Learning Deep Generative Models

Inference & Representation
Lecture 12

Rahul G. Krishnan

Fall 2015

NYU

Outline

Outline

1. Introduction
   - Variational Bound
   - Summary
2. Variational Inference
   - Latent Dirichlet Allocation
   - Learning LDA
   - Stochastic Variational Inference
3. Deep Generative Models
   - Bayesian Networks & Deep-Learning
   - Learning
   - Summary of DGMs
4. Summary

Overview of Lecture

1. Review mathematical concepts: Jensen's Inequality and the Maximum Likelihood (ML) principle
2. Learning as Optimization : Maximizing the Evidence Lower Bound (ELBO)
3. Learning in LDA
4. Stochastic Variational Inference
5. Learning Deep Generative Models
6. Summarize

Recap

- Jensen's Inequality: For concave f, we have

$$f(\mathbb{E}[X]) \geq \mathbb{E}[f(X)]$$

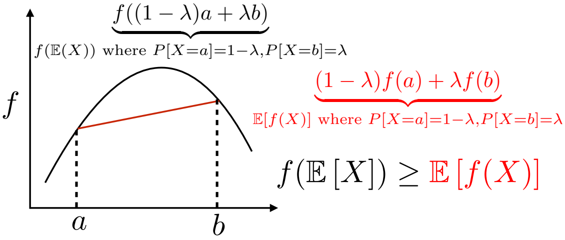
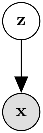


Figure: Jensen's Inequality

Recap

- We assume that for $\mathcal{D} = \{x_1, \ldots, x_N\}$, $x_i \sim p(x)$ i.i.d
- We hypothesize a model (with parameters θ) for how the data is generated
- The Maximum Likelihood Principle: $\max_\theta p(\mathcal{D};\theta) = \prod_{i=1}^N p(x_i;\theta)$
- Typically work with the log probability: i.e $\max_\theta \sum_{i=1}^N \log p(x_i;\theta)$

A simple Bayesian Network



- Lets start with a very simple generative model for our data
- We assume that the data is generated i.i.d as:

$$z \sim p(z) \qquad x \sim p(x|z)$$

- z is latent/hidden and x is observed

Bounding the Marginal Likelihood

- Log-Likelihood of a single datapoint $x \in \mathcal{D}$ under the model: $\log p(x; \theta)$
- Important: Assume $\exists q(z; \phi)$, (variational approximation)

$$
\begin{aligned}
\log p(x) &= \log \int_z p(x, z) \text{ (Multiply and divide by } q(z)) \\
&= \log \int_z \frac{q(z) p(x, z)}{q(z)} = \log \mathbb{E}_{z \sim q(z)} \left[\frac{p(x, z)}{q(z)} \right] \text{ (By Jensen's Inequality)} \\
&\geq \int_z q(z) \log \frac{p(x, z)}{q(z)} = \mathcal{L}(x; \theta, \phi) \\
&= \underbrace{\mathbb{E}_{q(z)} [\log p(x, z)]}_{\text{Expectation of Joint distribution}} + \underbrace{\mathrm{H}(q(z))}_{\text{Entropy}}
\end{aligned}
$$

Evidence Lower BOund (ELBO)/Variational Bound

- When is the lower bound tight?
- Look at: function - lower bound

$$\log p(x;\theta) - \mathcal{L}(x;\theta,\phi)$$

$$\begin{aligned}
&\log p(x) - \int_z q(z) \log \frac{p(x,z)}{q(z)} \\
&= \int_z q(z) \log p(x) - \int_z q(z) \log \frac{p(x,z)}{q(z)} \\
&= \int_z q(z) \log \frac{q(z)p(x)}{p(x,z)} \\
&= \mathrm{KL}(q(z;\phi)||p(z|x))
\end{aligned}$$

Evidence Lower BOund (ELBO)/Variational Bound

- We assumed the existance of $q(z;\phi)$
- What we just showed is that:

Key Point

The optimal $q(z;\phi)$ corresponds to the one that realizes $\mathrm{KL}(q(z;\phi)||p(z|x)) = 0 \iff q(z;\phi) = p(z|x)$

