CS5340: Uncertainty Modeling in AI

Tutorial 9

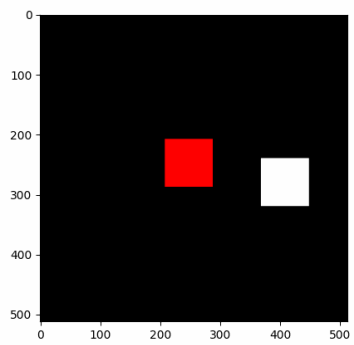
Released: Mar. 27, 2024

Sequential VAE

In Tutorial 7, we studied a state-space model where the transitions and emissions were linear functions, along with Gaussian random variables. This resulted in a linear Gaussian model, which made inference and learning tractable.

In this first part of the tutorial, we will extend the linear Gaussian model to the nonlinear regime: we will look at a (relatively) state-of-the-art Sequential VAE model. We will combine deep learning (neural networks) with the state-space model to yield a more expressive sequential model capable of learning from complex high-dimensional image observations.

Problem and Environment Suppose we have a robot (represented by the white rectangle) that moves in a 2D room. The task is to control the robot to reach the goal position (represented by the red rectangle) at the center of the room. We can directly control the velocity of the robot along x and y axes. However, we cannot directly observe the ground truth coordinates of the robot nor the goal. We only have access to high dimensional pixel (image) observations, as shown in the image below:



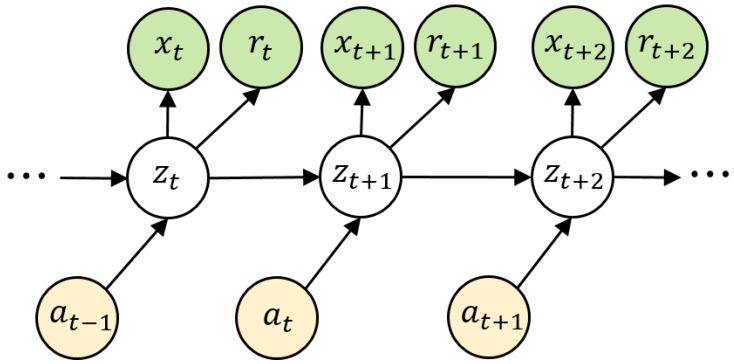
Note that the robot doesn't "know" it is the while square or that the goal is the red square. We have to learn these associations using data.

Overview Here, we will work through three main steps. First, we will construct a probabilistic model (directed graphical model) for this problem. Then we will use approximate inference to learn the parameters of this model. Finally, we will use this model to control the robot!

Learning the Model

Let us first define our State Space Model (SSM). Intuitively, we will model an agent that is sequentially taking actions in a world and receiving rewards and visual observations. The visual/image observation x_t at time t are generated from the latent state z_t. The model assumes Markovian transitions where the next state is conditioned upon the current state and the action a_t taken by the agent. Upon taking an action, the agent receives reward r_t. The goal of the agent is to maximize its rewards.

Problem 1. The problem above can be formulated as a Bayesian network:



Each of the factorized distributions are modelled using nonlinear functions:

- Transitions: $p_\theta(z_t|z_{t-1}, a_{t-1}) = p(z_t|f_\theta(z_{t-1}, a_{t-1}))$
- Observations: $p_\theta(x_t|z_t) = p(x_t|d_\theta(z_t))$
- Rewards: $p_\theta(r_t|z_t) = p(r_t|r_\theta(z_t))$

where f_θ, d_θ^m, r_θ are neural networks parameterized by θ. First, consider that the actions are always observed. Write out the factorization of the probability $p_\theta(x_{1:T}, r_{1:T}, z_{1:T}|a_{1:T-1})$ corresponding to the DGM above.

Problem 2. Learning this model is intractable due to the nonlinear transition, observation, and reward functions. We will perform variational inference to learn the parameters of the model. Assume we observe trajectories τ sampled from data distribution $p_d(\tau)$. Each trajectory is an observation $\tau = \{(x_t, r_t, a_t)\}_{t=1}^T$.

To obtain the maximum likelihood estimate (MLE) of the parameters θ, which of the following functions should we optimize?

