Variational Inference and Generative Models

CS 330
Course Reminders

Homework 3 due Monday next week.

Tutorial session on Thursday 4:30 pm

Be careful Azure usage — turning off machines when you are not using them!
This Week

A Bayesian perspective on meta-learning

Today: Approximate Bayesian inference via variational inference
Plan for Today

1. Latent variable models
2. Variational inference
3. Amortized variational inference
4. Example latent variables models

Goals
- Understand latent variable models in deep learning
- Understand how to use (amortized) variational inference

Part of (optional) Homework 4
Probabilistic models

\[ p(x) \]

\[ p(y|x) \]

Most commonly:
- probability values of discrete categorical distribution
- mean and variance of a Gaussian

But it could be other distributions!
How do we train probabilistic models?

the model: \( p_\theta(x) \)

the data: \( \mathcal{D} = \{x_1, x_2, x_3, \ldots, x_N\} \)

maximum likelihood fit:

\[
\theta \leftarrow \arg \max_\theta \frac{1}{N} \sum_i \log p_\theta(x_i)
\]

Easy to evaluate & differentiate for categorical or Gaussian distributions.

i.e. cross-entropy, MSE losses

Goal: Can we model and train more complex distributions?
When might we want more complex distributions?

- **generative models** of images, text, video, or other data
- represent **uncertainty over labels** (e.g. ambiguity arising from limited data, partial observability)
- represent **uncertainty over functions**

“HD Video: Riding a horse in the park at sunrise”
Meta-learning methods represent a deterministic $p(\phi_i|\mathcal{D}_i^{tr}, \theta)$ (i.e. a point estimate)

Why/when is this a problem?

Few-shot learning problems may be *ambiguous*. (even with prior)

Can we learn to *generate hypotheses* about the underlying function? 
- safety-critical few-shot learning 
  (e.g. medical imaging)  
- learning to *actively learn*  
- learning to *explore* in meta-RL

Important for:

Active learning w/ meta-learning: Woodward & Finn ’16, Konyushkova et al. ’17, Bachman et al. ’17

Goal: Can we model and train complex distributions?
Latent variable models: examples

$p(x)$

e.g. mixture model
Latent variable models: examples

\[ p(x) = \sum_z p(x|z)p(z) \]

- e.g. Gaussian
- mixture element
- e.g. mixture model
Latent variable models: examples

\[ p(x) = \sum_z p(x|z)p(z) \]

mixture element

e.g. Gaussian

\[ p(y|x) = \sum_z p(y|x, z)p(z|x) \]

e.g. mixture density network

\[ w_1, \mu_1, \sum_1, \ldots, w_N, \mu_N, \sigma_N \]

length of paper
Latent variable models in general

\[ p(x) = \int p(x|z)p(z)dz \]

💡 Key idea: represent complex distribution by composing two simple distributions
Latent variable models in general

\[ p(x) = \int p(x|z)p(z)dz \]

“easy” distribution (e.g., conditional Gaussian)

“easy” distribution (e.g., Gaussian)

💡 Key idea: represent complex distribution by composing two simple distributions

Questions:
1. Once trained, how do you generate a sample from \( p(x) \)?
2. How do you evaluate the likelihood of a given sample \( x_i \)?
How do we train latent variable models?

the model: \( p_\theta(x) \)

the data: \( \mathcal{D} = \{x_1, x_2, x_3, \ldots, x_N\} \)

maximum likelihood fit:

\[
\theta \leftarrow \arg \max_\theta \frac{1}{N} \sum_i \log p_\theta(x_i)
\]

\[
p(x) = \int p(x|z)p(z)dz
\]

\[
\theta \leftarrow \arg \max_\theta \frac{1}{N} \sum_i \log \left( \int p_\theta(x_i|z)p(z)dz \right)
\]

completely intractable
Flavors of Deep Latent Variable Models

Use latent variables:
- generative adversarial networks (GANs)
- variational autoencoders (VAEs)
- normalizing flow models
- diffusion models

Do not use latent variables:
- autoregressive models

All differ in how they are trained.

