EDGE: Explaining Deep Reinforcement Learning Policies

Chen Gong

18 October 2021

目录

本文 “EDGE: Explaining Deep Reinforcement Learning Policies” 于 2021 年发表于 NIPS，是宾州州立大学 Wenbo Guo 的工作，小编精读了好几篇他的文章了，很是崇拜这位作者，他的一作的工作感觉都是集中在 RL+testing 和 RL+Security 这块。

看到了知乎中的 “一个科学家” 的文章：https://zhuanlan.zhihu.com/p/377994641，其中提到了欣赏和提炼他人文章 insight 的重要性。此篇文章对我启发很大，于是决定之后的文章都思考其 insight。我思考，也询问了师兄关于 insight，motivation 和 idea 的区别的联系，insight 是我们从结果中发现的” 洞见”，而 motivation 是做这份研究的原因，而 idea 涵盖的特别广，它更像是解决 insight 的具体方法。举个例子，我们发现了一个 “insight”，神经网络是不鲁棒的，这个 “insight” 驱使我们增加神经网络的鲁棒性，i.e. “motivation”，然后提出了某方法来增加神经网络的鲁棒性，i.e.，“idea”。

之前的工作，也做过一些利用 RL 来进行 attack 的工作，之后也想继续这方面的研究。而攻击和测试是不太分家的，经常有工作 attack，defends 和 repair 一起做。

本文的 insight 在于，首先作者定义了一个好的强化学习解释问题，利用状态和动作的轨迹来预测奖励，从而寻找时间戳和时间戳之间的联系，找到对最终时刻奖励影响最大的时间戳来帮助 attacker 高效的攻击等等。或许在此类 RL 问题中，首先要找到一个好的故事，写这类文章对写作要求挺高的。而实现的方法上，看上去公式表达比较复杂，并且符号定义比较多。但是，得益于作者非常清晰的写作技巧，所以，实际读起来并不是很难，算法实现上比较清晰，但是也不知道是不是我对 VAE 比较熟悉。仔细读起来，自我感觉是本质上套了一个 VAE，用 VAE 来解决作者提出的问题。而 VAE 本身有很多的问题，比如隐藏层分布不一致；将先验和后验分布假设为 Gaussian，而 Gaussian 本身的表达能力有限，会损失很多的信息；很多改进 VAE 的问题都可以拿来改进这个算法。但是这样就没什么太大的意思了，再想想有没有什么 high-level 的 idea 吧，不想仅仅在对别人的工作进行修修补补。而我觉得关键还是找到一个好的解释或者攻击的问题，然后解决的方法可能并不需要那么的新。

1 Key idea

强化学习落地中一个重要的挑战就是智能体做出的动作的机理是一个黑盒，具有不可解释性，导致大家对智能体不太信任。为了解决这个问题，有一些文献得到了一些对于特定时刻智能体做出某动作的解释，具体的说，找到智能体观测的状态中哪些特征对智能体做出影响最大，但是这样的方法只是找到了环境特征和智能体动作的关系，依然缺乏理解策略本身的能力。换句话说，现有的方法并不能揭示智能体从游戏中获得的最终奖励，与游戏过程中的行动/状态之间的关系。因此，当智能体的任务失败时，无法帮助找到智能体的弱点。所以，作者的 insight 的在于解决此问题，解释 agent 最终获得的奖励和他经历的状态动作之间的关系。

实际上，作者的解释方法，就是一个 trajectory 中找到对最后智能体奖励产生最大影响的关键帧（个人觉得 Guo 在此处讲故事的能力真的很厉害，hhhh）。模糊一点的说法是，通过用一个自解释模型逼近目标 agent 的 MDP，并利用模型来识别重要的时间帧。具体做法，1. 对于个 well-trained 的智能体，首先收集一系列的轨迹和轨迹对应的累计奖励；2. 利用这些数据去拟合一个自解释模型，此模型输入是轨迹，输出的是累计奖励。3. 为了适应模拟 rl 的 MDP 过程这一问题，作者不是简单的用现成的自解释模型，而是做了一些改动（我有点期待他如何描述自己改动的逻辑，如何适应求解 MDP 过程这一个特定问题的）。首先选择一个修改的高斯过程，此高斯过程使用的是 customized deep additive 核函数（我还没往下看，但是，我猜这里是用了一个网络当 kernel function，hhhhh），来获得两方面

的关系：1. 时间帧之间；2. 轨迹（episode）之间。然后，利用变分法来对高斯过程进行回归。此方法被作者命名为：EDGE。小编觉得这篇文章的做法并不难，主要流程是**利用高斯过程来逼近智能体和环境的 MDP**。

