Week-9 Questions

Prof . Prathosh AP and Chandan J

July 2025

1. Consider the forward transition

$$q(x_t \mid x_{t-1}) = \mathcal{N}\big(\sqrt{\alpha_t}\, x_{t-1},\ \beta_t\big).$$

If $\alpha_t = 0.75$, $\beta_t = 0.25$, $x_{t-1} = 2$, and a sample $\epsilon = 1$, what is x_t

(A) 1.50

(B) 2.00

(C) 2.23

(D) 2.50

Answer: (C)
Explanation:

$$x_t = \sqrt{0.75} \cdot 2 + \sqrt{0.25} \cdot 1$$

$$\sqrt{0.75} \times 2 = 2 \times 0.8660 = 1.7320, \quad \sqrt{0.25} \times 1 = 0.5, \quad 1.7320 + 0.5 = 2.2320 \approx 2.23.$$

2. The marginal forward distribution is

$$q(x_t \mid x_0) = \mathcal{N}\big(\sqrt{\bar{\alpha}_t}\, x_0,\ 1 - \bar{\alpha}_t\big).$$

For $\bar{\alpha}_t = 0.36$ and $x_0 = 3$, what are the mean and variance?

(A) $\mu = 1.8,\ \sigma^2 = 0.64$

(B) $\mu = 1.2,\ \sigma^2 = 0.36$

(C) $\mu = 1.8,\ \sigma^2 = 0.36$

(D) $\mu = 1.2,\ \sigma^2 = 0.64$

Answer: (A)
Explanation: $\mu = \sqrt{0.36} \times 3 = 0.6 \times 3 = 1.8, \quad \sigma^2 = 1 - 0.36 = 0.64.$

3. In the mean-prediction VAE, Given $\alpha_t = 0.8$, $\beta_t = 0.2$, $\bar{\alpha}_t = 0.6$, $x_t = 1$, $\epsilon_\theta = 0.5$, compute μ_t. (Two decimals.)

(A) 0.85

(B) 0.94

(C) 1.00

(D) 1.05

Answer: (B) **Explanation:**

$$\mu_t = \frac{1}{\sqrt{\alpha_t}}\Big(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\,\epsilon_\theta\Big).$$

$$\frac{1}{\sqrt{0.8}}\Big(1 - \frac{0.2}{\sqrt{0.4}} \times 0.5\Big) = 1.1180 \times (1 - 0.3162 \times 0.5) = 1.1180 \times 0.8419 \approx 0.94.$$

4. In the reverse-diffusion update for noise estimation (with stochastic term),

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}}\Big(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\,\epsilon_\theta\Big) + \sqrt{\beta_t}\,z,$$

which option correctly matches this?

(A) $x_{t-1} = \dfrac{x_t - \beta_t\,\epsilon_\theta}{\sqrt{\alpha_t}} + \sqrt{\beta_t}\,z$

(B) $x_{t-1} = \dfrac{1}{\sqrt{\alpha_t}}\big(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta\big)$

(C) $x_{t-1} = \sqrt{\alpha_t}\,x_t + \sqrt{\beta_t}\,z$

(D) $x_{t-1} = \dfrac{1}{\sqrt{\alpha_t}}(x_t + \beta_t\epsilon_\theta) + z$

Answer: (A) **Explanation:** Only (A) matches both the mean and variance terms in the known reverse-diffusion formula.

5. Why is the training objective in DDPMs typically formulated as minimizing the mean squared error between the true noise ϵ and the predicted noise $\epsilon_\theta(x_t, t)$, rather than directly minimizing the error between the true posterior mean $\tilde{\mu}_t$ and the predicted mean $\mu_\theta(x_t, t)$?

(A) Optimizing noise leads to a simpler U-Net architecture that is easier to train.

(B) The noise prediction objective inherently simplifies the Kullback-Leibler (KL) divergence terms in the Evidence Lower Bound (ELBO), making it computationally more stable and directly proportional to the squared error of noise.

(C) The noise variable ϵ is directly observable and easily calculable during training, unlike $\tilde{\mu}_t$, which requires knowledge of x_0.

(D) Minimizing noise error results in less mode collapse compared to mean error minimization, especially in high-dimensional spaces.

