CARNEGIE MELLON UNIVERSITY 07-380
HOMEWORK 5

https://www.cs.cmu.edu/~07380

INSTRUCTIONS

- **Format:** Use the provided LaTeX template to write your answers in the appropriate locations within the *.tex files and then compile a pdf for submission. We try to mark these areas with STUDENT SOLUTION HERE comments. Make sure that you don't change the size or location of any of the answer boxes and that your answers are within the dedicated regions for each question/part. If you do not follow this format, we may deduct points.

  You may also digitally annotate the pdf. Illegible handwriting will lead to lost points. However, we suggest that you try to do at least some of your work directly in LaTeX.

- **How to submit written component:** Submit to Gradescope a pdf with your answers. Again, make sure your answer boxes are aligned with the original pdf template.

- **Policy:** See the course website for homework policies, including late policy, and academic integrity policies.

Name	
Andrew ID	
Hours to complete all components (nearest hour)	

1 [8 pts] Variational Autoencoders

1.1 [5 pts] Close to standard normal

Derive the KL divergence between a univariate Gaussian distribution with mean μ and variance σ^2 and a standard normal Gaussian (mean 0 and variance 1). That is, show the following:

$$D_{KL}(\mathcal{N}(\mu, \sigma^2) \parallel \mathcal{N}(0, 1)) = \frac{1}{2}\left(\sigma^2 + \mu^2 - 1 - \log(\sigma^2)\right)$$

Hints: The following will be useful: the definition of variance; and the fact that $\mathbb{E}[X^2] = \mu^2 + \sigma^2$

Your Answer

1.2 [3 pts] KL divergence is non-negative

Prove that KL divergence is always non-negative, i.e. show that:

$$D_{KL}(p(x) \parallel q(x)) \geq 0$$

In order to prove this, use the fact that the negative of the log function is a convex function and apply Jensen's inequality for convex functions.

Your Answer

2 [5 pts] Diffusion

2.1 [5 pts] Jumping straight to step t

The forward (noising) process is $q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}\left(\sqrt{\alpha_t}\,\mathbf{x}_{t-1}, (1-\alpha_t)\mathbf{I}\right)$ for $t = 1, \ldots, T$.

1. **[3 pts]** Show by induction that $q(\mathbf{x}_t \mid \mathbf{x}_0) = \mathcal{N}\left(\sqrt{\bar{\alpha}_t}\,\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I}\right)$, where $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$.
   Hint: the sum of independent Gaussians $\mathcal{N}(0, a\mathbf{I})$ and $\mathcal{N}(0, b\mathbf{I})$ is $\mathcal{N}(0, (a+b)\mathbf{I})$.

Your Answer

2. **[2 pts]** With $T = 1000$ and $\alpha_t = 1 - \beta_t$, where β_t increases linearly from 10^{-4} to 0.02, the product works out to $\bar{\alpha}_T \approx 4 \times 10^{-5}$ (you can check this with $\log(1-\beta) \approx -\beta$: $\log \bar{\alpha}_T \approx -\sum_t \beta_t \approx -10$). What does this value say about $\mathbf{x}_T$, and why does that let us start sampling from pure noise?

Your Answer

3 [7 pts] REINFORCE with Baseline

The following algorithm describes REINFORCE with a baseline, where we introduce a baseline function $b(s_t)$ by replacing $\bar{R}_t$ in REINFORCE with $\bar{R}_t - b(s_t)$.

A common choice for $b(s_t)$ is the value function $b(s_t) = V_\phi(s_t)$ (this is the baseline used in the pseudocode shown in lecture, and displayed here).

procedure REINFORCEWITHBASELINE($\eta_\theta, \eta_\phi, \gamma$)
  Initialize policy parameters θ and baseline parameters ϕ randomly
  while not converged **do**
    Sample a trajectory $\tau = [(s_0, a_0, r_0), \ldots, (s_T, a_T, r_T)]$ from π_θ
    $g_\theta \leftarrow \sum_{t=0}^{T} (\bar{R}_t - V_\phi(s_t)) \nabla_\theta \log \pi_\theta(a_t \mid s_t)$ where $\bar{R}_t = \sum_{t'=t}^{T} \gamma^{t'-t} r_{t'}$
    $g_\phi \leftarrow \sum_{t=0}^{T} \nabla_\phi \left(V_\phi(s_t) - \bar{R}_t\right)^2$
    $\theta \leftarrow \theta + \eta_\theta g_\theta$
    $\phi \leftarrow \phi - \eta_\phi g_\phi$
  end while
  return π_θ, V_ϕ
end procedure

We already know that REINFORCE is unbiased but suffers from high variance. Now, we will explore REINFORCE with a baseline.

