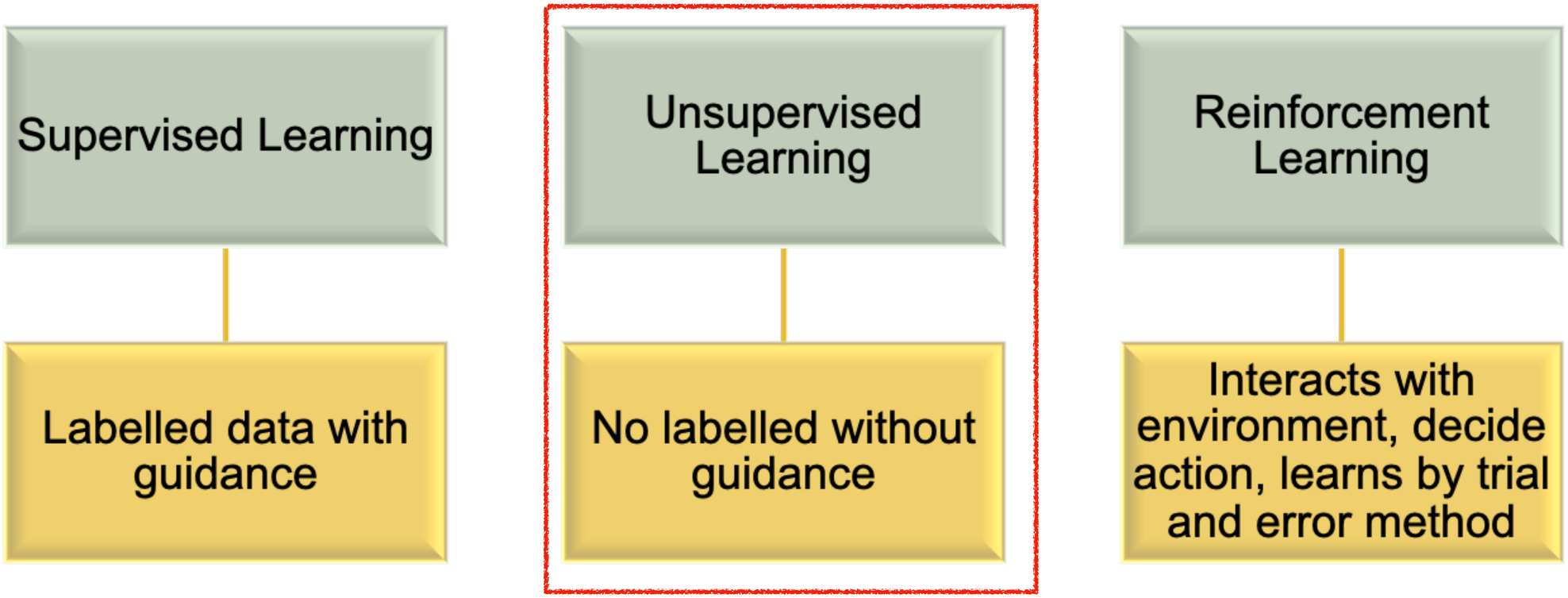
Clustering / Unsupervised Representation Learning

COMP3314 — Week 8

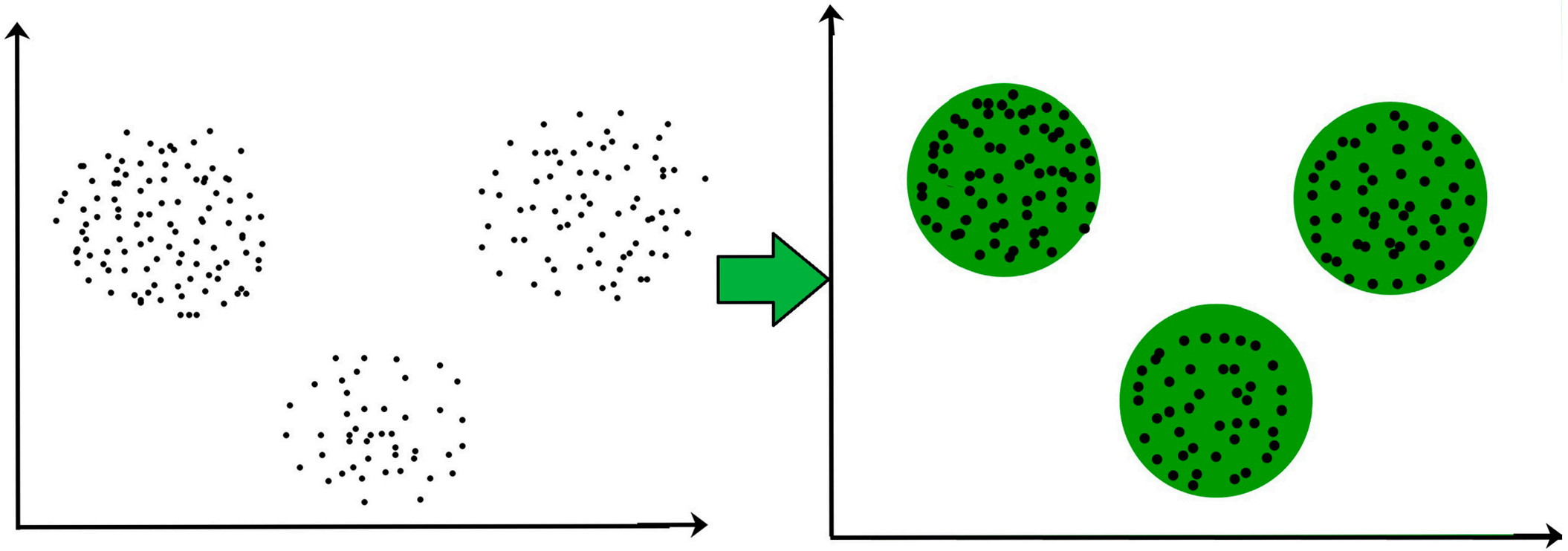
Lingpeng Kong
Department of Computer Science, The University of Hong Kong

Based on: Probabilistic Machine Learning by Kevin Murphy
Slides from: Saw Shier Nee with special thanks!

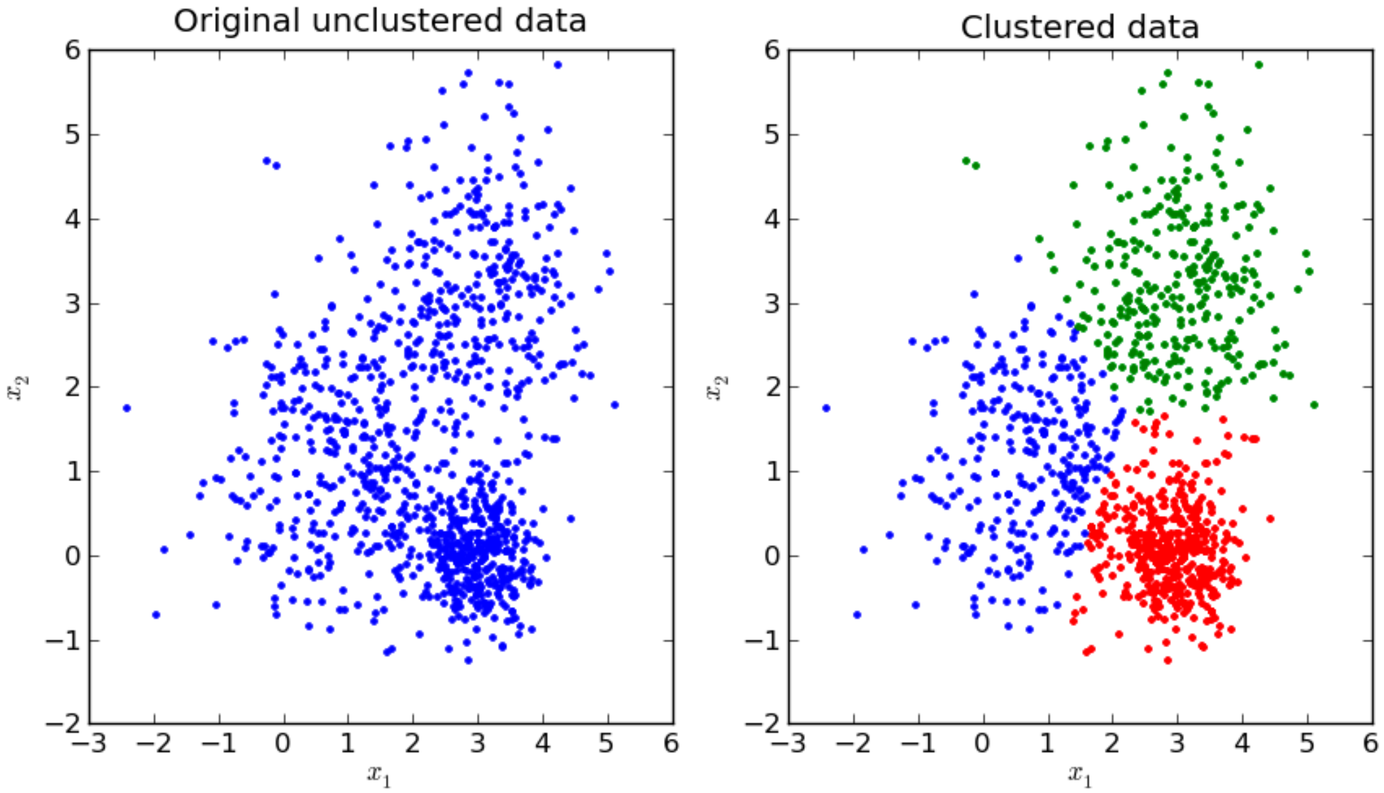
Unsupervised Learning



Clustering



Clustering



Is it even a well-defined machine learning problem?

