# Variational Autoencoders (VAEs)

## STATS 305C: Applied Statistics

Scott Linderman

May 11, 2023

# Where are we?

| Model | Algorithm | Application |
|---|---|---|
| Multivariate Normal Models | Conjugate Inference | Bayesian Linear Regression |
| Hierarchical Models | MCMC (MH & Gibbs) | Modeling Polling Data |
| Probabilistic PCA & Factor Analysis | MCMC (HMC) | Images Reconstruction |
| Mixture Models | EM & Variational Inference | Image Segmentation |
| Mixed Membership Models | Coordinate Ascent VI | Topic Modeling |
| Variational Autoencoders | Gradient-based VI | Image Generation |
| State Space Models | Message Passing | Segmenting Video Data |
| Bayesian Nonparametrics | Fancy MCMC | Modeling Neural Spike Trains |

# PCA as a linear autoencoder

Recall from Lecture 5 that PCA could be motivated as a linear autoencoder trained to minimize reconstruction error subject to having orthogonal weights.

# Deep autoencoders

Why restrict ourselves to **linear** autoencoders? The neural network community has used **deep autoencoders** (a.k.a. autoassociative networks) for nonlinear dimensionality reduction [LeCun, 1987, Bourlard and Kamp, 1988, Hinton and Zemel, 1993, Hinton and Salakhutdinov, 2006, Vincent et al., 2010]. See also, Goodfellow et al. [2016, Ch. 14].

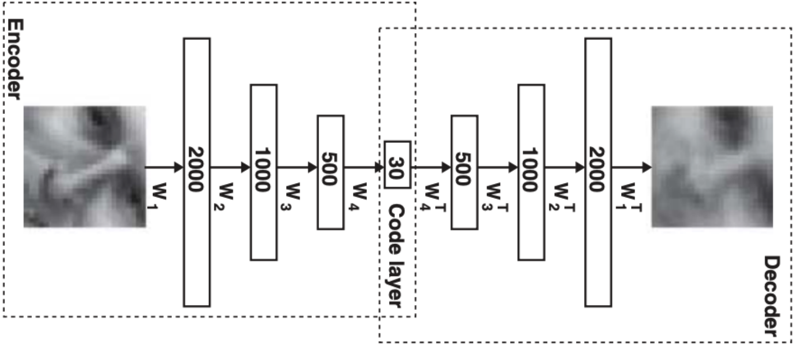
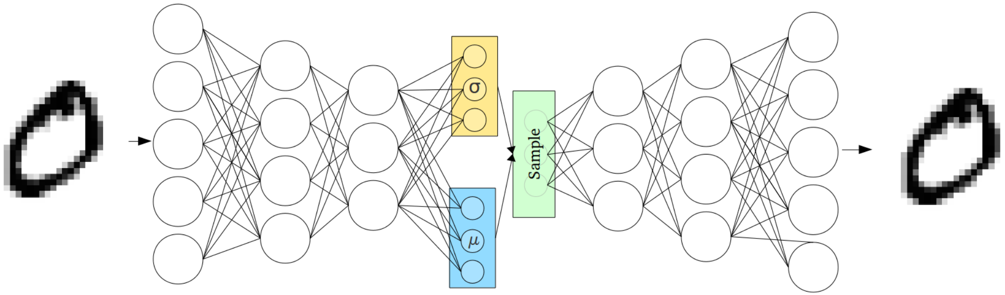



*Figure:* Figure from Hinton and Salakhutdinov [2006]

# Variational autoencoders as deep, stochastic, regularized autoencoders

Kingma and Welling [2014] and Rezende et al. [2014] concurrently developed what we now call **variational autoencoders**. The idea is to treat the hidden codes as random variables. As we will see, VAEs can be viewed as **deep generative models** combined with **amortized variational inference**.