作者总结的这样的自解释模型的优点有，(1) EDGE 可以帮助理解智能体的行为（**揭示动作和奖励直接的关系，之前的工作大多是建立状态和动作直接的关系，这也是作者认为的此篇文章的立足点**），(2) 帮助 attacker 高效的攻击，（3）帮助解释为什么智能体会犯错误，这样可以帮助提升智能体，（4）EDGE 可以帮助建立防御策略。

2 Background

这里对符号做一些简单的解释，有 N 段轨迹 $\mathcal{T} = \{\mathbf{X}^{(i)}, y_i\}_{i=1:N}$，其中第 i 段轨迹 $\mathbf{X}^{(i)} = \{s_t^{(i)}, a_t^{(i)}\}_{t=1:T}$。$y_i$ 是一段轨迹的最终奖励。**我们的目标是找出一段轨迹中的 top-K 的时间帧**。

2.1 Possible Solutions and Limitations

最简单的方法就是用每个时刻 t 的价值函数 $V(s_t)$，但是我们没有收集每个时刻的奖励，不能近似价值函数，也没有智能体价值函数的信息。一种非常自然的方法是采用 Seq2one 方法（RNN），用一段轨迹的状态-动作对作为输入来预测最终的奖励。利用该预测模型，可以利用 post-training 解释方法推导出时间步长重要性。然而，现有的 post-training 解释方法通常需要使用更 transparent 的模型逼近目标 DNN，这不可避免地会引入错误（不太懂这个意思，hhh）。并且，现有的 post-training 方法都很容易被攻击或者容易生成和模型独立的解释。还有利用自解释模型来预测最终的奖励，但是有很多很多的问题。而本文的工作中，考虑了两种 self-explanation 方法来拟合和解释时序数据，(1) 增强了注意力的 RNN，(2) rational net。具体来说，所有的模型都可以表示为：$g(\theta(\mathbf{x}) \odot \mathbf{x})$，其中 $\theta(\cdot)$ 表示一个 RNN 或者一个 attention 层，而 $g(\cdot)$ 表示预测的 RNN 网络。而 $\theta(\cdot)$ 可以被用来识别输入序列中的重要帧。而最近一些研究中，揭示 $\theta(\cdot)$ 给出的解释不能真实地反映序列预测模型 $g(\cdot)$ 学习到的关联 (即特征重要性)，导致在某些应用中，其保真度甚至低于 post-training 的解释。此外，这些模型并不是用来解释 RL 智能体的，也不能完全获得智能体轨迹之间的依赖关系。具体的说，同一个智能体产生的多条轨迹包含两方面的关系，同一个轨迹中不同时间步之间的关系，不同轨迹之间的关系。之前的方法无法获得不同估计之间的关系。

3 EDGE

首先，用 RNN 对一段轨迹中的图片进行 embedding，来建立不同帧之间的关系，类似于使用高维抽象信息作为网络的参数，而不是直接使用原像素作为输入。那么模型可以被写为 $g(f(x))$，其中，$\mathbf{x}$ 表示一段轨迹中的所有状态，$f(\mathbf{x})$ 是 RNN 的输出，i.e., 对 $\mathbf{x}$ 的一个特征提取，$g(\cdot)$ 是可解释性的预测模型。第二点，我们设计了一个高斯过程，作为特征提取器来获得不同时间帧和不同的轨迹之间的相关性。**而且 GP 相对于 DNN 的另一个优势是，GP 对输出信号的联合分布进行建模，从而能够获取输出信号的不确定性**。最后，建立了一个可解释的贝叶斯预测模型来推测最终奖励的分布，并且得到时间帧的重要性。