Evaluating Clustering Methods

It is hard to evaluate the quality of the output of any given method.

Intuition: assign points that are similar to the same cluster, and to ensure that points that are dissimilar are in different clusters.

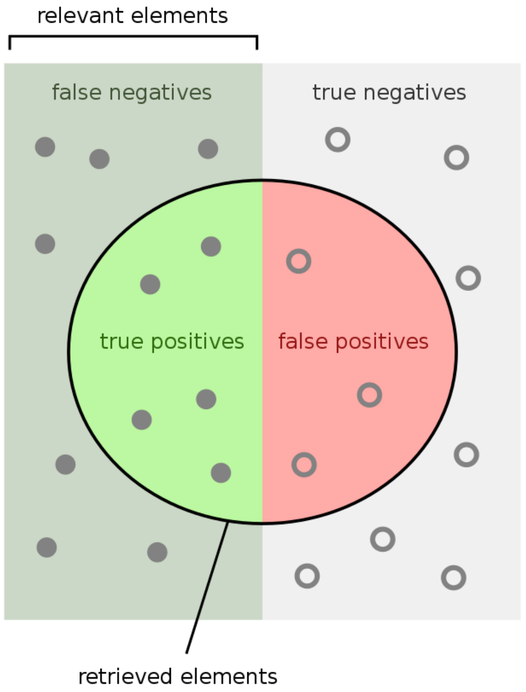
Rely on external form of data, where we have labels for each object.

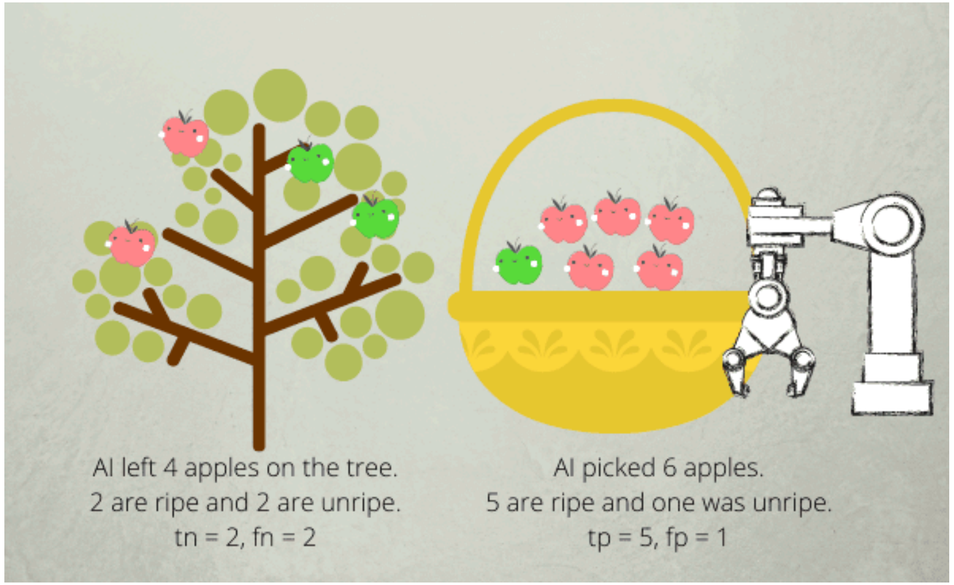
$$\text{purity} \triangleq \sum_i \frac{N_i}{N} p_i \qquad \frac{6}{17}\frac{5}{6} + \frac{6}{17}\frac{4}{6} + \frac{5}{17}\frac{3}{5} = \frac{5+4+3}{17} = 0.71$$

dominant class

range? Why it can be bad?

F-Score





https://deepai.org/machine-learning-glossary-and-terms/f-score

How many retrieved items are relevant?

Precision = 

How many relevant items are retrieved?

Recall = 

$$F_1 = \frac{2}{\mathrm{recall}^{-1} + \mathrm{precision}^{-1}} = 2\frac{\mathrm{precision} \cdot \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}} = \frac{2\mathrm{tp}}{2\mathrm{tp} + \mathrm{fp} + \mathrm{fn}}$$

Evaluating Clustering Methods

Rand index

$$R \triangleq \frac{TP + TN}{TP + FP + FN + TN}$$

	same cluster	different cluster
same label	TP	FN
different label	FP	TN

TP = C(5,2) + C(4,2) + C(3,2) + C(2,2) = 20

FP = 5*1 + 4*2 + 1*1 + 2*3 = 5 + 8 + 1 + 6 = 20

R = (20 + 72) / (20 + 20 + 24 + 72) = 0.68

K Means Clustering

Iterative over:

Assignment:

Assume there are K cluster centers $\boldsymbol{\mu}_k \in \mathbb{R}^D$

Each data point $\boldsymbol{x}_n \in \mathbb{R}^D$ is assigned to the closest center

$$z_n^* = \arg\min_k ||\boldsymbol{x}_n - \boldsymbol{\mu}_k||_2^2$$

Compute the centers:

Given the assignments of $\boldsymbol{x}_n$ to $z_n = k$

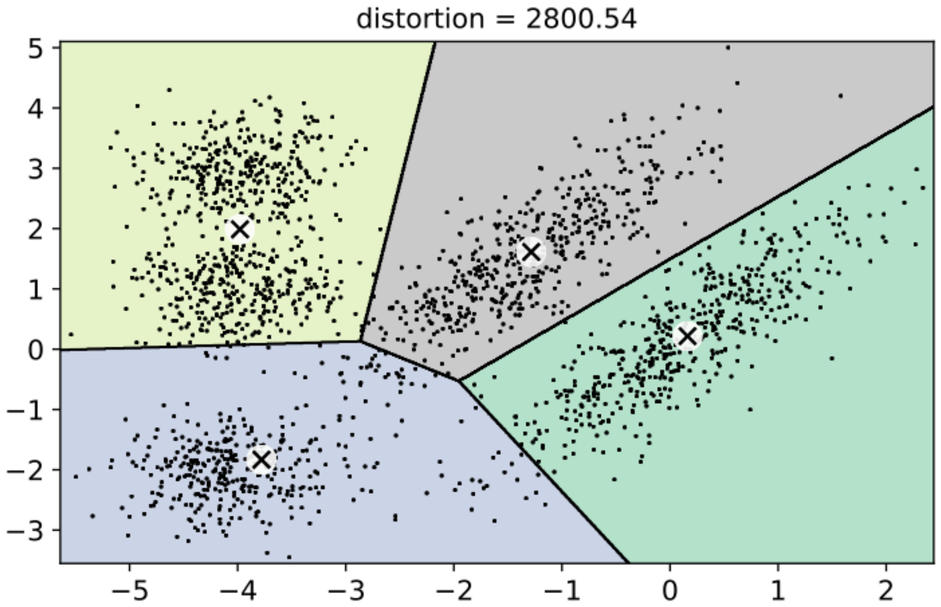
$$\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{n:z_n=k} \boldsymbol{x}_n$$

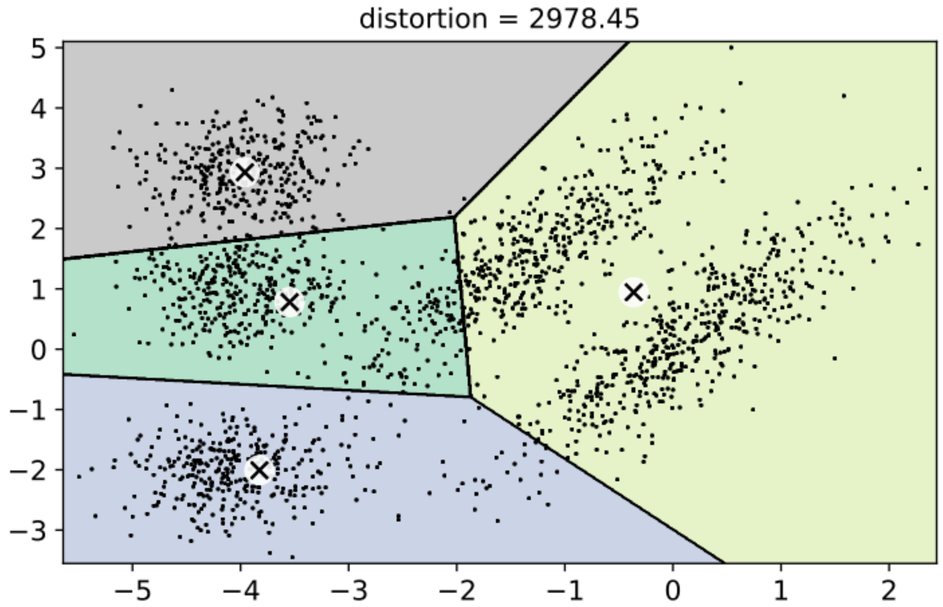
Multiple initialization!!

