Gen AI Intro & Text generation

What is Generative AI?

Generative AI refers to artificial intelligence models that create new content, such as text, images, music, or code, based on patterns in the data they were trained on.

How is Generative AI different from traditional AI?

Traditional AI focuses on recognizing patterns, making predictions, or performing specific tasks, while Generative AI creates novel outputs by learning the underlying distribution of data.

Name two applications of Generative AI in the industry.

- Content creation (e.g., generating articles, images, or videos).
- Drug discovery (e.g., designing new molecules).

What are some challenges associated with Generative AI?

- Ethical concerns, such as misuse or bias in outputs.
- Computational cost for training large models.
- Ensuring the quality and coherence of generated outputs.

Why is Generative AI important for modern applications?

It enables automation in creative tasks, enhances productivity, and powers innovative applications, such as personalized recommendations and realistic simulations.

What is probabilistic modeling in the context of Generative AI?

Probabilistic modeling involves using mathematical frameworks to predict and generate data by modeling the likelihood of different outcomes.

Define a generative model.

A generative model learns the underlying data distribution and generates new samples that resemble the original dataset.

Explain how an n-gram model works in text generation.

An n-gram model predicts the next word in a sequence based on the preceding $n-1$n-1$n-1$ words by analyzing word probabilities from a dataset.

What are the limitations of n-gram models?

- Limited context since they only consider a fixed number of preceding words.
- Sparse data issues, leading to poor performance on rare sequences.

How can you improve the performance of an n-gram model?

- Use smoothing techniques to handle zero probabilities.
- Increase the training dataset size.
- Consider hybrid approaches with neural networks.

What is the Markov assumption, and how does it apply to text generation?
The Markov assumption states that the future state depends only on the current state. In text generation, this means the next word depends only on a fixed number of previous words.
Why are probabilistic models important in generative AI?
They allow models to generate diverse and plausible outputs by simulating data distributions.
What is an autoencoder?
An autoencoder is a neural network designed to compress input data into a lower-dimensional latent space and reconstruct it back to the original form.
How does a VAE differ from a standard autoencoder?
A Variational Autoencoder (VAE) introduces a probabilistic latent space, enabling the generation of new, meaningful data samples.
Why are VAEs useful in generative modeling?
They allow for controlled data generation and interpolation between data points in the latent space.
What role does the decoder play in an autoencoder?
The decoder reconstructs the original data from the compressed latent representation.
How does the latent space affect text generation in a VAE?
The latent space encodes meaningful features, enabling the generation of diverse and coherent text samples.
What is the purpose of the Kullback-Leibler (KL) divergence term in VAEs?
It ensures the latent space follows a specific distribution, typically Gaussian, for smooth interpolation and controlled generation.
How can you prevent overfitting in a VAE?

- Use regularization techniques like dropout.
- Incorporate early stopping during training.
- Employ a sufficiently large and diverse dataset.

Explain why VAEs are commonly used for unsupervised learning tasks.
VAEs learn latent representations without labeled data, making them ideal for unsupervised tasks like anomaly detection and data synthesis.

What is a transformer model?

A transformer is a neural network architecture that uses self-attention mechanisms to process sequential data efficiently.

Explain the purpose of self-attention in transformers.

Self-attention allows the model to weigh the importance of different parts of the input sequence, enabling it to capture long-range dependencies.

How does a GPT model generate text?

GPT generates text by predicting the next token in a sequence based on previously generated tokens using a transformer architecture.

What are the key differences between a GPT model and an RNN?

- GPT uses transformers and self-attention, allowing parallel processing, whereas RNNs process sequences sequentially.
- GPT can handle long-range dependencies better than RNNs.

How does fine-tuning improve a pre-trained GPT model?

Fine-tuning adapts the model to specific tasks or domains by training it further on task-specific labeled data.

What is zero-shot learning in the context of GPT models?

Zero-shot learning enables GPT models to perform tasks without explicit task-specific training by interpreting instructions from prompts.

Describe how prompt engineering can impact GPT model performance.

Carefully designed prompts can guide the model to produce more accurate, relevant, and coherent outputs.

Why are large datasets essential for training GPT models?

Large datasets provide diverse examples, enabling the model to learn broader language patterns and improve generalization.

What are potential ethical concerns with GPT models?

- Generating biased or harmful content.
- Misinformation dissemination.
- Privacy issues from training on sensitive data.

How does the attention mechanism contribute to GPT's ability to handle long-range dependencies?

Attention mechanisms assign importance to relevant tokens across the sequence, enabling the model to focus on distant context.

What are some limitations of GPT models for real-world applications?

- High computational cost.
- Tendency to generate incorrect or nonsensical outputs.
- Vulnerability to biased training data.

How can GPT models be adapted for domain-specific text generation?
By fine-tuning the model on domain-specific datasets or leveraging prompt engineering tailored to the domain.
What are some common metrics for evaluating text generation quality?

- BLEU (Bilingual Evaluation Understudy).
- ROUGE (Recall-Oriented Understudy for Gisting Evaluation).
- Perplexity.

Explain the difference between deterministic and probabilistic text generation.

- Deterministic generation always produces the same output for a given input.
- Probabilistic generation uses randomness to create diverse outputs.

How does beam search improve text generation in language models?
Beam search explores multiple sequences simultaneously, keeping the most probable ones to improve the overall quality of generated text.