Probabilistic PCA and Variational autoencoders

Principal component analysis

Preliminary: Matrix factorization view

- Matrix factorization: $X \approx WZ + b1$
  - Data matrix: $X = [x_1; x_2; \ldots; x_N] \in \mathbb{R}^{d \times N}$
  - Weight matrix: $W = [w_1; w_2; \ldots; w_d] \in \mathbb{R}^{d \times d'}$
  - Code matrix: $Z = [z_1; z_2; \ldots; z_N] \in \mathbb{R}^{d' \times N}$
  - Bias vector: $b = \mathbb{E}[x] \in \mathbb{R}^d$
  - $1 = [1,\ldots,1]$

Principal component analysis

Preliminary: Matrix factorization view

- Principal componen analysis: $X \approx WZ + b1$
  - The code vectors have a zero mean and a diagonal covariance.
    - $\mathbb{E}[z] = 0$ and $\text{Cov}[z] = \text{diag}(\sigma^2) \in \mathbb{R}^{d' \times d'}$
  - Eigendecomposition of the (scaled) covariance of X
    - $\text{Cov}(X) = (X - b1)(X - b1)^\top \approx (WZ)(WZ)^\top = W(ZZ^\top)W^\top = W\text{Cov}(z)W^\top$
- It is tricky to extend principal component analysis under this perspective.

Principal component analysis

Probabilistic view

- Instead of the data matrix, consider each data vector separately
  - $X \approx WZ + b1 \iff x_n \approx Wz_n + b, \forall n = 1, \ldots, N$
- x_n is a (approximate) linear transformation of z_n by W and b.
- Instead of approximation, can we say it's noisy?
  - $x = Wz + b + \epsilon$
- Any observation x is noisy linear transformation of z.

Principal component analysis

Probabilistic view

- Any observation x is noisy linear transformation of z: $x = Wz + b + \epsilon$
- Where does z come from?
  - Recall: *the code vectors have a zero mean and a diagonal covariance*.
  - Assume z is a sample from a standard Normal distribution: $z \sim \mathcal{N}(0, 1^{d'})$
- Where does ϵ come from?
  - Zero-mean, scaled identity covariance: $\epsilon \sim \mathcal{N}(0, \sigma^2)$

Principal component analysis

Probabilistic view

- A probabilistic graphical model of principal component analysis
  - Unobserved: $z \sim \mathcal{N}(0, 1^{d'})$
  - Observed: $x \mid z \sim \mathcal{N}(Wz + b, \sigma^2 I)$
  - Parameters: W and b
- Joint probability $p(x, z) = p(x \mid z)p(z)$
- Marginal probability $p(x) = \int_{\mathbb{R}^{d'}} p(x \mid z)p(z)\mathrm{d}z$
- Posterior probability $p(z \mid x) = p(x \mid z)p(z) \Big/ p(x)$

Principal component analysis

Recap: Normal distributions

- Marginals and conditionals of Gaussians are Gaussians.
- Any Gaussian distribution is fully characterized by its *mean* and (co)*variance*.

Principal component analysis

Probabilistic view

- A probabilistic graphical model of principal component analysis
  - Unobserved: $z \sim \mathcal{N}(0, 1^{d'})$
  - Observed: $x \mid z \sim \mathcal{N}(Wz + b, \sigma^2 I)$
- Marginal distribution: derive them from $x = Wz + b + \epsilon$
  - $\mathbb{E}[x] = W\mathbb{E}[z] + b + \mathbb{E}[\epsilon] = b$
  - $\text{Cov}[x] = \mathbb{E}[(x - \mathbb{E}[x])(x - \mathbb{E}[x])^\top] = \mathbb{E}[(Wz + \epsilon)(Wz + \epsilon)^\top] = WW^\top + \sigma^2 I$

Principal component analysis

Probabilistic view: in the limit of no noise, it's all the same thing..

- When $\sigma^2 = 0$,
  - $\mathbb{E}[x] = W\mathbb{E}[z] + b + \mathbb{E}[\epsilon] = b$
  - $\text{Cov}[x] = WW^\top + \sigma^2 I = WW^\top$
- W can be found by the eigendecomposition of the covariance of x.

Principal component analysis

Probabilistic view: learning by maximum log-likelihood

- Log-likelihood:
  - $\sum_{n=1}^{N} \log p(x^n) = -\frac{1}{2}\sum_{n=1}^{N}(x^n - b)^\top (WW^\top + \sigma^2 I)^{-1}(x^n - b) - \frac{1}{2}\log\left|WW^\top + \sigma^2 I\right| - d' \log 2\pi$
- Maximize the log-likelihood w.r.t. W and b using gradient descent
  - The optimal $b = \frac{1}{N}\sum_{n=1}^{N} x^n$
  - The derivative of the log-determinant: $\frac{\partial \log\left|WW^\top + \sigma^2 I\right|}{\partial W} = 2\mathrm{Tr}\left[(WW^\top + \sigma^2 I)^{-1} W\right]$
- You can also solve it exactly and get an analytical solution: left for your own exercise!