3.1 整体方法

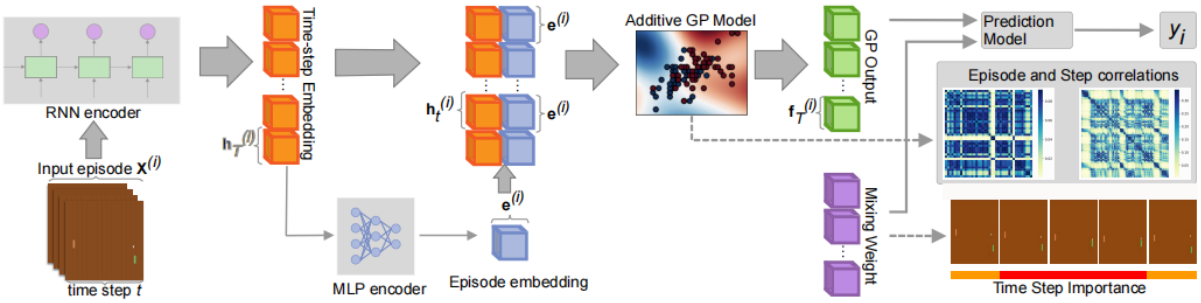

图 1: EDGE 算法流程图

正如图 1中描述的一样，对于目标智能体的一段轨迹 $\mathbf{X}^{(i)}$ (i.e., $\{(s,a)\}$)，首先将其输入到一个 RNN encoder 中，其输出为每一个时间帧的 embedding $\{\mathbf{h}_t^{(i)}\}_{t=1:T}$。同时，将最后一个时间帧的 embedding (i.e., $\mathbf{h}_T^{(i)}$) 输入一个简单的网络，得到 embedding (i.e., $\mathbf{e}^{(i)}$)。并将这两者输入到一个本文提出的高斯过程来获得一个表示层 $\mathbf{f}_{1:T}^{(i)}$（这里其实获得的是轨迹中每一帧的表示形式）。然后将表示层 $\mathbf{f}_{1:T}^{(i)}$ 输入到一个简单的线性回归模型来预测最终的奖励 $y^{(i)}$，那么每一帧对应的 $\mathbf{f}_{1:T}^{(i)}$ 都会有一个权重，这个权重对应的就是帧的重要性。

3.2 Additive GP with Deep Recurrent Kernels

高斯过程定义了一个关于函数的分布。GP 定义了非参数的函数先验 $f:\mathcal{X}\to\mathbb{R}$，并且此函数服从 GP 先验，$f\sim\mathcal{GP}(0,k_\gamma)$，其中，$k_\gamma(\cdot,\cdot)$ 是半正定的核函数，其参数记为 γ。而 $\mathbf{f}\in\mathbb{R}^N$ 服从多元高斯分布 $(\mathbf{f}|\mathbf{X})\sim\mathcal{N}(0,K_{XX})$。其中，$K_{XX}\in\mathbb{R}^{N\times N}$ 是协方差矩阵，$(K_{XX})_{ij}=k_\gamma(\mathbf{x}_i,\mathbf{x}_j)$。本文中，使用的是常见的 SE 核函数：$k_\gamma(\mathbf{x}_i,\mathbf{x}_j)=\exp\left(-\frac{1}{2}(\mathbf{x}_i-\mathbf{x}_j)^T\Theta_k(\mathbf{x}_i-\mathbf{x}_j)\right)$，其中 $\gamma=\Theta_k$。由于输入变量的维度过高，所以本文中采用先用 DNN 对输入变量进行降维，然后将 GP 应用于 DNN 的表示层的方法。

在本文，通过使用 RNN 作为核函数来获得轨迹中的顺序依赖关系。对于一个轨迹 $\mathbf{X}^{(i)}$，首先将动作和状态进行拼接 (i.e., $\mathbf{X}_t^{(i)}=[\mathbf{s}_t^{(i)},\mathbf{a}_t^{(i)}]$)，然后输入到 RNN h_ϕ，然后获得轨迹的表示层：$\{\mathbf{h}_t^{(i)}\}_{t=1:T}$，其中 $\mathbf{h}_t^{(i)}\in\mathbb{R}^q$ 是时间帧对动作状态对的 embedding。同时，通过将轨迹最后一个时间帧的隐藏层变量输入一个浅层 MLP 得到轨迹的 embedding $e_{\phi_1}\to\mathbf{e}^{(i)}\in\mathbb{R}^q$。在获得了 $\mathbf{s}_t^{(i)}$ 和 $\mathbf{e}^{(i)}$ 之后，利用 additive 的 GP 框架来获得同一个轨迹不同时间帧和不同轨迹之间的相关性。形式上说，additive GP 为 J 个独立 GP 的加权和，i.e., $f=\sum_J\alpha_j f_j$，且 $f_j\sim\mathcal{GP}(0,k_j)$，其中，协方差函数 k_j 通常为输入特征的一个子集。基于这种框架，我们构建了一个有两个高斯过程 f_t 和 f_e 组成的 additive GP 模型。具体来说，$f_t\sim\mathcal{GP}(0,k_{\gamma_t})$，建立了时间帧的关联性，其中 $k_{\gamma_t}(\mathbf{h}_t^{(i)},\mathbf{h}_t^{(j)})$ 表示在轨迹 i 的第 t 个时间帧和轨迹 j 的第 k 个时间帧之间的关联性。而 $f_e\sim\mathcal{GP}(0,k_{\gamma_e})$ 则来评估轨迹之间的相似性。具体的说，用 $k_{\gamma_e}(\mathbf{e}^{(i)},\mathbf{e}^{(j)})$ 表示 i 轨迹和 j 轨迹之间的关联性。最终，深度 additive GP 模型可以被表达为：$f=\alpha_t f_t+\alpha_e f_e$。对于给定的收集到的轨迹，$\mathbf{T}\in\mathbb{R}^{N\times T\times(d_s+d_a)}$，$\mathbf{f}\in\mathbb{R}^{NT}$ 为：$\mathbf{f}|\mathbf{X}\sim\mathcal{N}(0,k=\alpha_t^2k_{\gamma_t}+\alpha_e^2k_{\gamma_e})$，其中，$\mathbf{X}\in\mathbb{R}^{NT\times(d_s+d_a)}$ 是 $\mathbf{T}$ 矩阵的 flatten 矩阵。

