Generative Learning: Variational Autoencoders

Yossi Keshet

Bar-Ilan University

April 26, 2021
Outline

- What is Generative Learning?
- PixelRNN and PixelCNN
- Autoencoders
- Variational Autoencoders (VAE)
Supervised versus unsupervised learning

**Supervised learning**

- **Data:** \((x, y)\)
  - \(x\) is an instance, \(y\) its label
- **Goal:** learn the mapping \(x \rightarrow y\)
- **Examples:** classification, regression, object detection, image segmentation, etc.

**Unsupervised learning**

- **Data:** \(x\)
  - no labels are given
- **Goal:** learn the distribution that generates the data or some hidden or underlying structure of the data
- **Examples:** clustering, feature reduction, new image generation, etc.
Goal: Given (unlabeled) training data \( \{ x_i \}_{i=1}^{N} \), the goal is to learn the underlying generating distribution. This can be used to generate new samples from the same distribution.
Generative models

We would like to learn $p_{model}(x)$ that would be similar to $p_{data}(x)$.

Training data $\sim P_{data}(x)$  
Generated samples $\sim P_{model}(x)$
Model the density directly. The model is based on

1. the chain rule to decompose the likelihood of an example $x$ into a product of distributions:

$$p(x) = \prod_{i=1}^{N} p(x_i | x_{i-1}, x_{i-2}, \ldots, x_1)$$

2. maximizing the likelihood of training data
Fully visible Belief Network

Model the density directly. The model is based on

1. the chain rule to decompose the likelihood of an example \( x \) into a product of distributions:

\[
p(x) = \prod_{i=1}^{N} p(x_i | x_{i-1}, x_{i-2}, \ldots, x_1)
\]

2. maximizing the likelihood of training data

The probability \( p(x_i | x_{i-1}, x_{i-2}, \ldots, x_1) \) is complex to estimate. It can be estimated by some Markovian assumption, i.e., that the current sample \( x_i \) depends on the last \( L \) samples, and it can be estimated using a neural network.
PixelRNN (van der Oord et al. 2016)

- Generates image pixels starting from corner.
- Dependency on previous pixels modeled using an RNN (LSTM).
- Disadvantage: inference is slow because the image is generated sequentially.
PixelRNN (van der Oord et al. 2016)

- Generates image pixels starting from corner.
- Dependency on previous pixels modeled using an RNN (LSTM).
- Disadvantage: inference is slow because the image is generated sequentially.
PixelRNN (van der Oord et al. 2016)

▶ Generates image pixels starting from corner.
▶ Dependency on previous pixels modeled using an RNN (LSTM).
▶ Disadvantage: inference is slow because the image is generated sequentially.
PixelRNN (van der Oord et al. 2016)

- Generates image pixels starting from corner.
- Dependency on previous pixels modeled using an RNN (LSTM).
- Disadvantage: inference is slow because the image is generated sequentially.
PixelCNN (van der Oord et al. 2016)

- Also generates image pixels starting from corner.
- Dependency on previous pixels modeled using an CNN over the context region.
- Training is faster than PixelRNN, since the convolution operations can be parallelized.
- The inference is slow because the image is still generated sequentially.
Autoencoders
Autoencoders

- Unsupervised approach for learning a lower-dimensional feature representation from unlabeled training data.
- Learns a mapping from the data $x$ to a low-dimensional representation $z$.
- Low dimensional representation of $x$ can be used as a feature vector.
Autoencoders

- How can we learn this low-dimensional representation?
- We train a model that is used to reconstruct the original data.
Autoencoders

- How can we learn this low-dimensional representation?
- We train a model that is used to reconstruct the original data.

\[ \ell(x, \hat{x}) = \|x - \hat{x}\|^2 \]

The loss function doesn’t use any labels!
Autoencoders - Examples

left: 1st epoch, middle: 9th epoch, right: original
(after http://kvfrans.com/variational-autoencoders-explained/)
Autoencoders

- **Bottleneck hidden layer** forces the network to learn a compressed latent representation.
- **Reconstruction loss** forces the latent representation to capture and encode as much information as possible.
Variational Autoencoders (VAE)
VAE versus Autoencoders

In autoencoders the latent variable is generated deterministically by the encoder.
In VAE the latent representation is generated from a probabilistic model.