Principal component analysis

Probabilistic view

- A probabilistic graphical model of principal component analysis
  - Unobserved: $z \sim \mathcal{N}(0, 1^{d'})$
  - Observed: $x \mid z \sim \mathcal{N}(Wz + b, \sigma^2 I)$
- Joint distribution
  - Simply stack x and z and for a Gaussian distribution
    - Because their mean and covariance do *not* depend on each other.
  - $\mathbb{E}[[x; z]] = [b; 0]$, $\Sigma_{xx} = \text{Cov}[x] = WW^\top + \sigma^2 I$ and $\Sigma_{zz} = \text{Cov}[z] = I$
  - $\Sigma_{xz} = \mathbb{E}[(x - \mathbb{E}[x])(z - \mathbb{E}[z])^\top] = \mathbb{E}[(Wz + b - \epsilon - b)z^\top] = W\mathbb{E}[zz^\top] - \mathbb{E}[\epsilon z^\top] = W$
  - $\Sigma_{zx} = W^\top$

Principal component analysis

Probabilistic view: posterior inference

- A probabilistic graphical model of principal component analysis
  - Unobserved: $z \sim \mathcal{N}(0, 1^{d'})$
  - Observed: $x \mid z \sim \mathcal{N}(Wz + b, \sigma^2 I)$
- Posterior distribution
  - $\mathbb{E}[z \mid x] = 0 + \Sigma_{zx}\Sigma_{xx}^{-1}(x - b) = W^\top (WW^\top + \sigma^2 I)^{-1}(x - b)$
  - $\mathrm{Cov}[z \mid x] = \Sigma_{zz} - \Sigma_{zx}\Sigma_{xx}^{-1}\Sigma_{zx} = I - W^\top (WW^\top + \sigma^2 I)^{-1} W$
  - The covariance of $z \mid x$ does not depend on x.

Principal component analysis

Probabilistic view: in the limit of no noise, it's pseudo inverse.

- When $\sigma^2 = 0$,
  - $\mathbb{E}[z \mid x] = W^\top (WW^\top + \sigma^2 I)^{-1}(x - b) = W^\top (WW^\top)^{-1}(x - b)$
  - $\text{Cov}[z \mid x] = I - W^\top (WW^\top + \sigma^2 I)^{-1} W = I$

Nonlinear principal component analysis

Going beyond linear transformation

- Generalizing PCA
  - From $x = Wz + b + \epsilon$ to $x = f(z) + \epsilon$.
- In general, $[x; z]$ is not Gaussian jointly anymore
  - because the mean and covariance of x may be tied.
  - The marginal $p(x) = \int p(x \mid z)p(z)\mathrm{d}z$ may not be Gaussian.
  - The posterior $p(z \mid x) = \dfrac{p(x \mid z)p(z)}{\int p(x \mid z)p(z)\mathrm{d}z}$ may not be Gaussian.

Nonlinear principal component analysis

Going beyond linear transformation

- Nonlinear principal component analysis can be *more powerful* than PCA
  - From $x = Wz + b + \epsilon$ to $x = f(z) + \epsilon$.
- The marginal is a mixture of (infinitely many) Gaussians: $p(x) = \int p(x \mid z) p(z) \mathrm{d}z$
  - For every z, there is a Gaussian $p(x \mid z)$ defined on the input space.
  - This Gaussian $p(x \mid z)$ is weighted by $p(z)$.
  - But, it's *still constrained*, because f is shared by all the Gaussian components.

Nonlinear principal component analysis

Variational inference: using an approximate posterior distribution

- Given f in $x = f(z) + \epsilon$, what is $\mathbb{E}_{z|x;f}[F(z)]$?
  - Intractable in general, because $p(z \mid s)$ is intractable in general.
- What if we use a tractable proxy $q(z \mid x)$ to the exact posterior $p(z \mid x)$?
  - $q(z \mid x)$ is a distribution we know how to easily use with a set of parameters
    - *Gaussian*, Laplace, etc.
  - $q(z \mid x)$ should be similar to $p(z \mid x)$: low $\mathrm{KL}(q \| p) = -\int q(z \mid x) \log p(z \mid x) \mathrm{d}z + \mathscr{H}(q)$