这段作者比较清晰的把数据的 pipline 讲了一遍，看着复杂其实还是比较清晰的。

3.3 Prediction Model

本文使用线性回归作为预测模型的基础，其中回归系数反映了每个输入单元的重要性。具体来说，将 $\mathbf{f}$ 重新变回矩阵形式：$\mathbf{F} \in \mathbb{R}^{N\times T}$，其中，第 i 行 $\mathbf{F}^{(i)} \in \mathbb{R}^T$ 为第 i 段轨迹，用 GP encoding 后的结果。然后分别定义离散和连续两种最终奖励的条件似然。当 y_i 是连续的，使用经典的 GP 可以定义为 $y_i = \mathbf{F}^{(i)}\mathbf{w}^T + \epsilon_1$，其中，$\mathbf{w} \sim \mathbb{R}^{1\times T}$ 为权重，而 $\epsilon_i \sim \sigma^2$。而奖励的似然被定义为：$y_i|\mathbf{F}^{(i)} \sim \mathcal{N}(\mathbf{F}^{(i)}\mathbf{w}^T, \sigma^2)$。对于可能值有限的离散最终奖励，我们使用 softmax 预测模型进行分类。具体来说，我们定义 $y_i|\mathbf{F}^{(i)}$ 服从离散分布，$p(y_i = k|\mathbf{F}^{(i)}) = \frac{\exp((\mathbf{F}^{(i)}\mathbf{W})_k)}{\sum_k \exp((\mathbf{F}^{(i)}\mathbf{W})_k)}$，其中，$\mathbf{W} \sim \mathbb{R}^{K\times T}$。那么，可解释模型可以解释为：

$$\mathbf{f}\big|\mathbf{X} \sim \mathcal{N}\left(\mathbf{0}, k = \alpha_t^2 k_{\gamma_t} + \alpha_e^2 k_{\gamma_e}\right), \quad y_i\big|\,\mathbf{F}^{(i)} \sim \begin{cases} \mathrm{Cal}\left(\mathrm{softmax}\left(\mathbf{F}^{(i)}\mathbf{W}^T\right)\right) \\ \mathcal{N}\left(\mathbf{F}^{(i)}\mathbf{w}^T, \sigma^2\right) \end{cases} \tag{1}$$

那么，根据权重就可以得到对时间帧的重要性的评判。根据高相关性的时间步长往往对游戏结果具有联合效应 (相似重要性) 的观点，我们可以将全局解释与 $K_t(\mathbf{X}, \mathbf{X})$ 中的时间步长相关性结合起来，以获得对每个游戏的细粒度理解。具体的说，（**我发现这个作者，非常喜欢用一句话来 high-level 的解释，然后用 specifically 给出具体的解释**。）给定一个轨迹，可以用权重找到最重要的时间帧，我们可以确定与这些全局重要时间帧有高度相关的时间步长，并将它们放在一起作为对这一轨迹的局部解释。而轨迹之间的相关性 $K_e(\mathbf{X}, \mathbf{X})$ 可以用来一堆轨迹的种群特点，可以帮助分类出具有类似解释的轨迹。

3.4 后验推断和参数学习

3.4.1 带有诱导点的稀疏 GP

直接推断我们模型需要计算 $(K_{XX} + \sigma^2 I)^{-1}$，其计算复杂度为 $\mathcal{O}(NT^3)$，其计算复杂度非常的高。为了解决这个问题，采用一个基于 point 的变分推断方法。在较高的层次上，该方法简化了后验计算，将 X 中的有效样本数量从 NT 减少到 M，其中 M 为 point 的数量。具体的说，我们定义隐空间上的每个诱导点为：$\mathbf{z}_i \in \mathbb{R}^{2q}$，并且 $\mathbf{u}_i$ 为 GP 关于 z_i 的输出。那么，f 和 u 的联合先验和条件先验 f|u 为:

$$\mathbf{f}, \mathbf{u}|\mathbf{X}, \mathbf{Z} \sim \mathcal{N}\left(\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix}, \begin{bmatrix} K_{XX} & K_{XZ} \\ K_{XZ}^T & K_{ZZ} \end{bmatrix}\right), \mathbf{f}|\mathbf{u}, \mathbf{X}, \mathbf{Z} \sim \mathcal{N}\left(K_{XZ}K_{ZZ}^{-1}\mathbf{u}, K_{XX} - K_{XZ}K_{ZZ}^{-1}K_{XZ}^T\right) \tag{2}$$

