Ideas From
Deep Reinforcement Learning

Presenter: Kim Seung Hwan (overnap@khu.ac.kr)

Deep Reinforcement Learning

- Reinforcement learning (RL) is an area of machine learning concerned with how intelligent agents ought to take actions in an environment in order to maximize the notion of cumulative reward (Wikipedia definition)
- I took a reinforcement learning class last semester
- A few ideas that were borrowed into the CV field caught my eyes

Deep Reinforcement Learning

- Markov Process : $< S, P >$
- Markov Reward Process : $< S, P, R >$
- Markov Decision Process : $< S, A, P, R >$

- Markov property :

  “The future is independent of the past and dependent of the present”
- Markov assumption simplifies the problem

Deep Reinforcement Learning

- Let $V(s)$ be the value function – a function of the cumulative reward
- Bellman's equation : $E[V(s_t)] = E[R_t + \gamma V(s_{t+1})] = R_t + \gamma E[V(s_{t+1})]$

- Then, Let $Q(s, a)$ be the same as $V(s)$ but with a specific action
- $E[Q(s_t, a_t)] = R_t + \gamma E[Q(s_{t+1}, a^*_{t+1})]$
- * means it can be greedy argmax, average or etc.

- Let us train the $Q(s, a)$ modeled by DL
- With objective $Q(s_t, a_t) = E[R_t + \gamma Q(s_{t+1}, a^*_{t+1})]$

Deep Reinforcement Learning

- The model takes itself as the learning target
- It means that the object to be learned is moving

- This is called non-stationary problem, etc.
- It creates strong variance, like the mini-batch SGD
- We should manage the variance, just as we did with batch normalization for SGD

- Deep Q-Learning (DQN)

Deep Reinforcement Learning

- Let θ be the parameter of the model
- This is now the online/current parameter

- Let θ^- be the parameter of the 'target model'
- Then we can change the Q learning to:

$$Q(s_t, a_t, \theta) = E[R_t + \gamma Q(s_{t+1}, a^*_{t+1}, \theta^-)]$$

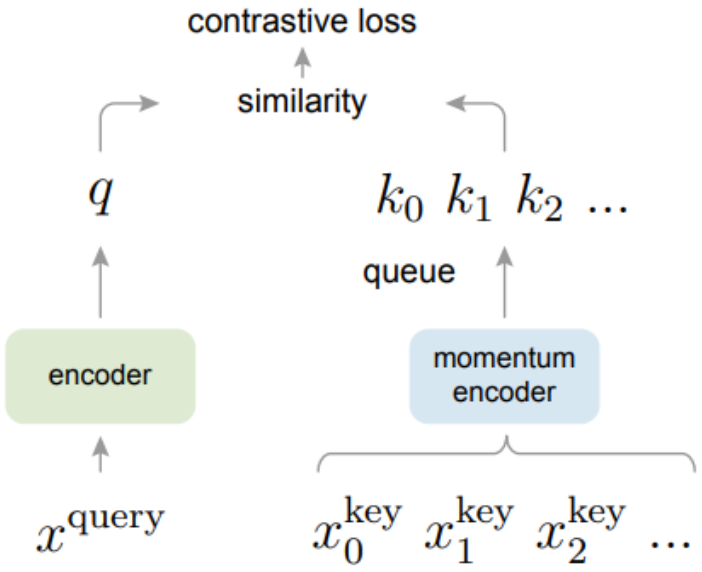
- The target model updates like:

$\theta^- = \theta$ at certain intervals (hard update)

$\theta^- = \tau\theta^- + (1 - \tau)\theta$ at certain intervals (soft/momentum update)

Contrastive Learning

- Momentum Contrast (MoCo)
- Bootstrap Your Own Latent (BYOL)
- Consistency Model



