Section 6: Variational Inference

March 3rd, March 4th

Author: Catherine Glossop

1 In Today's Section

A deeper dive into the topics discussed in last week's lecture:

1. Latent variable models
2. Variational Inference + Variational Autoencoders (VAEs)
3. Control as Inference

2 Latent variable models

Say you want to model some distribution, $p(x)$, maybe a conditional distribution, $p(y|x)$. (for example, $\pi(a|s)$). We may choose to believe there is some unobserved variable, z, such that,

$$p(x) = \int_z p(x|z)p(z)$$
$$p(y|z) = \int_z p(y|x,z)p(z)$$

where, z could correspond to some underlying information, such as the intent of an agent when encountering a multi-modal decision.

This form allows us to represent highly complex distributions, but becomes intractable to figure out, as sampling z, especially if it is arbitrarily large, to compute this integral, would be impossible to do in any reasonable amount of time.

Naturally, we could decide we want to represent $p(x)$ with parameters θ and optimize θ with the maximum likelihood objective,

$$\theta \leftarrow \arg\max_\theta \frac{1}{N} \sum_i \log p_\theta(x_i)$$
$$\theta \leftarrow \arg\max_\theta \frac{1}{N} \sum_i \log(\int_z p_\theta(x_i|z)p(z)))$$

Where i indexes into each of our data points. But we still have our intractable integral (and we would need to differentiate it!) and we cannot estimate it with samples because of the non-linearity of the log.

Say we instead want to optimize w.r.t an *expected log-likelihood* objective

$$\theta \leftarrow \arg\max_\theta \frac{1}{N} \sum_i E_{z\sim p(z|x_i)}[\log [p_\theta(x_i|z)p(z)]]$$

where the key here is we sample z from our *posterior distribution*, $z \sim p_\theta(z|x_i)$.

The fundamental problem we will discuss today is how we learn this posterior distribution.

3 Variational Inference

3.1 How do we learn a model that approximates $p(x)$?

Assume that we choose our learnable posterior distribution to be, $q_i(z)$, which corresponds to each data point, x_i, as a Gaussian distribution, just to make it tractable and convenient (but as as we will see, this doesn't matter). We want the q_is to approximate the true posterior distributions of $p(x)$.

Let's write out the equation for $\log p(x_i)$, and move things around a bit to get an expectation over $z \sim q_i(z)$ with our incorrect (but convenient) guess for $q_i(z)$.

$$
\begin{aligned}
\log p(x_i) &= \log \int_z p(x_i|z)p(z) \\
&= \log \int p(x_i|z)p(z)\frac{q_i(z)}{q_i(z)} && \text{Equivalent to multiplying by one} \\
&= \log E_{z\sim q_i(z)}\left[\frac{p(x_i|z)p(z)}{q_i(z)}\right] \\
&\geq E_{z\sim q_i(z)}\left[\log \frac{p(x_i|z)p(z)}{q_i(z)}\right] && \text{Apply Jensen's inequality} \\
&= E_{z\sim q_i(z)}\left[\log p(x_i|z) + \log p(z)\right] - E_{z\sim q_i(z)}[\log q_i(z)] && \log(\frac{x\cdot y}{z}) = \log(x) + \log(y) - \log(z) \\
&= E_{z\sim q_i(z)}\left[\log p(x_i|z) + \log p(z)\right] + \mathcal{H}(q_i) && \text{Second term is entropy}
\end{aligned}
$$

No matter what q_i is, a lower bound for $\log p(x_i)$ can be constructed, which, when the lower bound is increased, $\log p(x_i)$ will also increase. This is dependent on how *tight* the bound is, and the tighter the bound, the more easily $\log p(x_i)$ can be increased by pushing up on it.