K Means Clustering

Finding a local minimum of **distortion**

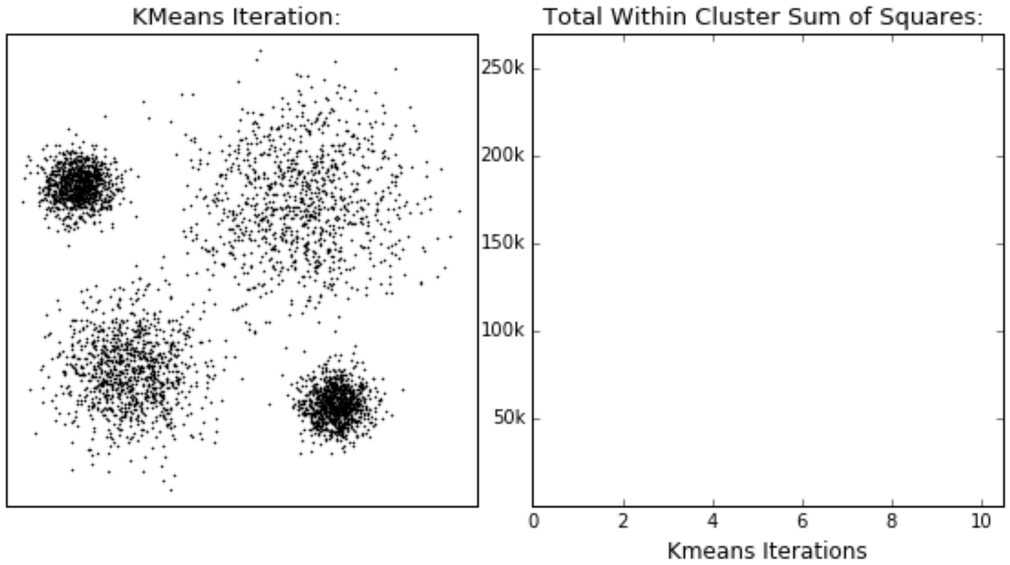
$$J(\mathbf{M}, \mathbf{Z}) = \sum_{n=1}^{N} ||\boldsymbol{x}_n - \boldsymbol{\mu}_{z_n}||^2 = ||\mathbf{X} - \mathbf{Z}\mathbf{M}^\mathsf{T}||_F^2$$

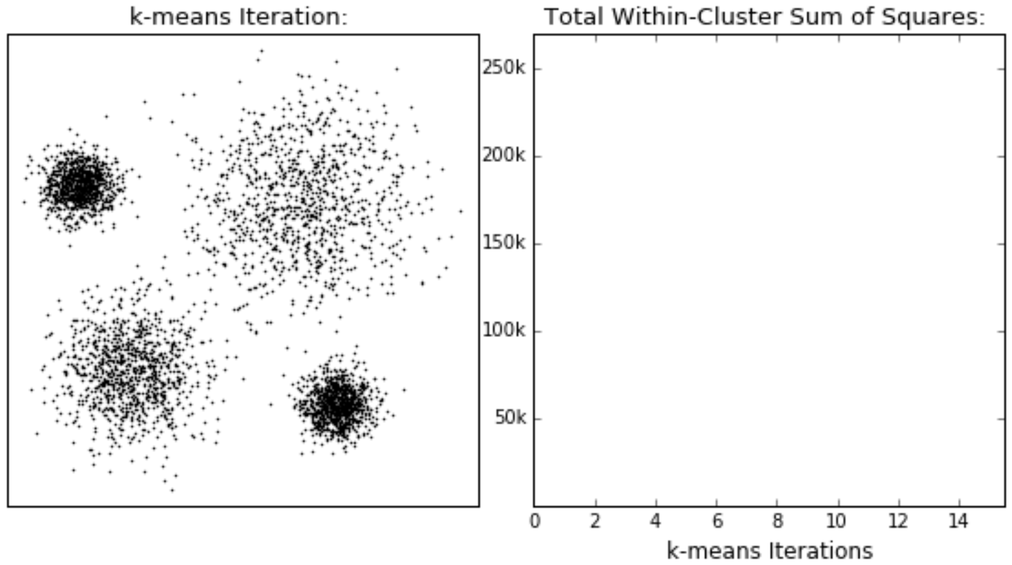




The resulting clustering is sensitive to the initialization!

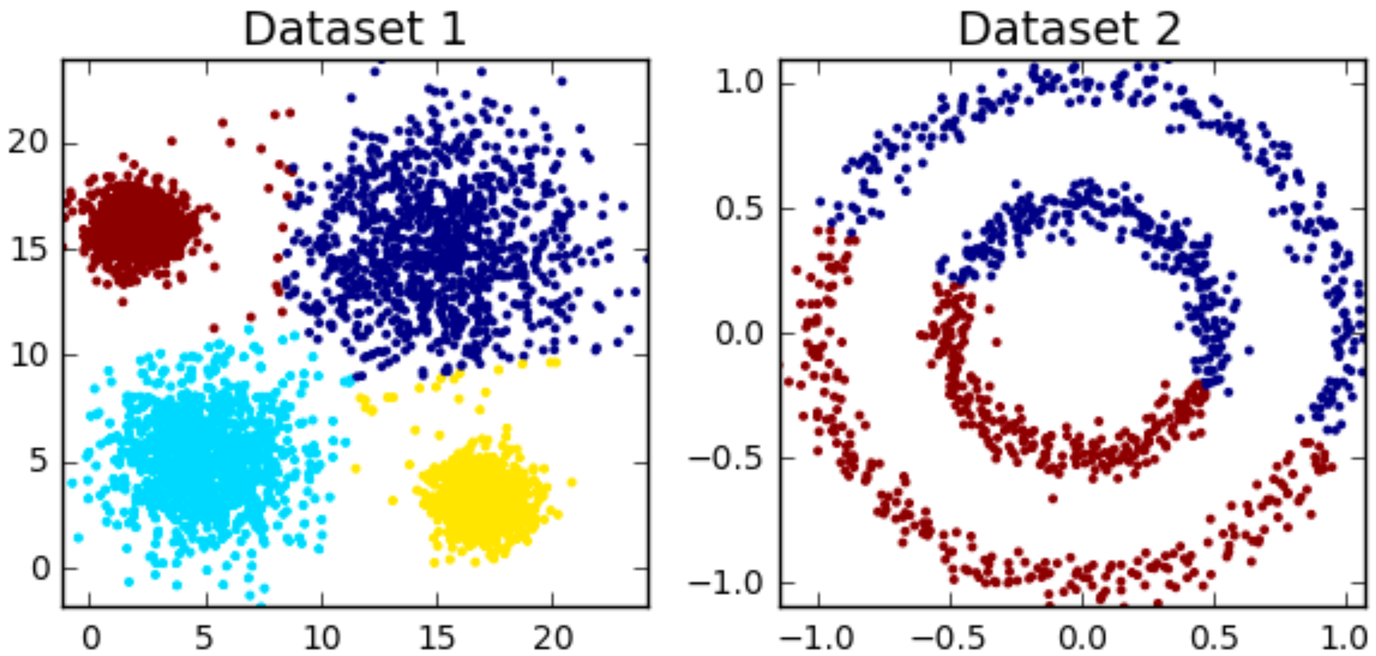
K Means Clustering





https://dashee87.github.io/data%20science/general/Clustering-with-Scikit-with-GIFs/

K Means Clustering



https://en.wikipedia.org/wiki/Spectral_clustering

https://dashee87.github.io/data%20science/general/Clustering-with-Scikit-with-GIFs/

K-means++ Algorithm

K-means is optimizing a non-convex objective, and hence needs to be initialized carefully.

Intuition: Pick the centers sequentially so as to try to “cover” the data.

At iteration t, we pick the next cluster center to be $\boldsymbol{x}_n$ with probability:

$$p(\boldsymbol{\mu}_t = \boldsymbol{x}_n) = \frac{D_{t-1}(\boldsymbol{x}_n)}{\sum_{n'=1}^{N} D_{t-1}(\boldsymbol{x}_{n'})}$$ farthest point clustering

where

$$D_t(\boldsymbol{x}) = \min_{k=1}^{t-1} ||\boldsymbol{x} - \boldsymbol{\mu}_k||_2^2$$ squared distance to the closest existing centroid

K-means++ Algorithm

K-means is optimizing a non-convex objective, and hence needs to be initialized carefully.