(recall generative pre-training lecture)
Variational Inference

A. Formulate a lower bound on the log likelihood objective.
B. Check how tight the bound is.
C. Variational inference -> Amortized variational inference
D. How to optimize
Estimating the log-likelihood

alternative: \textit{expected} log-likelihood:

\[ \theta \leftarrow \arg \max_{\theta} \frac{1}{N} \sum_{i} E_{z \sim p(z|x_i)} \left[ \log p_{\theta}(x_i, z) \right] \]

but... how do we calculate \( p(z|x_i) \)?

intuition: “guess” most likely \( z \) given \( x_i \), and pretend it’s the right one
...but there are many possible values of \( z \)
so use the distribution \( p(z|x_i) \)
The variational approximation

but... how do we calculate \( p(z|x_i) \)?

can bound \( \log p(x_i) \)!

\[
\log p(x_i) = \log \int_z p(x_i|z)p(z)
= \log \int_z p(x_i|z)p(z) \frac{q_i(z)}{q_i(z)}
= \log E_{z \sim q_i(z)} \left[ \frac{p(x_i|z)p(z)}{q_i(z)} \right]
\]

what if we approximate with \( q_i(z) = \mathcal{N}(\mu_i, \sigma_i) \)
The variational approximation

but... how do we calculate $p(z|x_i)$?

can bound $\log p(x_i)$!

$$\log p(x_i) = \log \int_z p(x_i|z)p(z)$$

$$= \log \int_z p(x_i|z)p(z) \frac{q_i(z)}{q_i(z)}$$

$$= \log E_{z \sim q_i(z)} \left[ \frac{p(x_i|z)p(z)}{q_i(z)} \right]$$

maximizing this maximizes $\log p(x_i)$

$$\geq E_{z \sim q_i(z)} \left[ \log \frac{p(x_i|z)p(z)}{q_i(z)} \right] = E_{z \sim q_i(z)} [\log p(x_i|z) + \log p(z)] + H_{q_i(z)}(z)[\log q_i(z)]$$

"evidence lower bound" (ELBO)
A brief aside...

Entropy:

\[
\mathcal{H}(p) = -E_{x \sim p(x)}[\log p(x)] = - \int x p(x) \log p(x) dx
\]

Intuition 1: how random is the random variable?
Intuition 2: how large is the log probability in expectation under itself

what do we expect this to do?

\[
E_{z \sim q_i(z)}[\log p(x_i|z) + \log p(z)] + \mathcal{H}(q_i)
\]

this maximizes the first part

this also maximizes the second part (makes it as wide as possible)
A brief aside...

KL-Divergence:

\[ D_{KL}(q \parallel p) = E_{x \sim q(x)} \left[ \log \frac{q(x)}{p(x)} \right] = E_{x \sim q(x)} \left[ \log q(x) \right] - E_{x \sim q(x)} \left[ \log p(x) \right] = -E_{x \sim q(x)} \left[ \log p(x) \right] - \mathcal{H}(q) \]

Intuition 1: how different are two distributions? e.g. when \( q = p \), KL divergence is 0

Intuition 2: how small is the expected log probability of one distribution under another, minus entropy?

why entropy?

![Diagram](image)
this maximizes the first part
this also maximizes the second part
(makes it as wide as possible)
How tight is the lower bound?

\[ \mathcal{L}_i(p, q_i) \quad \text{“evidence lower bound” (ELBO)} \]

\[ \log p(x_i) \geq E_{z \sim q_i(z)} \left[ \log p(x_i | z) + \log p(z) \right] + \mathcal{H}(q_i) \]

what makes a good \( q_i(z) \)?

intuition: \( q_i(z) \) should approximate \( p(z | x_i) \)

approximate in what sense?

compare in terms of KL-divergence: \( D_{KL}(q_i(z) || p(z|x)) \)

why?

\[
D_{KL}(q_i(z) || p(z|x_i)) = E_{z \sim q_i(z)} \left[ \log \frac{q_i(z)}{p(z|x_i)} \right] = E_{z \sim q_i(z)} \left[ \log \frac{q_i(z)p(x_i)}{p(x_i, z)} \right] \\
= -E_{z \sim q_i(z)} \left[ \log p(x_i | z) + \log p(z) \right] + E_{z \sim q_i(z)} \left[ \log q_i(z) \right] + E_{z \sim q_i(z)} \left[ \log p(x_i) \right] \\
= -E_{z \sim q_i(z)} \left[ \log p(x_i | z) + \log p(z) \right] - \mathcal{H}(q_i) + \log p(x_i) \\
= -\mathcal{L}_i(p, q_i) + \log p(x_i) \\
\log p(x_i) = D_{KL}(q_i(z) || p(z|x_i)) + \mathcal{L}_i(p, q_i) \quad \text{Note 1: If KL divergence is 0, then bound is tight.}
\]

\[ \log p(x_i) \geq \mathcal{L}_i(p, q_i) \]
How tight is the lower bound?