Jensen's Inequality: For any convex function, $\varphi(X)$,

$$\varphi(E[X]) \leq E[\varphi(X)]$$

For another perspective, let's look at the reverse KL. The reverse KL can be simplified to

$$
\begin{aligned}
D_{KL}(q||p) &= \sum_x q(x)\log\frac{q(x)}{p(x)} \\
&= \sum_x q(x)\log q(x) - q(x)\log p(x) \\
&= -E_{x\sim q(x)}[\log p(x)] + E_{x\sim q(x)}[\log q(x)] \\
&= -E_{x\sim q(x)}[\log p(x)] - \mathcal{H}(q)
\end{aligned}
$$

Entropy: The entropy for a given random variable, X, distributed according to $p(x)$, is defined as,

$$
\begin{aligned}
\mathcal{H}(p) &= -E_{x\sim p(x)}[\log p(x)] \\
&= -\int_x p(x)\log p(x)
\end{aligned}
$$

where it can be thought of as measuring the randomness of X or how big the log-likelihood of the distribution is in expectation under itself.

Our bound looks like the negative reverse KL between $p(x_i)$ and $q_i(z)$. Let's define this bound as

$$\mathcal{L}_i(p, q_i) = E_{z\sim q_i(z)}\left[\log p(x_i|z) + \log p(z)\right] + \mathcal{H}(q_i)$$

Let's try another method to arrive at a bound. What would make a good posterior? It should approximate the true posterior $p(z|x_i)$. Let's use KL divergence (reverse KL specifically).

$$
\begin{aligned}
D_{KL}(q_i(z)||p(z|x_i)) &= E_{z\sim q_i(z)}\left[\log \frac{q_i(z)}{p(z|x_i)}\right] \\
&= E_{z\sim q_i(z)}\left[\log \frac{q_i(z)p(x_i}{p(x_i,z)}\right] \\
&= -E_{z\sim q_i(z)}\left[\log p(x_i|z) + \log p(z)\right] + E_{z\sim q_i(z)}\left[\log q_i(z)\right] + E_{z\sim q_i(z)}[\log p(x_i)] \\
&= -E_{z\sim q_i(z)}\left[\log p(x_i|z) + \log p(z)\right] + \mathcal{H}(q_i) + \log p(x_i) \\
&= -\mathcal{L}_i(p, q_i) + \log p(x_i)
\end{aligned}
$$

We want to maximize $\log p(x_i)$, so let's rearrange

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

This is also a lower bound as:

- The KL is always positive, so $\mathcal{L} + D_{KL}$ must also be a lower bound.
- As the KL goes to zero, the bound will become tighter and when $q_i(z) = p(z|x_i)$ and $D_{KL} = 0$, this will be an equality!

KL divergence: The Kullback-Leibler (KL) divergence of two distributions, p(x) and q(x),

$$D_{KL}(p||q) = \sum_x p(x) \log \frac{p(x)}{q(x)}$$

where it can be thought of as measuring the randomness of X or how big the log-likelihood of the distribution is in expectation under itself.

This also means we can use the same objective to maximize p and make q_i close to $p(z|x_i)$. Say we represent $q_i(z) = \mathcal{N}(\mu_i, \sigma_i)$ and we parameterize $p(x_i|z) = p_\theta(x_i|z)$.

Algorithm 1 Variational Inference

Initialize θ, μ_i, σ_i
for x_i or mini-batch in $\mathcal{D}$ **do**
  Calculate $\nabla_\theta \mathcal{L}$:
    $z \sim q_i(z)$
    $\nabla_\theta \mathcal{L} \approx \nabla_\theta \log p_\theta(x_i|z)$
  $\theta \leftarrow \theta + \alpha \nabla_\theta \mathcal{L}$
  Calculate $\nabla_{\mu_i} \mathcal{L}, \nabla_{\sigma_i} \mathcal{L}$
  $\mu_i \leftarrow \mu_i + \alpha \nabla_{\mu_i} \mathcal{L}$
  $\sigma_i \leftarrow \sigma_i + \alpha \nabla_{\sigma_i} \mathcal{L}$
end for

But! As i corresponds to the number of data points, $\#parameters \propto i$.

3.2 Amortized Variational Inference

To make variational inference tractable, we can share variables (such as μ and σ) across data points. This is also valid as we essentially make $q_i(z) = q(z)$, and any q is valid per our bound. But as likely every data point does **not** have the same posterior, we represent it as a function x_i, $q_\varphi(z|x_i)$, parameterized by φ.