Intuition: Pick the centers sequentially so as to try to “cover” the data.

1. Randomly select the first centroid from the data points.

2. For each data point compute its distance from the nearest, previously chosen centroid.

3. Select the next centroid from the data points such that the probability of choosing a point as centroid is directly proportional to its distance from the nearest, previously chosen centroid. (i.e. the point having maximum distance from the nearest centroid is most likely to be selected next as a centroid)

4. Repeat steps 2 and 3 until k centroids have been sampled

K-medoids Algorithm

Estimate each cluster center μ_k by choosing the data example $x_n \in \mathcal{X}$ whose average dissimilarity to all other points in that cluster is minimal.

(1) More robust to outliers
(2) Can be applied to data that does not live in R^D, where the average is not well-defined

Algorithm 12: K-medoids algorithm

Initialize $m_{1:K}$ as a random subset of size K from $\{1, \ldots, N\}$;
repeat
  $z_n = \text{argmin}_k \, d(n, m_k)$ for $n = 1 : N$;
  $m_k = \text{argmin}_{n:z_n=k} \sum_{n':z_{n'}=k} d(n, n')$ for $k = 1 : K$;
until *converged*;

For each cluster k, look at all the points currently assigned to that cluster, and then set m_k to be the index of the media of that set.

For each point n, assign it to its closest medoid. $z_n = \text{argmin}_k \, D(n, k)$

Clustering Using Mixture Models

Gaussian mixture model (GMM)

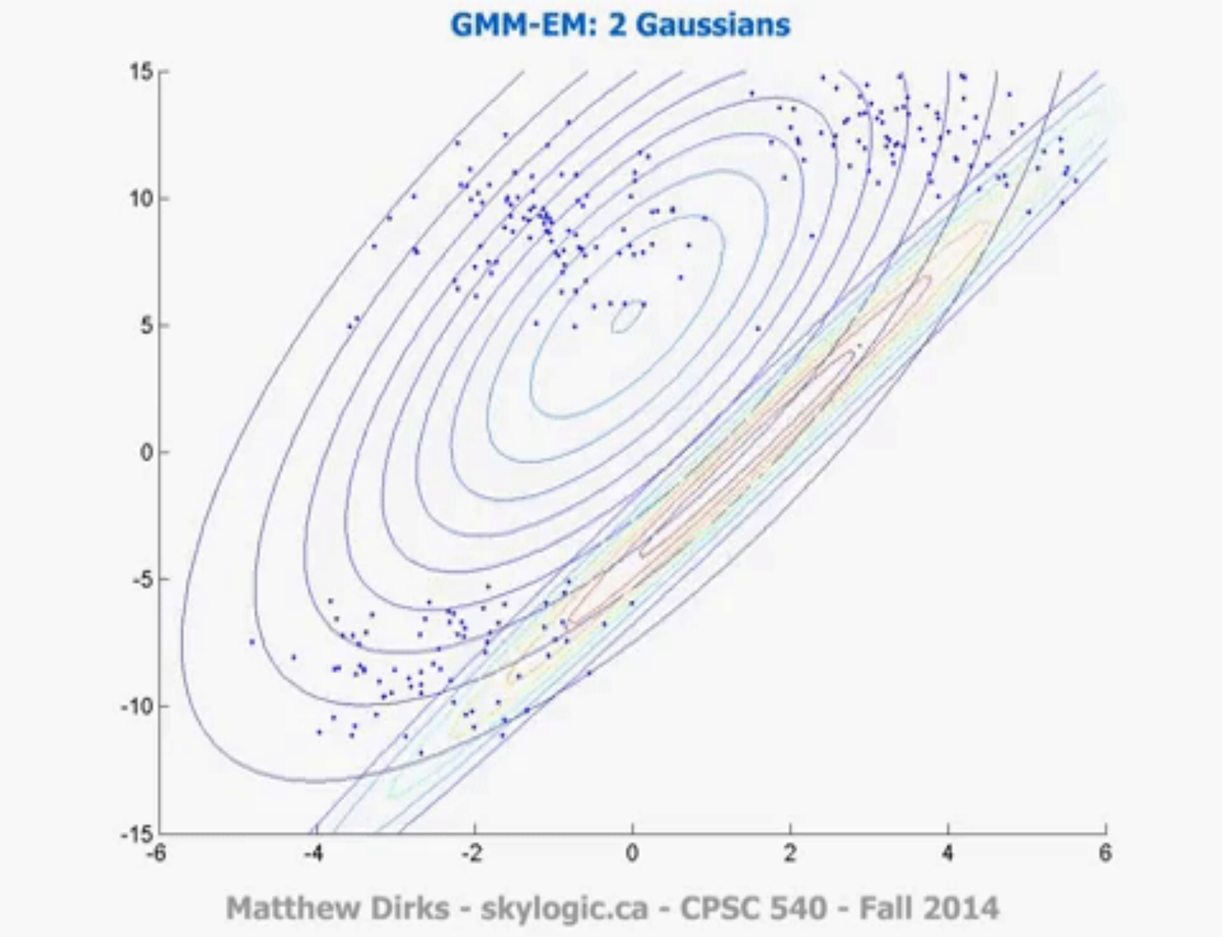
$$p(\boldsymbol{x}|\boldsymbol{\theta}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

We can use Bayes rule to compute the responsibility (posterior membership probability) of cluster k for data point x_n

$$r_{nk} \triangleq p(z_n = k|\boldsymbol{x}_n, \boldsymbol{\theta}) = \frac{p(z_n = k|\boldsymbol{\theta})p(\boldsymbol{x}_n|z_n = k, \boldsymbol{\theta})}{\sum_{k'=1}^{K} p(z_n = k'|\boldsymbol{\theta})p(\boldsymbol{x}_n|z_n = k', \boldsymbol{\theta})}$$

EM algorithm — similar to the one used in K-means

Clustering Using Mixture Models



K-means and EM

Iterative over:

Assignment (E-step):

Assume there are K cluster centers $\boldsymbol{\mu}_k \in \mathbb{R}^D$

Each data point $\boldsymbol{x}_n \in \mathbb{R}^D$ is assigned to the closest center

$$z_n^* = \arg\min_k ||\boldsymbol{x}_n - \boldsymbol{\mu}_k||_2^2$$

Compute the centers (M-Step):

Given the assignments of $\boldsymbol{x}_n$ to $z_n = k$

$$\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{n:z_n=k} \boldsymbol{x}_n$$

$$p(\boldsymbol{x}|\boldsymbol{\theta}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

Hard instead of soft

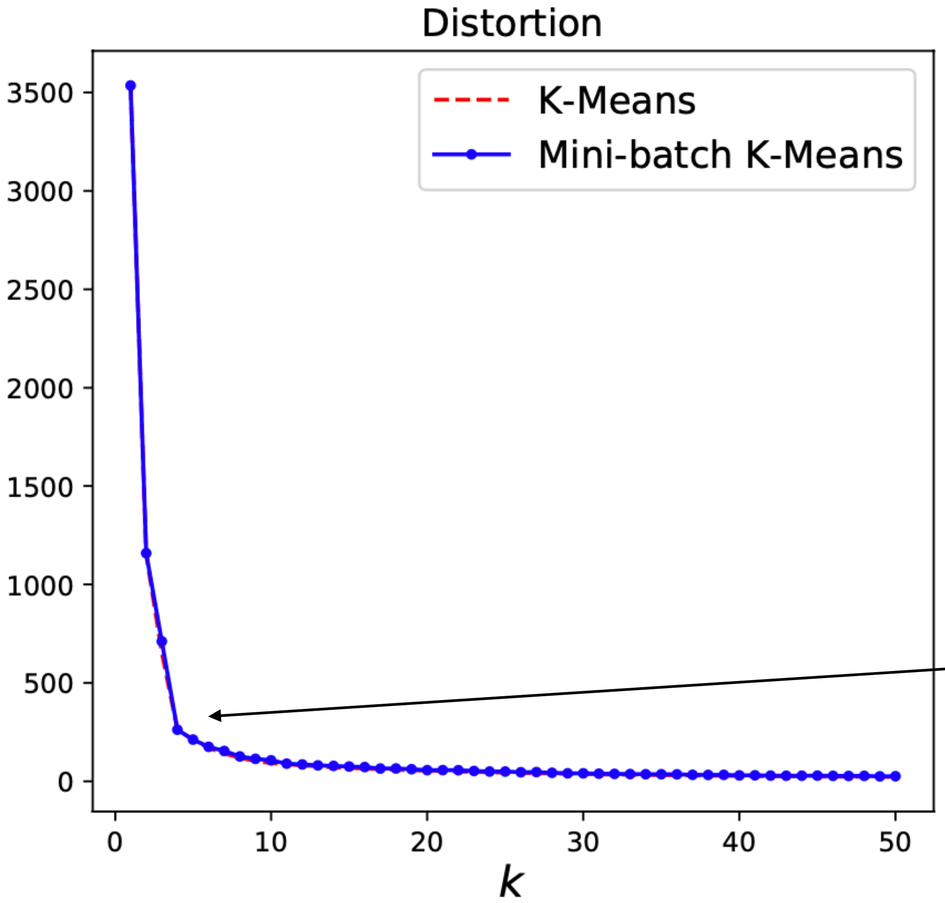
$$z_n^* = \text{argmax}_k \, r_{nk}$$

Simplify

$$\boldsymbol{\Sigma}_k = \mathbf{I}$$

$$\pi_k = 1/K$$

Choosing the Number of Clusters K



Minimize the distortion on a validation set?

$$\text{err}(\mathcal{D}_{\text{valid}}, K) = \frac{1}{|\mathcal{D}_{\text{valid}}|} \sum_{n \in \mathcal{D}_{\text{valid}}} ||\boldsymbol{x}_n - \hat{\boldsymbol{x}}_n||_2^2$$

Can the distortion figure tell some information?