$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}) = \mathbb{E}_{q(\mathbf{z}_n; \mathbf{x}_n, \boldsymbol{\phi})}[\log p(\mathbf{x}_n \mid \mathbf{z}_n; \boldsymbol{\theta})] - D_{\mathrm{KL}}(q(\mathbf{z}_n; \mathbf{x}_n, \boldsymbol{\phi}) \parallel p(\mathbf{z}_n)) \tag{1}$$

# Outline

- The generative model
- Variational expectation maximization
- Amortized inference

# The generative model

VAEs start with a "deep" but conceptually simple generative model,

$$
\begin{aligned}
\boldsymbol{z}_n &\sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}) && (2)\\
\boldsymbol{x}_n &\sim \mathcal{N}(g(\boldsymbol{z}_n; \boldsymbol{\theta}), \boldsymbol{I}) && (3)
\end{aligned}
$$

where $g : \mathbb{R}^H \to \mathbb{R}^D$ is a nonlinear mapping from $\boldsymbol{z}_n \in \mathbb{R}^H$ to $\mathbb{E}[\boldsymbol{x}_n] \in \mathbb{R}^D$, parameterized by $\boldsymbol{\theta}$.

We will assume $g$ is a simple **feedforward neural network** (a.k.a. multilayer perceptron) of the form,

$$
g(\boldsymbol{z}; \boldsymbol{\theta}) = g_L(g_{L-1}(\cdots g_1(\boldsymbol{z}) \cdots)) \tag{4}
$$

where each **layer** is a cascade of a linear mapping followed by an element-wise nonlinearity (except for the last layer, perhaps). For example,

$$
g_\ell(\boldsymbol{u}_\ell) = \mathrm{relu}(\boldsymbol{W}_\ell \boldsymbol{u}_\ell + \boldsymbol{b}_\ell); \qquad \mathrm{relu}(a) = \max(0, a). \tag{5}
$$

The generative parameters consist of the weights and biases, $\boldsymbol{\theta} = \{\boldsymbol{W}_\ell, \boldsymbol{b}_\ell\}_{\ell=1}^{L}$.

# Two goals

The **learning goal** is to find the parameters that **maximize the marginal probability of the data**,

$$\boldsymbol{\theta}^\star = \arg\max_{\boldsymbol{\theta}} p(\boldsymbol{X}; \boldsymbol{\theta}) \tag{6}$$

$$= \arg\max_{\boldsymbol{\theta}} \prod_{n=1}^{N} \int p(\boldsymbol{x}_n \mid \boldsymbol{z}_n; \boldsymbol{\theta}) \, p(\boldsymbol{z}_n; \boldsymbol{\theta}) \, \mathrm{d}\boldsymbol{z}_n \tag{7}$$

The **inference goal** is to find the **posterior distribution of latent variables**,

$$p(\boldsymbol{z}_n \mid \boldsymbol{x}_n; \boldsymbol{\theta}) = \frac{p(\boldsymbol{x}_n \mid \boldsymbol{z}_n; \boldsymbol{\theta}) \, p(\boldsymbol{z}_n; \boldsymbol{\theta})}{\int p(\boldsymbol{x}_n \mid \boldsymbol{z}'_n; \boldsymbol{\theta}) \, p(\boldsymbol{z}'_n; \boldsymbol{\theta}) \, \mathrm{d}\boldsymbol{z}'_n} \tag{8}$$

Both goals require an integral over $\boldsymbol{z}_n$, but that is intractable for deep generative models.

# The evidence lower bound (ELBO)

**Idea:** Use the ELBO to get a bound on the marginal probability and maximize that instead.

$$
\begin{aligned}
\log p(\boldsymbol{X}; \boldsymbol{\theta}) &= \sum_{n=1}^{N} \log p(\boldsymbol{x}_n; \boldsymbol{\theta}) && (9) \\
&\geq \sum_{n=1}^{N} \log p(\boldsymbol{x}_n; \boldsymbol{\theta}) - D_{\mathrm{KL}}\left(q_n(\boldsymbol{z}_n; \boldsymbol{\lambda}_n) \,\|\, p(\boldsymbol{z}_n \mid \boldsymbol{x}_n; \boldsymbol{\theta})\right) && (10) \\
&= \sum_{n=1}^{N} \underbrace{\mathbb{E}_{q_n(\boldsymbol{z}_n)}\left[\log p(\boldsymbol{x}_n, \boldsymbol{z}_n; \boldsymbol{\theta}) - \log q_n(\boldsymbol{z}_n; \boldsymbol{\lambda}_n)\right]}_{\text{“local ELBO”}} && (11) \\
&\triangleq \sum_{n=1}^{N} \mathscr{L}_n(\boldsymbol{\lambda}_n, \boldsymbol{\theta}) && (12) \\
&= \mathscr{L}(\boldsymbol{\lambda}, \boldsymbol{\theta}) && (13)
\end{aligned}
$$