Expectation-Maximization:
E-step: Update the expected values of latent variables (Optimize φ for $q_\varphi(z|x_i)$)
M-step: Maximize the expected log-likelihood (Optimize θ for $p_\theta(x_i|z)$)

This gives the new objective

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

$$\nabla_\varphi J(\varphi) \approx \frac{1}{M} \sum_j \nabla_\varphi \log q_\varphi(z_j|x_i)[\log p_\theta(x_i|z_j) + \log p(z_j)]$$

But we run into an expectation over samples from the model we are trying to optimize. We have two options: 1) policy gradient method, 2) the reparameterization trick (see table 1). We will choose the reparameterization trick, allowing us to rewrite our objective

$$\begin{aligned}\mathcal{L}_i &= E_{\epsilon\sim\mathcal{N}(0,1)}[\log p_\theta(x_i|\mu_\varphi(x_i)+\epsilon\sigma_\varphi(x_i))] - D_{KL}(q_\varphi(z|x_i)||p(z)) \\ &\approx \log p_\theta(x_i|\mu_\varphi(x_i)+\epsilon\sigma_\varphi(x_i)) - D_{KL}(q_\varphi(z|x_i)||p(z))\end{aligned}$$

3.3 A Latent Variable Model: Variational Autoencoder (VAE)

VAEs allow us to encode complex states into a simplified latent space (the zs), which will preserve the information from the xs while removing redundant information. We train two models:

1. Encoder $\rightarrow q_\varphi(z|x) = \mathcal{N}(\mu_\varphi(x), \sigma_\varphi(x))$
2. Decoder $\rightarrow p_\theta(x|z) = \mathcal{N}(\mu_\theta(z), \sigma_\theta(z))$

Note: typically $p(z)$ (the prior) is typically $p(z) = \mathcal{N}(0, I)$

Taking our equation from before (with the reparameterization trick), our full objective is

Objective for VAE (Evidence Lower Bound or ELBO):

$$\max_{\theta,\varphi} \frac{1}{N}\sum_i \log p_\theta(x_i|\mu_\varphi(x_i)+\epsilon\sigma_\varphi(x_i)) - D_{KL}(q_\varphi(z|x_i)||p(z))$$

where $\epsilon \sim \mathcal{N}(0,1)$

At test time:

$z \sim p(z)$ often $\mathcal{N}(0, I)$

$x \sim p_\theta(x|z)$

We can sample $z \sim p(z)$ at test time, even though we never sample from it at train time for updating p_θ, because we make q_φ look like the true posterior with the first term, while the second term makes it look like the prior ($p(z)$).

We can extend this to conditional models (modeling $p(y|x)$, or our favorite conditional model $\pi(a|s)$) with a simple modification to the previous objective

Objective for Conditional VAE:

$$\max_{\theta,\varphi} \frac{1}{N}\sum_i \log p_\theta(y_i|\mu_\varphi(x_i,y_i)+\epsilon\sigma_\varphi(x_i,y_i)) - D_{KL}(q_\varphi(z|x_i,y_i)||p(z|x_i))$$

where $\epsilon \sim \mathcal{N}(0,1)$

Property	**Policy Gradient Method**	**Reparameterization Trick**
$\nabla_\varphi J(\varphi) \approx$	$\frac{1}{M}\sum_j \nabla_\varphi \log q_\varphi(z_j\|x_i) f(x_i,z_j)$	$\frac{1}{M}\sum_j \nabla_\varphi f(x_i, \mu_\varphi(x_i)+\epsilon\sigma_\varphi(x_i))$
Variance	High	Low
Latent variable types	Continuous / Discrete	Continuous only

Table 1: Overview of methods to deal with gradients through samples. Here, $f(x_i, z_j)$ is just some function of x_i, z_j.

At test time:

$$z \sim p(z|x_i)$$
$$y \sim p_\theta(y|x_i, z)$$

Where the biggest thing we've changed is now **everything** is conditioned on x_i. Note that we can also modify our prior so be conditioned on x_i such that $z \sim p(z|x_i)$.