Sample from the Multivariate normal distribution $\mathcal{N}(\mu, \Sigma)$ to generate the latent sample.
VAE Optimization

VAE assumes that the data are generated by some random process, involving an unobserved continuous random variable $z$:

1. The value of $z^{(i)}$ is generated from a prior distribution $p_{\theta}(z)$.
2. The value of $x^{(i)}$ is generated from a conditional probability $p_{\theta}(x|z)$.
We can assume that $p_\theta(z)$ and $p_\theta(x|z)$ come from parametric families of distributions, or can be learned using neural network. Then:

$$p_\theta(x) = \int p_\theta(z)p_\theta(x|z)dz$$

The main problem: this is intractable – we can not generate the probability $p_\theta(x|z)$ for every $z$. The EM algorithm cannot be used since the conditional probability $p_\theta(z|x) = p_\theta(z)p_\theta(x|z)/p_\theta(x)$ is intractable as well.
Solution: In addition to decoder network modeling $p_{\theta}(x|z)$, define additional encoder network $q_{\phi}(z|x)$ that approximates $p_{\theta}(z|x)$. We derive a lower bound on the data likelihood that is tractable, which we can optimize.
VAE Optimization

\[ \log p_\theta(x^{(i)}) = \mathbb{E}_{z \sim q_\phi(z|x^{(i)})} \left[ \log p_\theta(x^{(i)}) \right] \quad p_\theta(x^{(i)}) \text{ does not depend on } z \]

\[ = \mathbb{E}_z \left[ \log \frac{p_\theta(x^{(i)}|z)p_\theta(z)}{p_\theta(z|x^{(i)})} \right] \quad \text{Bayes' rule} \]

\[ = \mathbb{E}_z \left[ \log \frac{p_\theta(x^{(i)}|z)p_\theta(z)}{p_\theta(z|x^{(i)})} \frac{q_\phi(z|x^{(i)})}{q_\phi(z|x^{(i)})} \right] \quad \text{Mul. and div. by same constant} \]

\[ = \mathbb{E}_z \left[ \log p_\theta(x^{(i)}|z) \right] - \mathbb{E}_z \left[ \log \frac{q_\phi(z|x^{(i)})}{p_\theta(z)} \right] + \mathbb{E}_z \left[ \log \frac{q_\phi(z|x^{(i)})}{p_\theta(z|x^{(i)})} \right] \]

\[ = \mathbb{E}_z \left[ \log p_\theta(x^{(i)}|z) \right] - D_{KL}(q_\phi(z|x^{(i)})||p_\theta(z)) + D_{KL}(q_\phi(z|x^{(i)})||p_\theta(z|x^{(i)})) \]

Decoder network estimates \( p_\theta(x|z) \) through sampling

This KL term (between Gaussians for encoder and \( z \) prior) has nice closed-form solution

\( p_\theta(z|x) \) is intractable and we can't compute this term – but it is always \( \geq 0 \)
\[
\log p_\theta(x^{(i)}) = \mathbb{E}_{z \sim q_\phi(z|x^{(i)})} \left[ \log p_\theta(x^{(i)}) \right]
\]

\[
= \mathbb{E}_z \left[ \log \frac{p_\theta(x^{(i)}|z)p_\theta(z)}{p_\theta(z|x^{(i)})} \right]
\]

\[
= \mathbb{E}_z \left[ \log \frac{p_\theta(x^{(i)}|z)p_\theta(z)}{p_\theta(z|x^{(i)})} \right] q_\phi(z|x^{(i)}) \]