Explanation: The ELBO for DDPMs involves a KL divergence term $D_{KL}(q(x_{t-1}|x_t, x_0)||p_\theta(x_{t-1}|x_t))$ Due to the Gaussian nature of these distributions, this KL term simplifies to a mean squared error between $\tilde{\mu}_t$ and μ_θ. Crucially, both $\tilde{\mu}_t$ and μ_θ can be expressed linearly in terms of the true noise ϵ and the predicted noise ϵ_θ respectively, with common scaling factors. This algebraic transformation shows that minimizing $\|\tilde{\mu}_t - \mu_\theta\|^2$ is equivalent to minimizing $\|\epsilon - \epsilon_\theta\|^2$ up to a positive constant factor. This simplified noise-prediction objective is both computationally efficient and leads to effective training. Option (A) is a simplification and not the core reason. Option (C) is partially true, but the core reason lies in the ELBO simplification. Option (D) is a desirable outcome, but not the primary *reason* for choosing this objective form.

Answer: (B)

6. Consider a DDPM forward process where $x_0 = 5$, $\bar{\alpha}_t = 0.81$, and the observed noisy data is $x_t = 4.6$. The forward process equation is $x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t}\epsilon$. If $\alpha_t = 0.9$ and $\beta_t = 0.1$, calculate the true noise ϵ that generated x_t from x_0, and then calculate the true posterior mean $\tilde{\mu}_t(x_t, x_0)$. Round ϵ to two decimal places and $\tilde{\mu}_t$ to three decimal places.

(A) $\epsilon = 0.23, \tilde{\mu}_t = 4.795$

(B) $\epsilon = 0.23, \tilde{\mu}_t = 4.600$

(C) $\epsilon = 0.10, \tilde{\mu}_t = 4.795$

(D) $\epsilon = 0.10, \tilde{\mu}_t = 4.600$

Explanation:

(a) **Calculate the true noise ϵ:** The forward process is $x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t}\epsilon$. Rearranging for ϵ: $\epsilon = \frac{x_t - \sqrt{\bar{\alpha}_t} x_0}{\sqrt{1-\bar{\alpha}_t}}$. Given: $x_0 = 5$, $x_t = 4.6$, $\bar{\alpha}_t = 0.81$. $\sqrt{\bar{\alpha}_t} = \sqrt{0.81} = 0.9$. $\sqrt{1 - \bar{\alpha}_t} = \sqrt{1 - 0.81} = \sqrt{0.19} \approx 0.435889$. $\epsilon = \frac{4.6 - (0.9 \times 5)}{0.435889} = \frac{4.6 - 4.5}{0.435889} = \frac{0.1}{0.435889} \approx 0.2294$. Rounding to two decimal places, $\epsilon \approx 0.23$.

(b) **Calculate the true posterior mean $\tilde{\mu}_t(x_t, x_0)$:** The formula for the true posterior mean (simplified in terms of ϵ) is $\tilde{\mu}_t(x_t, x_0) = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon\right)$. Given: $\alpha_t = 0.9$, $\beta_t = 0.1$, $\epsilon \approx 0.2294$. $\sqrt{\alpha_t} = \sqrt{0.9} \approx 0.948683$. $\tilde{\mu}_t = \frac{1}{0.948683}\left(4.6 - \frac{0.1}{0.435889} \times 0.2294\right)$ $\tilde{\mu}_t = 1.054093\,(4.6 - 0.22943 \times 0.2294)$ $\tilde{\mu}_t = 1.054093\,(4.6 - 0.05265)$ $\tilde{\mu}_t = 1.054093 \times 4.54735 \approx 4.7946$. Rounding to three decimal places, $\tilde{\mu}_t \approx 4.795$.

Answer: (A)

7. The U-Net architecture is a cornerstone for the noise prediction network $\epsilon_\theta(x_t, t)$ in DDPMs. What specific design feature of the U-Net is most crucial for its effectiveness in estimating high-resolution noise?

   (A) The exclusive use of residual connections throughout the entire network.

   (B) Its symmetric encoder-decoder structure combined with skip connections between corresponding resolution levels.

   (C) The pervasive application of batch normalization layers in every convolutional block.

   (D) The use of a final sigmoid activation function to constrain noise values between 0 and 1.