A. $\mathbb{E}_{p_d}[\log p(x_{1:T}, r_{1:T}|a_{1:T-1}; \theta)]$

B. $\mathbb{E}_{p_d}[\log p(x_{1:T}, r_{1:T}, z_{1:T}|a_{1:T-1}; \theta)]$

C. $\mathbb{E}_{p_d}[\log p(\theta|x_{1:T}, r_{1:T}, z_{1:T}, a_{1:T-1})]$

D. $\mathbb{E}_{p_d}[\log p(\theta, x_{1:T}|r_{1:T}, z_{1:T}, a_{1:T-1})]$

E. Any of the above would work.

Solve this problem before moving to the next one.

Problem 3. Note that the maximum likelihood estimation requires us to marginalize out the latent variables $z^i_{1:T}$ for each trajectory τ^i in a dataset $\mathcal{D}$. We will need the variational posterior q. Consider three choices:

A. $q(z^i_{1:T}) = \prod_{t=1} q(z^i_t)$ where the q's are Gaussian distribution that share the same parameters (mean and covariance).

B. $q(z^i_{1:T}) = \prod_{t=1} q^i_t(z^i_t)$ where each q^i_t is a Gaussian distribution with *different* parameters.

C. $q(z^i_{1:T}|x^i_{1:T}, a^i_{1:T-1}) = \prod_{t=1}^{T} q_\phi(z^i_t|g_\phi(x^i_{1:t}, a^i_{1:t-1}))$ where q_ϕ is a Gaussian distribution and the *inference network* $g_\phi(x_{1:t}, a_{1:t-1})$ is a neural network (usually a recurrent neural network like a LSTM or GRU) parameterized by ϕ that outputs the mean and covariance for each z^i_t. The inference networks provides the parameters for the mean and the covariance of the distributions.

Between A, B and C, which variational distribution is the least expressive? Which is the most expressive?

Problem 4. Consider the variational distribution given in C above, i.e.,

$$q(z^i_{1:T}|x^i_{1:T}, a^i_{1:T-1}) = \prod_{t=1}^{T} q_\phi(z^i_t|g_\phi(x^i_{1:t}, a^i_{1:t-1}))$$

where each q_ϕ is a Gaussian distribution and the *inference network* $g_\phi(x_t, z_{t-1}, a_{t-1})$ is a neural network parameterized by ϕ. The inference network provides the parameters for the mean and the covariance of the distributions. Is $q(z^i_{1:T}|x^i_{1:T}, a^i_{1:T-1})$ a multivariate Gaussian in general? Provide a brief justification.

Problem 5. Suppose we pick the inference network variational distribution given in C above. To simplify notation, we call this $q_\phi(z_t)$. Given all these distributions and trajectories $\tau \sim p_d(\tau)$, we seek to learn the parameters θ and ϕ. We optimize the evidence lower bound (ELBO) under the data distribution p_d using a variational distribution q_ϕ over the latent state variables z_t.

$$\mathbb{E}_{p_d}[\text{ELBO}] \leq \mathbb{E}_{p_d}[\log p_\theta(x_{1:T}, r_{1:T}|a_{1:T-1})] \tag{1}$$

where

$$\text{ELBO} = \sum_{t=1}^{T} \left(\mathop{\mathbb{E}}_{q_\phi(z_t)} [\log p_\theta(x_t|z_t)] + \mathop{\mathbb{E}}_{q_\phi(z_t)} [\log p_\theta(r_t|z_t)] \right) \tag{2}$$

$$- \sum_{t=2}^{T} \mathop{\mathbb{E}}_{q_\phi(z_{t-1})} [\text{KL}\left[q_\phi(z_t) \| p_\theta(z_t|z_{t-1}, a_{t-1})\right]] - \text{KL}\left[q_\phi(z_1) \| p_\theta(z_1)\right] \tag{3}$$

Note that we have dropped the explicit conditioning to reduce clutter in the above equation, i.e., $q_\phi(z_t) = q_\phi(z_t|x_{1:t}, a_{1:t-1})$. Derive the ELBO shown above.