Nonlinear principal component analysis

Variational lower bound

- $q(z \mid x)$ should be similar to $p(z \mid x)$: low $\mathrm{KL}(q \| p) = -\int q(z \mid x) \log p(z \mid x) \mathrm{d}z + \mathscr{H}(q)$
- Variational lowerbound
  1. $\mathrm{KL}(q \| p) = -\mathbb{E}_q[\log p(x \mid z)] + \mathrm{KL}(q(z \mid x) \| p(z)) + \log p(x)$
  2. $\log p(x) \geq \mathbb{E}_q[\log p(x \mid z)] - KL(q(z \mid x) \| p(z))$

Nonlinear principal component analysis

Variational lowerbound: Expectation-Maximization algorithm

- Expectation step: find q that minimize $\mathrm{KL}(q\|p)$
  - $\mathrm{KL}(q\|p) = -\mathbb{E}_q[\log p(x\,|\,z)] + \mathrm{KL}(q(z\,|\,x)\|p(z)) + \log p(x)$
  - Equivalent to $\min_q -\mathbb{E}_q[\log p(x\,|\,z)] + \mathrm{KL}(q(z\,|\,x)\|p(z))$
    - Because $\log p(x)$ is not dependent on q.
  - Goals:
    - Find a distribution over z that can easily recover the corresponding x.
    - Ensure that z likely under q is also likely under the prior $p(z)$.

Nonlinear principal component analysis

Variational lowerbound: Expectation-Maximization algorithm

- Maximization step: find f that maximizes $\log p(x)$ (or $\sum_{n=1}^{N} \log p(x^n)$ with $N > 1$)
  - Instead of $\log p(x)$, maximize its lowerbound given q:
    - A classical technique in optimization
    - $\log p(x) \geq \mathbb{E}_q[\log p(x \mid z)] - KL(q(z \mid x) \| p(z))$
    - $\max_f \mathbb{E}_q[\log p(x \mid z)]$, since q and $p(z)$ are fixed.
  - The goal: find f that makes x likely given z from the approximate posterior q.

Nonlinear principal component analysis

Variational lowerbound: Expectation-Maximization algorithm

- Expectation-maximization algorithm
  - Expectation step: $\min_q -\mathbb{E}_q[\log p(x\,|\,z)] + \mathrm{KL}(q(z\,|\,x)\|p(z))$
  - Maximization step: $\max_f \mathbb{E}_q[\log p(x\,|\,z)]$
- Solving them all together
  - $$\min_{f,q^1,\ldots,q^N} \frac{1}{N}\sum_{n=1}^{N} \mathbb{E}_{z\sim q(z|x^n)}\left[-\log p(x\,|\,z)\right] + \mathrm{KL}(q(z\,|\,x^n)\|p(z))$$
  - *Simultaneously* infer the posterior distribution $q(z\,|\,x^n)$ of each data point *and* estimate f

Nonlinear principal component analysis

Variational lowerbound: Amortized inference

- As N grows, this approach becomes pretty much impossible:
  - $$\min_{f,q^1,\ldots,q^N} \frac{1}{N} \sum_{n=1}^{N} \mathbb{E}_{z\sim q(z|x^n)} \left[-\log p(x^n \mid z)\right] + \mathrm{KL}(q(z \mid x^n) \| p(z))$$
  - There are N approximate posterior distributions to be estimated.
  - Each approximate posterior distribution requires solving
    - $$\min_{q^n} \mathbb{E}_{z\sim q(z|x^n)} \left[-\log p(x^n \mid z)\right] + \mathrm{KL}(q(z \mid x^n) \| p(z))$$

Nonlinear principal component analysis

Variational lowerbound: Amortized inference

- Each approximate posterior distribution requires solving
  - $\min_{q^n} \mathbb{E}_{z \sim q(z|x^n)} \left[-\log p(x^n \mid z) \right] + \mathrm{KL}(q(z \mid x^n) \| p(z))$
- Instead, we train a neural net g to solve this optimization problem
  - $\min_{g} \frac{1}{N} \sum_{n=1}^{N} \mathbb{E}_{z \sim q(z|g(x^n))} \left[-\log p(x^n \mid z) \right] + \mathrm{KL}(q(z \mid g(x^n)) \| p(z))$
  - Amortize the cost of per-example optimization in the future by spending a lot of time training g in advance.

Nonlinear principal component analysis

Variational lowerbound maximization with amortized inference

- The final objective function:
  - $$\min_{f,g} \frac{1}{N} \sum_{n=1}^{N} \mathbb{E}_{z \sim q(z|g(x^n))} \left[-\log p(x^n \,|\, f(z)) \right] + \mathrm{KL}(q(z \,|\, g(x^n)) \| p(z))$$
- But, how do we compute the gradient of the lower bound w.r.t. g?