4 Control as Inference

4.1 A model for human behavior

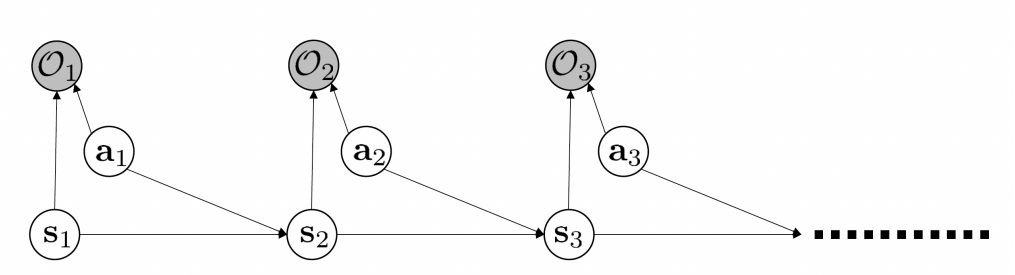

Figure 1: Dynamics with optimality variables

RL models rational agents that attempt to be optimal in every step. However, humans are often not perfectly optimal, and rather we have stochastic behavior where *good* behavior is most probably but *bad* behaviors still have non-zero probability.

Considering this, we can add binary *optimality* variables at each step, conditioned on state and action (see section 4.1), which we can observe. While previously our MDPs just modeled physics, these variables can now help us ascribe the agent's *intent* to the dynamics.

Let's assume we can model $p(\mathcal{O}_t|s_t, a_t) = \exp(r(s_t, a_t))$ and that $r(s_t, a_t) \leq 0$ so that $0 < p(\mathcal{O}_t|s_t, a_t) \leq 1$. What would the distribution of trajectories be given the agent tries to be optimal at all steps?

$$p(\tau|\mathcal{O}_{1:T}) = \frac{p(\tau, \mathcal{O}_{1:T})}{p(\mathcal{O}_{1:T})}$$

We can approximately say this is proportional to the probability of any physically possible trajectory, $p(\tau)$, times the probability of being optimal at every step

$$\begin{aligned}\frac{p(\tau, \mathcal{O}_{1:T})}{p(\mathcal{O}_{1:T})} &\propto p(\tau) \prod_t \exp(r(s_t, a_t)) \\ &= p(\tau) \exp\left(\sum_t r(s_t, a_t)\right)\end{aligned}$$

which means the agent is most likely to take the most rewarding trajectory (sum of rewards over the trajectory), and as the trajectory becomes less rewarding, it becomes exponentially less likely.

4.2 How can we use this model?

We can use this model to solve various inference problems:

- **Backward messages:** $\beta_t(s_t, a_t) = p(\mathcal{O}_{t:T}|s_t, a_t) \rightarrow$ How likely is the agent to be optimal at this step and all future steps given the current state and action?
- **Computing policies:** $p(a_t|s_t, \mathcal{O}_{1:T}) \rightarrow$ Given the state and optimality over the trajectory, what is the probability of a given action?
- **Forward messages:** $\alpha_t(s_t) = p(s_t|\mathcal{O}_{1:t-1}) \rightarrow$ What is the probability of a given state given how optimal you have been up until that state?

4.2.1 Backwards Messages

To be able to compute $\beta_t(s_t, a_t)$, we will set up a recursive method, going forward in time from t to $t+1$ all the way to T. At time T, our base case is $p(\mathcal{O}_T|s_T, a_T) = \exp(r(s_T, a_T))$.

Let's break up $\beta_t(s_t, a_t)$

$$\begin{aligned}
\beta_t(s_t, a_t) &= p(\mathcal{O}_{t:T}|s_t, a_t) \\
&= \int p(\mathcal{O}_{t:T}, s_{t+1}|s_t, a_t) ds_{t+1} && s_{t+1} \text{ is marginalized out, so equality} \\
&= \int p(\mathcal{O}_{t+1:T}|s_{t+1}) p(s_{t+1}|s_t, a_t) p(\mathcal{O}_t|s_t, a_t) ds_{t+1} && \mathcal{O}_{t:T} \text{ only depends on } s_{t+1}
\end{aligned}$$

We already have two terms that are known, $p(s_{t+1}|s_t, a_t)$, from our graphical model, and $p(\mathcal{O}_t|s_t, a_t) = \exp(r(s_t, a_t))$. Let's break down the first term.

$$p(\mathcal{O}_{t+1:T}|s_{t+1}) = \int p(\mathcal{O}_{t+1:T}|s_{t+1}, a_{t+1}) p(a_{t+1}|s_{t+1}) da_{t+1}$$

We've done the same marginalization trick as before to insert the actions into our equation. We also notice that the first term is

$$\beta_{t+1}(s_{t+1}, a_{t+1}) = p(\mathcal{O}_{t+1:T}|s_{t+1}, a_{t+1})$$

The second term seems to be a policy unconditioned on optimality, also known as the *action prior* of the agent. We can assume this is uniform (the agent is equally likely to take any action if it has no incentive to do anything in particular).