Planning and Control

With the ELBO, we can learn the parameters θ (and ϕ) using an off-the-shelf-optimizer like stochastic gradient descent (SGD). Once we have learnt the model, we can use it for planning/control. The idea is quite simple. Say we are at current time step t, we will use the model to simulate possible futures (up to some horizon $t + H$) and find actions that lead to the best return (the sum of discounted rewards).

$$\text{argmax}_{a_{t:t+H-1}} J = \text{E}_{p(z_{t+1:t+H}|a_{t:t+H-1},z_t)} \left[\sum_{k=1}^{H} \gamma^k r(z_{t+k}) \right] \tag{4}$$

We then take action a_t and then repeat the process. We'll use a simple zero-order method called the cross-entropy method (CEM), which we will demonstrate during tutorial (but will not go into detail here since it is beyond the scope of the course[1]).

[1]For those who are interested, check out: `https://jetnew.io/blog/2021/cem/`

Diffusion Model

Diffusion models are a class of latent variable generative models that have garnered significant attention, with prominent examples including DALL·E and Stable Diffusion for image generation. In this question, we shall explore a specific formulation of diffusion model known as *Denoising Diffusion Probabilistic Model* (DDPM).

Similar to various other generative models, the objective in DDPM is to learn a model p_θ for some data distribution q_{data}, from which we can sample with. There are two key components to DDPM: the forward diffusion process q and the reverse diffusion process p_θ. Both processes are formulated as Markov chains, as illustrated in Figure 1.

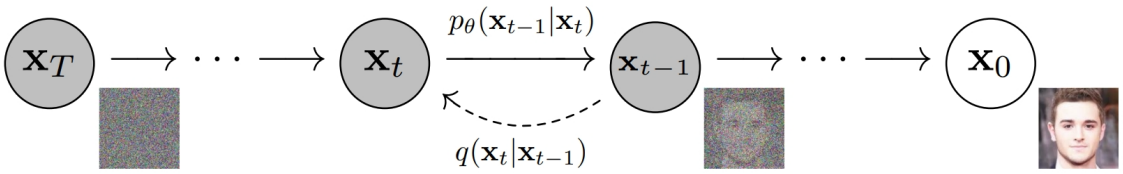


Figure 1: Graphical model for DDPM. The forward process q is denoted with dashed lines, and the reverse process p_θ is denoted with solid lines.

Intuitively, we will choose the forward process q to be a simple process that **does not require learning**, and we train the parameters θ of the model p_θ to reverse the process q.

Forward Process

The forward process q progressively adds Gaussian noise (over T time steps) to the data. More specifically, let $q(\mathbf{x}_0) = q_{\text{data}}(\mathbf{x}_0)$ be the data distribution and $\mathbf{x}_0 \sim q(\mathbf{x}_0)$. For each time step $t \in \{1, \ldots, T\}$, let

$$\mathbf{x}_t = \sqrt{\alpha_t}\mathbf{x}_{t-1} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_t \tag{5}$$

where $\{\alpha_t\}_{t=1}^T$ is a set of hyperparameters and $\boldsymbol{\epsilon}_t \sim \mathcal{N}(\boldsymbol{\epsilon}_t|0, \mathbf{I})$.

Problem 6. Fill in the blanks: $q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t|\_\_\_, \_\_\_)$.

Problem 7. With reference to Figure 1, write down the joint distribution $q(\mathbf{x}_0, \ldots, \mathbf{x}_T)$ in terms of $q(\mathbf{x}_0)$ and $q(\mathbf{x}_t|\mathbf{x}_{t-1})$.

Problem 8. Rather than sampling $\mathbf{x}_t$ by progressively adding noise over t time steps, we can directly sample $\mathbf{x}_t$ from $\mathbf{x}_0$. Let $\bar{\alpha}_t = \prod_{i=1}^t \alpha_i$. Prove that $q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t|\sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I})$.

Hint: You can prove this by induction with Equation 5 and the inductive hypothesis.

