DSC291: Advanced Statistical Natural Language Processing

Self-supervised Learning

Zhiting Hu

Lecture 6, April 14, 2022

UC San Diego
HALICIOĞLU DATA SCIENCE INSTITUTE

Outline

- Contrastive learning (a special self-supervised learning)
- Unsupervised Learning

Representation Learning with Contrastive Learning

Contrastive learning

- Take a data example x, sample a “positive” sample x_{pos} and “negative” samples x_{neg} in some way
- Then try fit a scoring model such that

$$score(x, x_{pos}) > score(x, x_{neg})$$

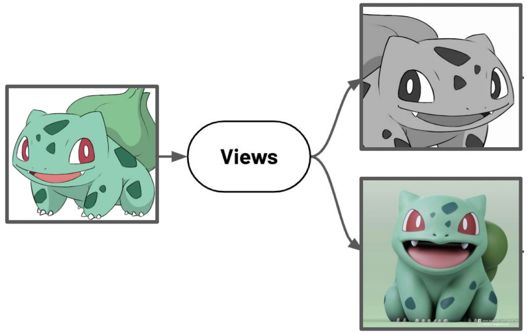
Credit: [CVPR 2021 Tutorial] Leave Those Nets Alone: Advances in Self-Supervised Learning

Contrastive learning

- Take a data example x, sample a "positive" sample x_{pos} and "negative" samples x_{neg} in some way

"positive" sample:

- Data of the same labels
- Data of the same pseudo-labels
- Augmented/distorted version of x
- Data that captures the same target from different views



"negative" sample:

- Randomly sampled data
- Hard negative sample mining

Contrastive learning

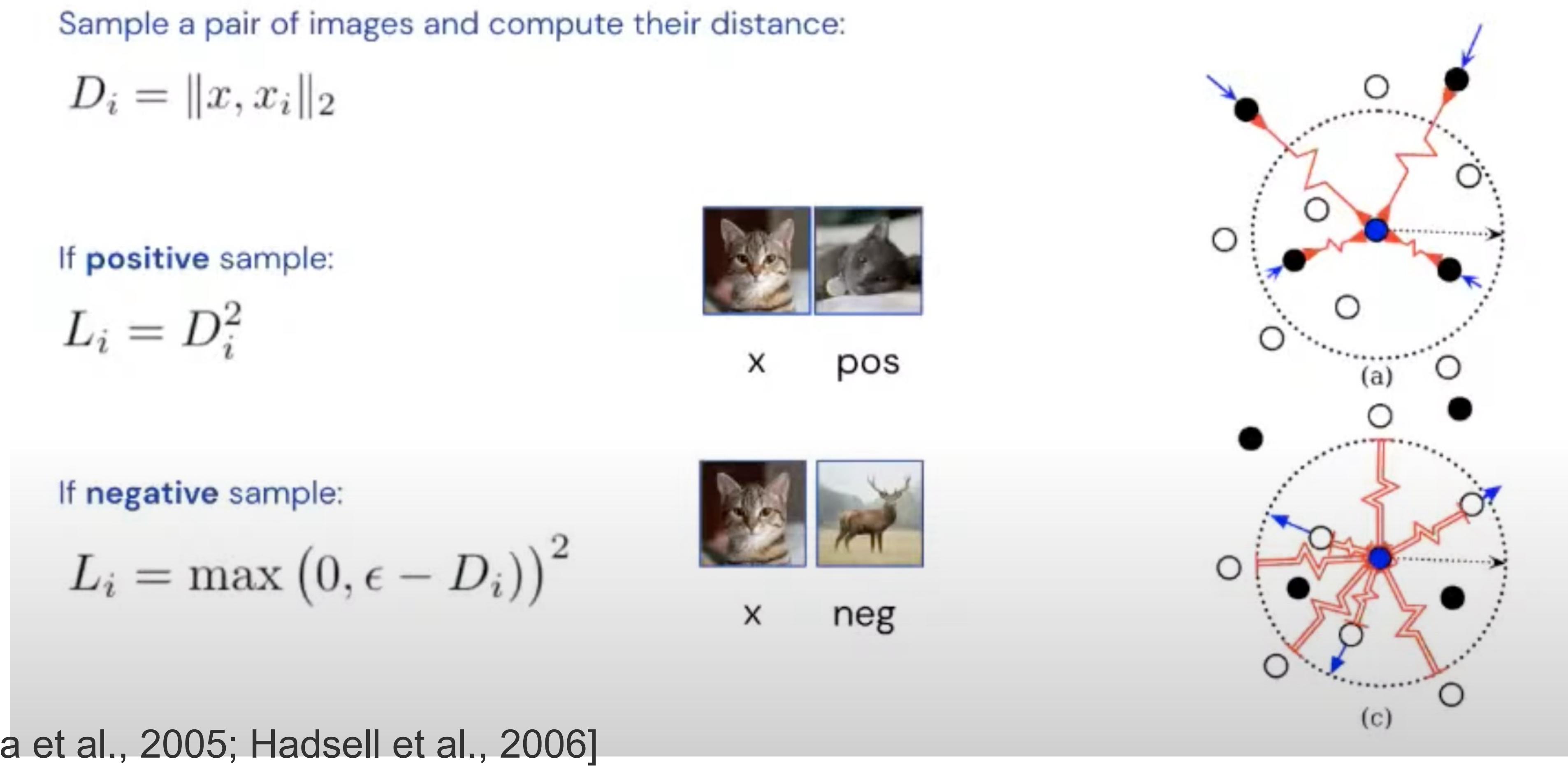
- Take a data example x, sample a “positive” sample x_{pos} and “negative” samples x_{neg} in some way
- Then try fit a scoring model such that

$$score(x, x_{pos}) > score(x, x_{neg})$$

Credit: [CVPR 2021 Tutorial] Leave Those Nets Alone: Advances in Self-Supervised Learning

Contrastive learning: Ex 1

Learning a similarity metric discriminatively



Sample a pair of images and compute their distance:

$$D_i = \|x, x_i\|_2$$

If **positive** sample:

$$L_i = D_i^2$$

If **negative** sample:

$$L_i = \max \big(0, \epsilon - D_i)\big)^2$$

[Chopra et al., 2005; Hadsell et al., 2006]

Credit: [CVPR 2021 Tutorial] Leave Those Nets Alone: Advances in Self-Supervised Learning

Common contrastive learning functions

- Contrastive loss (Chopra et al. 2005)
- Triplet loss (Schroff et al. 2015; FaceNet)
- Lifted structured loss (Song et al. 2015)
- Multi-class n-pair loss (Sohn 2016)
- Noise contrastive estimation (“NCE”; Gutmann & Hyvarinen 2010)
- InfoNCE (van den Oord, et al. 2018)
- Soft-nearest neighbors loss (Salakhutdinov & Hinton 2007, Frosst et al. 2019)

[Courtesy: Weng & Kim, NeurIPS 2021 tutorial]

Contrastive learning: Ex 2

- SimCSE ("Simple Contrastive learning of Sentence Embeddings"; Gao et al. 2021)
  - Predict a sentence from itself with only dropout noise
  - One sentence gets two different versions of dropout augmentations