\[ \mathcal{L}_i(p, q_i) \quad \text{“evidence lower bound” (ELBO)} \]

\[
\log p(x_i) \geq E_{z \sim q_i(z)}[\log p(x_i | z) + \log p(z)] + \mathcal{H}(q_i)
\]

what makes a good \( q_i(z) \)?

approximate in what sense?

why?

\[
D_{KL}(q_i(z) \| p(z|x_i)) = -\mathcal{L}_i(p, q_i) + \log p(x_i)
\]

Note 2: Maximizing \( L(p, q_i) \) w.r.t. \( q_i \) minimizes the KL divergence.

Note 1: If KL divergence is 0, then bound is tight.

\[
\log p(x_i) = D_{KL}(q_i(z) \| p(z|x_i)) + \mathcal{L}_i(p, q_i)
\]

Optimization objective: \[
\max_{\theta, q_i} \frac{1}{N} \sum_i \mathcal{L}_i(p_\theta, q_i)
\]
Optimizing the ELBO

\[ \mathcal{L}_i(p, q_i) \] “evidence lower bound” (ELBO)

\[
\log p(x_i) \geq \mathbb{E}_{z \sim q_i(z)} \left[ \log p_\theta(x_i | z) + \log p(z) \right] + \mathcal{H}(q_i)
\]

\[
\theta \leftarrow \arg \max_\theta \frac{1}{N} \sum_i \log p_\theta(x_i)
\]

for each \( x_i \) (or mini-batch):

- calculate \( \nabla_\theta \mathcal{L}_i(p, q_i) \):
  - sample \( z \sim q_i(z) \)
  - \( \nabla_\theta \mathcal{L}_i(p, q_i) \approx \nabla_\theta \log p_\theta(x_i | z) \)

\[
\theta \leftarrow \theta + \alpha \nabla_\theta \mathcal{L}_i(p, q_i)
\]

update \( q_i \) to maximize \( \mathcal{L}_i(p, q_i) \)

\[
\theta \leftarrow \arg \max_\theta \frac{1}{N} \sum_i \mathcal{L}_i(p, q_i)
\]

let’s say \( q_i(z) = \mathcal{N}(\mu_i, \sigma_i) \)

use gradient \( \nabla_{\mu_i} \mathcal{L}_i(p, q_i) \) and \( \nabla_{\sigma_i} \mathcal{L}_i(p, q_i) \)

gradient ascent on \( \mu_i, \sigma_i \)

how?
What’s the problem?

for each $x_i$ (or mini-batch):

calculate $\nabla_\theta \mathcal{L}_i(p, q_i)$:

sample $z \sim q_i(z)$

$\nabla_\theta \mathcal{L}_i(p, q_i) \approx \nabla_\theta \log p_\theta(x_i | z)$

$\theta \leftarrow \theta + \alpha \nabla_\theta \mathcal{L}_i(p, q_i)$

update $q_i$ to maximize $\mathcal{L}_i(p, q_i)$

Question: How many parameters are there?

in terms of $|\theta|, |z|, N$

let’s say $q_i(z) = \mathcal{N}(\mu_i, \sigma_i)$

use gradient $\nabla_{\mu_i} \mathcal{L}_i(p, q_i)$ and $\nabla_{\sigma_i} \mathcal{L}_i(p, q_i)$

gradient ascent on $\mu_i, \sigma_i$
What’s the problem?

for each $x_i$ (or mini-batch):

calculate $\nabla_\theta \mathcal{L}_i(p, q_i)$:

sample $z \sim q_i(z)$

$\nabla_\theta \mathcal{L}_i(p, q_i) \approx \nabla_\theta \log p_\theta(x_i | z)$

$\theta \leftarrow \theta + \alpha \nabla_\theta \mathcal{L}_i(p, q_i)$

update $q_i$ to maximize $\mathcal{L}_i(p, q_i)$

**Question:** How many parameters are there?

intuition: $q_i(z)$ should approximate $p(z | x_i)$

what if we learn a network $q_i(z) = q(z | x) \approx p(z | x_i)$?