As a result, we get the following recursion **and** an analogy to the RL concepts we've discussed before:

Backwards Messages:

Original Form (from graphical model)

for $t = T-1$ to 1:

$$\beta_t(s_t, a_t) = p(\mathcal{O}_t \mid s_t, a_t)\, \mathbb{E}_{s_{t+1} \sim p(s_{t+1}|s_t, a_t)}[\beta_{t+1}(s_{t+1})]$$

$$\beta_t(s_t) = \mathbb{E}_{a_t \sim p(a_t|s_t)}[\beta_t(s_t, a_t)]$$

RL form (still from graphical model!)

Let $V_t(s_t) = \log \beta_t(s_t)$, $Q_t(s_t, a_t) = \log \beta_t(s_t, a_t)$

$$V_t(s_t) = \log \int \exp(Q_t(s_t, a_t)) da_t$$

$$Q_t(s_t, a_t) = r(s_t, a_t) + \log E[\exp(V_{t+1}(s_{t+1}))]$$

The interesting thing about this relationship is that it kind of looks like a Bellman backup and our value iteration algorithm despite just being derived from a graphical model of dynamics! The differences are:

- "Soft max" instead of the hard max for $V_t(s_t)$
- Log expectation of the exponential of V_t instead of the regular expectation

However, by having $E[\exp(V_{t+1}(s_{t+1}))]$, we are implicitly being optimistic over stochastic transitions. We will get back to this when we discuss control as variational inference in this context.

If we do not have a uniform action prior, this is equivalent to changing the reward

$$\begin{aligned} V_t(s_t) &= \log \int \exp(Q_t(s_t, a_t) + \log p(a_t|s_t)) da_t \\ Q_t(s_t, a_t) &= r(s_t, a_t) + \log E[\exp(V_{t+1}(s_{t+1}))] \\ \tilde{Q}_t(s_t, a_t) &= r(s_t, a_t) + \log p(a_t|s_t) + \log E[\exp(V_{t+1}(s_{t+1}))] \end{aligned}$$

Intuitively, this is because if the action prior is not uniform, then there is some implicit reward for some actions over others. Therefore, we can just modify the reward.

4.2.2 Policy Computation

To get our policy, we simply compute $\pi(a_t|s_t) = p(a_t|s_t, \mathcal{O}_{1:T})$, the action when the agent is always optimal.

$$\begin{aligned} \pi(a_t|s_t) &= p(a_t|s_t, \mathcal{O}_{1:T}) \\ &= p(a_t|s_t, \mathcal{O}_{t:T}) && \text{Past does not matter} \\ &= \frac{p(a_t, s_t|\mathcal{O}_{t:T})}{p(s_t|\mathcal{O}_{t:T})} && \text{Conditional probability} \\ &= \frac{p(\mathcal{O}_{t:T}|s_t, a_t)p(a_t, s_t)/p(\mathcal{O}_{t:T})}{p(\mathcal{O}_{t:T}|s_t)p(s_t)/p(\mathcal{O}_{t:T})} && \text{Bayes' rule to top and bottom} \\ &= \frac{p(\mathcal{O}_{t:T}|s_t, a_t)}{p(\mathcal{O}_{t:T}|s_t)} \frac{p(a_t, s_t)}{p(s_t)} && p(\mathcal{O}_{t:T}) \text{ cancels and split} \\ &= \frac{\beta_t(s_t, a_t)}{\beta_t(s_t)} p(a_t|s_t) && \text{Sub in } \beta_t(s_t, a_t) \text{ and } \beta_t(s_t) \\ &= \frac{\beta_t(s_t, a_t)}{\beta_t(s_t)} && \text{Uniform action prior} \end{aligned}$$

Applying our same value function interpretation as before,

$$\begin{aligned} \pi(a_t|s_t) &= \frac{\exp(V_t(s_t))}{\exp(Q_t(s_t, a_t))} \\ &= \exp(V_t(s_t) - Q_t(s_t, a_t)) \\ &= \exp(A_t(s_t, a_t)) \end{aligned}$$

So our agent is most likely to take actions that provide high advantage, and as the action's advantage for the current state gets worse, it is exponentially less likely to take those actions.