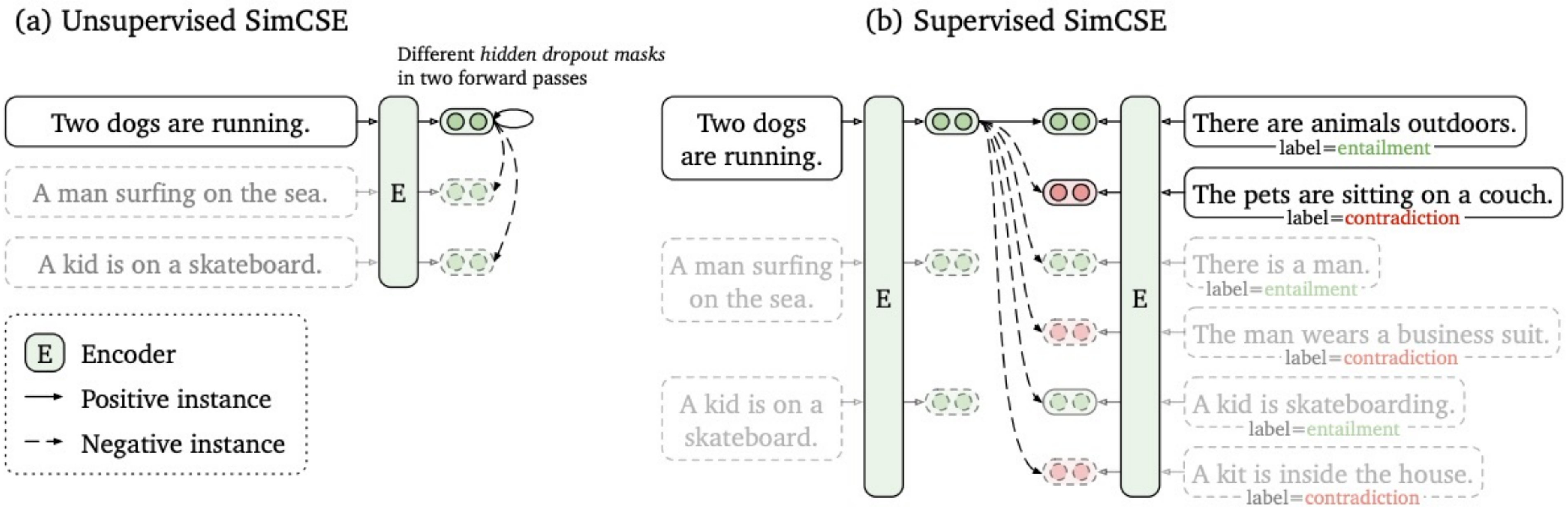
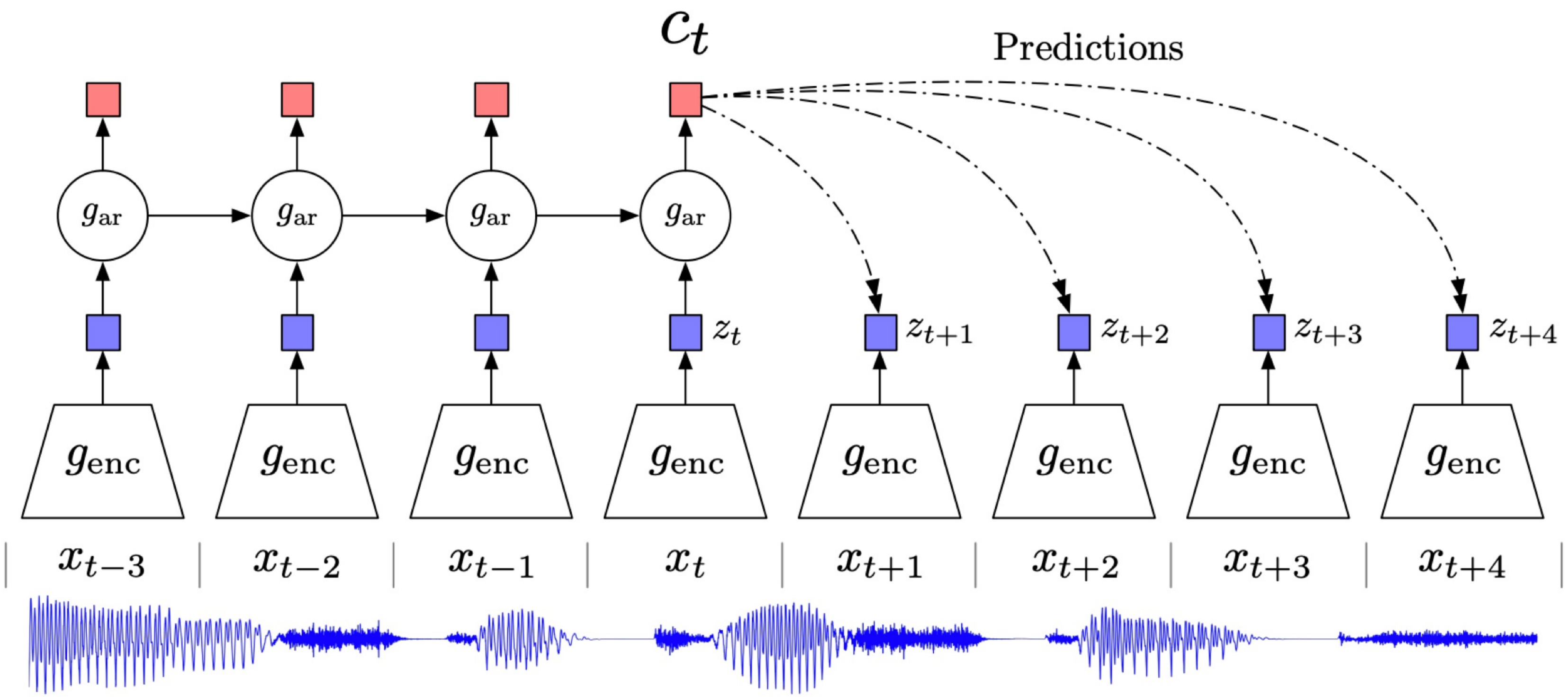


Figure 1: (a) Unsupervised SimCSE predicts the input sentence itself from in-batch negatives, with different hidden dropout masks applied. (b) Supervised SimCSE leverages the NLI datasets and takes the entailment (premise-hypothesis) pairs as positives, and contradiction pairs as well as other in-batch instances as negatives.

Contrastive learning: Ex 3 - InfoNCE

- The CPC model
  - c_t: context representation from history
  - x_{t+k} (or z_{t+k}): future target



[van den Oord et al., “Representation Learning with Contrastive Predictive Coding”]

InfoNCE loss

- Define scoring function $f_k > 0$
- The InfoNCE (Noise-Contrastive Estimation) loss:
  - Given $X = \{$ one positive sample from $p(x_{t+k} | c_t)$, $N - 1$ negative samples from the negative sampling distribution $p(x_{t+k})$ $\}$

$$\mathcal{L}_{\mathrm{N}} = - \mathop{\mathbb{E}}_{X} \left[\log \frac{f_k(x_{t+k}, c_t)}{\sum_{x_j \in X} f_k(x_j, c_t)} \right]$$

- InfoNCE is interesting because it’s effectively maximizing the mutual information between c_t and x_{t+k}

[van den Oord et al., “Representation Learning with Contrastive Predictive Coding”]

Mutual Information (MI)

- How much is our uncertainty about x reduced by knowing c ?

$$I(x;c) = \sum_{x,c} p(x,c) \log \frac{p(x,c)}{p(x)p(c)} = \sum_{x,c} p(x,c) \log \frac{p(x|c)}{p(x)}$$

$$= H(x) + H(c) - H(x,c)$$

$$= H(x) - H(x|c)$$

$$= KL(p(x,c) \,||\, p(x)p(c))$$

[van den Oord et al., “Representation Learning with Contrastive Predictive Coding”]

Minimizing InfoNCE <=> Maximzing MI

- InfoNCE loss

$$\mathcal{L}_{\mathrm{N}} = -\mathop{\mathbb{E}}_{X}\left[\log \frac{f_k(x_{t+k}, c_t)}{\sum_{x_j \in X} f_k(x_j, c_t)}\right]$$

- The loss is optimized when

$$f_k(x_{t+k}, c_t) \propto \frac{p(x_{t+k}|c_t)}{p(x_{t+k})}$$

  - Proof:

$$p(sample\ i\ is\ positive|X, c_t) = \frac{p(x_i|c_t)\prod_{l \neq i} p(x_l)}{\sum_{j=1}^{N} p(x_j|c_t)\prod_{l \neq j} p(x_l)}$$

$$= \frac{\frac{p(x_i|c_t)}{p(x_i)}}{\sum_{j=1}^{N} \frac{p(x_j|c_t)}{p(x_j)}}.$$

[van den Oord et al., "Representation Learning with Contrastive Predictive Coding"]

- How does this loss maximize the mutual information?

$$\mathcal{L}_{\mathrm{N}} = -\mathop{\mathbb{E}}_{X}\left[\log \frac{f_k(x_{t+k}, c_t)}{\sum_{x_j \in X} f_k(x_j, c_t)}\right]$$

[van den Oord et al., “Representation Learning with Contrastive Predictive Coding”]

- How does this loss maximize the mutual information?

$$\mathcal{L}_{\mathrm{N}} = -\underset{X}{\mathbb{E}}\left[\log \frac{f_k(x_{t+k}, c_t)}{\sum_{x_j \in X} f_k(x_j, c_t)}\right]$$

$$\mathcal{L}_{\mathrm{N}}^{\mathrm{opt}} = -\underset{X}{\mathbb{E}} \log\left[\frac{\frac{p(x_{t+k}|c_t)}{p(x_{t+k})}}{\frac{p(x_{t+k}|c_t)}{p(x_{t+k})} + \sum_{x_j \in X_{\mathrm{neg}}} \frac{p(x_j|c_t)}{p(x_j)}}\right]$$

Use proportionality condition

[van den Oord et al., “Representation Learning with Contrastive Predictive Coding”]

- How does this loss maximize the mutual information?

$$\mathcal{L}_{\mathrm{N}} = -\mathop{\mathbb{E}}_{X}\left[\log \frac{f_k(x_{t+k}, c_t)}{\sum_{x_j \in X} f_k(x_j, c_t)}\right]$$

$$\mathcal{L}_{\mathrm{N}}^{\mathrm{opt}} = -\mathop{\mathbb{E}}_{X} \log\left[\frac{\frac{p(x_{t+k}|c_t)}{p(x_{t+k})}}{\frac{p(x_{t+k}|c_t)}{p(x_{t+k})} + \sum_{x_j \in X_{\mathrm{neg}}} \frac{p(x_j|c_t)}{p(x_j)}}\right]$$

$$= \mathop{\mathbb{E}}_{X} \log\left[1 + \frac{p(x_{t+k})}{p(x_{t+k}|c_t)} \sum_{x_j \in X_{\mathrm{neg}}} \frac{p(x_j|c_t)}{p(x_j)}\right]$$

Take -ve inside log

[van den Oord et al., “Representation Learning with Contrastive Predictive Coding”]

