Gen AI Intro & Text generation

Assignment Questions

Gen AI Intro & Text generation

1. What is Generative AI?
2. How is Generative AI different from traditional AI?
3. Name two applications of Generative AI in the industry.
4. What are some challenges associated with Generative AI?
5. Why is Generative AI important for modern applications?
6. What is probabilistic modeling in the context of Generative AI?
7. Define a generative model.
8. Explain how an n-gram model works in text generation.
9. What are the limitations of n-gram models?
10. How can you improve the performance of an n-gram model?
11. What is the Markov assumption, and how does it apply to text generation?
12. Why are probabilistic models important in generative AI?
13. What is an autoencoder?
14. How does a VAE differ from a standard autoencoder?
15. Why are VAEs useful in generative modeling?
16. What role does the decoder play in an autoencoder?
17. How does the latent space affect text generation in a VAE?
18. What is the purpose of the Kullback-Leibler (KL) divergence term in VAEs?
19. How can you prevent overfitting in a VAE?
20. Explain why VAEs are commonly used for unsupervised learning tasks.
21. What is a transformer model?
22. Explain the purpose of self-attention in transformers.
23. How does a GPT model generate text?
24. What are the key differences between a GPT model and an RNN?
25. How does fine-tuning improve a pre-trained GPT model?
26. What is zero-shot learning in the context of GPT models?
27. Describe how prompt engineering can impact GPT model performance
28. Why are large datasets essential for training GPT models?
29. What are potential ethical concerns with GPT models?
30. How does the attention mechanism contribute to GPT's ability to handle long-range dependencies?
31. What are some limitations of GPT models for real-world applications?
32. How can GPT models be adapted for domain-specific text generation?
33. What are some common metrics for evaluating text generation quality?
34. Explain the difference between deterministic and probabilistic text generation.
35. How does beam search improve text generation in language models?

Practical

1. Write a code to generate a random sentence using probabilistic modeling (Markov Chain). Use the sentence "The cat is on the mat" as an example
2. Build a simple Autoencoder model using Keras to learn a compressed representation of a given sentence. Use a dataset of your choice.
3. Use the Hugging Face transformers library to fine-tune a pre-trained GPT-2 model on a custom text data and generate text.
4. Implement a text generation model using a simple Recurrent Neural Network (RNN) in Keras. Train the model on a custom data and generate a word
5. Write a program to generate a sequence of text using an LSTM-based model in TensorFlow, trained on a custom data of sentences.
6. Build a program that uses GPT-2 from Hugging Face to generate a story based on a custom prompt.
7. Write a code to implement a simple text generation model using a GRU-based architecture in Keras.
8. Create a script to implement GPT-2-based text generation with beam search decoding to generate text.
9. Implement a text generation script using GPT-2 with a custom temperature setting for diversity in output text.
10. Create a script to implement temperature sampling with GPT-2, experimenting with different values to generate creative text.
11. Implement a simple LSTM-based text generation model from scratch using Keras and train it on a custom data.
12. How can you implement text generation using it in a simple custom attention-based architecture?