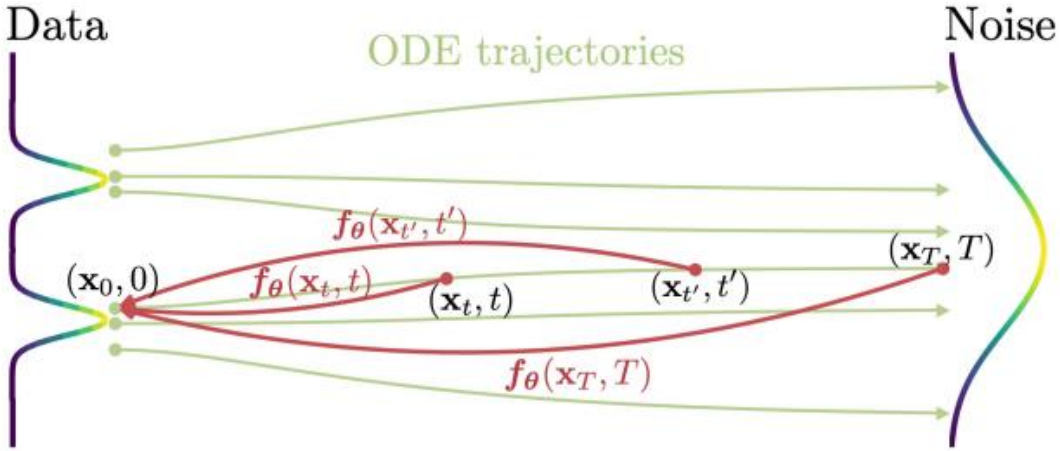


Figure 2: Consistency models are trained to map points on any trajectory of the PF ODE to the trajectory's origin.

Contrastive Learning

Algorithm 3 Consistency Training (CT)

Input: dataset $\mathcal{D}$, initial model parameter $\boldsymbol{\theta}$, learning rate η, step schedule $N(\cdot)$, EMA decay rate schedule $\mu(\cdot)$, $d(\cdot,\cdot)$, and $\lambda(\cdot)$

$\boldsymbol{\theta}^- \leftarrow \boldsymbol{\theta}$ and $k \leftarrow 0$

repeat

Sample $\mathbf{x} \sim \mathcal{D}$, and $n \sim \mathcal{U}[\![1, N(k)-1]\!]$

Sample $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$

$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\theta}^-) \leftarrow$

$\lambda(t_n) d(\boldsymbol{f}_{\boldsymbol{\theta}}(\mathbf{x} + t_{n+1}\mathbf{z}, t_{n+1}), \boldsymbol{f}_{\boldsymbol{\theta}^-}(\mathbf{x} + t_n \mathbf{z}, t_n))$

$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\theta}^-)$

$\boldsymbol{\theta}^- \leftarrow \text{stopgrad}(\mu(k)\boldsymbol{\theta}^- + (1-\mu(k))\boldsymbol{\theta})$

$k \leftarrow k+1$

until convergence

rate $0 \leqslant \mu < 1$, we perform the following update after each optimization step:

$$\boldsymbol{\theta}^- \leftarrow \text{stopgrad}(\mu\boldsymbol{\theta}^- + (1-\mu)\boldsymbol{\theta}). \tag{8}$$

The overall training procedure is summarized in Algorithm 2. In alignment with the convention in deep reinforcement learning (Mnih et al., 2013; 2015; Lillicrap et al., 2015) and momentum based contrastive learning (Grill et al., 2020; He et al., 2020), we refer to $\boldsymbol{f}_{\boldsymbol{\theta}^-}$ as the "target network", and $\boldsymbol{f}_{\boldsymbol{\theta}}$ as the "online network". We find that compared to simply setting $\boldsymbol{\theta}^- = \boldsymbol{\theta}$, the EMA update and "stopgrad" operator in Eq. (8) can greatly stabilize the training process and improve the final performance of the consistency model.

Actor Critic

- Actor $\pi_\theta(s)$ directed by Critic $Q_\phi(s,a) \approx Q^{\pi_\theta}(s,a)$
- Actor who decides actions and Critic who judges values learn from each other
- The variance will increase here again!

- There are numerous studies to solve this…
- Twin Delayed Deep Deterministic Policy Gradient (TD3)