Elbow!

Choosing the Number of Clusters K

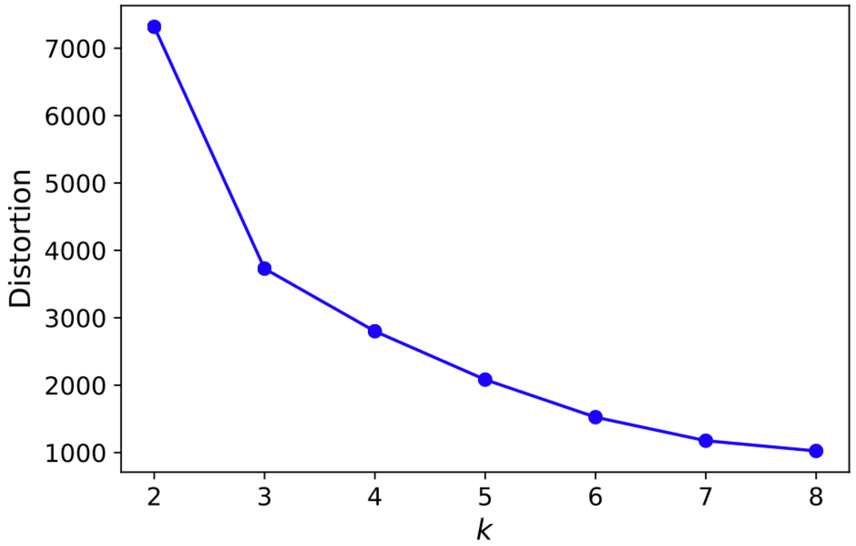
Maximizing the marginal likelihood

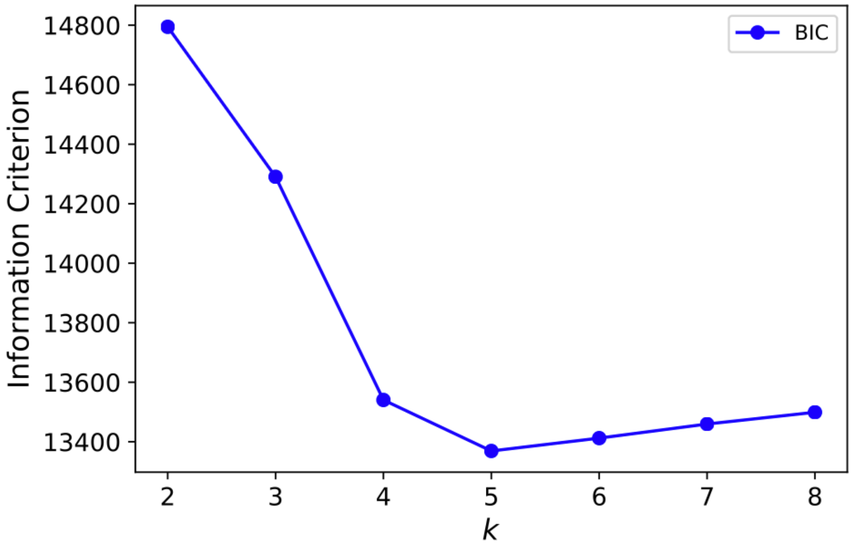
$$\mathrm{BIC}(K) = \log p(\mathcal{D}|\hat{\boldsymbol{\theta}}_k) - \frac{D_K}{2}\log(N)$$

BIC: Bayesian Information Criterion

MLE, fits K Gaussian to the clusters

Number of parameters in a model with K clusters





Vector Quantization (Optional)

A very simple approach to performing lossy compression of some real-valued vectors.

Replace each real-valued vector $\boldsymbol{x}_n$ with a discrete symbol $z_n \in \{1, \ldots, K\}$

$$\text{encode}(\boldsymbol{x}_n) = \arg\min_k ||\boldsymbol{x}_n - \boldsymbol{\mu}_k||^2$$

The quality of a codebook by computing the reconstruction error or distortion it induces:

$$J \triangleq \frac{1}{N}\sum_{n=1}^{N} ||\boldsymbol{x}_n - \text{decode}(\text{encode}(\boldsymbol{x}_n))||^2 = \frac{1}{N}\sum_{n=1}^{N} ||\boldsymbol{x}_n - \boldsymbol{\mu}_{z_n}||^2$$

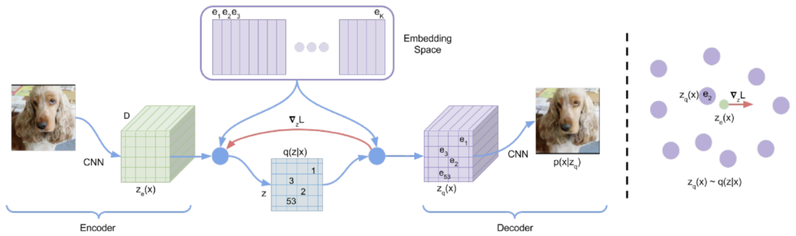
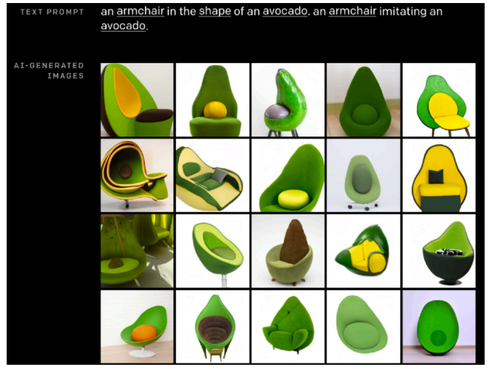
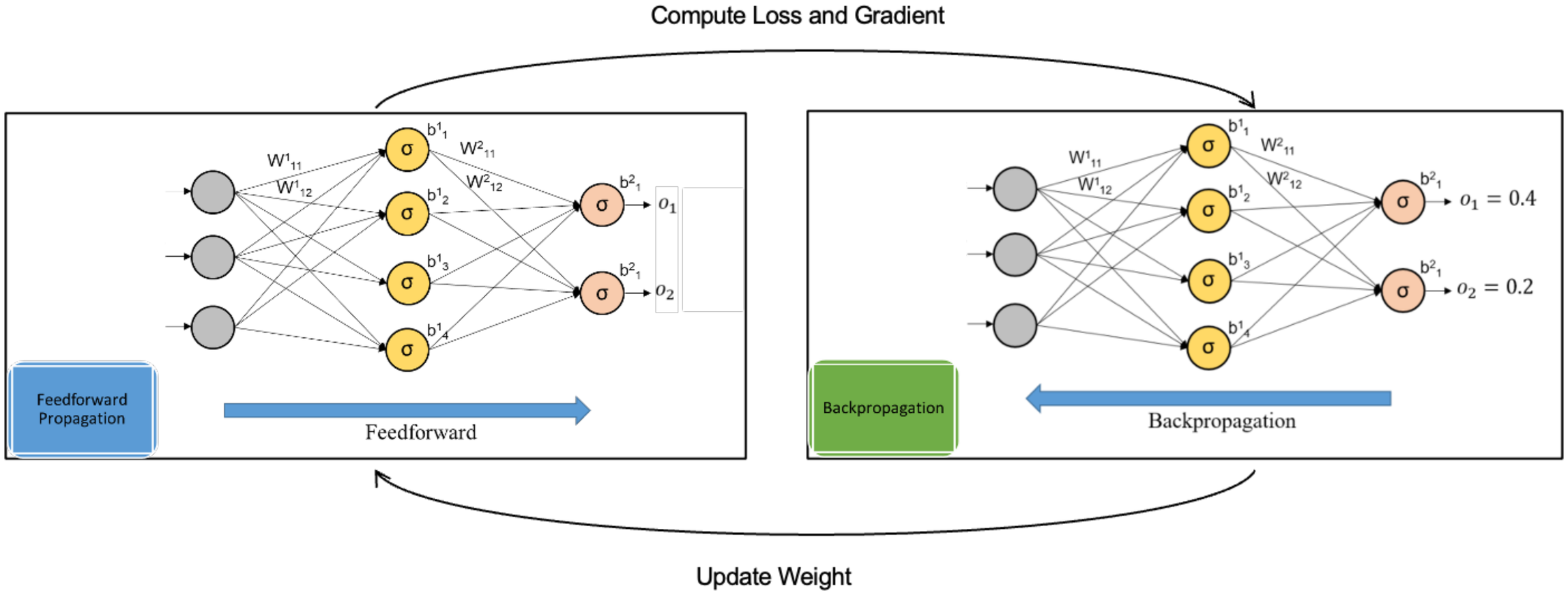