- How does this loss maximize the mutual information?

$$\mathcal{L}_{\mathrm{N}} = -\mathop{\mathbb{E}}_{X}\left[\log \frac{f_k(x_{t+k}, c_t)}{\sum_{x_j \in X} f_k(x_j, c_t)}\right]$$

$$\begin{aligned}
\mathcal{L}_{\mathrm{N}}^{\mathrm{opt}} &= -\mathop{\mathbb{E}}_{X} \log \left[\frac{\frac{p(x_{t+k}|c_t)}{p(x_{t+k})}}{\frac{p(x_{t+k}|c_t)}{p(x_{t+k})} + \sum_{x_j \in X_{\mathrm{neg}}} \frac{p(x_j|c_t)}{p(x_j)}}\right] \\
&= \mathop{\mathbb{E}}_{X} \log \left[1 + \frac{p(x_{t+k})}{p(x_{t+k}|c_t)} \sum_{x_j \in X_{\mathrm{neg}}} \frac{p(x_j|c_t)}{p(x_j)}\right] \\
&\approx \mathop{\mathbb{E}}_{X} \log \left[1 + \frac{p(x_{t+k})}{p(x_{t+k}|c_t)} (N-1) \mathop{\mathbb{E}}_{x_j} \frac{p(x_j|c_t)}{p(x_j)}\right]
\end{aligned}$$

This approximation becomes more accurate as N increases, so it is preferable to use large negative samples

[van den Oord et al., “Representation Learning with Contrastive Predictive Coding”]

- How does this loss maximize the mutual information?

$$\mathcal{L}_{\mathrm{N}} = -\mathop{\mathbb{E}}_{X}\left[\log \frac{f_k(x_{t+k}, c_t)}{\sum_{x_j \in X} f_k(x_j, c_t)}\right]$$

$$\begin{aligned}
\mathcal{L}_{\mathrm{N}}^{\mathrm{opt}} &= -\mathop{\mathbb{E}}_{X} \log \left[\frac{\frac{p(x_{t+k}|c_t)}{p(x_{t+k})}}{\frac{p(x_{t+k}|c_t)}{p(x_{t+k})} + \sum_{x_j \in X_{\mathrm{neg}}} \frac{p(x_j|c_t)}{p(x_j)}}\right] \\
&= \mathop{\mathbb{E}}_{X} \log \left[1 + \frac{p(x_{t+k})}{p(x_{t+k}|c_t)} \sum_{x_j \in X_{\mathrm{neg}}} \frac{p(x_j|c_t)}{p(x_j)}\right] \\
&\approx \mathop{\mathbb{E}}_{X} \log \left[1 + \frac{p(x_{t+k})}{p(x_{t+k}|c_t)} (N-1) \mathop{\mathbb{E}}_{x_j} \frac{p(x_j|c_t)}{p(x_j)}\right] \qquad = 1 \\
&= \mathop{\mathbb{E}}_{X} \log \left[1 + \frac{p(x_{t+k})}{p(x_{t+k}|c_t)} (N-1)\right]
\end{aligned}$$

[van den Oord et al., “Representation Learning with Contrastive Predictive Coding”]

- How does this loss maximize the mutual information?

$$\mathcal{L}_{\mathrm{N}} = -\mathop{\mathbb{E}}_{X}\left[\log \frac{f_k(x_{t+k}, c_t)}{\sum_{x_j \in X} f_k(x_j, c_t)}\right]$$

$$\begin{aligned}
\mathcal{L}_{\mathrm{N}}^{\mathrm{opt}} &= -\mathop{\mathbb{E}}_{X} \log \left[\frac{\frac{p(x_{t+k}|c_t)}{p(x_{t+k})}}{\frac{p(x_{t+k}|c_t)}{p(x_{t+k})} + \sum_{x_j \in X_{\mathrm{neg}}} \frac{p(x_j|c_t)}{p(x_j)}}\right] \\
&= \mathop{\mathbb{E}}_{X} \log \left[1 + \frac{p(x_{t+k})}{p(x_{t+k}|c_t)} \sum_{x_j \in X_{\mathrm{neg}}} \frac{p(x_j|c_t)}{p(x_j)}\right] \\
&\approx \mathop{\mathbb{E}}_{X} \log \left[1 + \frac{p(x_{t+k})}{p(x_{t+k}|c_t)} (N-1) \mathop{\mathbb{E}}_{x_j} \frac{p(x_j|c_t)}{p(x_j)}\right] \\
&= \mathop{\mathbb{E}}_{X} \log \left[1 + \frac{p(x_{t+k})}{p(x_{t+k}|c_t)} (N-1)\right] \\
&\geq \mathop{\mathbb{E}}_{X} \log \left[\frac{p(x_{t+k})}{p(x_{t+k}|c_t)} N\right] \\
&= -I(x_{t+k}, c_t) + \log(N),
\end{aligned}$$

[van den Oord et al., “Representation Learning with Contrastive Predictive Coding”]

- How does this loss maximize the mutual information?

$$\mathcal{L}_{\mathrm{N}} = -\mathbb{E}_{X}\left[\log \frac{f_k(x_{t+k}, c_t)}{\sum_{x_j \in X} f_k(x_j, c_t)}\right]$$

$$I(x_{t+k}, c_t) \geq \log(N) - \mathcal{L}_{\mathrm{N}}$$

[van den Oord et al., “Representation Learning with Contrastive Predictive Coding”]

Key Takeaways: Contrastive learning

- Contrastive learning is a way of doing self-supervised learning
- Positive/negative samples
- Mutual information

$$I(x;c) = \sum_{x,c} p(x,c) \log \frac{p(x,c)}{p(x)p(c)} = \sum_{x,c} p(x,c) \log \frac{p(x|c)}{p(x)}$$

$$= H(x) + H(c) - H(x,c)$$

$$= H(x) + H(x|c)$$

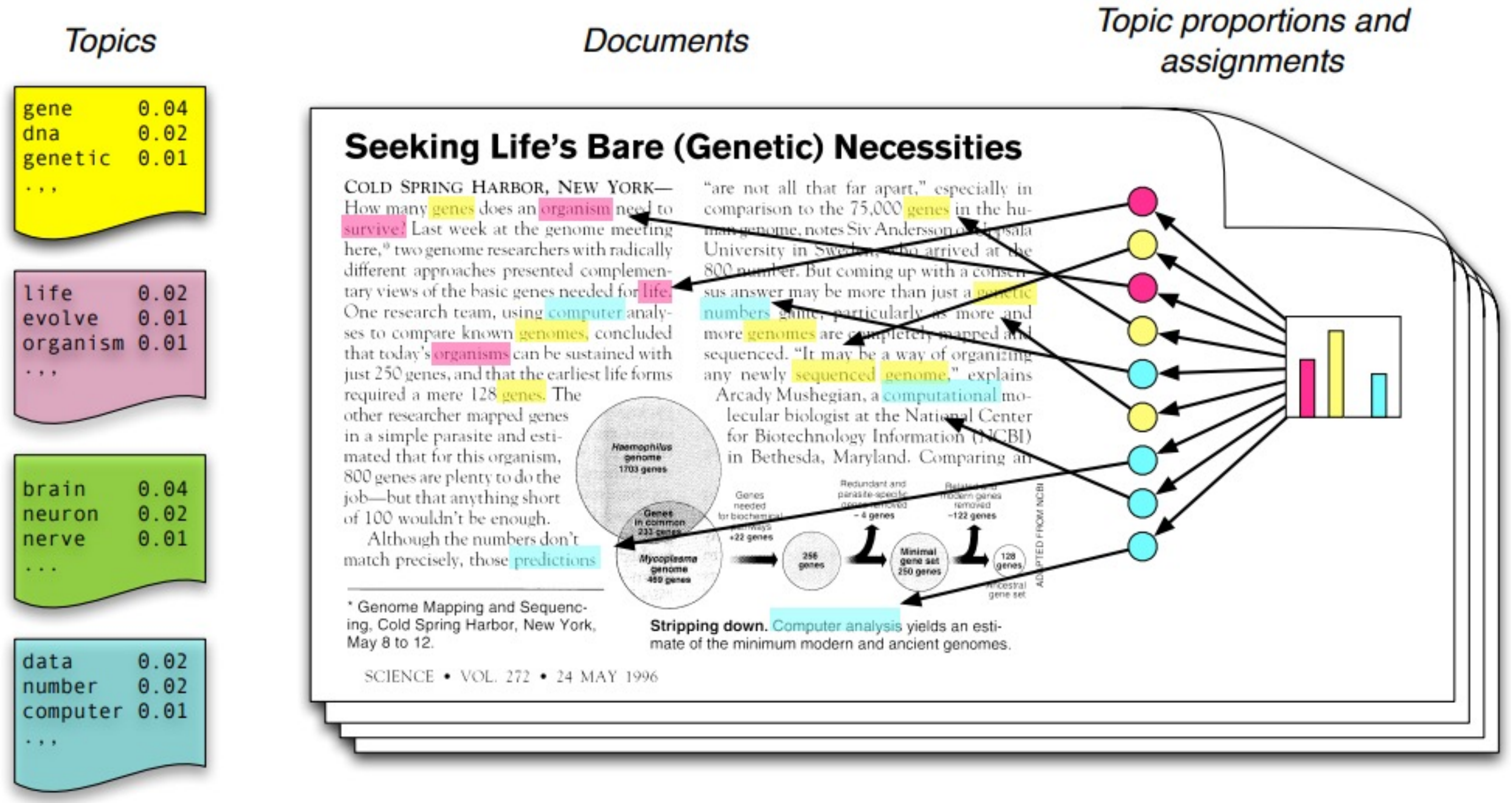
$$= KL(p(x,c) \,||\, p(x)p(c))$$

  - InfoNCE ⇔ MI

Representation Learning with Unsupervised Learning

Unsupervised Learning for Representations

- For text $\boldsymbol{x}$, derive a latent representation $\boldsymbol{z}$
  - with no annotation
- Example 1: Topic models (e.g., Latent Dirichlet Analysis, LDA)