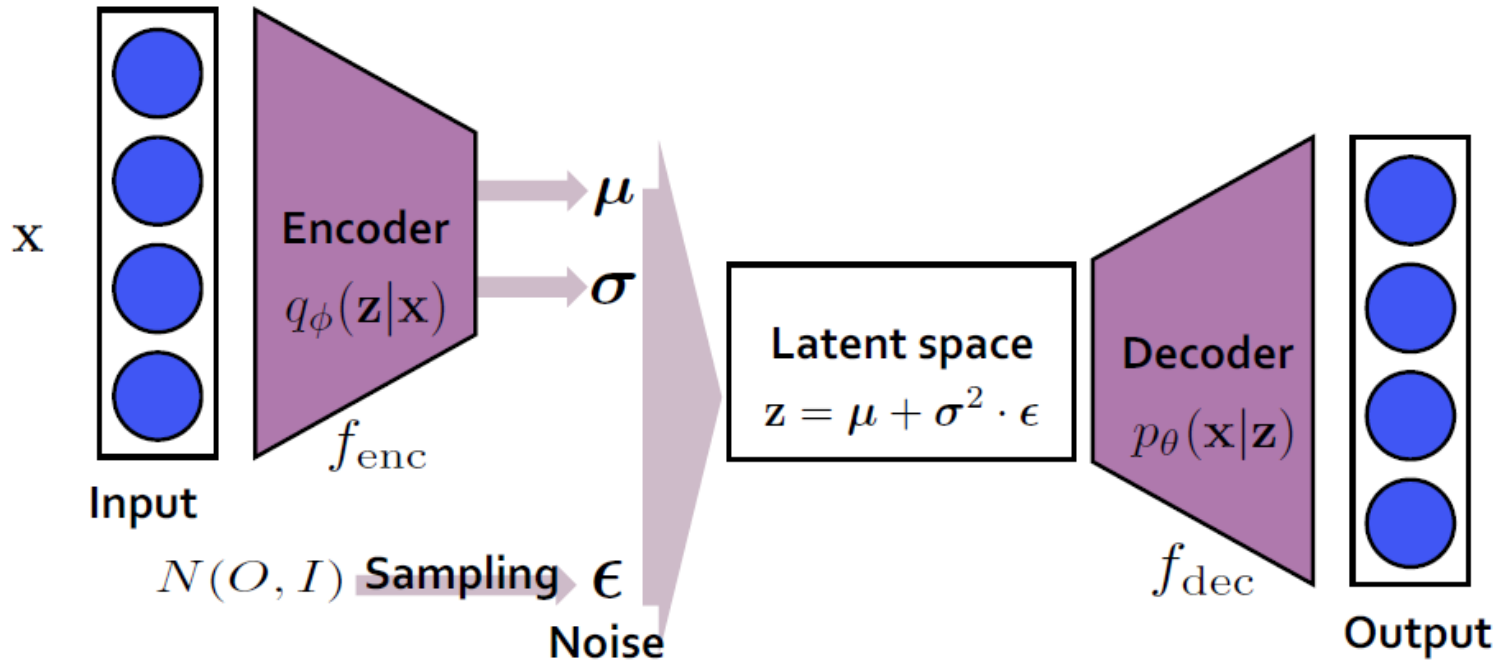
Actor Critic

- The variance explodes when both learn from each other in unstable states
- At least one of them should be stabilized

- Updates $Q_\phi(s, a)$ continuously, but $\pi_\theta(s)$ at intervals
- It just seems like it would slow down the training process
- But it works great anyway