其中，K_{XX}, K_{XZ}, K_{ZZ} 是协方差矩阵。这三个协方差矩阵，将应用我们的 additive 核函数于收集到的轨迹的时间桢和轨迹的 embedding 和 inducing point (我是真的不知道怎么翻译这个词比较好，而且看到这里我仍然没太懂这个 inducing point 是咋来的，不过我猜是用网络来计算一个表示) 来计算。注意观察公式 (2)，这样只需要求 K_{ZZ}^{-1}，将计算复杂度显著的由 $\mathcal{O}(NT^3)$ 缩减为 $\mathcal{O}(m^3)$。

3.4.2 变分推断和学习

我们的模型需要学习的参数：$\Theta_n = \{\phi, \phi_1\}$，GP 参数 $\Theta_k = \{\gamma_t, \gamma_e, \alpha_e, \alpha_t\}$，预测网络的参数 $\Theta_p = \{\mathbf{w}/\mathbf{W}, \sigma^2\}$，和 including points $\mathbf{Z} = \{\mathbf{z}_i\}_{i=1:M}$。我们使用贝叶斯方法，最大边缘分布似然函数：$\log p(y|\mathbf{X}, \mathbf{Z}, \Theta_n, \Theta_k, \Theta_p)$。最大化这个边缘似然函数非常的消耗计算资源，而且，如果是非高斯情况，这个基本不可解。为了实现对 $\log p(y|\mathbf{X}, \mathbf{Z}, \Theta_n, \Theta_k, \Theta_p)$ 的分解近似并实现高效学习，本文假设变分后验和 inducing 变量 $q(\mathbf{u}) \sim \mathcal{N}(\mu, \Sigma)$ 和已分解的联合后验 $q(\mathbf{f}, \mathbf{u}) = q(\mathbf{u})p(\mathbf{f}|\mathbf{u})$ 相关，其中 $p(\mathbf{f}|\mathbf{u})$ 在公

式 (2) 中。利用 Jensen 不等式，我们可以推出以下的 ELBO，

$$\log p(y|\mathbf{X},\mathbf{Z},\Theta_n,\Theta_k,\Theta_p) \geq \underbrace{\mathbb{E}_{q(\mathbf{f})}[\log p(y|\mathbf{f})] - \mathrm{KL}[q(\mathbf{u})\|p(\mathbf{u})]}_{\mathrm{ELBO}}, \tag{3}$$

其中，第一项是似然项，第二项是近似后验 $q(\mathbf{u})$ 和先验 $p(\mathbf{u})$ 之间的 KL 散度。最大化公式 (3) 的 ELBO 可以达到自动最大化边缘似然的目的 (这是变分推断里的常用套路)。证明过程如下所示。有几个地方相信大家可能有点疑问，关于第一行到第二行有一个关系：$p(y \mid \mathbf{f},\mathbf{X}) = p(y \mid \mathbf{f},,\mathbf{Z},\mathbf{X})$，这里 y 应该是只和 $\mathbf{f}$ 有关，实际上后面也是直接写成 $p(y \mid \mathbf{f})$。类似的写法文章中还要好几处。

Proof.