Problem 9. Suppose that $0 < \alpha_t < 1$ for all $t \in \{1, \ldots, T\}$ and T is sufficiently large. Justify why the distribution of the final time step $q(\mathbf{x}_T)$ is approximately $\mathcal{N}(\mathbf{x}_T|0, \mathbf{I})$.

Reverse Process

For each $t \in \{1, \ldots, T\}$, the reverse process $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$ attempts to "denoise" the noise that is added during the forward process $q(\mathbf{x}_t|\mathbf{x}_{t-1})$. More formally, this process starts from $\mathbf{x}_T$, which we sample from $\mathcal{N}(\mathbf{x}_T|0, \mathbf{I})$ using the justification made in the previous problem. Then, for each step $t \in \{1, \ldots, T\}$, it samples $\mathbf{x}_{t-1} \sim p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$ until we obtained a sample $\mathbf{x}_0$.

Problem 10. With reference to Figure 1, write down the joint distribution $p_\theta(\mathbf{x}_0, \ldots, \mathbf{x}_T)$ in terms of $p(\mathbf{x}_T)$ and $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$.

Training DDPM

To train the reverse process p_θ, we shall minimize the expected log-likelihood $\mathbb{E}_{\mathbf{x}_0 \sim q(\mathbf{x}_0)}[-\log p_\theta(\mathbf{x}_0)]$. Over the next few questions, we will progress through guided steps to derive the training objective for DDPM.

Problem 11. Denote $\mathbf{x}_{s:t} = \{\mathbf{x}_s, \ldots, \mathbf{x}_t\}$ for $s < t$. Justify that

$$\mathbb{E}_{\mathbf{x}_0 \sim q(\mathbf{x}_0)}[-\log p_\theta(\mathbf{x}_0)] \leq \mathbb{E}_{\mathbf{x}_{0:T} \sim q(\mathbf{x}_{0:T})}\left[-\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right].$$

Hint: Note that $\mathbf{x}_{1:T}$ are latent variables in our model.

Problem 12. By expanding p and q, show that

$$\mathbb{E}_{\mathbf{x}_{0:T} \sim q(\mathbf{x}_{0:T})}\left[-\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] = \mathbb{E}_{\mathbf{x}_{0:T} \sim q(\mathbf{x}_{0:T})}\left[-\log p(\mathbf{x}_T) - \sum_{t=2}^{T} \log \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_t|\mathbf{x}_{t-1})} - \log \frac{p_\theta(\mathbf{x}_0|\mathbf{x}_1)}{q(\mathbf{x}_1|\mathbf{x}_0)}\right].$$

Problem 13. Since the forward process is a Markov chain, by the Markovian property, we have $q(\mathbf{x}_t|\mathbf{x}_{t-1}) = q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0)$. Using this observation, continue the above derivation to show that

$$\mathbb{E}_{\mathbf{x}_{0:T} \sim q(\mathbf{x}_{0:T})}\left[-\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] = \mathbb{E}_{\mathbf{x}_{0:T} \sim q(\mathbf{x}_{0:T})}\left[-\log \frac{p(\mathbf{x}_T)}{q(\mathbf{x}_T|\mathbf{x}_0)} - \sum_{t=2}^{T} \log \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)} - \log p_\theta(\mathbf{x}_0|\mathbf{x}_1)\right].$$

Problem 14. Verify that

$$\begin{aligned}
&\mathbb{E}_{\mathbf{x}_{0:T} \sim q(\mathbf{x}_{0:T})}\left[-\log \frac{p(\mathbf{x}_T)}{q(\mathbf{x}_T|\mathbf{x}_0)} - \sum_{t=2}^{T} \log \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)} - \log p_\theta(\mathbf{x}_0|\mathbf{x}_1)\right] \\
=&\mathbb{E}_{\mathbf{x}_{0:T} \sim q(\mathbf{x}_{0:T})}\left[\underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_T|\mathbf{x}_0)\|p(\mathbf{x}_T))}_{L_T} + \sum_{t=2}^{T} \underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)\|p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t))}_{L_{t-1}} - \underbrace{\log p_\theta(\mathbf{x}_0|\mathbf{x}_1)}_{L_0}\right]. \qquad (6)
\end{aligned}$$