Training Dynamics

- Complex models for high-level tasks
- Generation models e.g. VAE, GAN



Training Dynamics

- Lagging Inference Networks And Posterior Collapse In Variational Autoencoders

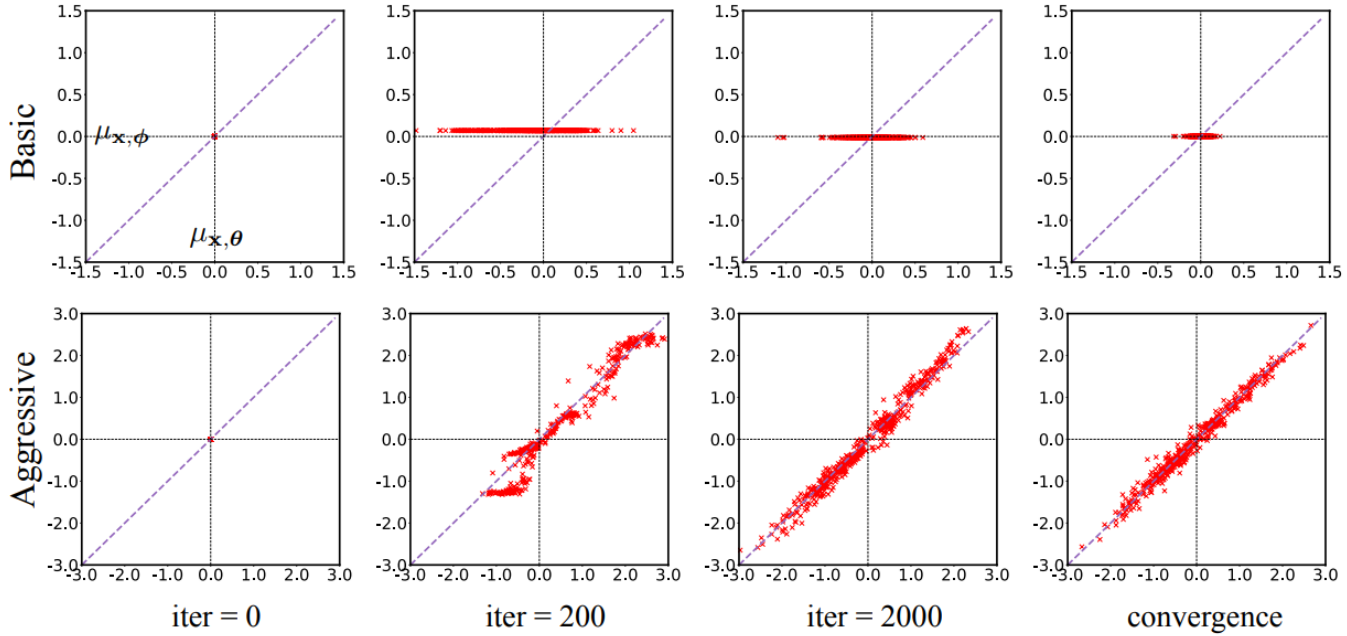


Figure 2: The projections of 500 data samples from a synthetic dataset on the posterior mean space over the course of training. "iter" denotes the number of updates of generators. The top row is from the basic VAE training, the bottom row is from our aggressive inference network training. The results show that while the approximate posterior is lagging far behind the true model posterior in basic VAE training, our aggressive training approach successfully moves the points onto the diagonal line and away from inference collapse.

Training Dynamics

- Lagging Inference Networks And Posterior Collapse In Variational Autoencoders

Algorithm 1 VAE training with controlled aggressive inference network optimization.

$\boldsymbol{\theta}, \boldsymbol{\phi} \leftarrow$ Initialize parameters
$aggressive \leftarrow$ TRUE
repeat
  if $aggressive$ **then**
    repeat ▷ [aggressive updates]
      $\mathbf{X} \leftarrow$ Random data minibatch
      Compute gradients $\boldsymbol{g}_{\boldsymbol{\phi}} \leftarrow \nabla_{\boldsymbol{\phi}} \mathcal{L}(\mathbf{X}; \boldsymbol{\theta}, \boldsymbol{\phi})$
      Update $\boldsymbol{\phi}$ using gradients $\boldsymbol{g}_{\boldsymbol{\phi}}$
    until convergence
    $\mathbf{X} \leftarrow$ Random data minibatch
    Compute gradients $\boldsymbol{g}_{\boldsymbol{\theta}} \leftarrow \nabla_{\boldsymbol{\theta}} \mathcal{L}(\mathbf{X}; \boldsymbol{\theta}, \boldsymbol{\phi})$
    Update $\boldsymbol{\theta}$ using gradients $\boldsymbol{g}_{\boldsymbol{\theta}}$
  else ▷ [basic VAE training]
    $\mathbf{X} \leftarrow$ Random data minibatch
    Compute gradients $\boldsymbol{g}_{\boldsymbol{\theta},\boldsymbol{\phi}} \leftarrow \nabla_{\boldsymbol{\phi},\boldsymbol{\theta}} \mathcal{L}(\mathbf{X}; \boldsymbol{\theta}, \boldsymbol{\phi})$
    Update $\boldsymbol{\theta}, \boldsymbol{\phi}$ using $\boldsymbol{g}_{\boldsymbol{\theta},\boldsymbol{\phi}}$
  end if
  Update $aggressive$ as discussed in Section 4.2
until convergence

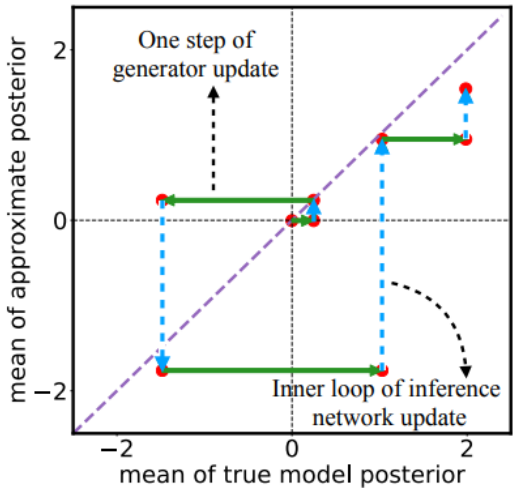


Figure 3: Trajectory of one data instance on the posterior mean space with our aggressive training procedure. Horizontal arrow denotes one step of generator update, and vertical arrow denotes the inner loop of inference network update. We note that the approximate posterior $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$ takes an aggressive step to catch up to the model posterior $p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$.

Actor Critic

- In the TD3 paper, the authors express an action value as $N(\mu, \sigma^2 I)$
- And they train this with reparameterization trick, citing VAE

- Each field influences each other
- Interesting
- But on the other hand, I feel the need for a wide and diverse study…

Thank you for listening

Presenter: Kim Seung Hwan (overnap@khu.ac.kr)