Topics
Documents
Topic proportions and assignments
gene 0.04
dna 0.02
genetic 0.01
...
life 0.02
evolve 0.01
organism 0.01
...
brain 0.04
neuron 0.02
nerve 0.01
...
data 0.02
number 0.02
computer 0.01
...
Haemophilus genome 1703 genes
Genes in common 233 genes
Mycoplasma genome 469 genes
Genes needed for biochemical pathways +22 genes
256 genes
Redundant and parasite-specific genes removed −4 genes
Minimal gene set 250 genes
Related and modern genes removed −122 genes
128 genes
Ancestral gene set
ADAPTED FROM NCBI

Seeking Life's Bare (Genetic) Necessities

COLD SPRING HARBOR, NEW YORK—How many genes does an organism need to survive? Last week at the genome meeting here,* two genome researchers with radically different approaches presented complementary views of the basic genes needed for life. One research team, using computer analyses to compare known genomes, concluded that today's organisms can be sustained with just 250 genes, and that the earliest life forms required a mere 128 genes. The other researcher mapped genes in a simple parasite and estimated that for this organism, 800 genes are plenty to do the job—but that anything short of 100 wouldn't be enough.

Although the numbers don't match precisely, those predictions "are not all that far apart," especially in comparison to the 75,000 genes in the human genome, notes Siv Andersson of Uppsala University in Sweden, who arrived at the 800 number. But coming up with a consensus answer may be more than just a genetic numbers game, particularly as more and more genomes are completely mapped and sequenced. "It may be a way of organizing any newly sequenced genome," explains Arcady Mushegian, a computational molecular biologist at the National Center for Biotechnology Information (NCBI) in Bethesda, Maryland. Comparing an

* Genome Mapping and Sequencing, Cold Spring Harbor, New York, May 8 to 12.

Stripping down. Computer analysis yields an estimate of the minimum modern and ancient genomes.

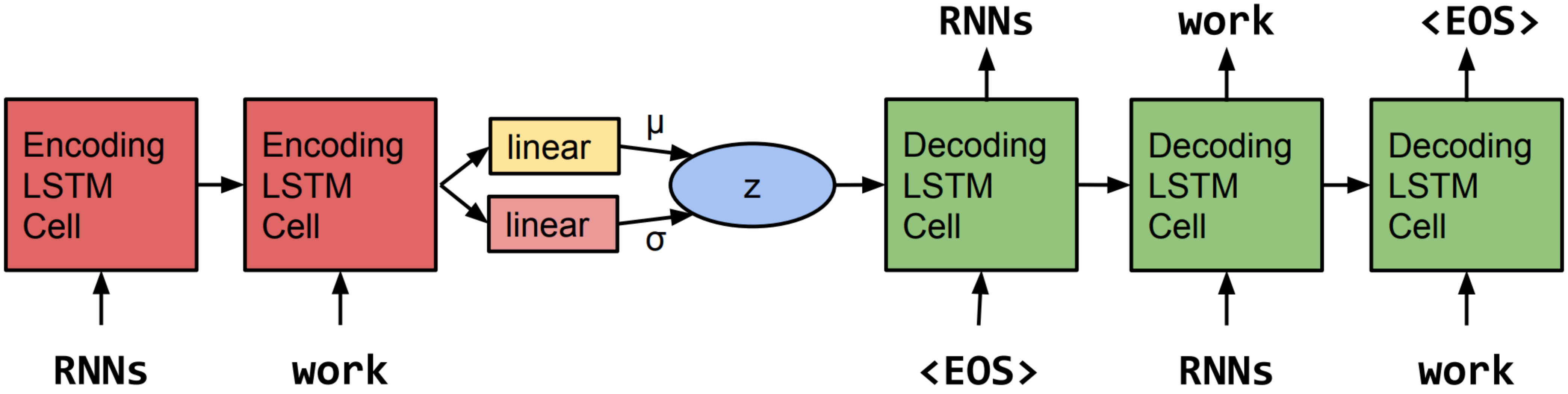
SCIENCE • VOL. 272 • 24 MAY 1996

- Each **document** is a mixture of corpus-wide **topics**
- Each **topic** is a distribution over **words**

[Blei et al., 2003, Latent Dirichlet Allocation]

Unsupervised Learning for Representations

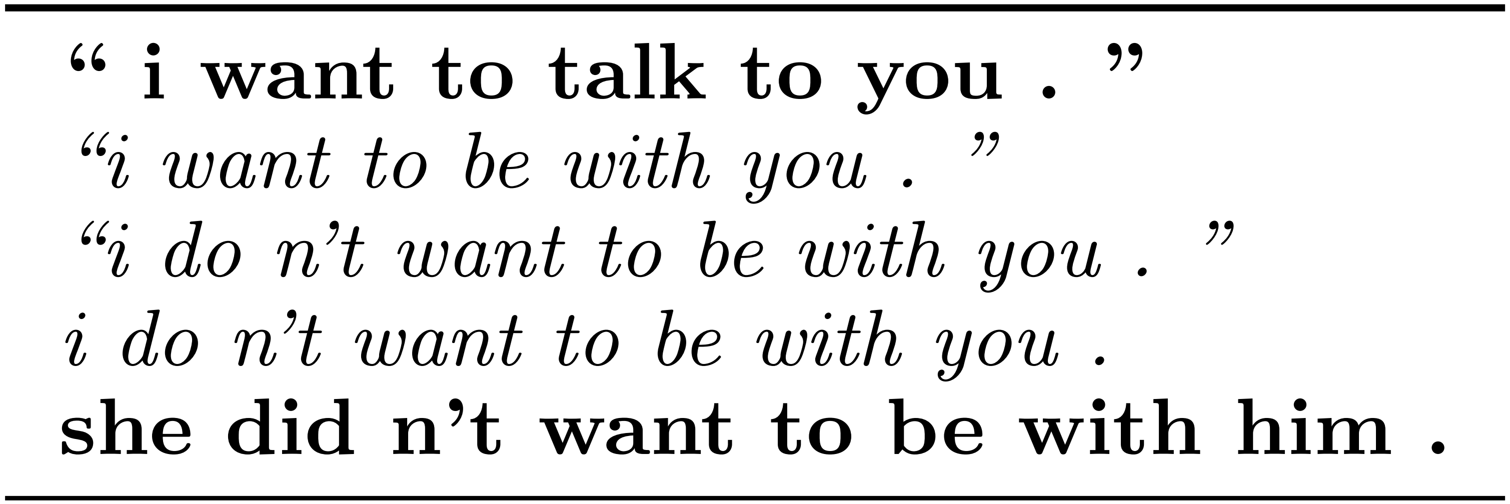
- For text $\boldsymbol{x}$, derive a latent representation $\boldsymbol{z}$
  - with no annotation
- Example 2: Variational Autoencoders (VAEs)



[Bowman et al., 2016 Generating Sentences from a Continuous Space]

Unsupervised Learning for Representations

- For text $\boldsymbol{x}$, derive a latent representation $\boldsymbol{z}$
  - with no annotation
- Example 2: Variational Autoencoders (VAEs)



“ 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 .

text interpolation with VAEs

[Bowman et al., 2016 Generating Sentences from a Continuous Space]

Unsupervised Learning

- Each instance has two parts:
  - observed variables $\boldsymbol{x}$
  - latent (unobserved) variables $\boldsymbol{z}$
  - A.k.a., "incomplete" data
- Want to learn a model $p_\theta(\boldsymbol{x}, \boldsymbol{z})$

[Content adapted from CMU 10-708]

Latent (unobserved) variables

- A variable can be unobserved (latent) because:
  - imaginary quantity: meant to provide some simplified and abstractive view of the data generation process
    - e.g., speech recognition models, mixture models, …