$$\begin{aligned}
\log p(y \mid \mathbf{X},\mathbf{Z}) &= \log \iint p(y,\mathbf{f},\mathbf{u} \mid \mathbf{X},\mathbf{Z})\mathrm{d}\mathbf{u}\mathrm{d}\mathbf{f} \\
&= \log \iint p(y \mid \mathbf{f},\mathbf{X})p(\mathbf{f},\mathbf{u} \mid \mathbf{X},\mathbf{Z})\mathrm{d}\mathbf{u}\mathrm{d}\mathbf{f} \\
&= \log \iint p(y \mid \mathbf{f},\mathbf{X})p(\mathbf{f},\mathbf{u} \mid \mathbf{X},\mathbf{Z})\frac{q(\mathbf{f},\mathbf{u})}{q(\mathbf{f},\mathbf{u})}\mathrm{d}\mathbf{u}\mathrm{d}\mathbf{f} \\
&= \log \iint p(y \mid \mathbf{f},\mathbf{X})q(\mathbf{f},\mathbf{u})\frac{p(\mathbf{f},\mathbf{u} \mid \mathbf{X},\mathbf{Z})}{q(\mathbf{f},\mathbf{u})}\mathrm{d}\mathbf{u}\mathrm{d}\mathbf{f} \\
&= \log \mathbb{E}_{q(\mathbf{f},\mathbf{u})}\left[p(y \mid \mathbf{f},\mathbf{X})\frac{p(\mathbf{f},\mathbf{u} \mid \mathbf{X},\mathbf{Z})}{q(\mathbf{f},\mathbf{u})}\right] \\
&\geq \mathbb{E}_{q(\mathbf{f},\mathbf{u})}\left[\log\left(p(y \mid \mathbf{f},\mathbf{X})\frac{p(\mathbf{f},\mathbf{u} \mid \mathbf{X},\mathbf{Z})}{q(\mathbf{f},\mathbf{u})}\right)\right] \\
&= \mathbb{E}_{q(\mathbf{f},\mathbf{u})}\left[\log p(y \mid \mathbf{f},\mathbf{X}) + \log\frac{p(\mathbf{f},\mathbf{u} \mid \mathbf{X},\mathbf{Z})}{q(\mathbf{f},\mathbf{u})}\right] \\
&= \mathbb{E}_{q(\mathbf{f},\mathbf{u})}[\log p(y \mid \mathbf{f},\mathbf{X})] - \mathbb{E}_{q(\mathbf{f},\mathbf{u})}\left[\log\frac{q(\mathbf{f},\mathbf{u})}{p(\mathbf{f},\mathbf{u} \mid \mathbf{X},\mathbf{Z})}\right] \\
&= \mathbb{E}_{q(\mathbf{f},\mathbf{u})}[\log p(y \mid \mathbf{f})] - \mathbb{E}_{q(\mathbf{f},\mathbf{u})}\left[\log\frac{p(\mathbf{f} \mid \mathbf{u})q(\mathbf{u})}{p(\mathbf{f} \mid \mathbf{u})p(\mathbf{u})}\right] \\
&= \mathbb{E}_{q(\mathbf{f},\mathbf{u})}[\log p(y \mid \mathbf{f})] - \int p(\mathbf{f} \mid \mathbf{u})\mathrm{d}\mathbf{f}\int\left[\log\frac{q(\mathbf{u})}{p(\mathbf{u})}\right]q(\mathbf{u})\mathrm{d}\mathbf{u} \\
&= \mathbb{E}_{q(\mathbf{f})}[\log p(y \mid \mathbf{f})] - \mathbb{KL}[q(\mathbf{u})\|p(\mathbf{u})],
\end{aligned}$$

并且，ELBO 也可以从最小化联合分布来推导（**啊这，这怎么感觉和 VAE 的套路一模一样，不过换了一种说法而已，VAE 也可以由直面联合分布推导出来。有兴趣的同学可以和 VAE 的推导对比一下试试。这里的 u 就类似于隐藏层变量**。），作者这里有一点笔误，这里做了一些小的变动，

$$\begin{aligned}
\mathbb{KL}[q(\mathbf{f},\mathbf{u})\|p(\mathbf{f},\mathbf{u} \mid y)] &= \iint q(\mathbf{f},\mathbf{u})\log\frac{q(\mathbf{f},\mathbf{u})}{p(\mathbf{f},\mathbf{u} \mid y)}\mathrm{d}\mathbf{u}\mathrm{d}\mathbf{f} \\
&= \iint q(\mathbf{f},\mathbf{u})\log\frac{q(\mathbf{f},\mathbf{u})p(y)}{p(y \mid \mathbf{f},\mathbf{u})p(\mathbf{f},\mathbf{u})}\mathrm{d}\mathbf{u}\mathrm{d}\mathbf{f} \\
&= \iint q(\mathbf{f},\mathbf{u})\left[\log\frac{1}{p(y \mid \mathbf{f},\mathbf{u})} + \log\frac{q(\mathbf{f},\mathbf{u})}{p(\mathbf{f},\mathbf{u})} + \log p(y)\right]\mathrm{d}\mathbf{u}\mathrm{d}\mathbf{f} \\
&= -\iint q(\mathbf{f},\mathbf{u})\left[\log p(y \mid \mathbf{f},\mathbf{u}) - \log\frac{q(\mathbf{f},\mathbf{u})}{p(\mathbf{f},\mathbf{u})}\right]\mathrm{d}\mathbf{u}\mathrm{d}\mathbf{f} + \log p(y) \\
&= -\iint q(\mathbf{f},\mathbf{u})\left[\log p(y \mid \mathbf{f},\mathbf{u}) - \log\frac{q(\mathbf{u})p(\mathbf{f}|\mathbf{u})}{p(\mathbf{u})p(\mathbf{f}|\mathbf{u})}\right]\mathrm{d}\mathbf{u}\mathrm{d}\mathbf{f} + \log p(y) \\
&= -\mathrm{ELBO} + \log p(y).
\end{aligned}$$