$z \quad p_\theta(x | z) \quad x \quad q_\phi(z | x) = \mathcal{N}(\mu_\phi(x), \sigma_\phi(x))$
Amortized Variational Inference

A. Formulate a lower bound on the log likelihood objective.
B. Check how tight the bound is.
C. Variational inference -> Amortized variational inference
D. How to optimize
What’s the problem?

for each $x_i$ (or mini-batch):

\[
\text{calculate } \nabla_{\theta} \mathcal{L}_i(p, q_i):
\]

\[
\text{sample } z \sim q_i(z)
\]

\[
\nabla_{\theta} \mathcal{L}_i(p, q_i) \approx \nabla_{\theta} \log p_{\theta}(x_i | z)
\]

\[
\theta \leftarrow \theta + \alpha \nabla_{\theta} \mathcal{L}_i(p, q_i)
\]

update $q_i$ to maximize $\mathcal{L}_i(p, q_i)$

**Question:** How many parameters are there?

\[
|\theta| + (|\mu_i| + |\sigma_i|) \times N
\]

intuition: $q_i(z)$ should approximate $p(z|x_i)$

what if we learn a network $q_i(z) = q(z|x_i) \approx p(z|x_i)$?

\[
q_{\phi}(z|x) = \mathcal{N}(\mu_{\phi}(x), \sigma_{\phi}(x))
\]
Amortized variational inference

\[ q_\phi(z|x) = \mathcal{N}(\mu_\phi(x), \sigma_\phi(x)) \]

\[ \mathcal{L}(p_\theta(x_i|z), q_\phi(z|x_i)) \]

\[
\log p(x_i) \geq E_{z \sim q_\phi(z|x_i)} \left[ \log p_\theta(x_i|z) + \log p(z) \right] + \mathcal{H}(q_\phi(z|x_i))
\]

for each \( x_i \) (or mini-batch):

- calculate \( \nabla_\theta \mathcal{L}(p_\theta(x_i|z), q_\phi(z|x_i)) \):
  - sample \( z \sim q_\phi(z|x_i) \)
  - \( \nabla_\theta \mathcal{L} \approx \nabla_\theta \log p_\theta(x_i|z) \)
  - \( \theta \leftarrow \theta + \alpha \nabla_\theta \mathcal{L} \)
  - \( \phi \leftarrow \phi + \alpha \nabla_\phi \mathcal{L} \)

how do we calculate this?
Amortized variational inference

for each $x_i$ (or mini-batch):

- calculate $\nabla_\theta \mathcal{L}(p_\theta(x_i|z), q_\phi(z|x_i))$:
  - sample $z \sim q_\phi(z|x_i)$
  - $\nabla_\theta \mathcal{L} \approx \nabla_\theta \log p_\theta(x_i|z)$

- $\theta \leftarrow \theta + \alpha \nabla_\theta \mathcal{L}$
- $\phi \leftarrow \phi + \alpha \nabla_\phi \mathcal{L}$

$q_\phi(z|x) = \mathcal{N}(\mu_\phi(x), \sigma_\phi(x))$

$\mathcal{L}_i = E_{z \sim q_\phi(z|x_i)}[\log p_\theta(x_i|z) + \log p(z)] + \mathcal{H}(q_\phi(z|x_i))$

$J(\phi) = E_{z \sim q_\phi(z|x_i)}[r(x_i, z)]$
The reparameterization trick

\[ J(\phi) = E_{z \sim q_\phi(z|x)}[r(x_i, z)] \]
\[ = E_{\epsilon \sim \mathcal{N}(0,1)}[r(x_i, \mu_\phi(x_i) + \epsilon \sigma_\phi(x_i))] \]

\[ q_\phi(z|x) = \mathcal{N}(\mu_\phi(x), \sigma_\phi(x)) \]
\[ z = \mu_\phi(x) + \epsilon \sigma_\phi(x) \]

estimating \( \nabla_\phi J(\phi) \):