Evidence Lower BOund (ELBO)/Variational Bound

- In order to estimate the liklihood of the entire dataset $\mathcal{D}$, we need $\sum_{i=1}^{N} \log p(x_i; \theta)$
- Summing up over datapoints we get:

$$\max_{\theta} \sum_{i=1}^{N} \log p(x_i; \theta) \geq \max_{\theta, \phi_1, \ldots, \phi_N} \underbrace{\sum_{i=1}^{N} \mathcal{L}(x_i; \theta, \phi_i)}_{ELBO}$$

- Note that we use a *different* ϕ_i for every data point

Outline

Summary

Learning as Optimization

Variational learning turns learning into an optimization problem, namely:

$$\max_{\theta,\phi_1,\ldots,\phi_N} \sum_{i=1}^{N} \mathcal{L}(x_i;\theta,\phi_i)$$

Summary

Optimal q

The optimal $q(z;\phi)$ used in the bound corresponds to the intractable posterior distribution $p(z|x)$

Summary

Approximating the Posterior

The better $q(z;\phi)$ can approximate the posterior, the smaller $KL(q(z;\phi)||p(z|x))$ we can achieve, the closer ELBO will be to $\log p(x;\theta)$

Outline

Generative Model

- Latent Dirichlet Allocation (LDA)

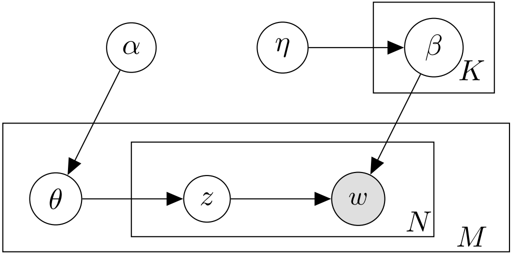


Figure: Generative Model for Latent Dirichlet Allocation

Generative Model

1. Sample global topics $\beta_k \sim \text{Dir}(\eta_k)$
2. For document $d = 1, \ldots, N$
3. Sample $\theta_d \sim \text{Dir}(\alpha)$
4. For each word $m = 1, \ldots, M$
5. Sample topic $z_{dm} \sim \text{Mult}(\theta_d)$
6. Sample word $w_{dm} \sim \text{Mult}(\beta_{z_{dm}})$

- $\mathbb{S}$ denotes the simplex
- V is the vocabulary and K is the number of topics
- $\theta_d \in \mathbb{S}^K$
- $\beta_{z_{dm}} \in \mathbb{S}^V$

Variational Distribution

- w are observed and z, β, θ are latent
- We will perform inference over z, β, θ
- As before, we will assume that there exists a distribution over our latent variables
- We will assume that our distribution factorizes (mean-field assumption)
- Variational Distribution:

$$q(\theta, z, \beta; \Phi) = q(\theta; \gamma) \left(\prod_{n=1}^{N} q(z_n; \phi_n) \right) \left(\prod_{k=1}^{K} q(\beta_k; \lambda_k) \right)$$

- Denote $\Phi = \{\gamma, \phi, \lambda\}$, the parameters of the variational approximation

Homework

- Your next homework assignment involves implementing a mean-field algorithm for inference in LDA
- Assume Topic-Word Probabilities $\beta_{1:K}$ observed and fixed, you won't have to infer these
- Perform inference over θ and z
- The following slides are to give you intuition and understanding on how to derive the updates for inference
- Read Blei *et al.* (2003) (particularly the appendix) for details on derivation

Outline

ELBO Derivation

For a single document, the joint distribution is:

$$\log p(\theta, z, w, \beta; \alpha, \eta)$$
$$= \log \left(\prod_{k=1}^{K} p(\beta_k; \eta) \prod_{d=1}^{D} \left[p(\theta_d; \alpha) \prod_{n=1}^{N} p(z_{dn}|\theta_d) p(w_n|z_{dn}, \beta) \right] \right)$$

ELBO Derivation

- Denote $\Phi = \{\gamma, \phi, \lambda\}$, the parameters of the variational approximation

For a single document, the bound on the log likelihood is:

$$\log p(w; \alpha, \eta) \geq \underbrace{\mathbb{E}_{q(\theta,z,\beta;\Phi)}\left[\log p(\theta, z, w, \beta; \alpha, \eta)\right] + H(q(\theta, z, \beta; \Phi))}_{\mathcal{L}(w;\alpha,\eta,\Phi)}$$