Reparametrization trick

Detour: the law of the unconscious statistician

- Let X be a continuous random variable with its density $p(X)$.
- We want to compute $\mathbb{E}[g(X)]$, but we don't know the density of $g(X)$.
- The law of the unconscious statistician tells us that
  - $\mathbb{E}[g(X)] = \int p(X)g(X)\mathrm{d}X$
    - That is, we *can* compute the expectation of $g(X)$ under the density of X.

Reparametrization trick

Detour: Gaussian reparametrization

- The law of the unconscious statistician: $\mathbb{E}[g(X)] = \int p(X)g(X)\mathrm{d}X$
- Let $X = \mu + \sigma \odot \epsilon$, where $\epsilon \sim \mathcal{N}(0, I^d)$
  - $\mathbb{E}_{X \sim \mathcal{N}(\mu, \mathrm{diag}(\sigma))}[g(X)] = \mathbb{E}_{\epsilon \sim \mathbb{N}(0, I^d)}[g(\mu + \sigma \odot \epsilon)]$
- Then, we can backpropagate through the sampling procedure:
  - $\nabla_\mu \mathbb{E}_{X \sim \mathcal{N}(\mu, \mathrm{diag}(\sigma^2)}[g(X)] = \nabla_\mu \mathbb{E}_{\epsilon \sim \mathcal{N}(0, I)}[g(\mu + \sigma \odot \epsilon)] = \mathbb{E}_{\epsilon \sim \mathcal{N}(0, I)}[\nabla_{\tilde{X}} g(\tilde{X}) \nabla_\mu \tilde{X}]$

Reparametrization trick

Detour: General reparametrization

- The law of the unconscious statistician: $\mathbb{E}[g(X)] = \int p(X)g(X)\mathrm{d}X$
- Let $X = A(\theta, \epsilon)$, where ϵ is independent of θ and A is differentiable w.r.t. θ:
  - $\mathbb{E}_X[g(X)] = \mathbb{E}_\epsilon[g(A(\theta, \epsilon)))]$
- Then, we can backpropagate through the sampling procedure:
  - $\nabla_\mu \mathbb{E}_X[g(X)] = \nabla_\mu \mathbb{E}_\epsilon[g(A(\theta, \epsilon))] = \mathbb{E}_\epsilon[\nabla_{\tilde{X}} g(\tilde{X}) \nabla_\theta \tilde{X}]$

Variational autoencoder

Variational lowerbound maximization with amortized inference and reparametrization

- With a reparametrizable distribution as q (e.g., Gaussian):
  - $$\min_{f,g} \frac{1}{N} \sum_{n=1}^{N} \mathbb{E}_{\epsilon} \left[-\log p(x^n \mid f(A(g(x^n), \epsilon))) \right] + \mathrm{KL}(q(z \mid g(x^n)) \| p(z))$$
- It is a regularized autoencoder with noisy bottleneck.
- We compute the gradient of the loss function w.r.t. g with backpropagation.
- We often refer to this as a *variational autoencoder*.

Variational autoencoder with discrete z

What if reparameterization is not possible?

- The original objective function without reparametrization:
  - $\min_{f,g} \frac{1}{N} \sum_{n=1}^{N} \mathbb{E}_{z \sim q(z|g(x^n))} \left[-\log p(x^n | f(z)) \right] + \mathrm{KL}(q(z | g(x^n)) \| p(z))$
- The gradient of the expectation is problematic:
  - $\nabla_g \mathbb{E}_{z \sim q(z|g(x))} \left[-\log p(x | f(z) \right] = -\int \nabla_g q(z | g(x)) \log p(x | f(z)) \mathrm{d}z$
- The log-derivative trick: $(\log f)' = \frac{f'}{f} \iff f' = f(\log f)'$

Variational autoencoder with discrete z

What if reparameterization is not possible?

- The gradient of the expectation with the log-derivative trick
  - $$-\int \nabla_g q(z \mid g(x)) \log p(x \mid f(z)) \mathrm{d}z = -\int q(z \mid g(x)) \log p(x \mid f(z)) \nabla_g \log q(z \mid g(x)) \mathrm{d}z$$
- Recall the definition of the expectation:
  - $$-\int q(z \mid g(x)) \log p(x \mid f(z)) \nabla_g \log q(z \mid g(x)) \mathrm{d}z = -\mathbb{E}_{z \sim q(z \mid g(x))} \left[\log p(x \mid f(z)) \nabla_g \log q(z \mid g(x)) \right]$$
- We use sample-based approximation:
  - $$-\mathbb{E}_{z \sim q(z \mid g(x))} \left[\log p(x \mid f(z)) \nabla_g \log q(z \mid g(x)) \right] = -\frac{1}{M} \sum_{m=1}^{M} \log p(x \mid f(z^m)) \nabla_g \log q(z^m \mid g(x))$$

Variational autoencoder with discrete z

What if reparameterization is not possible?