where $\boldsymbol{\lambda} = \{\boldsymbol{\lambda}_n\}_{n=1}^{N}$. Here, I've written the ELBO as a sum of “local ELBOs” $\mathscr{L}_n$.

# Optimal variational posterior

The ELBO is still maximized (and the bound is tight) when each $q_n$ is equal to the true posterior,

$$q_n(\boldsymbol{z}_n; \boldsymbol{\lambda}_n) = p(\boldsymbol{z}_n \mid \boldsymbol{x}_n, \boldsymbol{\theta}). \tag{14}$$

**Question:** The deep generative model above has a Gaussian prior on $\boldsymbol{z}_n$ and a Gaussian likelihood for $\boldsymbol{x}_n$ given $\boldsymbol{z}_n$. Why isn't the posterior Gaussian?

# Review: Gradient-based VI

Nevertheless, we can still constrain $q_n$ to be Gaussian and seek the best Gaussian approximation to the posterior. This is sometimes called **fixed-form**, **black-box**, or **automatic differentiation VI**.

For example, let,

$$\mathcal{Q} = \left\{q : q(\boldsymbol{z}; \boldsymbol{\lambda}) = \mathcal{N}\left(\boldsymbol{z} \mid \boldsymbol{\mu}, \mathrm{diag}(\boldsymbol{\sigma}^2)\right) \text{ for } \boldsymbol{\lambda} = (\boldsymbol{\mu}, \log \boldsymbol{\sigma}^2) \in \mathbb{R}^{2H}\right\} \tag{15}$$

Then, for fixed parameters $\boldsymbol{\theta}$, the best $q_n$ in this **variational family** is,

$$\begin{aligned} q_n^\star &= \underset{q_n \in \mathcal{Q}}{\arg\min}\, D_{\mathrm{KL}}\left(q_n(\boldsymbol{z}_n; \boldsymbol{\lambda}_n) \,\|\, p(\boldsymbol{z}_n \mid \boldsymbol{x}_n; \boldsymbol{\theta})\right) && (16) \\ &= \underset{\boldsymbol{\lambda}_n \in \mathbb{R}^{2H}}{\arg\max}\, \mathcal{L}_n(\boldsymbol{\lambda}_n, \boldsymbol{\theta}). && (17) \end{aligned}$$

We can maximize the ELBO with **stochastic gradient ascent** using unbiased estimates of the gradient, $\widehat{\nabla}_{\boldsymbol{\lambda}_n} \mathcal{L}(\boldsymbol{\lambda}_n, \boldsymbol{\theta})$, e.g., using the **score-function** or the **pathwise** gradient estimators.

# Variational expectation-maximization (vEM)

Now we can introduce a new algorithm: **variational expectation maximization**.

Repeat until either the ELBO or the parameters converges:

1. **M-step:** Set $\boldsymbol{\theta} \leftarrow \arg\max_{\boldsymbol{\theta}} \mathscr{L}(\boldsymbol{\lambda}, \boldsymbol{\theta})$
2. **E-step:** For $n = 1, \ldots, N$
    - Set $\boldsymbol{\lambda}_n \leftarrow \arg\max_{\boldsymbol{\lambda}_n \in \boldsymbol{\Lambda}} \mathscr{L}_n(\boldsymbol{\lambda}_n, \boldsymbol{\theta})$
3. Compute (an estimate of) the ELBO $\mathscr{L}(\boldsymbol{\lambda}, \boldsymbol{\theta})$.

Unfortunately, none of these steps will have closed form solutions, so we'll have to use approximations.