ELBO Derivation

- *Assumption:* The posterior distribution fully factorizes

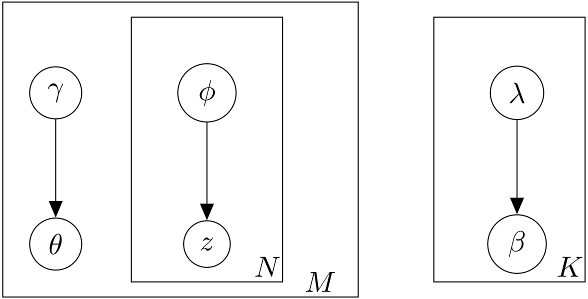


Figure: Plate model for Mean Field Approximation to LDA

ELBO Derivation

What $q(\theta, z, \beta; \Phi)$ do we use?

- Mean-field assumption:
  $q(\theta, z, \beta; \Phi) = q(\theta; \gamma) \left(\prod_{n=1}^{N} q(z_n; \phi_n) \right) \left(\prod_{k=1}^{K} q(\beta_k; \lambda_k) \right)$
- θ is a multinomial therefore γ is a Dirichlet parameter, likewise for β_k
- Each $z_n \in \{1, \ldots, K\}$, therefore ϕ_n represents the parameters of a Multinomial distribution

Variational EM

$$\mathcal{L}(w;\alpha,\eta,\Phi) = \mathbb{E}_{q(\theta,z,\beta;\Phi)}\left[\log p(\theta,z,w,\beta;\alpha,\eta)\right] + H(q(\theta,z,\beta;\Phi))$$

- $\mathcal{L}$ is a function of α, η, the parameters of the model and $\Phi = \{\gamma, \phi, \lambda\}$, the parameters of approximation to the posterior
- Variational EM
- Fix α, η. Approximate $\gamma^*, \phi^*, \lambda^*$ (mean-field inference)
- Fix $\gamma^*, \phi^*, \lambda^*$, Update α, η
- Unlike EM, variational EM not guaranteed to reach a local maximizer of $\mathcal{L}$

Variational EM

Deriving updates for Variational Inference in HW

1. See Appendix in Blei *et al.* (2003)
2. Expand the bound $\mathcal{L}$ using the factorization of the joint distribution and the form of the mean-field posterior
3. Isolate terms in $\mathcal{L}$ corresponding to variational parameters γ, ϕ.
4. Find γ^*, ϕ^* that maximize $\mathcal{L}(\gamma), \mathcal{L}(\phi)$

Outline

Variational Inference

Let us focus just on variational inference (E-step) for the moment.

- ϕ_{dn}^{k}: probability that word n in document d has topic k
- γ_d: posterior Dirichlet parameter for document d
- λ_k: posterior Dirichlet parameter for topic k

Variational Inference

Lets recall what the variational distribution looked like

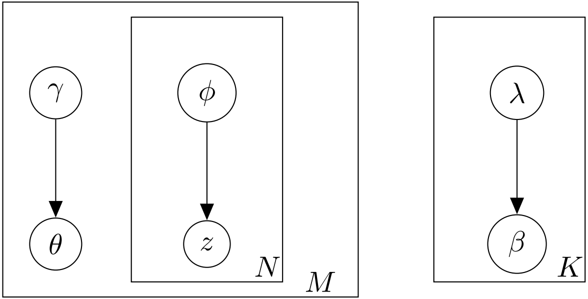


Figure: Plate model for Mean Field Approximation to LDA

Variational Inference

1. For a single document d
2. Repeat till convergence:
3. Update ϕ_{dn}^k for $n \in \{1, \ldots, N\}, k \in \{1, \ldots, K\}$
4. Update γ_d

- This process yields the *local* posterior parameters
- ϕ_{dn}^k gives us the probability that the nth work was drawn from topic k
- γ_d gives us a Dirichlet parameter. Samples from this distribution give us an estimate of the topic proportions in the document

Variational Inference

- We just saw the updates to the local variational parameters (local to every document)
- What about the update to λ, the global variational parameter (shared across all documents)