\[
= \mathbb{E}_z \left[ \log \frac{p_\theta(x^{(i)}|z)p_\theta(z)}{p_\theta(z|x^{(i)})} \right] q_\phi(z|x^{(i)})
\]

\[
= \mathbb{E}_z \left[ \log p_\theta(x^{(i)}|z) \right] - \mathbb{E}_z \left[ \log \frac{q_\phi(z|x^{(i)})}{p_\theta(z)} \right] + \mathbb{E}_z \left[ \log \frac{q_\phi(z|x^{(i)})}{p_\theta(z|x^{(i)})} \right]
\]

\[
= \mathbb{E}_z \left[ \log p_\theta(x^{(i)}|z) \right] - D_{KL}(q_\phi(z|x^{(i)} || p_\theta(z)) + D_{KL}(q_\phi(z|x^{(i)} || p_\theta(z|x^{(i)}))
\]

Decoder network estimates \( p_\theta(x|z) \) through sampling

This KL term (between Gaussians for encoder and \( z \) prior) has nice closed-form solution \( p_\theta(z|x) \) is intractable and we can’t compute this term – but it is always \( \geq 0 \)
\[
\log p_\theta(x^{(i)}) = \mathbb{E}_{z \sim q_\phi(z|x^{(i)})} \left[ \log p_\theta(x^{(i)}) \right] \\
= \mathbb{E}_z \left[ \log \frac{p_\theta(x^{(i)}|z)p_\theta(z)}{p_\theta(z|x^{(i)})} q_\phi(z|x^{(i)}) \right] \\
= \mathbb{E}_z \left[ \log p_\theta(x^{(i)}|z) \right] - \mathbb{E}_z \left[ \log \frac{q_\phi(z|x^{(i)})}{p_\theta(z)} \right] + \mathbb{E}_z \left[ \log \frac{q_\phi(z|x^{(i)})}{p_\theta(z|x^{(i)})} \right] \\
= \mathbb{E}_z \left[ \log p_\theta(x^{(i)}|z) \right] - D_{KL}(q_\phi(z|x^{(i)})||p_\theta(z)) + D_{KL}(q_\phi(z|x^{(i)})||p_\theta(z|x^{(i)}))
\]

Decoder network estimates \( p_\theta(x|z) \)

This KL term (between Gaussians for encoder and \( z \) prior) has nice closed-form solution

\( p_\theta(z|x) \) is intractable and we can't compute this term – but it is always \( \geq 0 \)
\[
\log p_{\theta}(x^{(i)}) = \mathbb{E}_{z \sim q_{\phi}(z|x^{(i)})} \left[ \log p_{\theta}(x^{(i)}) \right] \quad \text{\(p_{\theta}(x^{(i)})\) does not depend on } z
\]

\[
= \mathbb{E}_z \left[ \log \frac{p_{\theta}(x^{(i)}|z)p_{\theta}(z)}{p_{\theta}(z|x^{(i)})} \right] \quad \text{Bayes’ rule}
\]

\[
= \mathbb{E}_z \left[ \log \frac{p_{\theta}(x^{(i)}|z)p_{\theta}(z)}{p_{\theta}(z|x^{(i)})} \frac{q_{\phi}(z|x^{(i)})}{q_{\phi}(z|x^{(i)})} \right] \quad \text{Mul. and div. by same constant}
\]

\[
= \mathbb{E}_z \left[ \log p_{\theta}(x^{(i)}|z) \right] - \mathbb{E}_z \left[ \log \frac{q_{\phi}(z|x^{(i)})}{p_{\theta}(z)} \right] + \mathbb{E}_z \left[ \log \frac{q_{\phi}(z|x^{(i)})}{p_{\theta}(z|x^{(i)})} \right]
\]

\[
= \mathbb{E}_z \left[ \log p_{\theta}(x^{(i)}|z) \right] - D_{KL}(q_{\phi}(z|x^{(i)})||p_{\theta}(z)) + D_{KL}(q_{\phi}(z|x^{(i)})||p_{\theta}(z|x^{(i)}))
\]

Decoder network estimates \(p_{\theta}(x|z)\) through sampling.