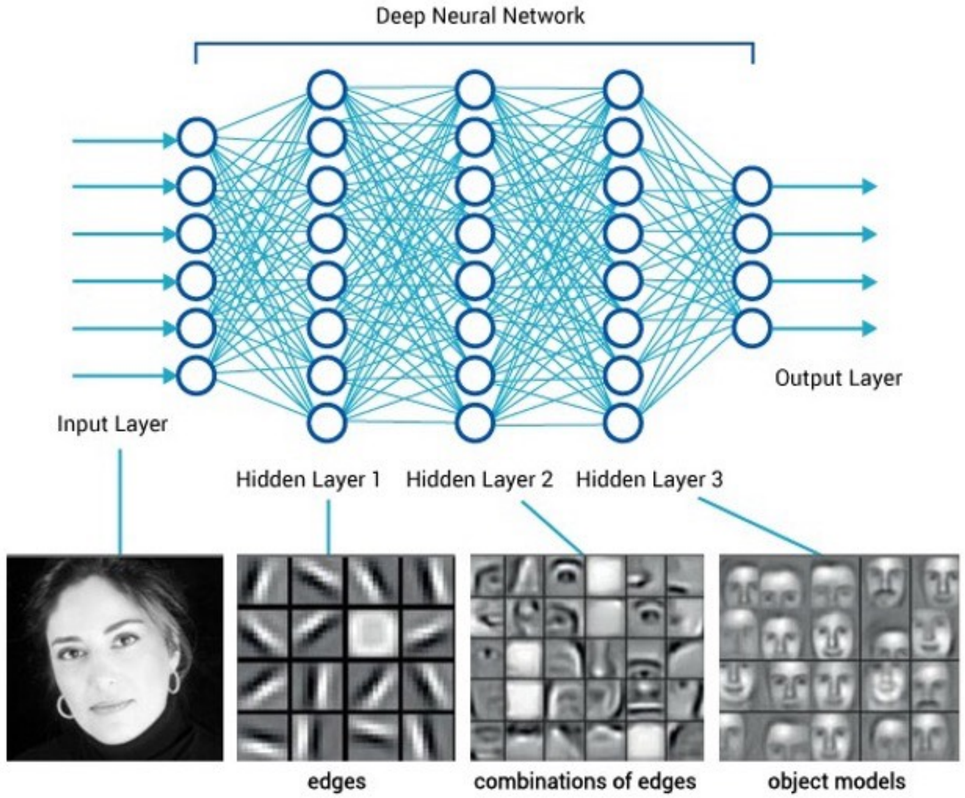
Figure 1: Left: A figure describing the VQ-VAE. Right: Visualisation of the embedding space. The output of the encoder $z(x)$ is mapped to the nearest point e_2. The gradient $\nabla_z L$ (in red) will push the encoder to change its output, which could alter the configuration in the next forward pass.

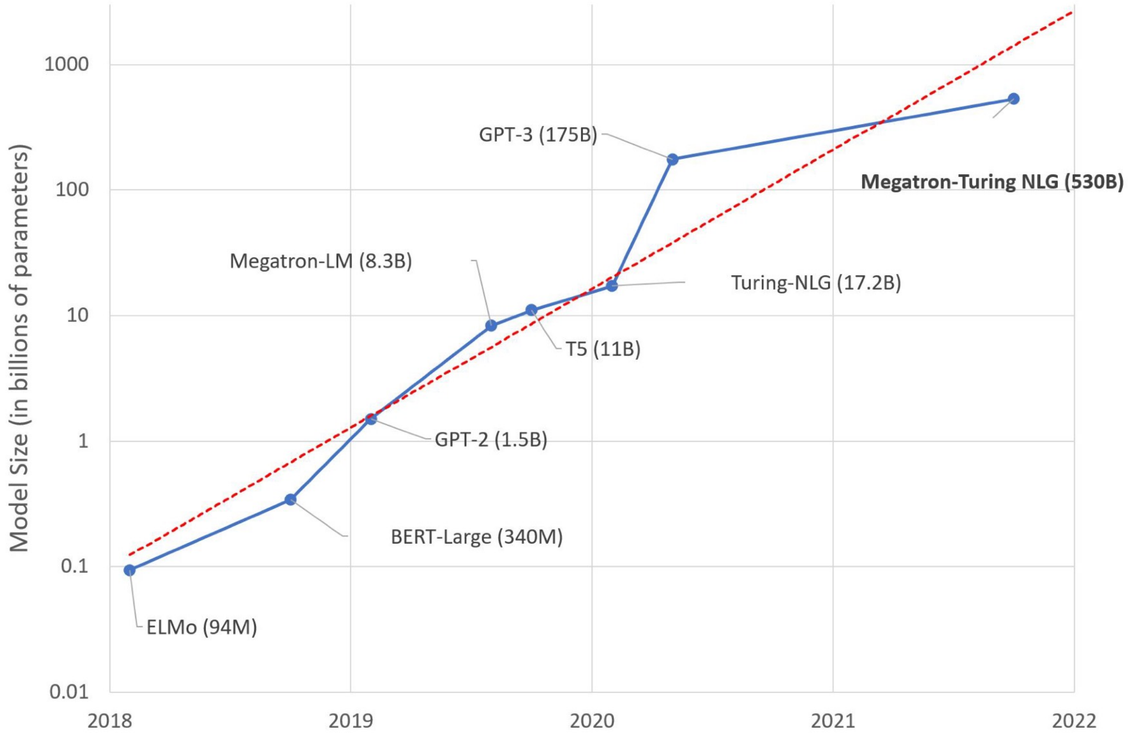


Recap: Feedforward Neural Network



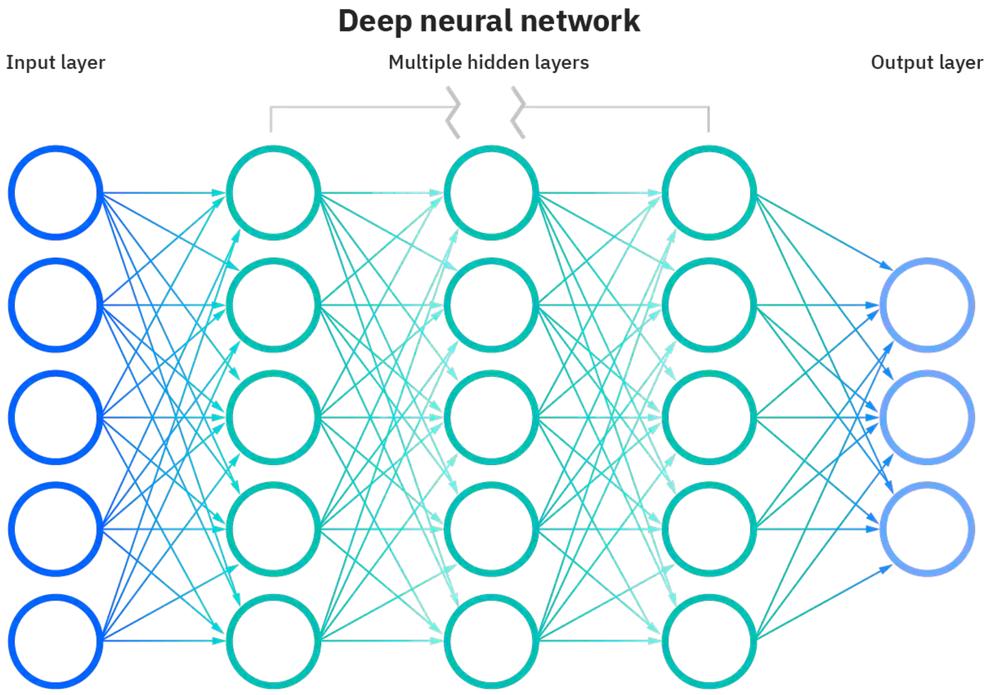
Recap: Larger Networks





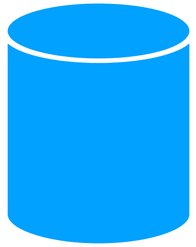
Microsoft Research Blog. Oct 6, 2021.

What do we get from this?