Variational Inference

The posterior over β uses local posterior parameters from every document

1. For all documents $d = 1, \ldots, M$, repeat:
2. Update ϕ_{dn}^k for $n \in \{1, \ldots, N\}, k \in \{1, \ldots, K\}$
3. Update γ_d
4. Update $\lambda_k \leftarrow \underbrace{\eta}_{\text{Prior over } \beta_k} + \sum_{d=1}^{D} \sum_{n=1}^{N} \phi_{dn}^k w_{dn}$ for $k = \{1, \ldots, K\}$
5. **The update to λ_k uses ϕ from *every* document in the corpus**

Inefficiencies in the Algorithm

- As M (the number of documents) increases, inference becomes increasingly inefficient
- Step 4 requires you to process the entire dataset before updating λ_k
- Can we do better?

Stochastic Variational Inference

Key Point

Instead of waiting to process the *entire* corpus before updating λ, why don't we replicate the update from a *single* document M times.

SVI Pseudocode

1. for $t = 1, \ldots, T$
2. Sample a document d from the dataset
3. Repeat till convergence:
4. Update ϕ_{dn}^{k} for $n \in \{1, \ldots, N\}, k \in \{1, \ldots, K\}$
5. Update γ_d
6. $$\hat{\lambda}_k \leftarrow \underbrace{\eta}_{\text{Prior over } \beta_k} + \underbrace{M \sum_{n=1}^{N} \phi_{dn}^{k} w_{dn}}_{\text{Multiply the update by } M},$$
   $k = \{1, \ldots, K\}$
7. Set $\lambda^t \leftarrow (1 - \rho_t)\lambda^{t-1} + \rho_t \hat{\lambda}$

- ρ_t is the adaptive learning rate

SVI Pseudocode

- For $k = \{1, \ldots, K\}$:

$$\hat{\lambda}_k \leftarrow \eta + M \sum_{n=1}^{N} \phi_{dn}^{k} w_{dn}$$

- $\hat{\lambda}_k$ is the estimate of the variational parameter
- We update λ^t to be a weighted sum of its previous value and the proposed estimate.

What do we gain?

- Lets us scale up to much larger datasets
- Faster convergence

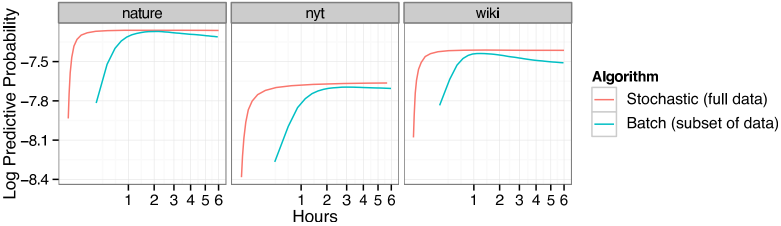
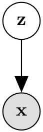


Figure: Per word predictive probability for 100-topic LDA. SVI converges faster than batch variational inference. Taken from Hoffman *et al.* (2013)

Outline

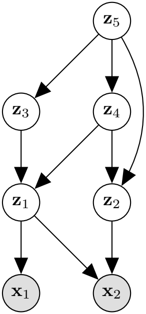
Deep Generative Model

- Can we give an efficient learning algorithm for bayesian networks like this:



Deep Generative Model

- Or deeper latent variable models like this?

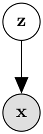


Outline

Outline

- Reset the notation from LDA, we're starting afresh
- First, a simple model to learn the technique, then a more complex latent variable model

Simple Generative Model



$$z \sim p(z) \qquad x \sim p(x|z)$$

- Assume that θ are the parameters of the generative model
- Includes the parameters of the prior $p(z)$ and the conditional $p(x|z)$

New methods for Learning

Based on recent work in learning graphical models (Rezende *et al.* , 2014), (Kingma & Welling, 2013)

- In variational EM, every point in our dataset had an associated set of posterior parameters

New Methods for Learning

- We'll use a *single* variational approximation for *all* datapoints

New Methods for Learning

- We'll use a *single* variational approximation for *all* datapoints
- To do that, we will *learn* a **conditional, parametric** function

New Methods for Learning

- We'll use a *single* variational approximation for *all* datapoints
- To do that, we will *learn* a **conditional, parametric** function
- The output of this *function* will be the parameters of the variational distribution