# Generic M-step with stochastic gradient ascent

- For exponential family mixture models and simple factor analysis, the M-steps had closed form. For deep generative models, we need a more general approach.
- If the parameters are unconstrained and the ELBO is differentiable wrt $\boldsymbol{\theta}$, we can use **stochastic gradient ascent**.

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha \nabla_{\boldsymbol{\theta}} \mathscr{L}(q, \boldsymbol{\theta}) = \boldsymbol{\theta} + \alpha \sum_{n=1}^{N} \mathbb{E}_{q(\boldsymbol{z}_n; \boldsymbol{\lambda}_n)} \left[ \nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{x}_n, \boldsymbol{z}_n; \boldsymbol{\theta}) \right] \tag{18}$$

- Note that the expected gradient wrt $\boldsymbol{\theta}$ can be computed using ordinary Monte Carlo — no fancy gradient estimators necessary!
- Likewise, we can use **mini-batches** of data to approximate the sum over data points.

# Variational expectation-maximization (vEM)

Now we can introduce a new algorithm: **variational expectation maximization**.

Repeat until either the ELBO or the parameters converge:

1. **M-step:** Set $\boldsymbol{\theta} \leftarrow \arg\max_{\boldsymbol{\theta}} \mathscr{L}(q, \boldsymbol{\theta})$
2. **E-step:** For $n = 1, \ldots, N$
   - Set $\boldsymbol{\lambda}_n \leftarrow \arg\max_{\boldsymbol{\lambda}_n \in \boldsymbol{\Lambda}} \mathscr{L}_n(\boldsymbol{\lambda}_n, \boldsymbol{\theta})$
3. Compute the ELBO $\mathscr{L}(q, \boldsymbol{\theta})$.

Unfortunately, none of these steps will have closed form solutions, so we'll have to use approximations.

# The variational E-step

- Assume $\mathcal{Q}$ is the family of Gaussian distributions with diagonal covariance: $q_n(\boldsymbol{z}_n) = \mathcal{N}(\boldsymbol{z}_n \mid \boldsymbol{\mu}_n, \mathrm{diag}(\boldsymbol{\sigma}_n^2))$, with **variational parameters** $\boldsymbol{\lambda}_n = (\boldsymbol{\mu}_n, \log \boldsymbol{\sigma}_n^2) \in \mathbb{R}^{2H}$.
- To perform SGD, we need an unbiased estimate of the gradient of the local ELBO, but

$$\nabla_{\boldsymbol{\lambda}_n} \mathcal{L}_n(\boldsymbol{\lambda}_n, \boldsymbol{\theta}) = \nabla_{\boldsymbol{\lambda}_n} \mathbb{E}_{q(\boldsymbol{z}_n; \boldsymbol{\lambda}_n)} [\log p(\boldsymbol{x}_n, \boldsymbol{z}_n; \boldsymbol{\theta}) - \log q(\boldsymbol{z}_n; \boldsymbol{\lambda}_n)] \tag{19}$$

$$\neq \mathbb{E}_{q(\boldsymbol{z}_n; \boldsymbol{\lambda}_n)} \left[ \nabla_{\boldsymbol{\lambda}_n} (\log p(\boldsymbol{x}_n, \boldsymbol{z}_n; \boldsymbol{\theta}) - \log q(\boldsymbol{z}_n; \boldsymbol{\lambda}_n)) \right]. \tag{20}$$

- Last lecture we introduced the score-function and pathwise gradient estimators to tackle this problem. For example,

$$\nabla_{\boldsymbol{\lambda}_n} \mathcal{L}_n(\boldsymbol{\lambda}_n, \boldsymbol{\theta}) = \nabla_{\boldsymbol{\lambda}_n} \mathbb{E}_{q(\boldsymbol{z}_n; \boldsymbol{\lambda}_n)} [\log p(\boldsymbol{x}_n, \boldsymbol{z}_n; \boldsymbol{\theta}) - \log q(\boldsymbol{z}_n; \boldsymbol{\lambda}_n)] \tag{21}$$

$$= \mathbb{E}_{\boldsymbol{\epsilon}_n \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})} \left[ \nabla_{\boldsymbol{\lambda}_n} (\log p(\boldsymbol{x}_n, r(\boldsymbol{\lambda}_n, \boldsymbol{\epsilon}_n); \boldsymbol{\theta}) - \log q(r(\boldsymbol{\lambda}_n, \boldsymbol{\epsilon}_n); \boldsymbol{\lambda}_n)) \right] \tag{22}$$

where $r(\boldsymbol{\lambda}_n, \boldsymbol{\epsilon}_n) = \boldsymbol{\mu}_n + \boldsymbol{\sigma}_n \boldsymbol{\epsilon}_n$.