$\{w_{ij}^{l}\}$

Case #1:



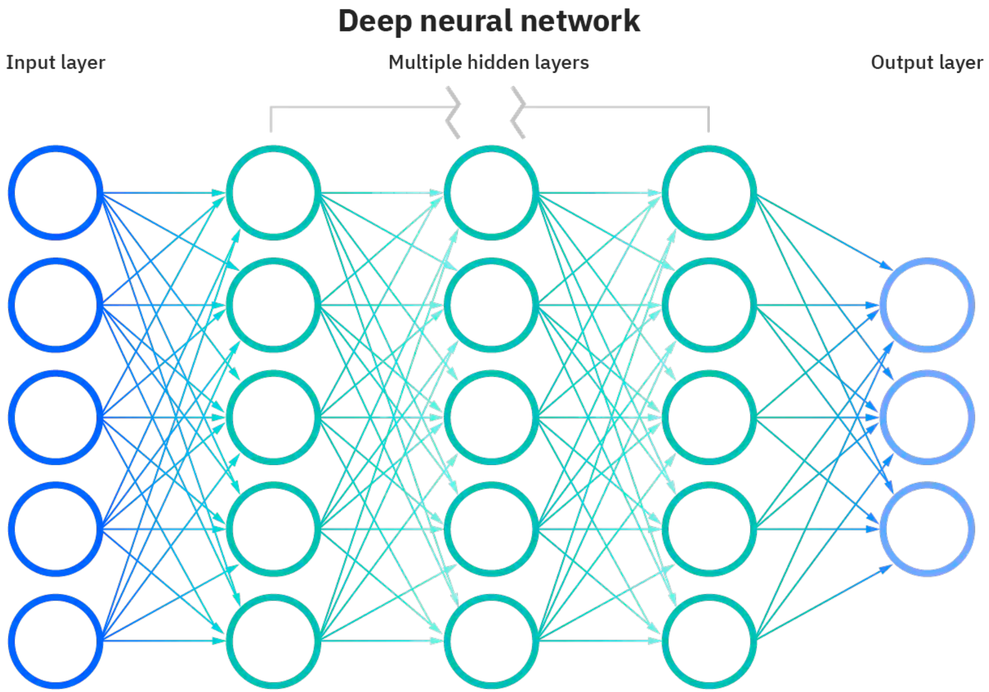
$$\mathcal{D}_{\text{train}} = \{(\boldsymbol{x}^{(n)}, \boldsymbol{y}^{(n)})\}_{n=1}^{N_{\text{train}}}$$

Results:

$$f_{\boldsymbol{\theta}}(\boldsymbol{x}) \rightarrow \boldsymbol{y}$$

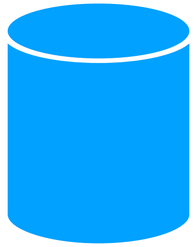
“product”

What do we get from this?



$\{w_{ij}^l\}$

Case #2:



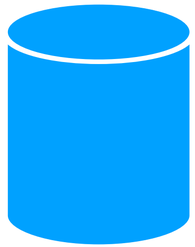
$$\mathcal{D}_{\text{train}} = \{(\boldsymbol{x}^{(n)}, \boldsymbol{y}^{(n)})\}_{n=1}^{N_{\text{train}}}$$

Results:

$\{w_{ij}^l\}$

“product”

“Unsupervised” Representation Learning



$$\mathcal{D}_{\text{train}} = \{(\boldsymbol{x}^{(n)}, \boldsymbol{y}^{(n)})\}_{n=1}^{N_{\text{train}}}$$

In supervised learning setting — annotation

design “unsupervised” learning tasks

original sentence:

I wish you all the very best

unsupervised learning task:

output: very $\boldsymbol{y}$

input: I wish you all the <mask> best $\boldsymbol{x}$

“Unsupervised” Representation Learning

original sentence:

I wish you all the very best

unsupervised learning task:

input: I wish output: you

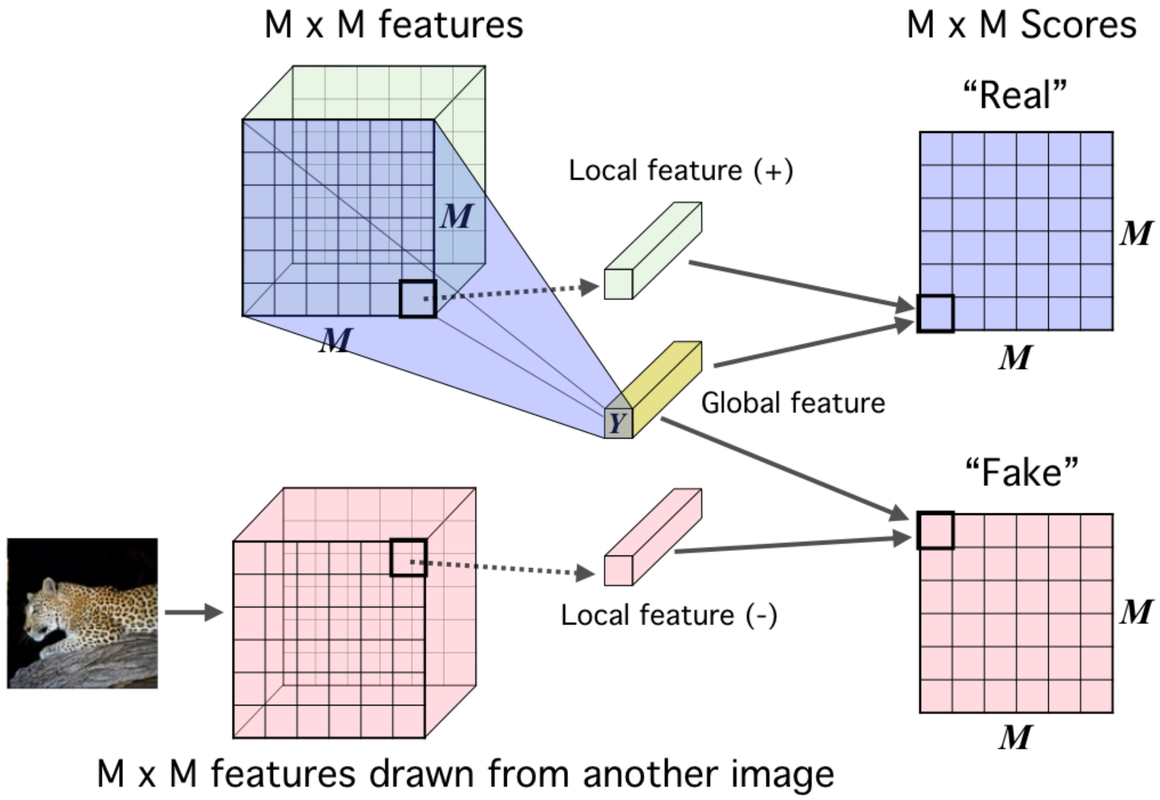
input: I wish you output: all

input: I wish you all output: the

⋮

Language Modeling Task

contrastive learning tasks:



The Outcome of Unsupervised Representation Learning

$$f_{\boldsymbol{\theta}}(\boldsymbol{x}) \rightarrow \boldsymbol{y}$$

$$\boldsymbol{\theta}$$

input: I wish you all output: the

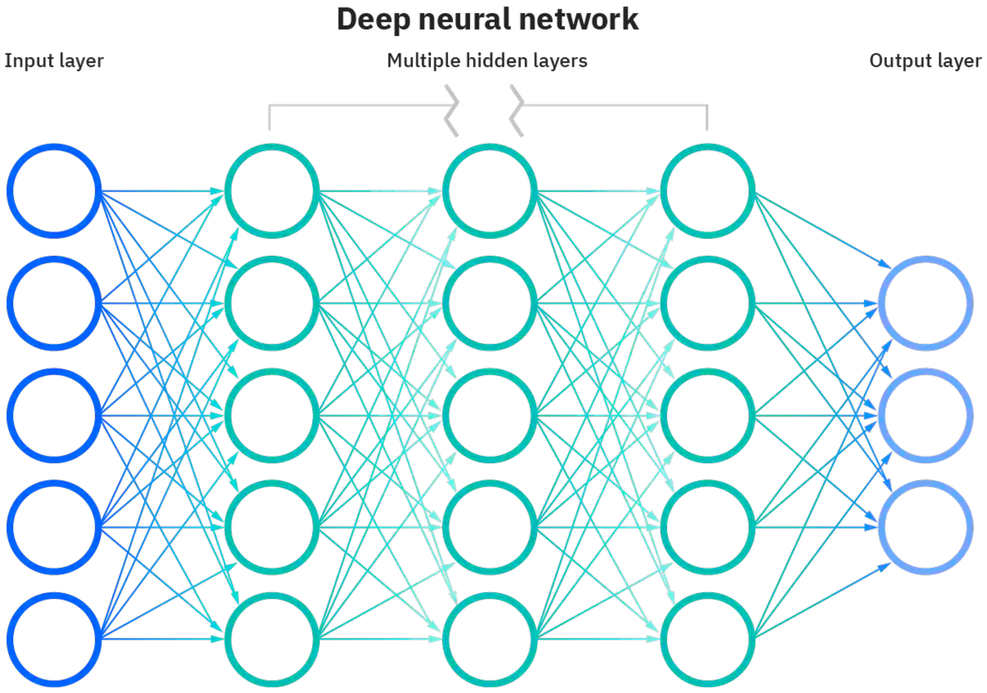
You need to do a better job understanding me.