This KL term (between Gaussians for encoder and \(z\) prior) has nice closed-form solution.

\(p_{\theta}(z|x)\) is intractable and we can’t compute this term – but it is always \(\geq 0\).
$$\log p_\theta(x^{(i)}) = \mathbb{E}_{z \sim q_\phi(z|x^{(i)})} \left[ \log p_\theta(x^{(i)}) \right] \quad \text{ } p_\theta(x^{(i)}) \text{ does not depend on } z$$

$$= \mathbb{E}_z \left[ \log p_\theta(x^{(i)}|z)p_\theta(z) \middle/ p_\theta(z|x^{(i)}) \right] \quad \text{Bayes' rule}$$

$$= \mathbb{E}_z \left[ \log \frac{p_\theta(x^{(i)}|z)p_\theta(z)}{p_\theta(z|x^{(i)})} \frac{q_\phi(z|x^{(i)})}{q_\phi(z|x^{(i)})} \right] \quad \text{Mul. and div. by same constant}$$

$$= \mathbb{E}_z \left[ \log p_\theta(x^{(i)}|z) \right] - \mathbb{E}_z \left[ \log \frac{q_\phi(z|x^{(i)})}{p_\theta(z)} \right] + \mathbb{E}_z \left[ \log \frac{p_\theta(z|x^{(i)})}{q_\phi(z|x^{(i)})} \right]$$

$$= \mathbb{E}_z \left[ \log p_\theta(x^{(i)}|z) \right] - D_{KL}(q_\phi(z|x^{(i)})||p_\theta(z)) + D_{KL}(q_\phi(z|x^{(i)})||p_\theta(z|x^{(i)}))$$

Decoder network estimates $p_\theta(x|z)$ through sampling

This KL term (between Gaussians for encoder and z prior) has nice closed-form solution

$p_\theta(z|x)$ is intractable and we can’t compute this term – but it is always $\geq 0$
VAE Optimization

\[
\log p_\theta(x^{(i)}) = \mathbb{E}_{z \sim q_\phi(z|x^{(i)})} \left[ \log p_\theta(x^{(i)}) \right] \\
= \mathbb{E}_z \left[ \log \frac{p_\theta(x^{(i)}|z)p_\theta(z)}{p_\theta(z|x^{(i)})} \right] \\
= \mathbb{E}_z \left[ \log \frac{p_\theta(x^{(i)}|z)p_\theta(z)}{p_\theta(z|x^{(i)})} \frac{q_\phi(z|x^{(i)})}{q_\phi(z|x^{(i)})} \right] \\
= \mathbb{E}_z \left[ \log p_\theta(x^{(i)}|z) \right] - \mathbb{E}_z \left[ \log \frac{q_\phi(z|x^{(i)})}{p_\theta(z)} \right] + \mathbb{E}_z \left[ \log \frac{q_\phi(z|x^{(i)})}{p_\theta(z|x^{(i)})} \right] \\
= \mathbb{E}_z \left[ \log p_\theta(x^{(i)}|z) \right] - D_{KL}(q_\phi(z|x^{(i)})||p_\theta(z)) + D_{KL}(q_\phi(z|x^{(i)})||p_\theta(z|x^{(i)}))
\]

This KL term (between Gaussians for encoder and z prior) has nice closed-form solution. But it is intractable and we can't compute this term – but it is always \( \geq 0 \)