To make the distribution softer or harder with respect to our value functions, we can apply a temperature, α, that when increased softens the policy and when decreased sharpens it. As $\alpha \to 0$, the policy goes to the greedy policy.

$$\pi(a_t|s_t) = \exp(\frac{1}{\alpha} A_t(s_t, a_t))$$

4.2.3 Forward messages

Forward messages can be used to figure out what our state distribution will look like given how optimally the agent behaves. We will go through the same steps as for backward messages, with the base case that $\alpha_1(s_1) = p(s_1)$ is usually known.

$$\begin{aligned}
\alpha_t(s_t) &= \int p(s_t, s_{t-1}, a_{t-1}|\mathcal{O}_{1:t-1})ds_{t-1}da_{t-1} \\
&= \int p(s_t|s_{t-1}, a_{t-1}, \mathcal{O}_{1:t-1})p(a_{t-1}|s_{t-1}, \mathcal{O}_{1:t-1})p(s_{t-1}|\mathcal{O}_{1:t-1})ds_{t-1}da_{t-1} && \text{Break up the probability} \\
&= \int p(s_t|s_{t-1}, a_{t-1})p(a_{t-1,\mathcal{O}_{t-1}}|s_{t-1})p(s_{t-1}|\mathcal{O}_{1:t-1})ds_{t-1}da_{t-1} && \text{Transition indep. of optimality}
\end{aligned}$$

We know $p(s_t|s_{t-1}, a_{t-1})$. Just looking at the second two terms

$$\begin{aligned}
p(a_{t-1}|s_{t-1}, \mathcal{O}_{t-1})p(s_{t-1}|\mathcal{O}_{1:t-1}) &= \frac{p(\mathcal{O}_{t-1}|s_{t-1}, a_{t-1})p(a_{t-1}|s_{t-1})}{p(\mathcal{O}_{t-1})}\frac{p(\mathcal{O}_{t-1})p(s_{t-1}|\mathcal{O}_{1:t-1})}{p(\mathcal{O}_{t-1}|\mathcal{O}_{1:t-2})} && \text{Bayes' rule} \\
&= p(\mathcal{O}_{t-1}|s_{t-1}, a_{t-1})p(a_{t-1}|s_{t-1})\frac{\alpha_{t-1}(s_{t-1})}{p(\mathcal{O}_{t-1}|\mathcal{O}_{1:t-2})} && \text{Cancel and sub in } \alpha_{t-1}
\end{aligned}$$

Now a lot of these things look pretty familiar. $p(\mathcal{O}_{t-1}|s_{t-1}, a_{t-1}) = \exp(r(s_{t-1}, a_{t-1}))$ and $p(s_{t-1}|a_{t-1})$ is our action prior, which we assume is uniform.

To write it out fully

Forward Messages:

$$\alpha_t(s_t) = \int p(s_t|s_{t-1}, a_{t-1}) \exp(r(s_{t-1}, a_{t-1}))\frac{\alpha_{t-1}(s_{t-1})}{p(\mathcal{O}_{t-1}|\mathcal{O}_{1:t-2})}$$

What is most important to observe is that we can write the distribution when optimal over the entire trajectory as

$$\begin{aligned}
p(s_t|\mathcal{O}_{1:T}) &= \frac{p(s_t, \mathcal{O}_{1:T})}{p(\mathcal{O}_{1:T})} && \text{Conditional probability} \\
&= \frac{p(\mathcal{O}_{t:T}|s_t)p(s_t, \mathcal{O}_{1:t-1})}{p(\mathcal{O}_{1:T})} && \text{Bayes' rule} \\
&\propto \beta_t(s_t)\alpha_t(s_t) && \text{Split joint into conditional}
\end{aligned}$$

This means that our states will lie in the overlap of the "cones" of the backward and forward messages.