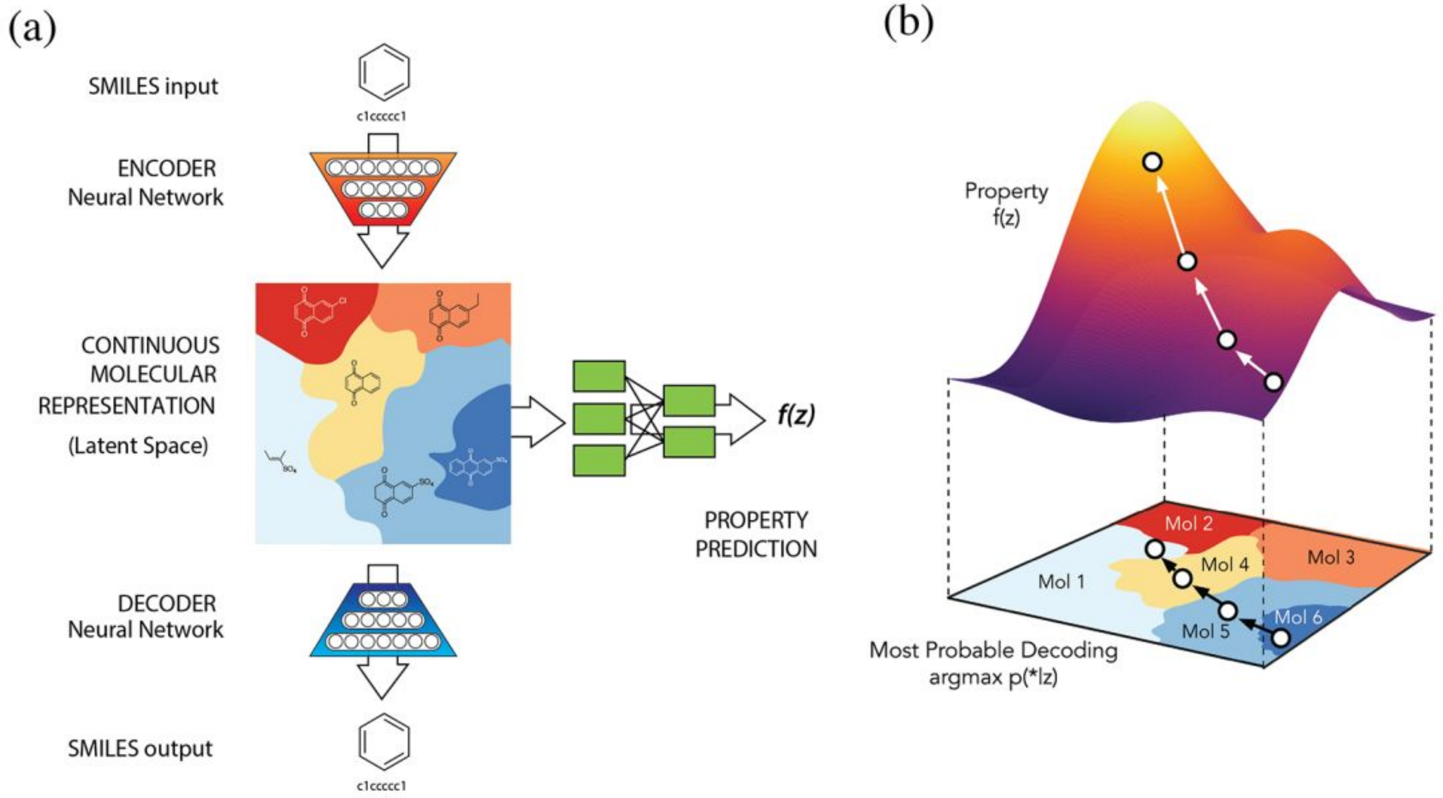
(image source)

Application: mutation effect prediction


Bayesian variational autoencoder
Inferring constraints at each position by learning the distribution of sequences in evolutionary data
One-hot encoding of MSA sequences
We sample from the approx. posterior
VAE reconstruction
Evolutionary index
$E_v \sim -\log \frac{P(x_v|\theta)}{P(x_{WT}|\theta)}$
Approximating the negative log-likelihood ratio of mutant versus wild type

Frazer, J., Notin, P., Dias, M. *et al.* Disease variant prediction with deep generative models of evolutionary data. *Nature* 599, 91–95 (2021). https://doi.org/10.1038/s41586-021-04043-8

Application: molecule design



Gómez-Bombarelli, Rafael, et al. "Automatic chemical design using a data-driven continuous representation of molecules." *ACS central science* 4.2 (2018): 268-276.

Summary of today

- PCA
  - Linear dimensionality reduction
  - Maximize variance
- Autoencoders
  - Nonlinear dimensionality reduction
  - Minimize reconstruction error
- VAE
  - Problistics generative model
  - Regularized latent space