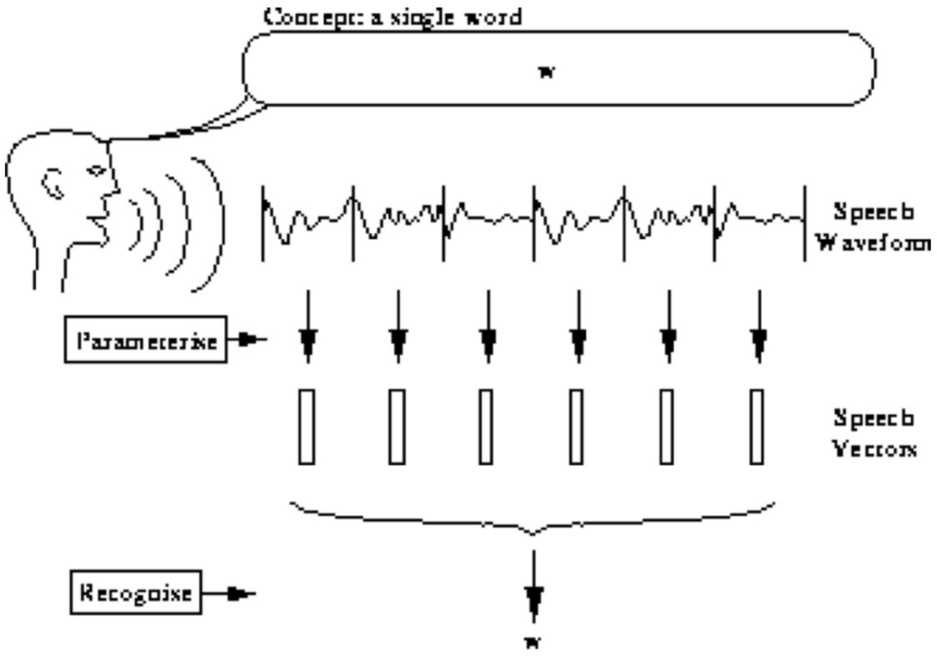
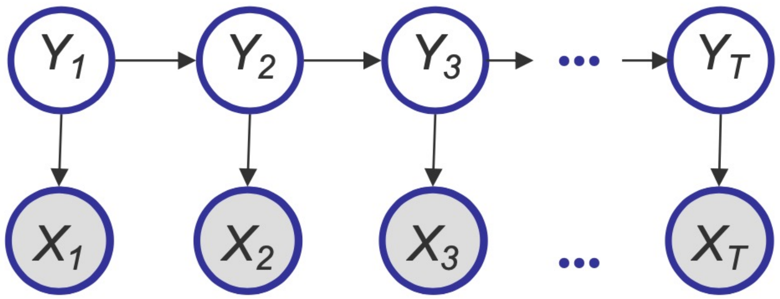
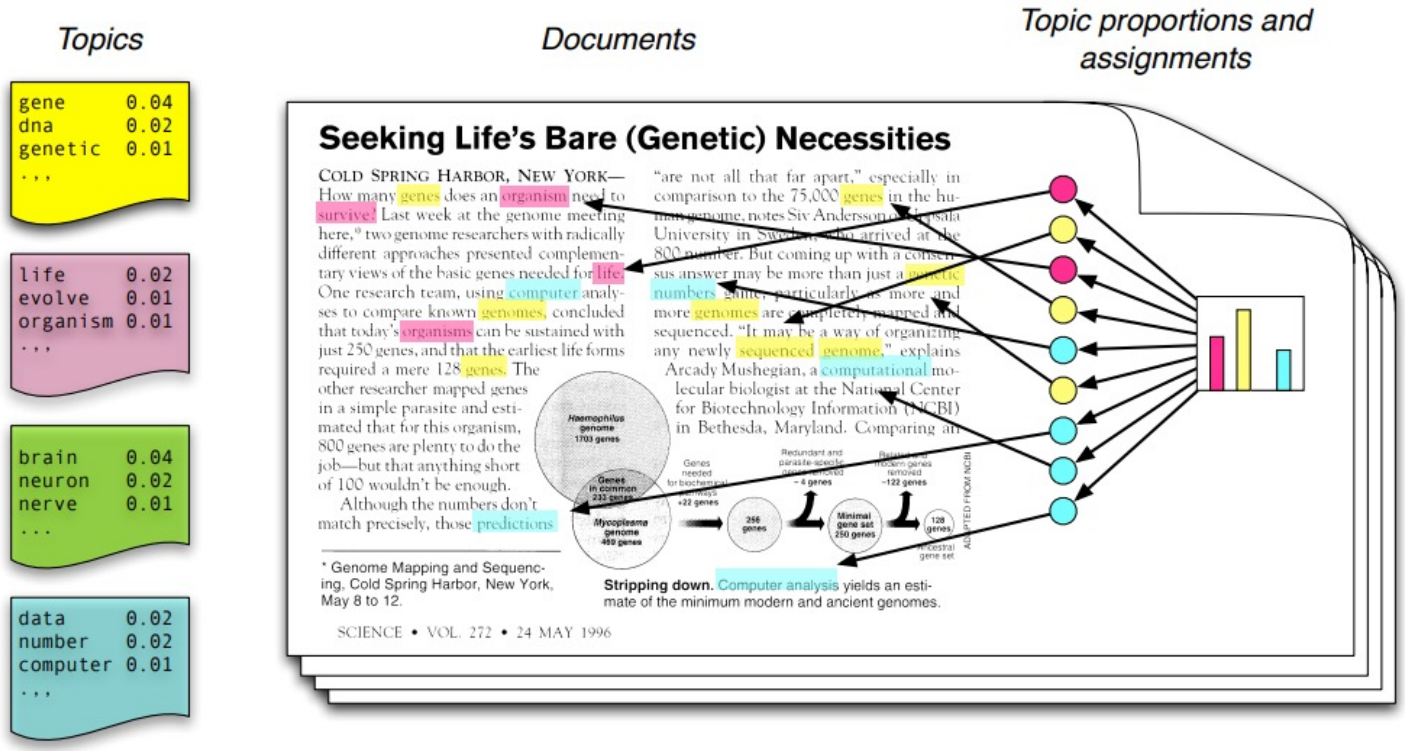


Fig. 1.2 Isolated Word Problem



Latent (unobserved) variables

- A variable can be unobserved (latent) because:
    - imaginary quantity: meant to provide some simplified and abstractive view of the data generation process
        - e.g., speech recognition models, mixture models, …


Topics
Documents
Topic proportions and assignments
gene 0.04
dna 0.02
genetic 0.01
...
life 0.02
evolve 0.01
organism 0.01
...
brain 0.04
neuron 0.02
nerve 0.01
...
data 0.02
number 0.02
computer 0.01
...
Haemophilus genome 1703 genes
Genes in common 233 genes
Mycoplasma genome 469 genes
Genes needed for biochemical pathways +22 genes
256 genes
Redundant and parasite-specific genes removed −4 genes
Minimal gene set 250 genes
Related and modern genes removed −122 genes
128 genes
Ancestral gene set
ADAPTED FROM NCBI

Seeking Life's Bare (Genetic) Necessities

COLD SPRING HARBOR, NEW YORK— How many genes does an organism need to survive? Last week at the genome meeting here,* two genome researchers with radically different approaches presented complementary views of the basic genes needed for life. One research team, using computer analyses to compare known genomes, concluded that today's organisms can be sustained with just 250 genes, and that the earliest life forms required a mere 128 genes. The other researcher mapped genes in a simple parasite and estimated that for this organism, 800 genes are plenty to do the job—but that anything short of 100 wouldn't be enough.

Although the numbers don't match precisely, those predictions "are not all that far apart," especially in comparison to the 75,000 genes in the human genome, notes Siv Andersson of Uppsala University in Sweden, who arrived at the 800 number. But coming up with a consensus answer may be more than just a genetic numbers game, particularly as more and more genomes are completely mapped and sequenced. "It may be a way of organizing any newly sequenced genome," explains Arcady Mushegian, a computational molecular biologist at the National Center for Biotechnology Information (NCBI) in Bethesda, Maryland. Comparing an

Stripping down. Computer analysis yields an estimate of the minimum modern and ancient genomes.

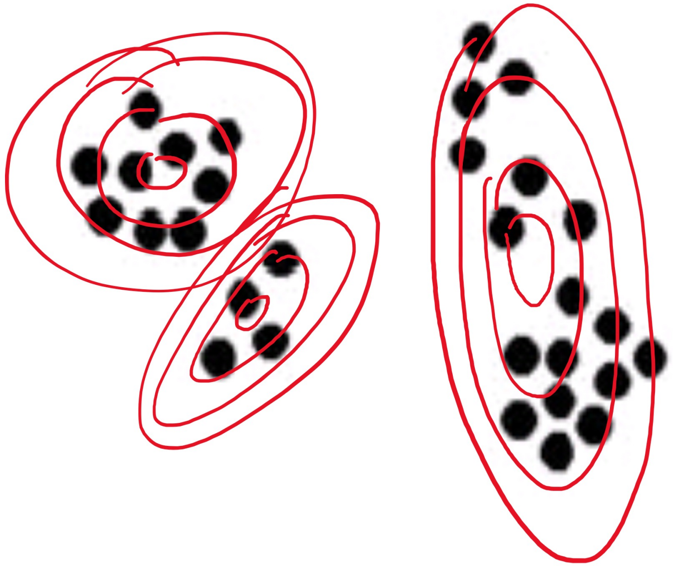
* Genome Mapping and Sequencing, Cold Spring Harbor, New York, May 8 to 12.