但是，如果指。我们首先计算 $\mathbf{f}$ 的边际变分后验分布，记为 $q(\mathbf{f}) = \mathcal{N}(\mu_f, \Sigma_f)$。那么下一个问题是，怎么求 μ_f 和 Σ_f。这里使用了一个白化的操作来实现协方差矩阵的对角化，来达到简便运算。具体来说，首先定义 $\mathbf{u} = \mathbf{L}\mathbf{v}$，而 $\mathbf{L}\mathbf{L}^T = K_{ZZ}$ 并且 $p(\mathbf{v}) = \mathcal{N}(0, \mathbf{I})$。所以，我们首先定义关于 $\mathbf{v}$ 的变分分布，$q(\mathbf{v}) = \mathcal{N}(\mu_v, \mathbf{S})$。那么可以得到，$q(\mathbf{u}) = \mathcal{N}(L\mu_v, L\mathbf{S}L^T)$。而，$q(\mathbf{f}, \mathbf{u}) = p(\mathbf{f}|\mathbf{u})q(\mathbf{u})$，$p(\mathbf{f}|\mathbf{u}) = \mathcal{N}(K_{XZ}K_{ZZ}^{-1}\mathbf{u}, K_{XX} - K_{XZ}K_{ZZ}^{-1}K_{XZ}^T)$，那么可以计算出 $q(\mathbf{f})$，

$$q(\mathbf{f}) = \int p(\mathbf{f} \mid \mathbf{u})q(\mathbf{u})\mathrm{d}\mathbf{u} = \mathcal{N}\left(K_{XZ}K_{ZZ}^{-1/2}\mu_v, K_{XX} + K_{XZ}K_{ZZ}^{-1/2}(\mathbf{S} - \mathbf{I})K_{ZZ}^{-1/2}K_{XZ}^T\right) \tag{4}$$

那么，$\nu_f = K_{XZ}K_{ZZ}^{-1/2}\mu_v$，$\Sigma_f = K_{XX} + K_{XZ}K_{ZZ}^{-1/2}(\mathbf{S} - \mathbf{I})K_{ZZ}^{-1/2}K_{XZ}^T$。害，明明 $q(\mathbf{f}, \mathbf{u}) = q(\mathbf{f}|\mathbf{u})q(\mathbf{u})$，而作者文章中写的是 $q(\mathbf{f}, \mathbf{u}) = p(\mathbf{f}|\mathbf{u})q(\mathbf{u})$，这里是用了一个近似，基于假设，$q(\mathbf{f}, \mathbf{u}) \approx p(\mathbf{f}, \mathbf{u})$。同样，近似一下也有 $q(\mathbf{f}) \approx \mathcal{N}(\mu_f, K_{XZ}K_{ZZ}^{-1/2}(\mathbf{S} - \mathbf{I})K_{ZZ}^{-1/2}K_{XZ}^T)$. 这样，白化之后，变分参数由 $\{\mu, \Sigma\}$ 转变为了 $\mu_v, \mathbf{S}$，然后 ELBO 中的 KL 项变成了 $\mathbb{KL}(q(\mathbf{v}) \| p(\mathbf{v}))$。（原来用 VAE 可以这么说，太牛了，这过程基本和 VAE 没太大的区别，换了一个说法！）

在计算出了 $q(\mathbf{f})$ 之后，可以采用重参数化技巧来实现从 $q(\mathbf{f})$ 中采样，也就是利用标准高斯实现从参数复杂的高斯分布中采样，其实这样做的主要目的是因为需要采样的分布中包含需要优化的参数，利用重参数化技巧可以优化到这一部分参数。定义 $\mathbf{f} = v(\epsilon_f) = \mu_f + \mathbf{L}_f\epsilon_f$，其中 $\epsilon \sim \mathcal{N}(0, \mathbf{I})$，$\mathbf{L}_f\mathbf{L}_f^{\mathrm{T}} = \Sigma_f$。这样，

$$\mathbb{E}_{q(\mathbf{f})}[\log p(y \mid \mathbf{f})] = \mathbb{E}_{\epsilon \sim p(\epsilon_f)}[\log p(y \mid v(\epsilon_f))] \approx \frac{1}{B}\sum_b \sum_i p(y_i|(\mathbf{F}^{(i)})^{(b)}) \tag{5}$$

注意 $\mathbf{f}$ 是 $\mathbf{F}$ 的 flatten 的形式。对于回归模型，可以直接计算出 ELBO 中 likelihood 项的解析形式，这里我们推导出了期望条件分布的解析解。具体的说，回归模型为，

$$\mathbf{f}|\mathbf{X} \sim \mathcal{N}(0, k = \alpha_t^2 k_{\gamma_t} + \alpha_e^2 k_{\gamma_e}), \quad y_i|\mathrm{f}^{(i)} \sim \mathcal{N}(\mathbf{f}^{(i)}\mathbf{w}^T, \sigma^2) \tag{6}$$