New Methods for Learning

- We'll use a *single* variational approximation for *all* datapoints
- To do that, we will *learn* a **conditional, parametric** function
- The output of this *function* will be the parameters of the variational distribution
- We will approximate the posterior with this distribution

New Methods for Learning

- We'll use a *single* variational approximation for *all* datapoints
- To do that, we will *learn* a **conditional, parametric** function
- The output of this *function* will be the parameters of the variational distribution
- We will approximate the posterior with this distribution
- So previously the $q(z)$ we assumed will now be $q_\phi(z|x)$

New Methods for Learning

- We'll use a *single* variational approximation for *all* datapoints
- To do that, we will *learn* a **conditional, parametric** function
- The output of this *function* will be the parameters of the variational distribution
- We will approximate the posterior with this distribution
- So previously the $q(z)$ we assumed will now be $q_\phi(z|x)$
- For every x, we get a different set of posterior parameters

New Methods for Learning

- We'll use a *single* variational approximation for *all* datapoints
- To do that, we will *learn* a **conditional, parametric** function
- The output of this *function* will be the parameters of the variational distribution
- We will approximate the posterior with this distribution
- So previously the $q(z)$ we assumed will now be $q_\phi(z|x)$
- For every x, we get a different set of posterior parameters
- Optimization Problem: $\max_{\phi,\theta} \sum_{i=1}^{N} \mathcal{L}(x_i;\theta,\phi)$

ELBO

$$
\begin{aligned}
\mathcal{L}(x;\theta,\phi) &= \int_z q_\phi(z|x) \log \frac{p_\theta(x,z)}{q_\phi(z|x)} \\
&= \int_z q_\phi(z|x) \log p_\theta(x|z) - \int_z q_\phi(z|x) \log \frac{q_\phi(z|x)}{p_\theta(z)} \\
&= \underbrace{\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x,z)]}_{\text{Expectation of Joint Distribution}} + \underbrace{\mathrm{H}(q_\phi(z|x))}_{\text{Entropy of } q_\phi(z|x)}
\end{aligned} \tag{1}
$$

Key Points

Parametric $q(z|x;\phi)$

We're going to learn a conditional parametric approximation $q(z|x;\phi)$ to $p(z|x)$, the posterior distribution.

Shared ϕ

Learning a conditional model, $q(z|x;\phi)$ where ϕ will be shared for all x

Gradient Ascent

We're going to perform joint optimization of θ, ϕ on $\max_{\theta,\phi} \sum_{i=1}^{N} \mathcal{L}(x_i;\theta,\phi)$

Plate Model

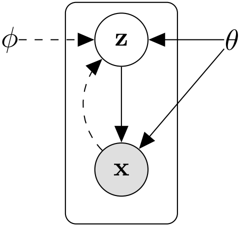


Figure: Learning DGMs

Use Stochastic Gradient Ascent to learn this model

Putting it all together

$$\mathcal{L}(x;\theta,\phi) = \mathbb{E}_{z\sim q_\phi(z|x)}\left[\log p_\theta(x,z)\right] + \mathrm{H}(q_\phi(z|x))$$

- Step 1: Sample a datapoint from dataset: $x \sim \mathcal{D}$
- Posterior Inference: Evaluate $q_\phi(z|x)$ to obtain parameters of posterior
- Step 2: Sample $z_{1:K} \sim q_\phi(z|x)$

Putting it all together

$$\mathcal{L}(x;\theta,\phi) = \underbrace{\mathbb{E}_{z\sim q_\phi(z|x)}\left[\log p_\theta(x,z)\right]}_{\text{(a)}} + \underbrace{\mathrm{H}(q_\phi(z|x))}_{(b)}$$

- Step 3: Estimate ELBO
- Approximate (a) as a Monte Carlo estimate over K samples
- (b) typically an analytic function of ϕ

Putting it all together

Compute gradients of $\mathcal{L}(x;\theta,\phi) = \underbrace{\mathbb{E}_{z\sim q_\phi(z|x)}\left[\log p_\theta(x,z)\right]}_{\text{(a)}} + \underbrace{\mathrm{H}(q_\phi(z|x))}_{(b)}$