Walkthrough: the baseline does not introduce bias (nothing to submit)

The estimate g_θ above differs from REINFORCE's by the baseline term $-\sum_t b(s_t)\, \nabla_\theta \log \pi_\theta(a_t \mid s_t)$. The estimator therefore stays unbiased exactly when this term has expectation zero over trajectories $\tau \sim P_\theta(\tau)$. The trajectory distribution factorizes as

$$P_\theta(\tau) = p(s_0) \prod_t \pi_\theta(a_t \mid s_t)\, p(s_{t+1} \mid s_t, a_t),$$

so the state and the action at step t of a sampled trajectory have the joint distribution $P_\theta(s_t)\, \pi_\theta(a_t \mid s_t)$: the state is reached first, with a marginal distribution $P_\theta(s_t)$ that depends on θ through all of the earlier actions, and then the action is drawn from the policy at that state. We can therefore take the expectation in two stages (iterated expectations), first over s_t and then over a_t given s_t:

$$\begin{aligned}
&\mathbb{E}_{\tau \sim P_\theta(\tau)}[b(s_t)\, \nabla_\theta \log \pi_\theta(a_t \mid s_t)] \\
&\quad = \mathbb{E}_{s_t \sim P_\theta(s_t)}\big[\, \mathbb{E}_{a_t \sim \pi_\theta(\cdot \mid s_t)}[b(s_t)\, \nabla_\theta \log \pi_\theta(a_t \mid s_t) \mid s_t]\big] \\
&\quad = \mathbb{E}_{s_t \sim P_\theta(s_t)}\left[b(s_t) \sum_a \pi_\theta(a \mid s_t)\, \nabla_\theta \log \pi_\theta(a \mid s_t) \right] \\
&\quad = \mathbb{E}_{s_t \sim P_\theta(s_t)}\left[b(s_t) \sum_a \nabla_\theta \pi_\theta(a \mid s_t) \right] = \mathbb{E}_{s_t \sim P_\theta(s_t)}\left[b(s_t)\, \nabla_\theta \sum_a \pi_\theta(a \mid s_t) \right] \\
&\quad = \mathbb{E}_{s_t \sim P_\theta(s_t)}[b(s_t)\, \nabla_\theta 1] = 0.
\end{aligned}$$

The second line is the inner expectation over the action given the state; the third uses that $b(s_t)$ does not depend on a_t and that $\pi_\theta \nabla_\theta \log \pi_\theta = \nabla_\theta \pi_\theta$; the last two use that the policy's probabilities sum to one for every state and every θ. The inner expectation is zero for every state s_t, so the outer expectation is an average of zeros, whatever the (θ-dependent) distribution of s_t is. Summing over t, the baseline term has expectation zero, so $\mathbb{E}[g_\theta]$ is the same with or without the baseline: REINFORCE with a baseline is still unbiased.

1. **[2 pts]** The walkthrough above shows that subtracting the baseline leaves the expected gradient unchanged. Why do we use the baseline, then? Why does the baseline reduce variance?

   Your Answer

2. **[3 pts]** For each of the following baselines, give one pro and one con.

   (a) $b(s) = V_\phi(s)$

   Pro

   Con

   (b) $b(s) = c$, a constant

   Pro

   Con

3. **[2 pts]** What if $b(s, a) = f(s, a)$ (the baseline also depends on the action)? Should we use this baseline?
   Hint: Refer to the walkthrough derivation above.

   ◯ Yes ◯ No

   Why/why not?

   Your Answer

4 [18 pts] Programming: Latent Variable Models and Diffusion

The programming portion of this assignment is the notebook `hw5.ipynb` (Google Colab preferred, or a local Jupyter notebook), in which you implement PCA, an autoencoder and a VAE, then a denoising diffusion probabilistic model (DDPM) on the pixels and in the latent space of your autoencoder. The notebook is graded by the Gradescope autograder; the questions below ask for figures and observations from the sections of the notebook marked **WRITTEN** (their training cells are guarded by `RUN_LONG_EXPERIMENTS` and the autograder does not run them). Section 4.1 collects the generated samples of every model, Section 4.2 the 2-D embeddings and the latent interpolations, and Section 4.3 the diffusion processes and the training curves. The training runs take a few minutes each on a Colab T4 GPU (*Runtime* → *Change runtime type*), but the pixel DDPM takes close to an hour on a CPU, so plan ahead. Set the flag back to `False` before you submit the notebook.

Include your figures with `\includegraphics` inside the answer boxes (save them from the notebook, e.g. right-click a plot in Colab) and keep each box at its original size.

4.1 [8 pts] Generated samples

1. **[2 pts]** Train PCA, the autoencoder and the VAE with a latent dimension of 2 using the provided training code and hyperparameters, and include the row of 10 samples from each model (the WRITTEN rows at the end of the Q1, Q2 and Q3 training sections).

