DSAI SET 6

EURON

1. What are Variational Autoencoders (VAEs), and how do they differ from traditional autoencoders?

Answer:

Variational Autoencoders (VAEs) are generative models that use deep learning techniques to model complex data distributions. Unlike traditional autoencoders, which compress and reconstruct data, VAEs introduce a probabilistic layer that maps input data into a latent space as a probability distribution rather than a fixed vector.

Key Differences:

- **Latent Space Representation:** VAEs encode data into a latent space as probability distributions (mean and variance), whereas traditional autoencoders use deterministic representations.
- **Generative Capability:** VAEs can sample new data points from the latent space, making them useful for generation tasks, unlike autoencoders, which merely reconstruct inputs.
- **Regularization via KL Divergence:** VAEs impose a regularization constraint using the Kullback–Leibler (KL) divergence to ensure latent variables follow a standard normal distribution.

---

2. How does the reparameterization trick work in VAEs?

Answer:

The reparameterization trick is used in VAEs to enable gradient-based optimization during backpropagation. Since VAEs map inputs to a probability distribution, direct sampling from it is non-differentiable. The trick involves rewriting the sampling process:

Given:

$$z = \mu + \sigma \cdot \epsilon, \text{ where } \epsilon \sim N(0,1)$$

Steps:

1. The encoder outputs a mean (μ) and standard deviation (σ).
2. A noise term (ϵ) is sampled from a standard normal distribution.
3. The latent variable (z) is computed using z=μ+σ·e ensuring differentiability.

This allows VAEs to be trained end-to-end using standard gradient descent.

---

3. What is the basic structure of a Generative Adversarial Network (GAN)?

Answer:

A GAN consists of two neural networks competing against each other:

- **Generator (G):** Creates synthetic data from random noise.
- **Discriminator (D):** Distinguishes between real and fake data.

Training Process:

1. The generator takes random noise as input and produces synthetic samples.
2. The discriminator evaluates both real and generated samples and assigns probabilities.
3. The generator is trained to produce more realistic data, while the discriminator improves in distinguishing real from fake.
4. The process continues iteratively, refining both networks.

The loss function used in GANs is typically a **min-max optimization**:

$$\min_G \max_D \mathbb{E}_{x \sim P_{data}}[\log D(x)] + \mathbb{E}_{z \sim P_z}[\log(1 - D(G(z)))]$$

where Pdata is the real data distribution, and Pz is the noise distribution.

---

4. What are some common challenges when training GANs?

Answer:

1. **Mode Collapse:** The generator produces only a limited variety of outputs instead of capturing the full data distribution.
2. **Training Instability:** The balance between generator and discriminator is crucial; if one overpowers the other, training collapses.
3. **Vanishing Gradients:** The discriminator may become too good, providing gradients too close to zero, which halts generator learning.
4. **Lack of Evaluation Metrics:** Unlike supervised learning, it is difficult to measure the quality of GAN-generated samples.
5. **Hyperparameter Sensitivity:** GANs require careful tuning of learning rates, batch sizes, and architectural choices.

---

5. How do Diffusion Models work in generative AI?

Answer:

Diffusion Models generate images by reversing a gradual noising process. They learn to **denoise** an image iteratively.

Process:

1. **Forward Diffusion:** A clean image is progressively corrupted by adding Gaussian noise.
2. **Reverse Diffusion:** A deep neural network learns to predict and remove noise step by step to reconstruct the original image.

Key Features:

- **High-quality image generation (e.g., DALL·E, Stable Diffusion).**
- **Stable training compared to GANs.**
- **Better diversity of generated samples.**

Mathematically, the forward diffusion process is:

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{\alpha_t}x_{t-1}, (1-\alpha_t)I)$$

where αt controls the noise schedule.

---

6. What is the difference between Pre-training and Fine-tuning in LLMs?

Answer:

- **Pre-training:** The model is trained on large-scale datasets (e.g., web pages, books) in a self-supervised manner to learn general knowledge and language structures.
- **Fine-tuning:** The pre-trained model is adapted to specific tasks (e.g., sentiment analysis, medical diagnosis) by training on task-specific data.

Example: GPT models are pre-trained on massive corpora using next-token prediction. Then, they can be fine-tuned on specialized datasets like legal documents or medical texts.

---

7. What is prompt engineering, and why is it important in LLMs?

Answer:

Prompt engineering is the art of designing input prompts to guide the output of language models effectively.

Importance:

- Controls the behavior of the model without retraining.
- Helps generate more relevant responses.
- Improves efficiency by reducing computational overhead.

Example:

- **Weak Prompt:** "Tell me about AI."
- **Strong Prompt:** "Explain AI in simple terms with real-world applications."

---

8. How do Chatbots like ChatGPT work?

Answer:

Chatbots like ChatGPT are based on Transformer architectures and trained using:

1. **Pre-training:** Learning from vast amounts of text data.
2. **Fine-tuning:** Refining with supervised datasets.
3. **Reinforcement Learning with Human Feedback (RLHF):** Aligning responses with human expectations.

They use **autoregressive decoding**, where each generated token is conditioned on previous tokens.

---

9. What are the ethical concerns with generative AI?

Answer:

1. **Misinformation Generation:** Fake news and deepfakes.
2. **Bias in AI Models:** Prejudices from training data.
3. **Intellectual Property Issues:** Generating content based on copyrighted works.
4. **Data Privacy Risks:** Potential leaks of sensitive information.

---

10. How does Stable Diffusion differ from DALL·E?

Answer:

- **Stable Diffusion:** Open-source, runs locally, uses latent diffusion models.
- **DALL·E:** Proprietary, API-based, diffusion-based but trained with a different architecture.