- The original objective function without reparametrization:
  - $\min_{f,g} \frac{1}{N} \sum_{n=1}^{N} \mathbb{E}_{z \sim q(z|g(x^n))} \left[-\log p(x^n \mid f(z)) \right] + \mathrm{KL}(q(z \mid g(x^n)) \| p(z))$
- REINFORCE estimator:
  - $\nabla_g \mathbb{E}_{z \sim q(z|g(x))} \left[-\log p(x \mid f(z)) \right] \approx -\frac{1}{M} \log p(x \mid f(z^n)) \nabla_g \log q(z^n \mid g(x))$
- It's always a good idea to use a continuous, reparametrizable latent variables.

What do f and g do?
Interpreting the variational autoencoder

- **Decoding**: f tells us a set of likely observations $p(x \mid f(z))$ given a latent configuration z.
- **Encoding**: g tells us which set of latent configurations $q(z \mid g(x))$ have likely resulted in the observation x.

Regularization in VAE

Detour: KL Divergence

- Kullback-Leibler (KL) divergence is perhaps the most widely used (asymmetric) divergence between two distributions in machine learning
  - $\mathrm{KL}(q\|p) = \int_z q(z) \log \frac{q(z)}{p(z)} \mathrm{d}z \geq 0$
  - KL divergence is 0, when $q = p$.

Regularization in VAE

Detour: KL Divergence

- KL divergence can be rewritten as $\mathrm{KL}(q||p) = -\mathbb{E}_{z\sim q}\left[\log p(z)\right] - \mathscr{H}(q)$
  - 1st term: the log-probability of z, which is likely under q, under p.
  - 2nd term: the entropy of q
- The lower KL divergence implies that
  - 1st term: any z likely under q must be likely under p, but z likely under p is not necessarily likely under q.
  - 2nd term: the support of q must be as large as possible.