- sample \( \epsilon_1, \ldots, \epsilon_M \) from \( \mathcal{N}(0,1) \) (a single sample works well!)

\[ \nabla_\phi J(\phi) \approx \frac{1}{M} \sum_j \nabla_\phi r(x_i, \mu_\phi(x_i) + \epsilon_j \sigma_\phi(x_i)) \]

+ Very simple to implement
+ Low variance
- Only continuous latent variables

Discrete latent variables:
- vector quantization & straight-through estimator (“VQ-VAE”)
- policy gradients / “REINFORCE”
Another way to look at everything...

\[ \mathcal{L}_i = E_{z \sim q_\phi(z|x_i)}[\log p_\theta(x_i|z) + \log p(z)] + \mathcal{H}(q_\phi(z|x_i)) \]

\[ = E_{z \sim q_\phi(z|x_i)}[\log p_\theta(x_i|z)] + E_{z \sim q_\phi(z|x_i)}[\log p(z)] + \mathcal{H}(q_\phi(z|x_i)) \]

\[ - D_{KL}(q_\phi(z|x_i)||p(z)) \quad \text{this has a convenient analytical form for Gaussians} \]

\[ = E_{z \sim q_\phi(z|x_i)}[\log p_\theta(x_i|z)] - D_{KL}(q_\phi(z|x_i)||p(z)) \]

\[ = E_{\epsilon \sim \mathcal{N}(0,1)}[\log p_\theta(x_i|\mu_\phi(x_i) + \epsilon\sigma_\phi(x_i))]} - D_{KL}(q_\phi(z|x_i)||p(z)) \]

\[ \approx \log p_\theta(x_i|\mu_\phi(x_i) + \epsilon\sigma_\phi(x_i)) - D_{KL}(q_\phi(z|x_i)||p(z)) \]
Example Models
The variational autoencoder

\[ q_\phi(z|x) = \mathcal{N}(\mu_\phi(x), \sigma_\phi(x)) \]

\[ p_\theta(x|z) = \mathcal{N}(\mu_\theta(z), \sigma_\theta(z)) \]

\[ x_i \xrightarrow{\phi} \mu_\phi(x_i) \xrightarrow{\sigma_\phi(x_i)} \mu_\phi(x_i) + \epsilon \sigma_\phi(x_i) = z \xrightarrow{\theta} p_\theta(x|z) \]

\[ \max_{\theta, \phi} \frac{1}{N} \sum_i \log p_\theta(x_i|\mu_\phi(x_i) + \epsilon \sigma_\phi(x_i)) - D_{\text{KL}}(q_\phi(z|x_i)||p(z)) \]
Using the variational autoencoder

\[ q_\phi(z|x) = \mathcal{N}(\mu_\phi(x), \sigma_\phi(x)) \]

\[ p_\theta(x|z) = \mathcal{N}(\mu_\theta(z), \sigma_\theta(z)) \]

\[ p(x) = \int p(x|z)p(z)dz \]

why does this work?

\[ \mathcal{L}_i = E_{z \sim q_\phi(z|x_i)}[\log p_\theta(x_i|z)] - D_{KL}(q_\phi(z|x_i) \| p(z)) \]

sampling:

\[ z \sim p(z) \]
\[ x \sim p(x|z) \]
Conditional models

\[ \mathcal{L}_i = E_{z \sim q_\phi(z|x_i, y_i)}[\log p_{\theta}(y_i|x_i, z) + \log p(z|x_i)] + \mathcal{H}(q_\phi(z|x_i, y_i)) \]

just like before, only now generating \( y_i \) and everything is conditioned on \( x_i \)

at test time:

\[ z \sim p(z|x_i) \]
\[ y \sim p(y|x_i, z) \]

can optionally depend on \( x \)

**Images from Razavi, van den Oord, Vinyals. Generating Diverse High-Fidelity Images with VQ-VAE-2. ‘19**
Plan for Today

1. Latent variable models
2. Variational inference
3. Amortized variational inference
4. Example latent variables models

Goals
- Understand latent variable models in deep learning
- Understand how to use (amortized) variational inference
Course Reminders

Homework 3 due Monday next week.

Tutorial session on **Thursday 4:30 pm**

**Next time:** Bayesian meta-learning

Be careful Azure usage — turning off machines when you are not using them!