Decoder network estimates \( p_\theta(x|z) \) through sampling.
$$\log p_\theta(x^{(i)}) = \mathbb{E}_{z \sim q_\phi(z|x^{(i)})} \left[ \log p_\theta(x^{(i)}) \right] \quad p_\theta(x^{(i)}) \text{ does not depend on } z$$

$$= \mathbb{E}_z \left[ \log \frac{p_\theta(x^{(i)}|z)p_\theta(z)}{p_\theta(z|x^{(i)})} \right] \quad \text{Bayes’ rule}$$

$$= \mathbb{E}_z \left[ \log \frac{p_\theta(x^{(i)}|z)p_\theta(z)}{p_\theta(z|x^{(i)})} \frac{q_\phi(z|x^{(i)})}{q_\phi(z|x^{(i)})} \right] \quad \text{Mul. and div. by same constant}$$

$$= \mathbb{E}_z \left[ \log p_\theta(x^{(i)}|z) \right] - \mathbb{E}_z \left[ \log \frac{q_\phi(z|x^{(i)})}{p_\theta(z)} \right] + \mathbb{E}_z \left[ \log \frac{q_\phi(z|x^{(i)})}{p_\theta(z|x^{(i)})} \right]$$

$$= \mathbb{E}_z \left[ \log p_\theta(x^{(i)}|z) \right] - D_{KL}(q_\phi(z|x^{(i)})||p_\theta(z)) + D_{KL}(q_\phi(z|x^{(i)})||p_\theta(z|x^{(i)}))$$

Decoder network estimates $p_\theta(x|z)$ through sampling

This KL term (between Gaussians for encoder and $z$ prior) has nice closed-form solution $p_\theta(z|x)$ is intractable and we can’t compute this term – but it is always $\geq 0$
\[
\log p_{\theta}(x^{(i)}) = \mathbb{E}_{z \sim q_{\phi}(z|x^{(i)})} \left[ \log p_{\theta}(x^{(i)}) \right] \\
= \mathbb{E}_z \left[ \log \frac{p_{\theta}(x^{(i)}|z)p_{\theta}(z)}{p_{\theta}(z|x^{(i)})} \right] \\
= \mathbb{E}_z \left[ \log \frac{p_{\theta}(x^{(i)}|z)p_{\theta}(z)}{p_{\theta}(z|x^{(i)})} \frac{q_{\phi}(z|x^{(i)})}{q_{\phi}(z|x^{(i)})} \right] \\
= \mathbb{E}_z \left[ \log p_{\theta}(x^{(i)}|z) \right] - \mathbb{E}_z \left[ \log \frac{q_{\phi}(z|x^{(i)})}{p_{\theta}(z)} \right] + \mathbb{E}_z \left[ \log \frac{q_{\phi}(z|x^{(i)})}{p_{\theta}(z|x^{(i)})} \right]
\]

This KL term (between Gaussians for encoder and z prior) has nice closed-form solution. \( p_{\theta}(z|x) \) is intractable and we can’t compute this term – but it is always \( \geq 0 \).

Decoder network estimates \( p_{\theta}(x|z) \) through sampling.
VAE Optimization

Overall we developed Evidence Lower BOund (ELBO):

$$\log p_\theta(x^{(i)}) \geq \mathbb{E}_z \left[ \log p_\theta(x^{(i)}|z) \right] - D_{KL}(q_\phi(z|x^{(i)})||p_\theta(z))$$

Define the loss

$$\ell(x^{(i)}, \theta, \phi) = \mathbb{E}_z \left[ \log p_\theta(x^{(i)}|z) \right] - D_{KL}(q_\phi(z|x^{(i)})||p_\theta(z))$$

The first term is the reconstruction loss. The second term direct the posterior distribution close to prior, and it can be thought of a regularization.

During training we minimize this loss for both $\theta$ and $\phi$. 