# Variational expectation-maximization (vEM)

Now we can add some detail to our variational expectation maximization algorithm.

Repeat until either the ELBO or the parameters converges:

1. **M-step:** Set $\boldsymbol{\theta} \leftarrow \arg\max_{\boldsymbol{\theta}} \mathscr{L}(q, \boldsymbol{\theta})$ [with stochastic gradient ascent on the ELBO]
2. **E-step:** For $n = 1, \ldots, N$
   - ~~Set $q_n \leftarrow \arg\max_{q_n \in \mathscr{Q}} \mathscr{L}_n(q_n, \boldsymbol{\theta})$~~
   - Set $\boldsymbol{\lambda}_n \leftarrow \arg\max_{\boldsymbol{\lambda}_n} \mathscr{L}_n(\boldsymbol{\lambda}_n, \boldsymbol{\theta})$
     [with stochastic gradient ascent on the local ELBO using either the score function estimator or the pathwise gradient estimator]
3. Compute the ELBO $\mathscr{L}(q, \boldsymbol{\theta})$. [with Monte Carlo]

# Amortized inference with recognition networks

- Note that vEM involves a costly E-step to find the variational parameters $\boldsymbol{\lambda}_n$ for each data point. This could involve many steps of stochastic gradient descent inside just the E-step!
- With a finite computational budget, we might be better off doing more gradient steps on $\boldsymbol{\theta}$ and fewer on the local variational parameters.
- Note that the optimal variational parameters are just a function of the data point and the model parameters,

$$\boldsymbol{\lambda}_n^\star = \arg\min_{\boldsymbol{\lambda}_n} D_{\mathrm{KL}}\left(q(\boldsymbol{z}_n; \boldsymbol{\lambda}_n) \,\|\, p(\boldsymbol{z}_n \mid \boldsymbol{x}_n, \boldsymbol{\theta})\right) \triangleq f^\star(\boldsymbol{x}_n, \boldsymbol{\theta}). \tag{23}$$

  for some implicit and generally nonlinear function $f^\star$.

# Amortized inference with recognition networks II

- VAEs learn an approximation to $f^{\star}(\boldsymbol{x}_n, \boldsymbol{\theta})$ with an **inference network**, a.k.a. **recognition network** or **encoder**.
- The inference network is (yet another) neural network that takes in a data point $\boldsymbol{x}_n$ and outputs variational parameters $\boldsymbol{z}_n$,

$$\boldsymbol{\lambda}_n \approx f(\boldsymbol{x}_n, \boldsymbol{\phi}), \tag{24}$$

  where $\boldsymbol{\phi}$ are the weights of the network.
- The advantage is that the inference network is very fast; in the E-step, we simply need to pass a data point through the network to obtain the variational parameters.
- The disadvantage is the output will not minimize the KL divergence. However, in practice we might tolerate a worse variational posterior and a weaker lower bound if it buys us more updates of $\boldsymbol{\theta}$.

# Amortization and approximation gaps

Cremer et al. [2018] consider the relative effects of the **amortization gap** and the **approximation gap** on variational EM.