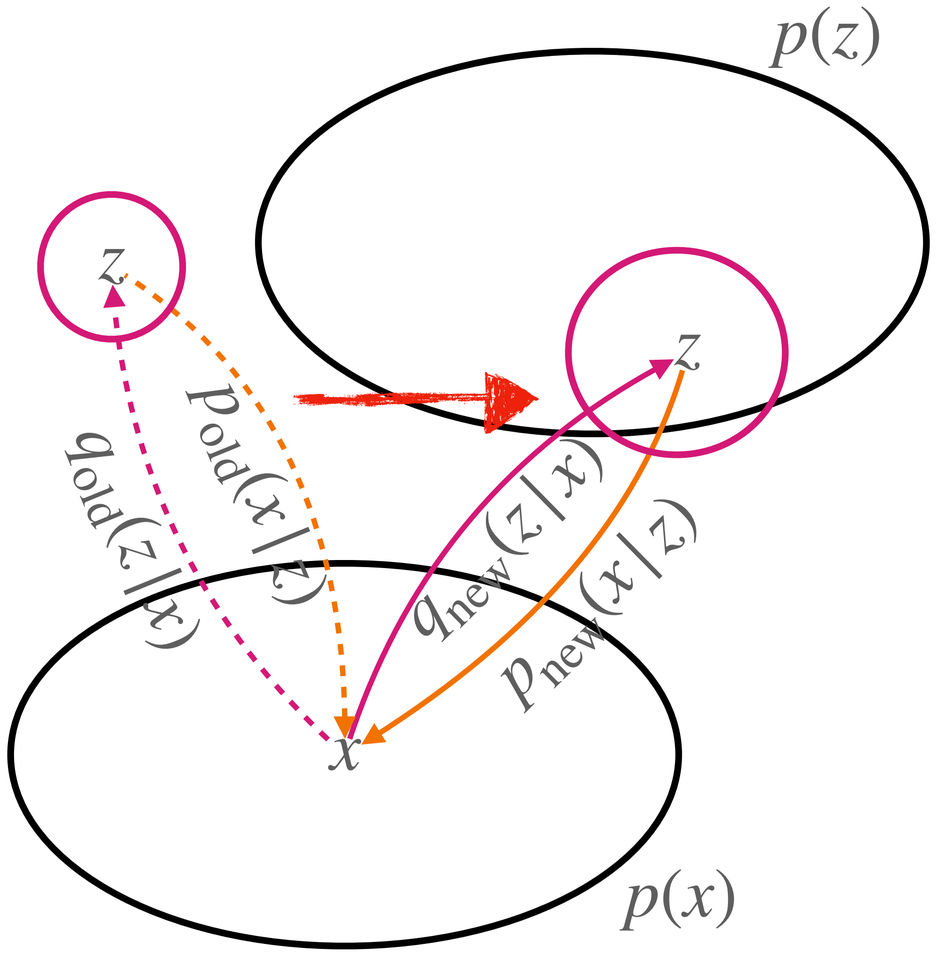
Regularization in VAE

KL Divergence from q to p: KL$(q\|p)$

- $-\mathbb{E}_{z\sim q}\left[\log p(z)\right]$: any z likely under q must be likely under p, but not vice versa.
  - q chooses z likely under the prior among those that may have generated x,
  - because we do not care about z unlikely under the prior.
- $-\mathscr{H}(q)$: the support of q must be as large as possible.
  - q must find the largest set of z that would have generated x,
  - because we do not want to leave any z likely under the prior hanging in the air.

Regularization in VAE

KL Divergence from q to p: KL$(q\|p)$



Regularized autoencoders

Autoencoding + Regularization

- The variational lower bound can be understood as the sum of
  - The (negative) reconstruction error: $\mathbb{E}_{z\sim q(z|x)}[\log p(x\,|\,z)]$
  - The regularization term: $\mathrm{KL}(q(z\,|\,x)\|p(z))$
- In other words, it's autoencoding + regularization
- $$\frac{1}{N}\sum_{n=1}^{N}\mathbb{E}_{z\sim z|x^n}[\log p(x^n\,|\,z)] - \frac{1}{N}\sum_{n=1}^{N}\mathrm{KL}(q(z\,|\,x^n)\|p(z))$$

Regularized autoencoders

VAE Regularization

- VAE Regularization
  - $\frac{1}{N}\sum_{n=1}^{N} \mathrm{KL}(q(z|x^n)\|p(z)) = -\frac{1}{N}\sum_{n=1}^{N}\int_z q(z|x^n)\log p(z) + \frac{1}{N}\sum_{n=1}^{N}\int_z q(z|x^n)\log q(z|x^n)$
- The first term can be rewritten as
  1. $\frac{1}{N}\sum_{n=1}^{N}\int_z q(z|x^n)\log p(z) = \int_z \left(\sum_{n=1}^{N}\frac{1}{N}q(z|x^n)\right)\log p(z)$

Regularized autoencoders

VAE Regularization

- VAE Regularization
  - $\frac{1}{N}\sum_{n=1}^{N} \mathrm{KL}(q(z\,|\,x^n)\|p(z)) = -\int_z \left(\sum_{n=1}^{N}\frac{1}{N}q(z\,|\,x^n)\right)\log p(z) + \frac{1}{N}\sum_{n=1}^{N}\int_z q(z\,|\,x^n)\log q(z\,|\,x^n)$
- Bounding the second term
  1. $\int_z \left(\sum_{n=1}^{N}\frac{1}{N}q(z\,|\,x^n)\right)\log\left(\sum_{n'=1}^{N}\frac{1}{N}q(z\,|\,x^{n'})\right) = \int_z \left(\sum_{n=1}^{N}\frac{1}{N}q(z\,|\,x^n)\right)\log\left(\sum_{n'=1}^{N}q(z\,|\,x^{n'})\right) - \log N$
  2. $\int_z \left(\sum_{n=1}^{N}\frac{1}{N}q(z\,|\,x^n)\right)\log\left(\sum_{n'=1}^{N}q(z\,|\,x^{n'})\right) - \log N \geq \int_z \left(\sum_{n=1}^{N}\frac{1}{N}q(z\,|\,x^n)\log q(z\,|\,x^n)\right) - \log N$
  3. $\int_z \left(\sum_{n=1}^{N}\frac{1}{N}q(z\,|\,x^n)\log q(z\,|\,x^n)\right) - \log N = \frac{1}{N}\sum_{n=1}^{N}\int_z q(z\,|\,x^n)\log q(z\,|\,x^n) - \log N$

Regularized autoencoders

VAE Regularization

- VAE Regularization
  - $\frac{1}{N}\sum_{n=1}^{N} \mathrm{KL}(q(z|x^n)\|p(z)) \leq -\int_z \left(\sum_{n=1}^{N}\frac{1}{N}q(z|x^n)\right)\log p(z) + \int_z \left(\sum_{n=1}^{N}\frac{1}{N}q(z|x^n)\right)\log\left(\sum_{n=1}^{N}\frac{1}{N}q(z|x^n)\right)$
  - $\frac{1}{N}\sum_{n=1}^{N} \mathrm{KL}(q(z|x^n)\|p(z)) \leq \mathrm{KL}\left(\sum_{n=1}^{N}\frac{1}{N}q(z|x^n)\middle\|p(z)\right)$
- It minimizes the lowerbound to the KL from the aggregate posterior to the prior distribution.
  - Can we instead directly minimize the latter?