The second term $D_{KL}(q_\phi(z|x^{(i)})||p_\theta(z))$ is the KL divergence between two multivariate Gaussians. It can be computed analytically as follows, where we denoted $q_\phi(z|x^{(i)}) \sim \mathcal{N}(\mu_q, \Sigma_q)$ and $p_\theta(z) \sim \mathcal{N}(\mu_p, \Sigma_p)$. The expectation is with respect to $z \sim q_\phi(z|x^{(i)})$, and $D$ is the dimension of $z$:

$$D_{KL}(q||p) = \mathbb{E}_q [\log q - \log p]$$

$$= \mathbb{E} \left[ \frac{1}{2} \log \frac{|\Sigma_p|}{|\Sigma_q|} - \frac{1}{2} (z - \mu_q)^\top \Sigma_q^{-1} (z - \mu_q) + \frac{1}{2} (z - \mu_p)^\top \Sigma_p^{-1} (z - \mu_p) \right]$$

$$= \frac{1}{2} \log \frac{|\Sigma_p|}{|\Sigma_q|} - \text{tr} \left\{ \mathbb{E} \left[ (z - \mu_q)(z - \mu_q)^\top \Sigma_q^{-1} \right] \right\} + \mathbb{E} \left[ \frac{1}{2} (z - \mu_p)^\top \Sigma_p^{-1} (z - \mu_p) \right]$$

$$= \frac{1}{2} \log \frac{|\Sigma_p|}{|\Sigma_q|} - \text{tr} \{ I_D \} + \frac{1}{2} (\mu_q - \mu_p)^\top \Sigma_p^{-1} (\mu_q - \mu_p) + \frac{1}{2} \text{tr} \{ \Sigma_p^{-1} \Sigma_q \}$$

$$= \frac{1}{2} \left[ \log \frac{|\Sigma_p|}{|\Sigma_q|} - D + \text{tr} \{ \Sigma_p^{-1} \Sigma_q \} + (\mu_q - \mu_p)^\top \Sigma_p^{-1} (\mu_q - \mu_p) \right]$$
VAE Optimization – re-parametrization of the sampling layer

Another concern is how to propagate the gradients when $z$ is a sampled variable:

Solution: re-parametrization of $z$. For example, when $D = 1$ then $z \sim \mathcal{N}(\mu, \sigma^2)$. We can represent it as $z = \mu + \sigma \odot \epsilon$, where $\epsilon \sim \mathcal{N}(0, 1)$. 
When \( z \sim \mathcal{N}(\mu, \sigma^2) \). We can re-parametrize it as \( z = \mu + \sigma \odot \epsilon \), where \( \epsilon \sim \mathcal{N}(0, 1) \):
VAE Training

During training, for every minibatch of input data: compute this forward pass, and then backprop.
Image generation using VAE

Use decoder network. Now sample $z$ from prior $\mathcal{N}(0, I)$.

- Sample $x|z$ from $x|z \sim \mathcal{N}(\mu_{x|z}, \Sigma_{x|z})$.
- Decoder: $p_\theta(x|z)$.
- Sample $z$ from $z \sim \mathcal{N}(0, I)$. 

\[ \hat{X} \]
Image generation examples

Vary $z_1$

Vary $z_2$
Image generation examples

Vary $z_1$ (degree of smile)

Vary $z_2$ (head pose)
Image generation examples

32 x 32 CIFAR-10 (Dirk Kingma et al. 2016)
Image generation examples

Labeled faces in the wild (Anders Larsen et al. 2017)
VAE - Summary

Probabilistic spin to traditional autoencoders, which allows generating data
Defines an intractable density and solves it by optimizing a (variational) lower bound

**Pros:**
- Principled approach to generative models
- Allows inference of $q(z|x)$, can be useful feature representation for other tasks

**Cons:**
- Maximizes lower bound of likelihood: okay, but not as good evaluation as PixelRNN/PixelCNN
- Samples blurrier and lower quality compared to state-of-the-art (GANs)

**Active areas of research:**
- More flexible approximations, e.g. richer approximate posterior instead of diagonal Gaussian, e.g., Gaussian Mixture Models (GMMs)
- Incorporating structure in latent variables, e.g., Categorical Distributions