Problem 15. The objective is composed of three sets of terms: L_T, $L_{1:T-1}$, L_0. The term L_T does not have any learnable parameters, so we can ignore it. We defer the discussion of L_0 to Section 3.3 of arXiv:2006.11239. In this problem, let us focus on

$$L_{t-1} = \mathbb{E}_{\mathbf{x}_{0:T} \sim q(\mathbf{x}_{0:T})}[D_{\mathrm{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \| p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t))]$$

for each $t \in \{2, \dots, T\}$. It can be shown that $q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)$ is a Gaussian distribution $\mathcal{N}(\mathbf{x}_{t-1}|\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0), \tilde{\alpha}_t \mathbf{I})$ where

$$\begin{aligned}
\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0) &= \frac{\sqrt{\alpha_t}(1 - \bar{\alpha}_{t-1})}{1 - \bar{\alpha}_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}(1 - \alpha_t)}{1 - \bar{\alpha}_t}\mathbf{x}_0 \\
\tilde{\alpha}_t(\mathbf{x}_t, \mathbf{x}_0) &= \frac{(1 - \alpha_t)(1 - \bar{\alpha}_{t-1})}{1 - \bar{\alpha}_t}.
\end{aligned}$$

The full derivation of this can be found in Equation 71 to 84 of arXiv:2208.11970.

Suppose we model $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$ as $\mathcal{N}(\mathbf{x}_{t-1}|\hat{\boldsymbol{\mu}}_\theta(\mathbf{x}_t, t), \tilde{\alpha}_t \mathbf{I})$ where $\hat{\boldsymbol{\mu}}_\theta(\mathbf{x}_t, t)$ is a learnable function. Given that the closed-form expression for KL divergence between two Gaussian distribution is

$$D_{\mathrm{KL}}(\mathcal{N}(\cdot|\boldsymbol{\mu}_1, \sigma_1 \mathbf{I}) \| \mathcal{N}(\cdot|\boldsymbol{\mu}_2, \sigma_2 \mathbf{I})) = \frac{1}{2}\left[k \log \frac{\sigma_2^2}{\sigma_1^2} - k + \frac{\|\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\|_2^2}{\sigma_2^2} + k\frac{\sigma_1^2}{\sigma_2^2}\right],$$

where k is the number of dimensions of $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$, show that

$$L_{t-1} = \mathbb{E}_{\mathbf{x}_0, \mathbf{x}_t \sim q(\mathbf{x}_0, \mathbf{x}_t)}\left[\frac{1}{2\tilde{\alpha}_t}\left\|\tilde{\boldsymbol{\mu}}(\mathbf{x}_t, \mathbf{x}_0) - \hat{\boldsymbol{\mu}}_\theta(\mathbf{x}_t, t)\right\|_2^2\right].$$

Problem 16. Suppose we parameterize the learned function $\hat{\boldsymbol{\mu}}_\theta(\mathbf{x}_t, t)$ to match the form of $\tilde{\boldsymbol{\mu}}_\theta(\mathbf{x}_t, \mathbf{x}_0)$. In particular, we let

$$\hat{\boldsymbol{\mu}}_\theta(\mathbf{x}_t, t) = \frac{\sqrt{\alpha_t}(1 - \bar{\alpha}_{t-1})}{1 - \bar{\alpha}_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}(1 - \alpha_t)}{1 - \bar{\alpha}_t}\hat{\mathbf{x}}_\theta(\mathbf{x}_t, t)$$

where $\hat{\mathbf{x}}_\theta(\mathbf{x}_t, t)$ is a learnable function. Using this parameterization, show that

$$\mathbb{E}_{\mathbf{x}_0, \mathbf{x}_t \sim q(\mathbf{x}_0, \mathbf{x}_t)}\left[\frac{1}{2\tilde{\alpha}_t}\frac{\sqrt{\bar{\alpha}_{t-1}}(1 - \alpha_t)}{1 - \bar{\alpha}_t}\left\|\mathbf{x}_0 - \hat{\mathbf{x}}_\theta(\mathbf{x}_t, t)\right\|_2^2\right].$$

Problem 17. In practice, how can we minimize L_{t-1}?