- Step 4: Compute gradients: $\nabla_\theta \mathcal{L}(x;\theta,\phi), \nabla_\phi \mathcal{L}(x;\theta,\phi)$
- First look at gradients with respect to θ
- $\nabla_\theta \mathcal{L}(x;\theta,\phi) = \mathbb{E}_z\left[\nabla_\theta \log p(x,z;\theta)\right] + \nabla_\theta \mathrm{H}(q_\phi(z|x)||p(z))$
- We approximate these gradients using a Monte-Carlo estimator with the K samples

Putting it all together

Compute gradients of $\mathcal{L}(x;\theta,\phi) = \underbrace{\mathbb{E}_{z\sim q_\phi(z|x)}\left[\log p_\theta(x,z)\right]}_{\text{(a)}} + \underbrace{\mathrm{H}(q_\phi(z|x))}_{(b)}$

- Step 4: Compute gradients: $\nabla_\theta \mathcal{L}(x;\theta,\phi), \nabla_\phi \mathcal{L}(x;\theta,\phi)$
- Now look at gradients with respect to ϕ
- As before, what we would like is to move the gradient into the expectation and approximate it with a Monte-Carlo estimator
- The issue is that the expectation also depends on ϕ

Putting it all together

Recent Work

- What we want: $\nabla \mathbb{E}[f] = \mathbb{E}\left[\nabla \tilde{f}\right]$
- We can write the gradient of an expectation as an expectation of gradients Ranganath *et al.* (2014); Kingma & Welling (2013); Rezende *et al.* (2014)

Putting it all together

Compute gradients of $\mathcal{L}(x;\theta,\phi) = \underbrace{\mathbb{E}_{z\sim q_\phi(z|x)}\left[\log p_\theta(x,z)\right]}_{\text{(a)}} + \underbrace{\mathrm{H}(q_\phi(z|x))}_{(b)}$

- Step 4: Compute gradients: $\nabla_\theta \mathcal{L}(x;\theta,\phi), \nabla_\phi \mathcal{L}(x;\theta,\phi)$
- Write the gradient of an expectation as an expectation of gradients Ranganath *et al.* (2014); Kingma & Welling (2013); Rezende *et al.* (2014)
- We approximate the gradients using a Monte-Carlo estimator with the K samples

Putting it all together

Update θ and ϕ

- Step 5: Update parameters:

$$\theta \leftarrow \theta + \eta_\theta \nabla_\theta \mathcal{L}(x; \theta, \phi)$$

and

$$\phi \leftarrow \phi + \eta_\phi \nabla_\phi \mathcal{L}(x; \theta, \phi)$$

Putting it all together

Pseudocode

- Step 1: Sample a datapoint from dataset: $x \sim \mathcal{D}$
- Step 2: Perform posterior inference: Sample $z_{1:K} \sim q(z|x;\phi)$
- Step 3: Estimate ELBO
- Step 4: Approximate gradients: $\nabla_\theta \mathcal{L}(x;\theta,\phi), \nabla_\phi \mathcal{L}(x;\theta,\phi)$ (Gradients are Monte Carlo estimates over K samples)
- Step 5: Update parameters:

$$\theta \leftarrow \theta + \eta_\theta \nabla_\theta \mathcal{L}(x;\theta,\phi)$$

  and

$$\phi \leftarrow \phi + \eta_\phi \nabla_\phi \mathcal{L}(x;\theta,\phi)$$

- Step 6: Go to Step 1

Gaussian DGMs

- This is a very general framework capable of learning many different kinds of graphical models
- Lets consider a simple set of DGMs is where priors and the conditionals are Gaussian

Assumption on $q(z|x)$

$q(z|x)$

Assume $q(z|x;\phi)$ approximates the posterior with a Gaussian distribution $z \sim \mathcal{N}(\mu(x;\phi), \Sigma(x;\phi))$

- $p(x,z) = p(z)p(x|z)$

$$\mathcal{L}(x;\theta,\phi) = \underbrace{\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x,z)]}_{\text{Function of } \theta,\phi} + \underbrace{\mathrm{H}(q_\phi(z|x))}_{\text{Function of } \phi}$$

- For Multivariate Gaussian distributions of dimension D:

$$H(q_\phi(z|x)) = \frac{1}{2}D\left[1+\log 2\pi\right] + \frac{1}{2}|det\Sigma(x;\phi)|$$