   Explanation: The U-Net's power, particularly for tasks requiring precise spatial output like image segmentation or denoising, comes from its unique structure. The encoder path progressively downsamples the input, capturing high-level semantic features. The decoder path then upsamples these features to reconstruct the output. The critical element is the "skip connections" (also known as "concatenation connections"). These connections directly pass feature maps from the encoder path to the decoder path at corresponding resolutions. This allows the decoder to recover fine-grained spatial details that might be lost during the downsampling process, which is essential for accurately estimating high-resolution noise. Option (A) describes residual connections, which are beneficial but not unique to U-Net's core strength. Option (C) refers to batch normalization, a regularization technique aiding training, not the primary structural advantage. Option (D) is incorrect, as noise in DDPMs is typically unbounded Gaussian.

   Answer: (B)

8. In a PyTorch implementation of the DDPM training loop, assuming `noise_pred` is the output of your ϵ_θ model (predicted noise) and `noise_true` is the actual noise added to x_0 to get x_t (ground truth noise), which line of code correctly computes the loss term that needs to be minimized?

   (A) `loss = torch.mean(torch.abs(noise_pred - noise_true))`

   (B) `loss = torch.sum((noise_pred - noise_true)**2)`

   (C) `loss = torch.nn.functional.mse_loss(noise_pred, noise_true)`

   (D) `loss = torch.nn.functional.cross_entropy(noise_pred, noise_true)`

   Explanation: The core training objective of DDPMs simplifies to minimizing the mean squared error (MSE) between the predicted noise and the true noise. This is a standard regression problem. Option (A) computes the Mean Absolute Error (MAE) or L1 loss, not MSE. Option (B) computes the Sum of Squared Errors (SSE), not the Mean

Squared Error, as it lacks division by the number of elements. Option (C) `torch.nn.functional.mse_loss` is the standard PyTorch function for calculating the Mean Squared Error, which is precisely the loss objective for noise prediction in DDPMs. Option (D) `torch.nn.functional.cross_entropy` is used for classification tasks where the output represents log-probabilities over classes and the target is class labels. This is inappropriate for noise prediction, which is a regression task.

Answer: (C)

9. Consider the pseudocode for the reverse sampling (generation) process in a DDPM implementation. Which of the following best describes the typical iterative loop structure?

   (A) Iterate from $t = 0$ to T (total timesteps), gradually adding noise to an original data sample x_0 to produce x_T.

   (B) Iterate from $t = T$ down to 1, predicting the noise ϵ_θ at each step for the current noisy sample x_t and using it to iteratively denoise x_t into x_{t-1}.

   (C) Perform a single forward pass through a decoder network to directly generate x_0 from a randomly sampled latent vector.

   (D) Iterate from $t = 0$ to T, updating the model parameters at each step based on the reconstruction error of the generated samples.

   Explanation: The DDPM generation (inference) process is an iterative denoising process. It starts with a pure noise sample ($x_T \sim \mathcal{N}(0, I)$) and gradually transforms it into a clean data sample (x_0). This is achieved by stepping backward through the diffusion process. At each step t (from T down to 1), the model uses its learned noise predictor $\epsilon_\theta(x_t, t)$ to estimate the noise in x_t and then samples x_{t-1} using the reverse diffusion formula. Option (A) describes the forward diffusion process (training data corruption). Option (C) describes how a VAE decoder or a GAN generator might operate. Option (D) describes the training loop (optimization of model parameters), not the sampling (inference) process.

   Answer: (B)

10. Consider a simple 2-step forward diffusion process: $x_1 \sim \mathcal{N}(\sqrt{\alpha_1}x_0, \beta_1)$ $x_2 \sim \mathcal{N}(\sqrt{\alpha_2}x_1, \beta_2)$ If $\alpha_1 = 0.9$, $\beta_1 = 0.1$, $\alpha_2 = 0.8$, $\beta_2 = 0.2$, what is the total variance of x_2 given x_0? (i.e., $1 - \bar{\alpha}_2$)

    (A) 0.28

    (B) 0.32

    (C) 0.20

    (D) 0.18

Explanation: The total variance of x_t given x_0 in a DDPM is $1 - \bar{\alpha}_t$. The term $\bar{\alpha}_t$ is the cumulative product of α_i from $i = 1$ to t: $\bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i$. For $t = 2$, $\bar{\alpha}_2 = \alpha_1 \times \alpha_2$. Given: $\alpha_1 = 0.9$, $\alpha_2 = 0.8$. $\bar{\alpha}_2 = 0.9 \times 0.8 = 0.72$. The total variance of x_2 given x_0 is $1 - \bar{\alpha}_2 = 1 - 0.72 = 0.28$.

Answer: (A)