Recap: energy-based models

Energy-based generative adversarial networks

- Start from a Boltzmann/Gibbs distribution
  - $\log p(x) = -E(x;\theta) - \log \int \exp(-E(x';\theta))\mathrm{d}x'$
- Maximum likelihood learning with SGD
  - $\nabla_\theta \log p(x) = -\nabla_\theta E(x;\theta) + \int \frac{e^{-E(x';\theta)}\nabla_\theta E(x';\theta)}{\int e^{-E(x'';\theta)}\mathrm{d}x''}\mathrm{d}x' = -\nabla_\theta E(x;\theta) + \mathbb{E}_{x'\sim p(x')}\left[\nabla_\theta E(x';\theta)\right]$
    - Learning minimizes the difference between the energy gradient under the data distribution and model distribution.

Recap: energy-based models

Energy-based generative adversarial networks

- Maximum likelihood learning with SGD
  - $\nabla_\theta \log p(x) = -\nabla_\theta E(x;\theta) + \mathbb{E}_{x' \sim p(x')}\left[\nabla_\theta E(x';\theta)\right]$
  - The 2nd term (negative phase) is often intractable to compute exactly:
  - Thus, MC approximation: $\mathbb{E}_{x' \sim p(x')}\left[\nabla_\theta E(x';\theta)\right] \approx \frac{1}{M}\sum_{m=1}^{M} \nabla_\theta E(x^m;\theta)$
  - But, sampling is also *very* difficult: I've wasted two years of my life on it.

Recap: energy-based models

Energy-based generative adversarial networks

- The purpose of sampling is to find (negative) samples with low energy.
- Instead, can directly generate negative samples by learning a generator?
  - $\min_g \mathbb{E}_{z\sim\mathcal{N}(0,I^{d'})}[E(g(z);\theta)]$, where $g : \mathbb{R}^{d'} \to \mathcal{X}$
  - Using SGD & MC: $\frac{1}{M}\sum_{m=1}^{M} \nabla_g E(g(z^m);\theta)$

Recap: energy-based models

Energy-based generative adversarial networks

- Energy-based generative adversarial networks
- Saddle-point optimization
  - Find a point that is minimum along some directions and is maximum along some other directions
  - A game between the energy function (discriminator) and the generator.
  - $$\max_{\theta} \min_{g} \frac{1}{|B|} \sum_{x \in B} - E(x;\theta) + \frac{1}{M} \sum_{m=1}^{M} \left[E(g(z^m);\theta) \right]$$

Recap: energy-based models

Energy-based generative adversarial networks

- Energy-based generative adversarial networks
- Near the equilibrium,
  - **The generator distribution** $p_g(x)$ **matches** the Boltzmann distribution $p(x;\theta)$, where
    - $p_g(x) = \int p(z)\mathcal{N}(x; g(z), \epsilon^2 I)\mathrm{d}z$
    - $p(x;\theta) = \exp(-E(x;\theta)) \Big/ \int \exp(-E(x';\theta))\mathrm{d}x'$
  - The Boltzmann distribution $p(x;\theta)$ matches **the target distribution** $p^*(x)$

Wasserstein Autoencoders

Replace KL divergence with the energy-based GAN

- Regularized autoencoders: autoencoding + distribution matching
  - $\frac{1}{N}\sum_{n=1}^{N}\mathbb{E}_{z\sim z|x^n}[\log p(x^n\,|\,z)] - \text{Div}\left(\sum_{n=1}^{N}\frac{1}{N}q(z\,|\,x^n)\middle\|p(z)\right)$
- Wassertein autoencoder uses the GAN instead of KL divergence.
  - Stochastic objective function:
    - $\max_{f,\theta}\min_{g}\frac{1}{N}\sum_{n=1}^{N}\frac{1}{B}\sum_{b=1}^{B}\log p(x^n\,|\,z^{n,b};f) - \frac{1}{M}\sum_{m=1}^{M}E(z^m;\theta) + \frac{1}{NB}\sum_{m=1}^{M}\sum_{b=1}^{B}\left[E(z^{n,b};\theta)\right]$
    - $z^{n,b}\sim q(z\,|\,x^n)$ and $z^m\sim p(z)$

Wasserstein Autoencoders

- Tolstikhin et al. [2017]