PCA samples with latent dimension 2

Autoencoder samples with latent dimension 2

VAE samples with latent dimension 2

2. **[3 pts]** Train PCA, the autoencoder and the VAE with a latent dimension of 16 (the WRITTEN latent variable model experiments), the DDPM (the WRITTEN DDPM experiments) and the latent diffuser (the WRITTEN latent diffusion experiments) with the provided hyperparameters, and include the row of 10 samples from each of the five models.

PCA samples with latent dimension 16

Autoencoder samples with latent dimension 16

VAE samples with latent dimension 16

DDPM samples

Latent diffusion samples

3. **[1 pts]** Compare the quality of the samples generated from the 2-D and 16-D latent spaces. Which latent dimension seems to give better results? Why do you think that is?

Your Answer

4. **[2 pts]** Compare the samples of the four latent dimension 16 and diffusion models (autoencoder, VAE, DDPM and latent diffusion) and explain the differences you see in terms of how each technique generates an image.

Your Answer

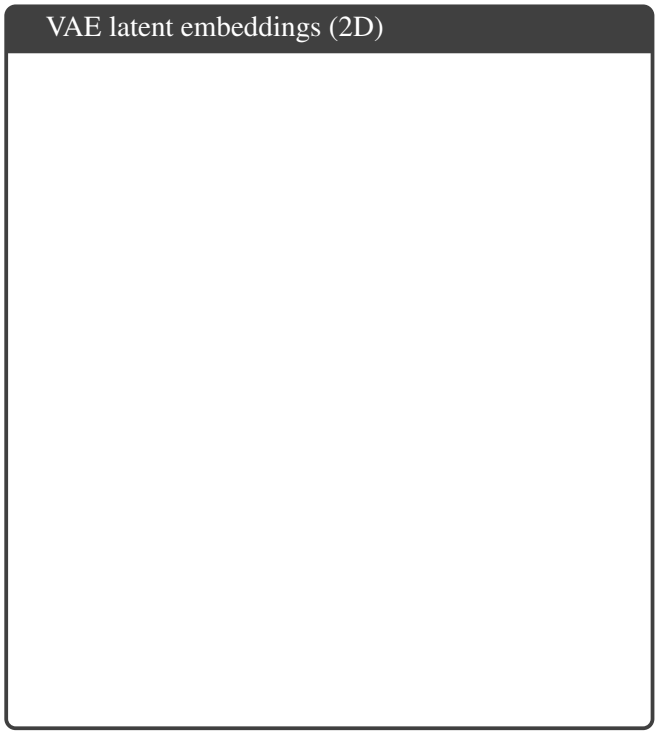
4.2 [6 pts] The latent space: 2-D embeddings and interpolation

1. **[1 pts]** Using the latent dimension 2 models for PCA, the autoencoder and the VAE, encode all training data and plot the 2-D latent encodings for each method (the code for this is provided in the WRITTEN latent variable model experiments). Include your three plots below.

PCA latent embeddings (2D)

Autoencoder latent embeddings (2D)

VAE latent embeddings (2D)



2. **[2 pts]** What do you observe about the 2-D latent embeddings for each model? How do they differ, and what might explain the differences in structure?

Your Answer

3. **[1 pts]** For each of PCA, the autoencoder and the VAE with latent dimension 16, include your row of 10 images decoded from the linearly interpolated latent vectors between a left-facing and a right-facing training sprite (see the WRITTEN latent variable model experiments for detailed instructions).

PCA latent interpolation

Autoencoder latent interpolation

VAE latent interpolation

4. **[2 pts]** Compare the interpolation results above for the different techniques. Why do you think the different techniques worked well or poorly?

Your Answer

4.3 [4 pts] Diffusion: the reverse and forward processes and training

1. **[1 pts]** Include the row of reverse-process snapshots ($\mathbf{x}_t$ after the step at $t = 999, 900, \ldots, 100, 0$) produced by the last cell of the WRITTEN DDPM experiments.

Reverse process

2. **[2 pts]** Include the row of noisy versions of one training sprite produced by the forward-process cell ($t = 0, 25, 50, 100, 200, 400, 700, 999$). At which of these timesteps can you no longer tell what the sprite looked like? Give that t and the value of $\sqrt{\bar{\alpha}_t}$ there, to two decimal places (your `DDPM` object stores the schedule).

Forward process

t

$\sqrt{\bar{\alpha}_t}$

3. **[1 pts]** Include the training / validation loss curves of the DDPM (20 epochs) and of the latent diffuser (50 epochs).

DDPM loss curve

Latent diffuser loss curve

5 Collaboration Policy

After you have completed all other components of this assignment, report your answers to the following collaboration questions.

1. Did you receive any help whatsoever from anyone in solving this assignment? If so, include full details including names of people who helped you and the exact nature of help you received.

Your Answer

2. Did you give any help whatsoever to anyone in solving this assignment? If so, include full details including names of people you helped and the exact nature of help you offered.

Your Answer

3. Did you find or come across code that implements any part of this assignment? If so, include full details including the source of the code and how you used it in the assignment.

Your Answer