Location and Scale transformations

- We'll need one more tool in our toolbox. This is specific to Gaussian latent variable models.
- In some cases, we can sample from distribution A and transform the samples to appear as if they came from distribution B.
- Easy to see in the univariate Gaussian case
- $z \sim \mathcal{N}(\mu, \sigma^2)$ is equivalent to $z = \mu + \sigma\epsilon$ where $\epsilon \sim \mathcal{N}(0, 1)$
- Therefore:

$$\mathbb{E}_{z\sim\mathcal{N}(\mu,\sigma^2)}\left[f(z)\right] = \mathbb{E}_{\epsilon\sim\mathcal{N}(0,1)}\left[f(\mu + \epsilon\sigma)\right]$$

Gradients of $\mathcal{L}(x;\theta,\phi)$

- Gradients with respect to θ

$$\begin{aligned}\nabla_\theta \mathcal{L} &= \nabla_\theta \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x,z)] + \nabla_\theta H_\phi \\ &= \mathbb{E}_{q_\phi(z|x)}[\nabla_\theta \log p_\theta(x,z)]\end{aligned}$$

- Gradients with respect to ϕ

$$\begin{aligned}&\text{Define } \Sigma_\phi(x) := \mathrm{R}_\phi(x)\mathrm{R}_\phi(x)^T \\ \nabla_\phi \mathcal{L} &= \nabla_\phi \mathbb{E}_{z\sim q_\phi(z|x)}[\log p_\theta(x,z)] + \nabla_\phi \mathrm{H}_\phi \\ &\text{(Using Location and Scale Transformation)} \\ &= \nabla_\phi \mathbb{E}_{\epsilon\sim\mathcal{N}(0;\mathbb{I})}[\log p_\theta(x, \mu_\phi(x) + \mathrm{R}_\phi(x)\epsilon)] + \nabla_\phi \mathrm{H}_\phi \\ &= \mathbb{E}_{\epsilon\sim\mathcal{N}(0;\mathbb{I})}[\nabla_\phi \log p_\theta(x, \mu_\phi(x) + \mathrm{R}_\phi(x)\epsilon)] + \nabla_\phi \mathrm{H}_\phi\end{aligned}$$

Gradients of $\mathcal{L}(x;\theta,\phi)$

The gradients are expectations!
We approximate them with a Monte-Carlo estimate

- Gradients with respect to θ: for $z \sim q_\phi(z|x)$

$$\nabla_\theta \mathcal{L} = \mathbb{E}_{z\sim q_\phi(z|x)}[\nabla_\theta \log p_\theta(x,z)] = \frac{1}{K}\sum_{k=1}^{K}[\nabla_\theta \log p_\theta(x,z_k)]$$

- Gradients with respect to ϕ: for $\epsilon \sim \mathcal{N}(0;\mathbb{I})$

$$\begin{aligned}\nabla_\phi \mathcal{L} &= \mathbb{E}_{\epsilon\sim\mathcal{N}(0;\mathbb{I})}[\nabla_\phi \log p_\theta(x,\mu_\phi(x)+\mathrm{R}_\phi(x)\epsilon)] + \nabla_\phi \mathrm{H}_\phi \\ &= \frac{1}{K}\sum_{k=1}^{K}[\nabla_\phi \log p_\theta(x,\mu_\phi(x)+\mathrm{R}_\phi(x)\epsilon_k)] + \nabla_\phi \mathrm{H}_\phi\end{aligned}$$

Learning: A graphical view

- Lets see a pictoral representation of this process for a single data point x

Learning: A graphical view

- For a given datapoint x, do inference to infer the parameters that form the approximation to the posterior
- At this point, we can evaluate the entropy $H(q_\phi(z|x))$

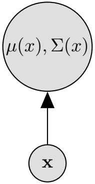


Figure: Step 1 & 2: Sampling datapoint & inferring $\mu(x), \Sigma(x)$

Learning: A graphical view

- Sample $z_{1:K}$ from posterior $z_{1:K} \sim \mathcal{N}(\mu(x), \Sigma(x))$
- Now, we have a fully observed bayesian network