SCIENCE • VOL. 272 • 24 MAY 1996

- Each **document** is a mixture of corpus-wide **topics**
- Each **topic** is a distribution over **words**

Latent (unobserved) variables

- A variable can be unobserved (latent) because:
  - imaginary quantity: meant to provide some simplified and abstractive view of the data generation process
    - e.g., speech recognition models, mixture models, ...



Clustering

Latent (unobserved) variables

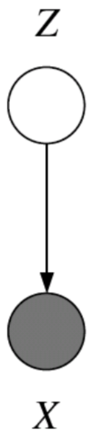
- A variable can be unobserved (latent) because:
  - imaginary quantity: meant to provide some simplified and abstractive view of the data generation process
    - e.g., speech recognition models, mixture models, …
  - a real-world object (and/or phenomena), but difficult or impossible to measure
    - e.g., the temperature of a star, causes of a disease, evolutionary ancestors …
  - a real-world object (and/or phenomena), but sometimes wasn't measured, because of faulty sensors, etc.
- Discrete latent variables can be used to partition/cluster data into sub-groups
- Continuous latent variables (factors) can be used for dimensionality reduction (e.g., factor analysis, etc.)

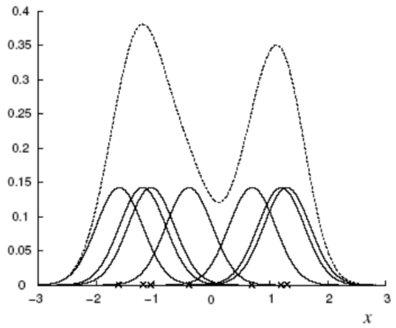
Example: Gaussian Mixture Models (GMMs)

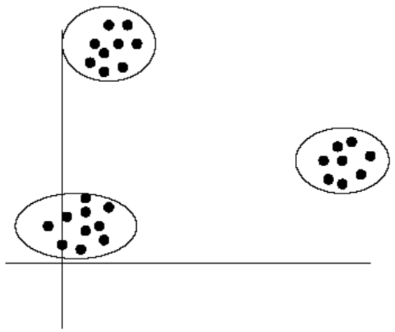
- Consider a mixture of K Gaussian components:

$$p(x_n|\mu,\Sigma)=\sum_k \pi_k N(x,|\,\mu_k,\Sigma_k)$$

mixture proportion   mixture component

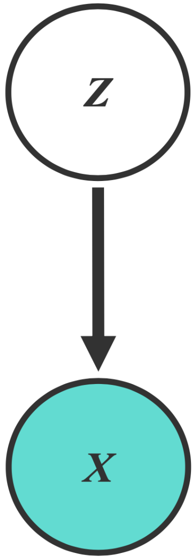






- This model can be used for unsupervised clustering.
    - This model has been used to discover new kinds of stars in astronomical data, etc.

Example: Gaussian Mixture Models (GMMs)



- Consider a mixture of K Gaussian components:
  - ❑ Z is a latent class indicator vector:

$$p(z_n) = \mathrm{multi}(z_n : \pi) = \prod_k (\pi_k)^{z_n^k}$$

  - ❑ X is a conditional Gaussian variable with a class-specific mean/covariance

$$p(x_n \mid z_n^k = 1, \mu, \Sigma) = \frac{1}{(2\pi)^{m/2} |\Sigma_k|^{1/2}} \exp\left\{ -\tfrac{1}{2} (x_n - \mu_k)^T \Sigma_k^{-1} (x_n - \mu_k) \right\}$$

  - ❑ The likelihood of a sample:

Parameters to be learned:

mixture component

mixture proportion

$$\begin{aligned} p(x_n|\mu,\Sigma) &= \sum\nolimits_k p(z^k = 1 \mid \pi) p(x,\mid z^k = 1, \mu, \Sigma) \\ &= \sum\nolimits_{z_n} \prod\nolimits_k \left((\pi_k)^{z_n^k} N(x_n : \mu_k, \Sigma_k)^{z_n^k} \right) = \sum\nolimits_k \pi_k N(x,\mid \mu_k, \Sigma_k) \end{aligned}$$

Example: Gaussian Mixture Models (GMMs)

- Consider a mixture of K Gaussian components: $p(x_n|\mu,\Sigma)=\sum_k \pi_k N(x,|\mu_k,\Sigma_k)$
- Recall MLE for completely observed data
  - Data log-likelihood: $$\ell(\boldsymbol{\theta};D)=\log\prod_n p(z_n,x_n)=\log\prod_n p(z_n\,|\,\pi)p(x_n\,|\,z_n,\mu,\sigma)$$ $$=\sum_n\log\prod_k \pi_k^{z_n^k}+\sum_n\log\prod_k N(x_n;\mu_k,\sigma)^{z_n^k}$$ $$=\sum_n\sum_k z_n^k\log\pi_k-\sum_n\sum_k z_n^k\frac{1}{2\sigma^2}(x_n-\mu_k)^2+C$$
  - MLE:

$$\hat{\pi}_{k,MLE}=\arg\max_\pi \ell(\boldsymbol{\theta};D),$$

$$\hat{\mu}_{k,MLE}=\arg\max_\mu \ell(\boldsymbol{\theta};D) \qquad \Rightarrow \quad \hat{\mu}_{k,MLE}=\frac{\sum_n z_n^k x_n}{\sum_n z_n^k}$$

$$\hat{\sigma}_{k,MLE}=\arg\max_\sigma \ell(\boldsymbol{\theta};D)$$

- What if we do not know z_n?

Why is Learning Harder?

- Complete log likelihood: if both $\boldsymbol{x}$ and $\boldsymbol{z}$ can be observed, then

$$\ell_c(\theta; \boldsymbol{x}, \boldsymbol{z}) = \log p(\boldsymbol{x}, \boldsymbol{z}|\theta) = \log p(\boldsymbol{z}|\theta_z) + \log p(\boldsymbol{x}|\boldsymbol{z}, \theta_x)$$

  - Decomposes into a sum of factors, the parameter for each factor can be estimated separately
- But given that $\boldsymbol{z}$ is not observed, $\ell_c(\theta; \boldsymbol{x}, \boldsymbol{z})$ is a random quantity, cannot be maximized directly
- Incomplete (or marginal) log likelihood: with $\boldsymbol{z}$ unobserved, our objective becomes the log of a marginal probability:

$$\ell(\theta; \boldsymbol{x}) = \log p(\boldsymbol{x}|\theta) = \log \sum_z p(\boldsymbol{x}, \boldsymbol{z}|\theta)$$

  - All parameters become coupled together
  - In other models when $\boldsymbol{z}$ is complex (continuous) variables (as we'll see later), marginalization over z is intractable.

Expectation Maximization (EM)

- For any distribution $q(\boldsymbol{z}|\boldsymbol{x})$, define expected complete log likelihood:

$$\mathbb{E}_q[\ell_c(\theta; \boldsymbol{x}, \boldsymbol{z})] = \sum_{z} q(\boldsymbol{z}|\boldsymbol{x}) \log p(\boldsymbol{x}, \boldsymbol{z}|\theta)$$

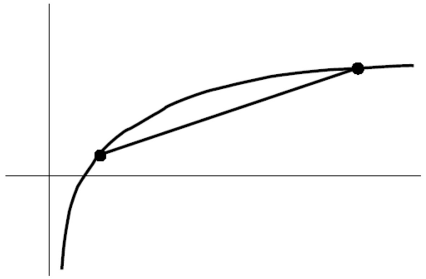
  - A deterministic function of θ
  - Inherit the factorizability of $\ell_c(\theta; \boldsymbol{x}, \boldsymbol{z})$
- Use this as the surrogate objective
- Does maximizing this surrogate yield a maximizer of the likelihood?

Expectation Maximization (EM)

- For any distribution $q(\boldsymbol{z}|\boldsymbol{x})$, define expected complete log likelihood:

$$\mathbb{E}_q[\ell_c(\theta;\boldsymbol{x},\boldsymbol{z})] = \sum_z q(\boldsymbol{z}|\boldsymbol{x}) \log p(\boldsymbol{x},\boldsymbol{z}|\theta)$$

- Jensen's inequality



$$\begin{aligned}
\ell(\theta;x) &= \log p(x \mid \theta) \\
&= \log \sum_z p(x,z \mid \theta) \\
&= \log \sum_z q(z \mid x) \frac{p(x,z \mid \theta)}{q(z \mid x)} \\
&\boxed{\geq} \sum_z q(z \mid x) \log \frac{p(x,z \mid \theta)}{q(z \mid x)} \qquad \text{Evidence Lower Bound (ELBO)} \\
&= \sum_z q(z \mid x) \log p(x,z \mid \theta) - \sum_z q(z \mid x) \log q(z \mid x) \\
&= \mathbb{E}_q[\ell_c(\theta;\boldsymbol{x},\boldsymbol{z})] + H(q)
\end{aligned}$$

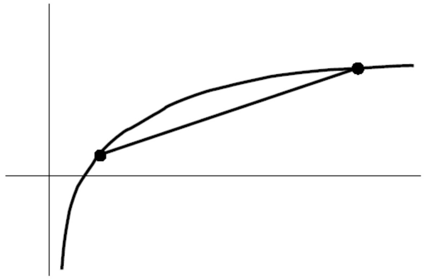
Expectation Maximization (EM)