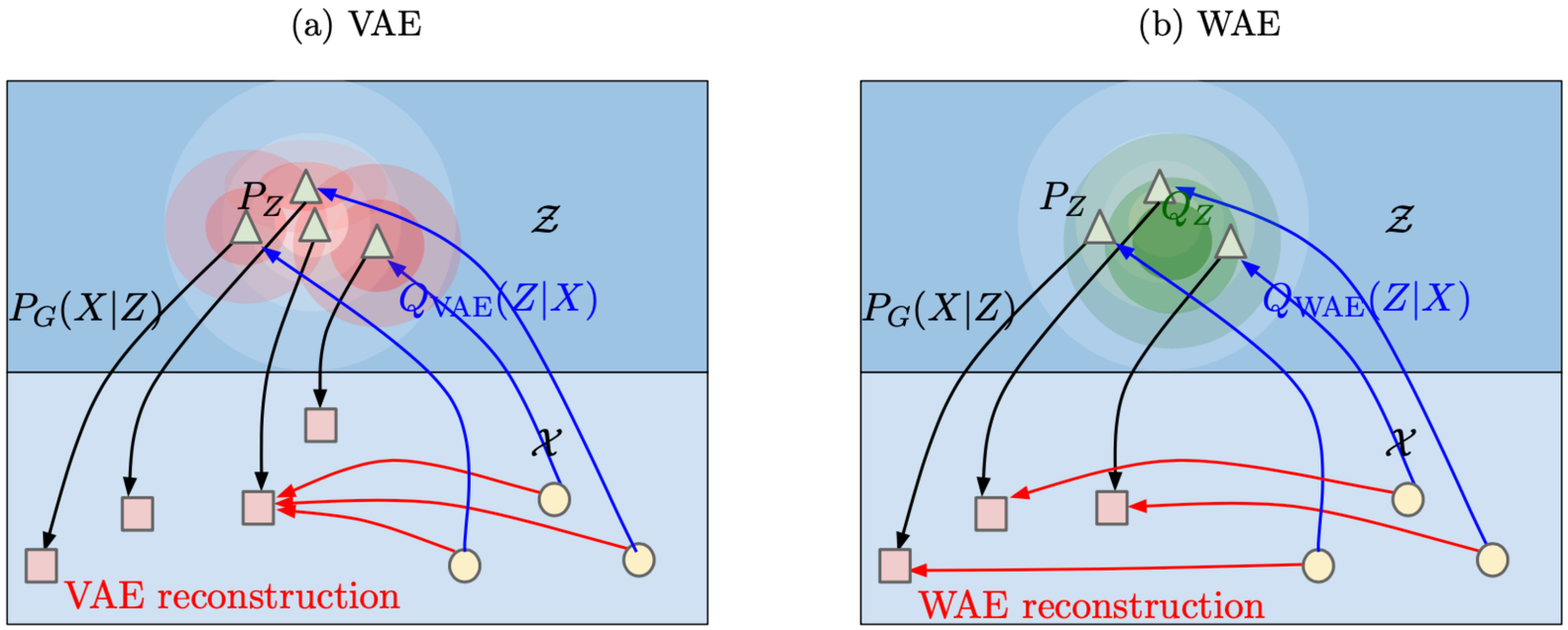


Figure 1: Both VAE and WAE minimize two terms: the reconstruction cost and the regularizer penalizing discrepancy between P_Z and distribution induced by the encoder Q. VAE forces $Q(Z|X = x)$ to match P_Z for all the different input examples x drawn from P_X. This is illustrated on picture (a), where every single red ball is forced to match P_Z depicted as the white shape. Red balls start intersecting, which leads to problems with reconstruction. In contrast, WAE forces the continuous mixture $Q_Z := \int Q(Z|X)dP_X$ to match P_Z, as depicted with the green ball in picture (b). As a result latent codes of different examples get a chance to stay far away from each other, promoting a better reconstruction.

Bonus: Normalizing flows

Change of variables

- Instead of regularization, impose a set of constraints on f (the decoder):
  - f is a bijective, differentiable function: the encoder is then f^{-1}.
  - It is efficient to compute the Jacobian of f^{-1}.
- We can compute the probability density of x exactly by a change of variable:
  - $$p(x) = p(z = f^{-1}(x)) \left| \frac{\partial f^{-1}}{\partial x} \right|$$

Bonus: Normalizing flows

Exact learning and inference

- Learning: $\max_f \frac{1}{N} \sum_{n=1}^{N} \log p(z = f^{-1}(x^n)) + \log \left| \frac{\partial f^{-1}}{\partial x^n} \right|$
- Inference
  - Posterior inference: $z = f^{-1}(x)$
  - Sampling: $f(z)$, where $z \sim \mathcal{N}(0, 1^d)$
- Limitations
  - The choice of f is non-trivial.
  - It cannot assign 0 probability to any configuration.