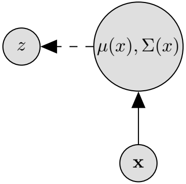


Figure: Step 2: Sampling z

Learning: A graphical view

- Evaluate ELBO, ie.
  $\mathcal{L}(x;\theta,\phi) = \mathbb{E}_{z\sim q_\phi(z|x)}\left[\log p_\theta(x,z)\right] + \mathrm{H}(q_\phi(z|x))$

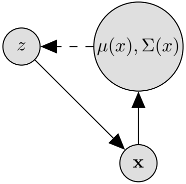


Figure: Step 3: Evaluating ELBO

Learning: A graphical view

- Compute $\nabla_\theta \mathcal{L}(x;\theta,\phi) = \mathbb{E}_{z\sim q_\phi(z|x)}\left[\nabla_\theta \log_\theta p(x,z)\right]$

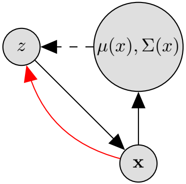


Figure: Step 4: Compute Gradients

Learning: A graphical view

- Use the Location and Scale Transformations:
- Compute

$$\nabla_\phi \mathcal{L}(x;\theta,\phi) = \mathbb{E}_{\epsilon\sim\mathcal{N}(0;\mathbb{I})}\left[\nabla_\phi \log p(x, \mu(x;\phi) + \mathrm{R}(x;\phi)\epsilon)\right] + \nabla_\phi \mathrm{H}(q_\phi(z|x))$$

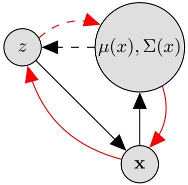
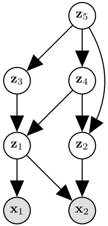


Figure: Step 4: Compute Gradients

Easy to Learn

- Specific forms of these models also go by the name Variational Autoencoders.
- There are ways to learn non-Gaussian graphical models (not covered)
- Easily implemented in popular libraries such as Torch/Theano!
- There is a Torch implementation you can play around with in: `https://github.com/clinicalml/dgm`

Combining Deep Learning with Graphical Models



- We haven't yet talked about the parameterizations of the conditional distributions (in both p and q)
- One possibility is to use a neural network. Results in a powerful, highly non-linear transformation

Generating Digits from MNIST

Figure: Generating MNIST Digits (Kingma & Welling, 2013)

Generating Faces

- With a DGM trained on images of faces, lets look at how the samples vary as we move around in the latent dimension
- Traversing the face manifold (Radford, 2015)
- Morphing Faces (Dumoulin, 2015)
- Many more such examples!

Outline

1. Introduction
   - Variational Bound
   - Summary
2. Variational Inference
   - Latent Dirichlet Allocation
   - Learning LDA
   - Stochastic Variational Inference
3. Deep Generative Models
   - Bayesian Networks & Deep-Learning
   - Learning
   - Summary of DGMs
4. Summary

Limitations of DGMs

- New methods allow us to learn a broad and powerful class of generative models.
  1. Can be tricky to learn.
  2. No theoretical guarantees on the optimization problem.
  3. Interpretability: Does z really *mean* anything? Can you & I put a name to the quantity it represents?

Summary

- Theres a lot more to do! Active area of research.
  - **Probabilistic Programming:** If I can write out my graphical model, can I automatically learn it using techniques from stochastic variational inference?
  - **Tightening the bound on** $\log p(x)$**:** How can we form better and more complex approximations to the posterior distributions?

References I

Blei, David M., Ng, Andrew Y., & Jordan, Michael I. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research*.

Dumoulin, Vincent. 2015. Morphing Faces. `http://vdumoulin.github.io/morphing_faces/online_demo.html`.

Hoffman, Matthew D., Blei, David M., Wang, Chong, & Paisley, John William. 2013. Stochastic variational inference. *Journal of Machine Learning Research*.

Kingma, Diederik P, & Welling, Max. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Radford, Alec. 2015. Morphing Faces. `https://www.youtube.com/watch?v=XNZIN7Jh3Sg`.

References II

Ranganath, Rajesh, Gerrish, Sean, & Blei, David M. 2014. Black Box Variational Inference. *In: Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics, AISTATS 2014, Reykjavik, Iceland, April 22-25, 2014.*

Rezende, Danilo Jimenez, Mohamed, Shakir, & Wierstra, Daan. 2014. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082.*