IT2011 - Artificial Intelligence and Machine Learning

Department of Information Technology, Faculty of Computing

Year 2 semester 1 (2025)

Tutorial 02

Introduction to Generative AI and LLM Applications

Section A: Knowledge Check

1. Define the following terms (1 mark each):

   a. Generative AI
   b. Large Language Model (LLM)
   c. Prompt Engineering
   d. Hugging Face
   e. OpenAI
   f. System Prompt
   g. User Prompt
   h. Agentic AI
   i. Predictive AI
   j. Discriminative AI
   k. Model Distillation
   l. Small Language Models
   m. Custom Models
   n. Training a Large Language Model
   o. Fine-tuning an LLM

2. Match the following tools with their functions (1 mark each):

Tool or Model	Description
Hugging Face Transformers	A. Python library for using and fine-tuning pretrained models
OpenAI API	B. Interface for accessing OpenAI models like GPT-4
ChatGPT	C. A chatbot built using large language models by OpenAI
Hugging Face Spaces	D. Platform for deploying and sharing AI/ML demos and apps
GPT-4	E. A state-of-the-art large language model developed by OpenAI

3. State whether the following statements are True or False (1 mark each):

   a. Prompt engineering involves crafting effective queries for models.
   b. Hugging Face can only host models built by OpenAI.
   c. LLMs can be fine-tuned for specific applications.
   d. Generative AI is limited to text-based outputs only.
   e. APIs provided by OpenAI can generate code snippets.
   f. Diffusion models are used for image generation.
   g. Text-to-speech models convert textual input into spoken audio.
   h. Agentic AI can autonomously perform tasks.
   i. Closed models are freely accessible and modifiable by the public.
   j. Predictive AI anticipates future outcomes based on historical data.
   k. Model distillation reduces the size of large models while retaining their capabilities.
   l. Small language models cannot be effective for domain-specific tasks.
   m. Fine-tuning adjusts a model to improve its performance on specific tasks.

Section B: Short Answer Questions

4. Explain the difference between general prompting and prompt engineering. Provide an example scenario for each.

5. Briefly describe how Hugging Face supports generative AI development. Mention at least two specific features.

6.

```
import openai


openai.api_key = "your-api-key"

response = openai.ChatCompletion.create(
    model="gpt-4",
    messages=[
        {"role": "system", "content": "You are a helpful
assistant with deep knowledge in physics."},
        {"role": "user", "content": "Explain how black holes are
formed."}
    ]
)

print(response['choices'][0]['message']['content'])
```

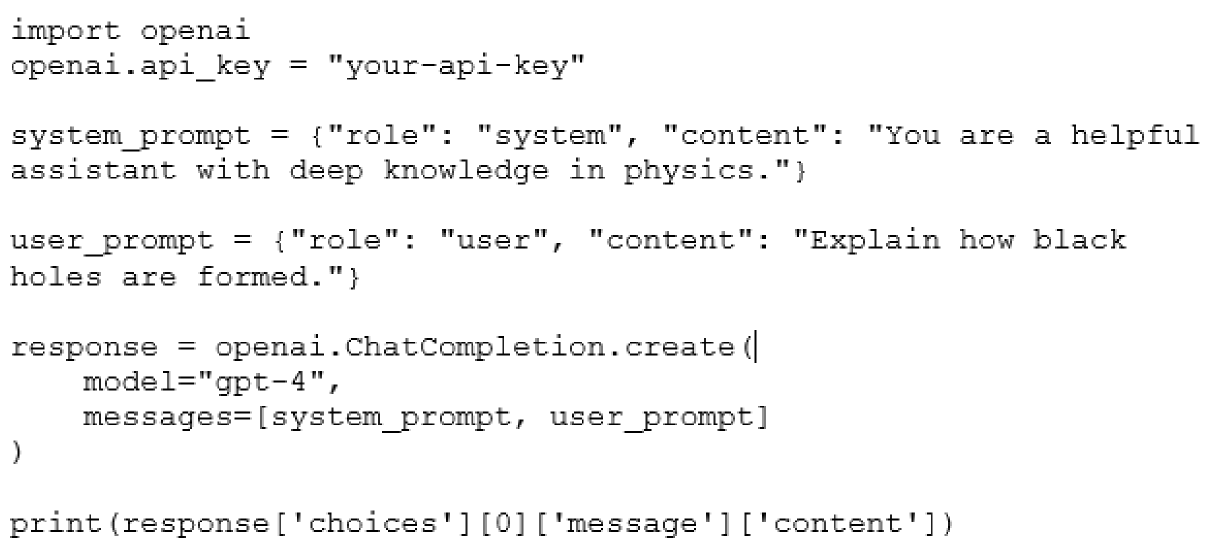
   a.What is the purpose of the system prompt?
   b. What is the role of the user prompt?
   c. How do these prompts influence the model's behavior?
   d. If the output is not satisfactory, what changes can be made to improve the result?

e. How would you structure a prompt to simulate a subject-matter expert responding to a user's question?

7. Discuss two ways in which AI can enhance educational practices. Provide clear examples.
8. Differentiate between predictive, discriminative, and generative AI, providing a clear example for each.
9. Explain model distillation, its benefits, and provide an example scenario of its use.
10. What does training a Large Language Model involve, and why is it resource-intensive? Provide an example of a known model.
11. Define fine-tuning of an LLM and describe how it differs from training from scratch. Provide one practical example scenario such as fine-tuning GPT-3 for medical diagnostics.

Section C: Prompt Engineering Activity

12. Describe a systematic approach for developing good prompts. Include the following steps with detailed examples:



```
import openai
openai.api_key = "your-api-key"

system_prompt = {"role": "system", "content": "You are a helpful
assistant with deep knowledge in physics."}

user_prompt = {"role": "user", "content": "Explain how black
holes are formed."}

response = openai.ChatCompletion.create(
    model="gpt-4",
    messages=[system_prompt, user_prompt]
)

print(response['choices'][0]['message']['content'])
```

a. Define clear objectives for the prompt (e.g., summarizing a news article).
b. Identify the target output format (e.g., bullet points).
c. Specify constraints and limitations clearly (e.g., summary must be under 100 words).
d. Use iterative refinement (show an initial prompt and refined version).
e. Validate the effectiveness through testing (provide an example evaluation scenario).

13. Given Python code demonstrating how to use the OpenAI API for connecting and interacting with an LLM, explain each step, including the roles of system and user prompts.

Section D: Application-Based Activity

14. An educational app wants to provide instant feedback on student essays.

    a. Which generative AI model type would you recommend? Why?
    b. List three key prompt considerations to improve the quality of feedback.
    c. Provide a sample prompt suitable for generating constructive essay feedback.

Section E: Mini Case Study

15. Imagine a university is developing a personalized AI tutor for students.

    a. What role can generative AI play in this scenario? List three specific examples.
    b. Explain how using Hugging Face and OpenAI could support the development of this AI tutor.
    c. If the university wants to implement real-time interactions, would you recommend using an API-based LLM or a locally deployed Hugging Face model? Justify your choice.
    d. Identify and explain one ethical consideration the university must address while using generative AI.

16. Discuss two limitations of generative AI models and suggest practical ways to address these limitations in applications.

17. Describe how small language models and custom models can be tailored for specific applications such as health or law. Provide one example for each.