- For any distribution $q(\boldsymbol{z}|\boldsymbol{x})$, define expected complete log likelihood:

$$\mathbb{E}_q[\ell_c(\theta;\boldsymbol{x},\boldsymbol{z})] = \sum_z q(\boldsymbol{z}|\boldsymbol{x}) \log p(\boldsymbol{x},\boldsymbol{z}|\theta)$$

- Jensen's inequality

$$\begin{aligned}\ell(\theta;x) &= \log p(x \mid \theta) \\ &= \log \sum_z p(x,z \mid \theta) \\ &= \log \sum_z q(z \mid x) \frac{p(x,z \mid \theta)}{q(z \mid x)} \\ &\geq \sum_z q(z \mid x) \log \frac{p(x,z \mid \theta)}{q(z \mid x)}\end{aligned}$$



- Indeed we have

$$\ell(\theta;\boldsymbol{x}) = \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p(\boldsymbol{x},\boldsymbol{z}|\theta)}{q(\boldsymbol{z}|\boldsymbol{x})}\right] + \mathrm{KL}\big(q(\boldsymbol{z}|\boldsymbol{x}) \,||\, p(\boldsymbol{z}|\boldsymbol{x},\theta)\big)$$

Lower Bound and Free Energy

- For fixed data $\boldsymbol{x}$, define a functional called the (variational) free energy:

$$F(q,\theta) = -\mathbb{E}_q[\ell_c(\theta;\boldsymbol{x},\boldsymbol{z})] - H(q) \geq \ell(\theta;\boldsymbol{x})$$

- The EM algorithm is coordinate-decent on F
  - At each step t:
    - E-step: $q^{t+1} = \arg\min_q F\left(q, \theta^t\right)$
    - M-step: $\theta^{t+1} = \arg\min_\theta F\left(q^{t+1}, \theta^t\right)$

E-step: minimization of $F(q,\theta)$ w.r.t q

- Claim:

$$q^{t+1} = \text{argmin}_q\ F(q,\theta^t) = p(\boldsymbol{z}|\boldsymbol{x},\theta^t)$$

  - This is the posterior distribution over the latent variables given the data and the current parameters.

- Proof (easy): recall

$$\underbrace{\ell(\theta^t;\boldsymbol{x})}_{\text{Independent of } q} = \underbrace{\mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x})}\left[\log\frac{p(\boldsymbol{x},\boldsymbol{z}|\theta^t)}{q(\boldsymbol{z}|\boldsymbol{x})}\right]}_{-F(q,\theta^t)} + \underbrace{\mathrm{KL}\big(q(\boldsymbol{z}|\boldsymbol{x})\ ||\ p(\boldsymbol{z}|\boldsymbol{x},\theta^t)\big)}_{\geq 0}$$

  - $F(q,\theta^t)$ is minimized when $\mathrm{KL}\big(q(\boldsymbol{z}|\boldsymbol{x})\ ||\ p(\boldsymbol{z}|\boldsymbol{x},\theta^t)\big) = 0$, which is achieved only when $q(\boldsymbol{z}|\boldsymbol{x}) = p(\boldsymbol{z}|\boldsymbol{x},\theta^t)$

M-step: minimization of $F(q,\theta)$ w.r.t $\boldsymbol{\theta}$

- Note that the free energy breaks into two terms:

$$F(q,\theta) = -\mathbb{E}_q[\ell_c(\theta;\boldsymbol{x},\boldsymbol{z})] - H(q) \geq \ell(\theta;\boldsymbol{x})$$

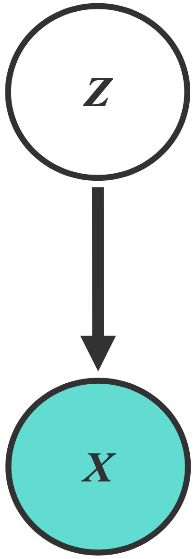
  - The first term is the expected complete log likelihood and the second term, which does not depend on q, is the entropy.

- Thus, in the M-step, maximizing with respect to θ for fixed q we only need to consider the first term:

$$\theta^{t+1} = \mathrm{argmax}_\theta\, \mathbb{E}_q[\ell_c(\theta;\boldsymbol{x},\boldsymbol{z})] = \mathrm{argmax}_\theta \sum_z q^{t+1}(\boldsymbol{z}|\boldsymbol{x}) \log p(\boldsymbol{x},\boldsymbol{z}|\theta)$$

  - Under optimal q^{t+1}, this is equivalent to solving a standard MLE of fully observed model $p(\boldsymbol{x},\boldsymbol{z}|\theta)$, with z replaced by its expectation w.r.t $p(\boldsymbol{z}|\boldsymbol{x},\theta^t)$

Example: Gaussian Mixture Models (GMMs)



- Consider a mixture of K Gaussian components:
    - ❑ Z is a latent class indicator vector:

$$p(z_n) = \text{multi}(z_n : \pi) = \prod_k (\pi_k)^{z_n^k}$$

    - ❑ X is a conditional Gaussian variable with a class-specific mean/covariance

$$p(x_n \mid z_n^k = 1, \mu, \Sigma) = \frac{1}{(2\pi)^{m/2}|\Sigma_k|^{1/2}} \exp\left\{-\tfrac{1}{2}(x_n - \mu_k)^T \Sigma_k^{-1}(x_n - \mu_k)\right\}$$

    - ❑ The likelihood of a sample:

mixture component

mixture proportion

$$\begin{aligned} p(x_n|\mu,\Sigma) &= \sum_k p(z^k = 1 \mid \pi)\, p(x, \mid z^k = 1, \mu, \Sigma) \\ &= \sum_{z_n} \prod_k \left((\pi_k)^{z_n^k} N(x_n : \mu_k, \Sigma_k)^{z_n^k}\right) = \sum_k \pi_k N(x, \mid \mu_k, \Sigma_k) \end{aligned}$$

Example: Gaussian Mixture Models (GMMs)

- Consider a mixture of K Gaussian components
- The expected complete log likelihood

$$\mathbb{E}_q\left[\ell_c(\boldsymbol{\theta}; x, z)\right] = \sum_n \mathbb{E}_q\left[\log p\left(z_n \mid \pi\right)\right] + \sum_n \mathbb{E}_q\left[\log p\left(x_n \mid z_n, \mu, \Sigma\right)\right]$$

$$= \sum_n \sum_k \mathbb{E}_q\left[z_n^k\right] \log \pi_k - \frac{1}{2} \sum_n \sum_k \mathbb{E}_q\left[z_n^k\right] \left(\left(x_n - \mu_k\right)^T \Sigma_k^{-1} \left(x_n - \mu_k\right) + \log\left|\Sigma_k\right| + C\right)$$

- E-step: computing the posterior of z_n given the current estimate of the parameters (i.e., π , μ, Σ)

↗ $p(z_n^k = 1, x, \mu^{(t)}, \Sigma^{(t)})$

$$p(z_n^k = 1 \mid x, \mu^{(t)}, \Sigma^{(t)}) = \frac{\pi_k^{(t)} N(x_n, \mid \mu_k^{(t)}, \Sigma_k^{(t)})}{\sum_i \pi_i^{(t)} N(x_n, \mid \mu_i^{(t)}, \Sigma_i^{(t)})}$$

↘ $p(x, \mu^{(t)}, \Sigma^{(t)})$

Example: Gaussian Mixture Models (GMMs)

- M-step: computing the parameters given the current estimate of z_n

$$\pi_k^* = \arg\max\langle l_c(\boldsymbol{\theta})\rangle, \qquad \Rightarrow \frac{\partial}{\partial \pi_k}\langle l_c(\boldsymbol{\theta})\rangle = 0, \forall k, \quad \text{s.t.} \sum_{\text{k}} \pi_k = 1$$

$$\Rightarrow \quad \pi_k^* = \sum\nolimits_n \langle z_n^k \rangle_{q^{(t)}} \Big/ N = \sum\nolimits_n \tau_n^{k(t)} \Big/ N = \langle n_k \rangle \Big/ N$$

$$\mu_k^* = \arg\max\langle l(\boldsymbol{\theta})\rangle, \qquad \Rightarrow \quad \mu_k^{(t+1)} = \frac{\sum_n \tau_n^{k(t)} x_n}{\sum_n \tau_n^{k(t)}}$$

$$\Sigma_k^* = \arg\max\langle l(\boldsymbol{\theta})\rangle, \qquad \Rightarrow \quad \Sigma_k^{(t+1)} = \frac{\sum_n \tau_n^{k(t)} (x_n - \mu_k^{(t+1)})(x_n - \mu_k^{(t+1)})^T}{\sum_n \tau_n^{k(t)}}$$