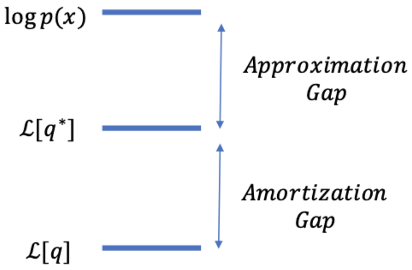



*Figure 1.* Gaps in Inference

# Linear VAEs

**Question:** What does the optimal encoder network look like for a VAE with a linear generative model,

$$\boldsymbol{z}_n \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}) \tag{25}$$

$$\boldsymbol{x}_n \sim \mathcal{N}(\boldsymbol{W}\boldsymbol{z}_n + \boldsymbol{b}, \boldsymbol{I}) \tag{26}$$

# Putting it all together

Logically, I find it helpful to distinguish between the E and M steps, but with recognition networks and stochastic gradient ascent, the line is blurred.

The final algorithm looks like this. Repeat until either the ELBO or the parameters converges:

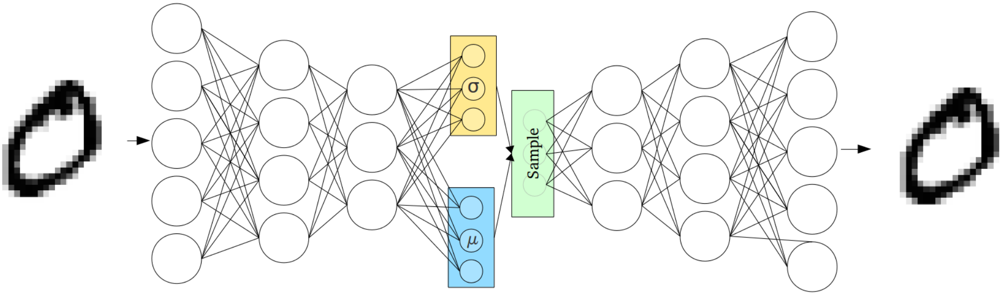
1. Sample data point $n \sim \text{Unif}(1, \dots, N)$. [Or a minibatch of data points.]
2. Estimate the local ELBO $\mathscr{L}_n(\boldsymbol{\phi}, \boldsymbol{\theta})$ with Monte Carlo. [Note: it is a function of $\boldsymbol{\phi}$ instead of $\boldsymbol{\lambda}_n$.]
3. Compute unbiased Monte Carlo estimates of the gradients $\widehat{\nabla}_{\boldsymbol{\theta}} \mathscr{L}_n(\boldsymbol{\phi}, \boldsymbol{\theta})$ and $\widehat{\nabla}_{\boldsymbol{\phi}} \mathscr{L}_n(\boldsymbol{\phi}, \boldsymbol{\theta})$. [The latter requires the score function or pathwise gradient estimator.]
4. Set

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha_i \widehat{\nabla}_{\boldsymbol{\theta}} \mathscr{L}_n(\boldsymbol{\phi}, \boldsymbol{\theta}) \tag{27}$$

$$\boldsymbol{\phi} \leftarrow \boldsymbol{\phi} + \alpha_i \widehat{\nabla}_{\boldsymbol{\phi}} \mathscr{L}_n(\boldsymbol{\phi}, \boldsymbol{\theta}) \tag{28}$$

with step size $\alpha_i$ decreasing over iterations $i$ according to a valid schedule.

# VAEs from an autoencoder perspective




From `https://towardsdatascience.com/intuitively-understanding-variational-autoencoders-1bfe67eb5daf`

# References I

Yann LeCun. *Modeles connexionnistes de lapprentissage*. PhD thesis, These de Doctorat, Universite Paris, 1987.

Hervé Bourlard and Yves Kamp. Auto-association by multilayer perceptrons and singular value decomposition. *Biological cybernetics*, 59(4):291–294, 1988.

Geoffrey E Hinton and Richard Zemel. Autoencoders, minimum description length and helmholtz free energy. *Advances in neural information processing systems*, 6, 1993.

G E Hinton and R R Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, July 2006.

Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, Pierre-Antoine Manzagol, and Léon Bottou. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research*, 11(12), 2010.

# References II


Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

D P Kingma and M Welling. Auto-encoding variational Bayes. *arXiv preprint arXiv:1312.6114*, 2014.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. January 2014.

Chris Cremer, Xuechen Li, and David Duvenaud. Inference suboptimality in variational autoencoders. In *International Conference on Machine Learning*, pages 1078–1086. PMLR, 2018.