将高斯分布展开写可以得到，

$$\begin{aligned}\mathbb{E}_{q(\mathbf{f})}[\log p(y \mid \mathbf{f})] = \mathbb{E}_{q(\mathbf{f})}[\log p(y \mid \mathbf{f})] &= \mathbb{E}_{q(\mathbf{f})}\left[\frac{-N}{2}\left[\log\sigma^2 + \log 2\pi + \frac{1}{\sigma^2}\left(\mathbf{y} - \mathbf{F}\mathbf{w}^T\right)^T\left(\mathbf{y} - \mathbf{F}\mathbf{w}^T\right)\right]\right] \\ &= \frac{-N}{2}\left[\log\sigma^2 + \log 2\pi + \frac{1}{\sigma^2}\mathbb{E}_{q(\mathbf{f})}\left[\left(\mathbf{y} - \mathbf{F}\mathbf{w}^T\right)^T\left(\mathbf{y} - \mathbf{F}\mathbf{w}^T\right)\right]\right]\end{aligned} \tag{7}$$

其中，$\left(\mathbf{y} - \mathbf{F}\mathbf{w}^T\right)^T\left(\mathbf{y} - \mathbf{F}\mathbf{w}^T\right)$ 可以被计算为：

$$\begin{aligned}\mathbb{E}_{q(\mathbf{f})}\left[\left(\mathbf{y} - \mathbf{F}\mathbf{w}^T\right)^T\left(\mathbf{y} - \mathbf{F}\mathbf{w}^T\right)\right] &= \mathbb{E}_{q(\mathbf{f})}\left[\mathbf{y}^T\mathbf{y} - \mathbf{w}\mathbf{F}^T\mathbf{y} - \mathbf{y}^T\mathbf{F}\mathbf{w}^T + \mathbf{w}\mathbf{F}^T\mathbf{F}\mathbf{w}^T\right] \\ &= \mathbf{y}^T\mathbf{y} - \mathbf{w}\nu_f^T\mathbf{y} - \mathbf{y}^T\nu_f\mathbf{w}^T + \mathbf{w}\mathbb{E}_{q(\mathbf{f})}\left[\mathbf{F}^T\mathbf{F}\right]\mathbf{w}^T \\ &= \sum_i\left(y_i^2 - 2\nu_f^{(i)}\mathbf{w}^T\right) + \mathbf{w}\mathbb{E}_{q(\mathbf{f})}\left[\mathbf{F}^T\mathbf{F}\right]\mathbf{w}^T\end{aligned} \tag{8}$$

其中，$\nu_f \in \mathbb{R}^{N\times T}$ 是 μ_f 的矩阵形式。$\nu_f^{(i)} \in \mathbb{R}^{1\times T}$ 是 ν_f 矩阵的第 i 行，表示变分后验 $\mathbf{F}^{(i)}$ 的均值。而后一项则等于，

$$\mathbb{E}_{q(\mathbf{f})}[\mathbf{F}^T\mathbf{F}] = \sum_N\left[\Sigma_f^{(i)} + (\nu_f^{(i)})^T\nu_f^{(i)}\right] \tag{9}$$

其中，$\Sigma_f^{(i)} \in \mathbb{R}^{N\times T}$ 为变分后验 $\mathbf{F}^{(i)}$ 的协方差矩阵。那么，$\mathbb{E}_{q(\mathbf{f})}[\log p(y \mid \mathbf{f})]$ 可以写为，

$$\mathbb{E}_{q(\mathbf{f})}[\log p(y \mid \mathbf{f})] = -\frac{N}{2}\left[\log\sigma^2 + \log 2\pi + \frac{1}{\sigma^2}\sum_i\left((y_i - \nu_f^{(i)}\mathbf{w}^T)^2 + \mathbf{w}\Sigma_f^{(i)}\mathbf{w}^T\right)\right] \tag{10}$$

这样，利用随机梯度下降来最大化 ELBO 可以有效的求解模型参数：$\{\Theta_n, \Theta_k, \Theta_k, \Theta_p, \mathbf{Z}, \{\mu_v, \mathbf{S}\}\}$。

4 Discussion

虽然，这篇文章看着方法很复杂，其实和变分的文章相比，此文章用的方法并不难，而且用的变分方法也比较的老，在算法细节上没有什么改动，用法上是有改动的。其实，这篇文章读起来这么的清晰也得益于作者非常高超的写作功底，Guo Wenbo 的文章读上去都非常的清晰，而且实验量都非常大。而此类和强化学习测试，攻击的相关的工作，非常的关键一点是找到一个合适的问题，而可能解决这类问题的重要性会相对来说低一点。而所解决的问题一般会有一个 setting，而描述 setting 的合理性非常的重要，有时候可能找到了一个好的问题，比提出解决方法更加重要。