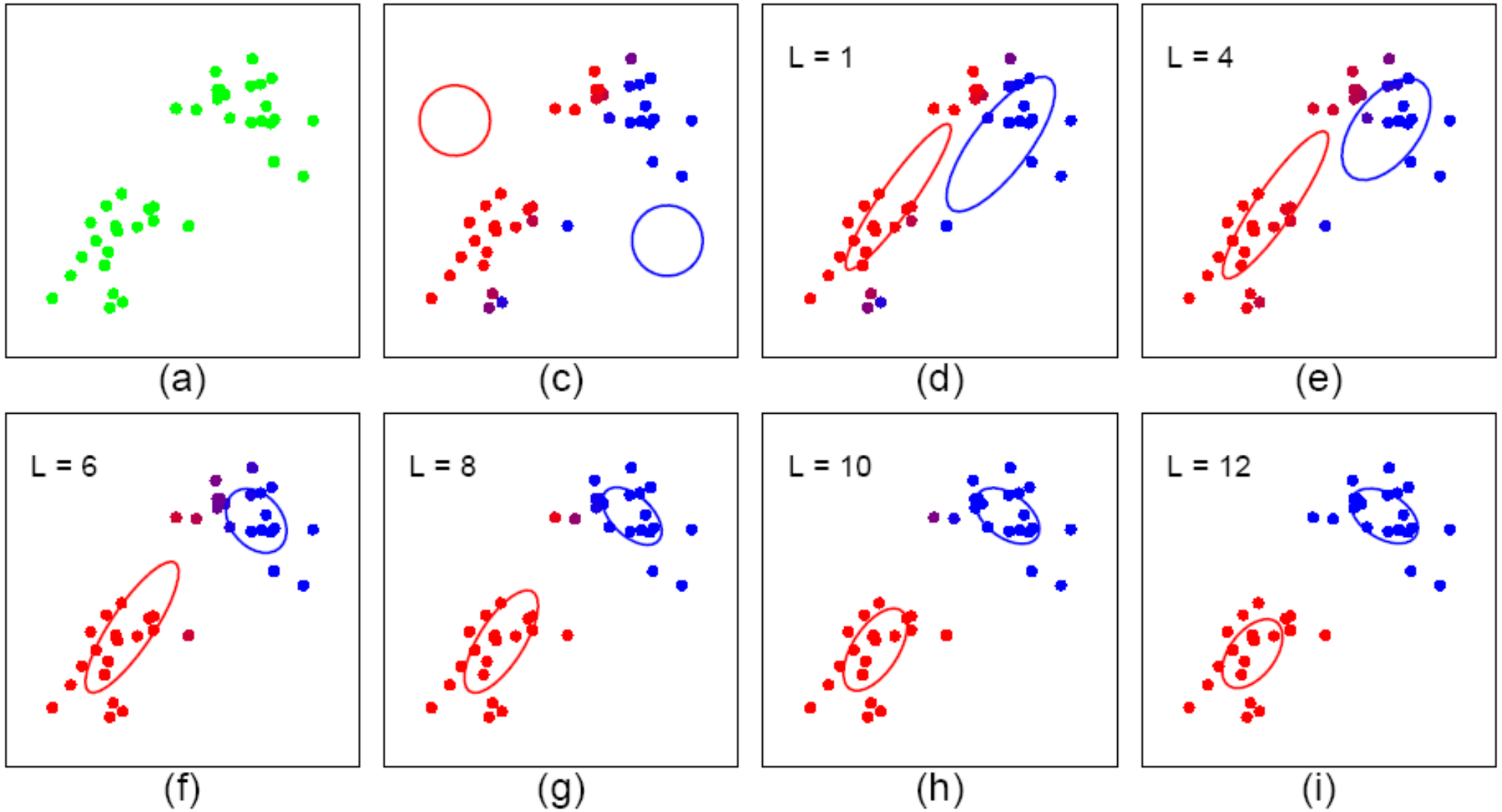
Fact:

$$\frac{\partial \log |\mathrm{A}^{-1}|}{\partial \mathrm{A}^{-1}} = \mathrm{A}^T$$

$$\frac{\partial \mathbf{x}^T \mathrm{A}\mathbf{x}}{\partial \mathrm{A}} = \mathbf{x}\mathbf{x}^T$$

Example: Gaussian Mixture Models (GMMs)

- Start: "guess" the centroid μ_k and covariance Σ_k of each of the K clusters
- Loop:



(a) (c) (d) (e)

(f) (g) (h) (i)

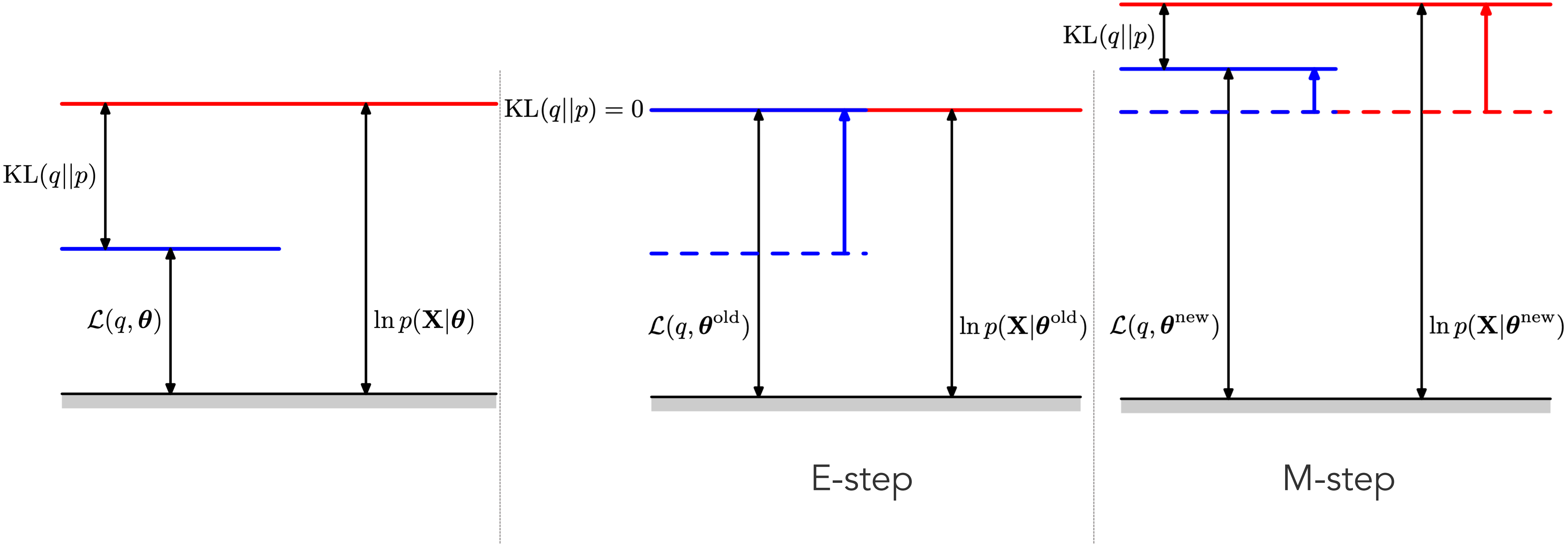
Summary: EM Algorithm

- A way of maximizing likelihood function for latent variable models. Finds MLE of parameters when the original (hard) problem can be broken up into two (easy) pieces
  - Estimate some "missing" or "unobserved" data from observed data and current parameters.
  - Using this "complete" data, find the maximum likelihood parameter estimates.

- Alternate between filling in the latent variables using the best guess (posterior) and updating the parameters based on this guess:
  - E-step: $q^{t+1} = \arg\min_{q} F\left(q, \theta^{t}\right)$
  - M-step: $\theta^{t+1} = \arg\min_{\theta} F\left(q^{t+1}, \theta^{t}\right)$

Each EM iteration guarantees to improve the likelihood

$$\ell(\theta;\boldsymbol{x}) = \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x})}\left[\log\frac{p(\boldsymbol{x},\boldsymbol{z}|\theta)}{q(\boldsymbol{z}|\boldsymbol{x})}\right] + \mathrm{KL}\big(q(\boldsymbol{z}|\boldsymbol{x})\,||\,p(\boldsymbol{z}|\boldsymbol{x},\theta)\big)$$


$\mathrm{KL}(q||p)$
$\mathcal{L}(q,\boldsymbol{\theta})$
$\ln p(\mathbf{X}|\boldsymbol{\theta})$
$\mathrm{KL}(q||p) = 0$
$\mathcal{L}(q,\boldsymbol{\theta}^{\mathrm{old}})$
$\ln p(\mathbf{X}|\boldsymbol{\theta}^{\mathrm{old}})$
$\mathrm{KL}(q||p)$
$\mathcal{L}(q,\boldsymbol{\theta}^{\mathrm{new}})$
$\ln p(\mathbf{X}|\boldsymbol{\theta}^{\mathrm{new}})$

E-step

M-step

[PRML, Chap 9.4]

EM Variants

- Sparse EM
  - Do not re-compute exactly the posterior probability on each data point under all models, because it is almost zero.
  - Instead keep an “active list” which you update every once in a while.

- Generalized (Incomplete) EM:
  - It might be hard to find the ML parameters in the M-step, even given the completed data. We can still make progress by doing an M-step that improves the likelihood a bit (e.g. gradient step).

Key Takeaways

- Unsupervised learning
  - Maximum likelihood estimation (MLE) with latent variables
  - EM algorithm for MLE
    - Expected complete log likelihood
    - Evidence lower bound (ELBO)
    - Coordinate ascent: E-step, M-step

Questions?