Noted.

Yeah, make a note of that.

Here's your note:

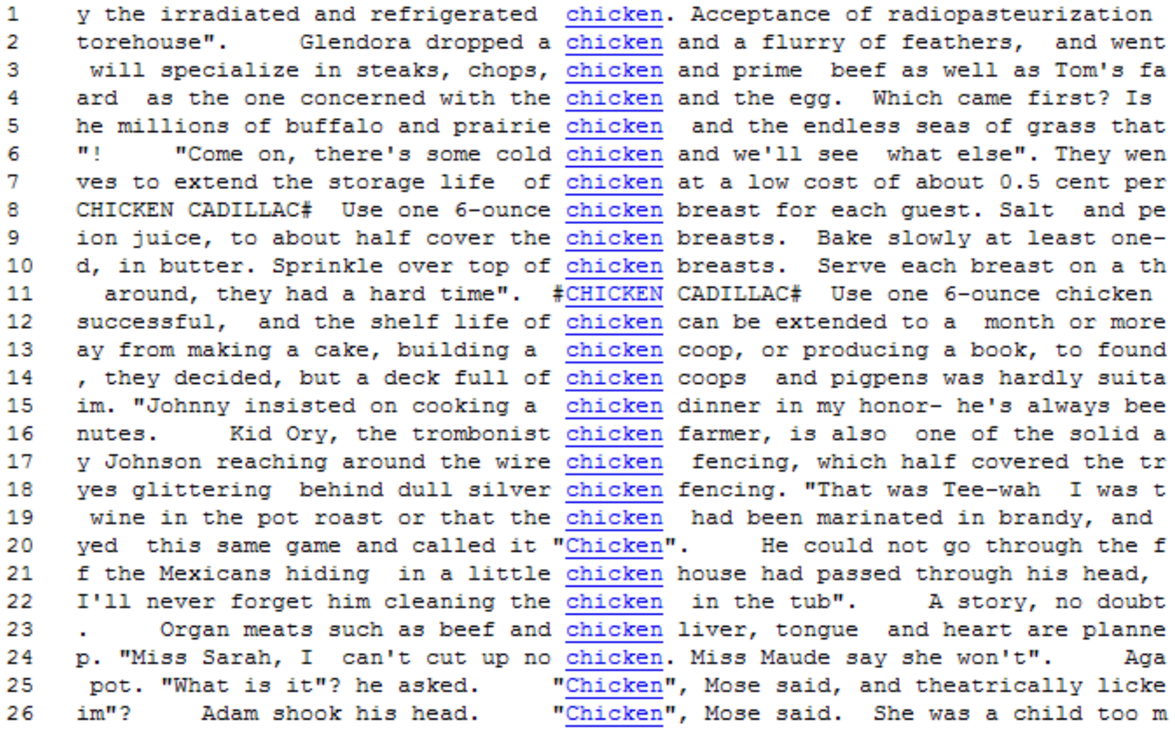
of that



$$\{w_{ij}^{l}\}$$

$$g_{\{\boldsymbol{\theta},\boldsymbol{\theta'}\}}(\boldsymbol{x}) \rightarrow \boldsymbol{y}$$ sentiment analysis?

Why this is useful?



Where is “chicken” ?

I want to eat some fruits today.
Therefore, I will eat <mask> tonight.

apple

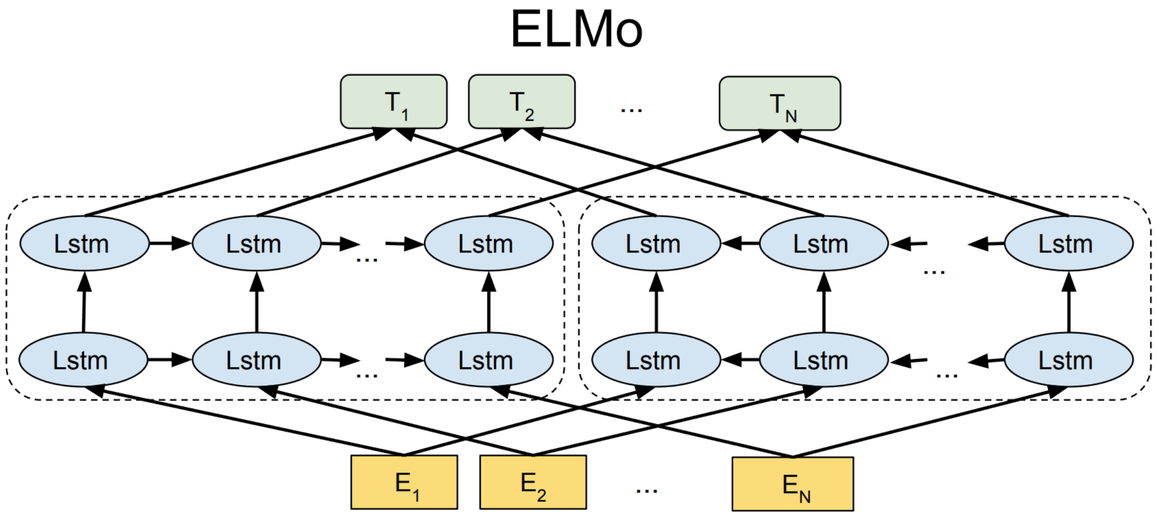
watermelon

θ representation -> semantic meaning

“transfer”

useful in other tasks

ELMo (Embeddings from Language Models)



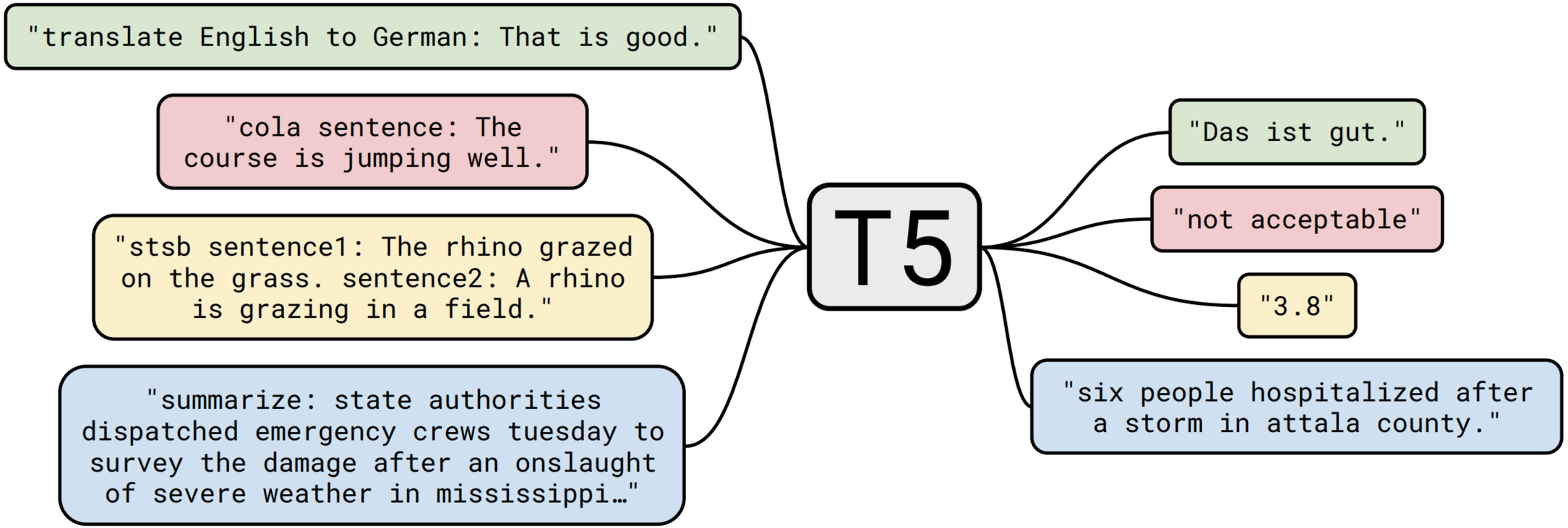
$$\sum_{k=1}^{N} \Big(\log p(t_k \mid t_1, \ldots, t_{k-1}; \Theta_x, \overrightarrow{\Theta}_{LSTM}, \Theta_s) + \log p(t_k \mid t_{k+1}, \ldots, t_N; \Theta_x, \overleftarrow{\Theta}_{LSTM}, \Theta_s) \Big)$$

Bidirectional Language Model

Peters et al., 2018

T5 (Text-to-Text Transfer Transformer)

Multi-task learning — using one set of parameters for many different tasks



Raffel et al., 2020