

Lab 1 : Introduction to ChatGPT

Introduction:

AI models like **ChatGPT** are transforming in the way we interact with technology. Developed by **OpenAI**, ChatGPT is a conversational AI built on the **GPT (Generative Pre-trained Transformer)** architecture. It can understand, generate, and respond to text in a human-like manner.

How It Works (In Simple Terms):

ChatGPT uses the **Transformer architecture**, which was introduced in 2017. It's trained on a large amount of text data and uses **self-attention** mechanisms to understand context. The training happens in two main stages:

1. **Pre-training:** The model learns general language patterns from massive datasets.
2. **Fine-tuning:** The model is improved using feedback from real human responses (using RLHF – Reinforcement Learning from Human Feedback).

Visualization tools like **attention maps** help us understand how the model "pays attention" to different parts of input text while generating responses.

Key Capabilities of ChatGPT:

- Natural language conversations
- Writing help (essays, stories, code, emails)
- Code generation and debugging (Python, C++, Java, etc.)
- Summarization and language translation
- Image understanding (in GPT-4o)

Real life scenario: Use Case: College Student Support (Education Sector).

A student is having difficulty comprehending the Merge Sort algorithm in Data Structures.

What ChatGPT do:

- The student initiates ChatGPT and enters:
- "Explain merge sort in simple terms with an example in C++."

How ChatGPT Reacts:

- ChatGPT rapidly processes the question, knows the subject, and comes up with an easy-to-understand explanation accompanied by C++ code for merge sort. It can even provide step-by-step output or discuss time complexity.

Conclusion:

ChatGPT is a powerful example of how AI is changing industries. As an AI & DS student, learning how it works helps me understand real-world AI implementation. It's not just about theory anymore—tools like ChatGPT show us how AI can solve real problems across fields like education, business, and healthcare.

Lab 2:Types of Prompts and Prompt Engineering Basics

A **prompt** is the input or instruction we give to an AI model like ChatGPT. How well the model responds depends on how the prompt is written. Designing effective prompts is called prompt engineering.

Prompt Engineering is about designing prompts that give accurate and useful outputs.

Key principles :

1. Be Clear and Specific
2. Set the Format
3. Use Examples (Few-Shot)
4. Add Constraints
5. Test and Improve

Prompt Type	Description	Best For
Instructional	Direct command	Writing, tasks
Interrogative	Question format	Getting explanations or facts
Zero-Shot	Task with no examples	Simple or common tasks
Few-Shot	Task with 1-3 examples	Custom formats,creative tasks

Instructional Prompts (Tell ChatGPT what to do)

Prompt	Expected Result
1. Write a poem about rain.	Generates a short poem with a rain theme.
2. Create a Python program that adds two numbers.	Produces code with input and addition logic.
3. Summarize the paragraph in 2 lines.	Gives a brief summary.
4. List 5 advantages of online learning.	A clear bullet list.
5. Explain Newton's third law in simple words.	Gives a simplified explanation suitable for students.
<ul style="list-style-type: none">● Result: Outputs are clear and direct because the instruction is specific.	

Interrogative Prompts (Ask a question)

Prompt	Expected Result
1. What is machine learning?	Short explanation of ML.
2. How does a transformer model work?	Describes attention mechanisms and architecture.
3. Why is data cleaning important?	Lists reasons and benefits of data preprocessing.
4. When was GPT-4 released?	Gives release date and related info.
5. Where is AI used in real life?	Lists multiple domains like health, business, education, etc.

- **Result:** Answers are **factual or explanatory**, similar to how you'd answer a quiz.

Zero-Shot Prompts (No examples given)

Prompt	Expected Result
1. Translate this to Spanish: "Good morning."	"Buenos días."
2. Make a headline for a news article about AI in education.	"AI Revolutionizes the Classroom Experience"
3. Summarize this: "AI is growing fast and helping many industries."	"AI is expanding and supporting various fields."
4. Write a tagline for a food delivery app.	"Delicious food at your doorstep."
5. Recommend a movie similar to Inception.	"Try Interstellar or Tenet."

- **Result:** Works well if the task is common or clear, but less consistent for creative or tricky prompts.

Few-Shot Prompts (Give examples first)

Prompt	Expected Result
1. Translate English to French: Hello → Bonjour Thank you → Merci Good night →	Bonne nuit
2. Summarize in 5 words: The cat ran fast → Fast running cat The car was red →	Red-colored car
3. Generate startup names: AI photo editor → SnapSmart Food delivery →	QuickBite
4. Correct grammar: She go to school → She goes to school He like apple →	He likes apple
5. Classify sentiment: I love this! → Positive This is terrible →	Negative

- **Result:** More accurate and on-target, especially for specific formats, styles, or patterns. Few-shot prompts help ChatGPT “understand” the context better.

Lab 3: Precision Prompting for Information Extraction

You are an information extraction assistant.

Your goal is to read the provided text and extract the required fields into a structured format.

Only return the result in the specified format — no explanations.

Fields to extract:

- Name
- Email
- Phone Number
- Skills (list)
- Education (list of {Degree, Institution, Year})
- Work Experience (list of {Role, Company, Years})

Return your output in ****JSON format**** exactly as shown:

```
{
  "Name": "",
  "Email": "",
  "Phone Number": "",
  "Skills": [],
  "Education": [
    {
      "Degree": "",
      "Institution": "",
      "Year": ""
    }
  ],
  "Work Experience": [
    {
      "Role": "",
      "Company": "",
      "Years": ""
    }
  ]
}
```

SAMPLE INPUT PARAGRAPH:

Johnathan Smith is a software engineer based in Pune. You can contact him at johnsmith.dev@gmail.com or on +91-9876543210. He is skilled in Python, Java, and cloud computing.

He graduated with a B.Tech in Computer Science from Savitribai Phule Pune University in 2020 and later completed an M.Tech in Artificial Intelligence from IIT Bombay in 2022. He has worked as a Backend Developer at Infosys for 2 years and as a Cloud Engineer at TCS for 1 year.

Sample JSON format :

```
{
  "Name": "Johnathan Smith",
  "Email": "johnsmith.dev@gmail.com",
  "Phone Number": "+91-9876543210",
  "Skills": ["Python", "Java", "Cloud Computing"],
  "Education": [
    {
      "Degree": "B.Tech in Computer Science",
      "Institution": "Savitribai Phule Pune University",
      "Year": "2020"
    },
    {
      "Degree": "M.Tech in Artificial Intelligence",
      "Institution": "IIT Bombay",
      "Year": "2022"
    }
  ],
  "Work Experience": [
    {
      "Role": "Backend Developer",
      "Company": "Infosys",
      "Years": "2"
    },
    {
      "Role": "Cloud Engineer",
      "Company": "TCS",
      "Years": "1"
    }
  ]
}
```

Sample tabular format :

<u>Field</u>	<u>Details</u>
<u>Name</u>	<u>Johnathan Smith</u>

Email **johnsmith.dev@gmail.com**

Phone Number **+91-9876543210**

Skills **Python, Java, Cloud Computing**

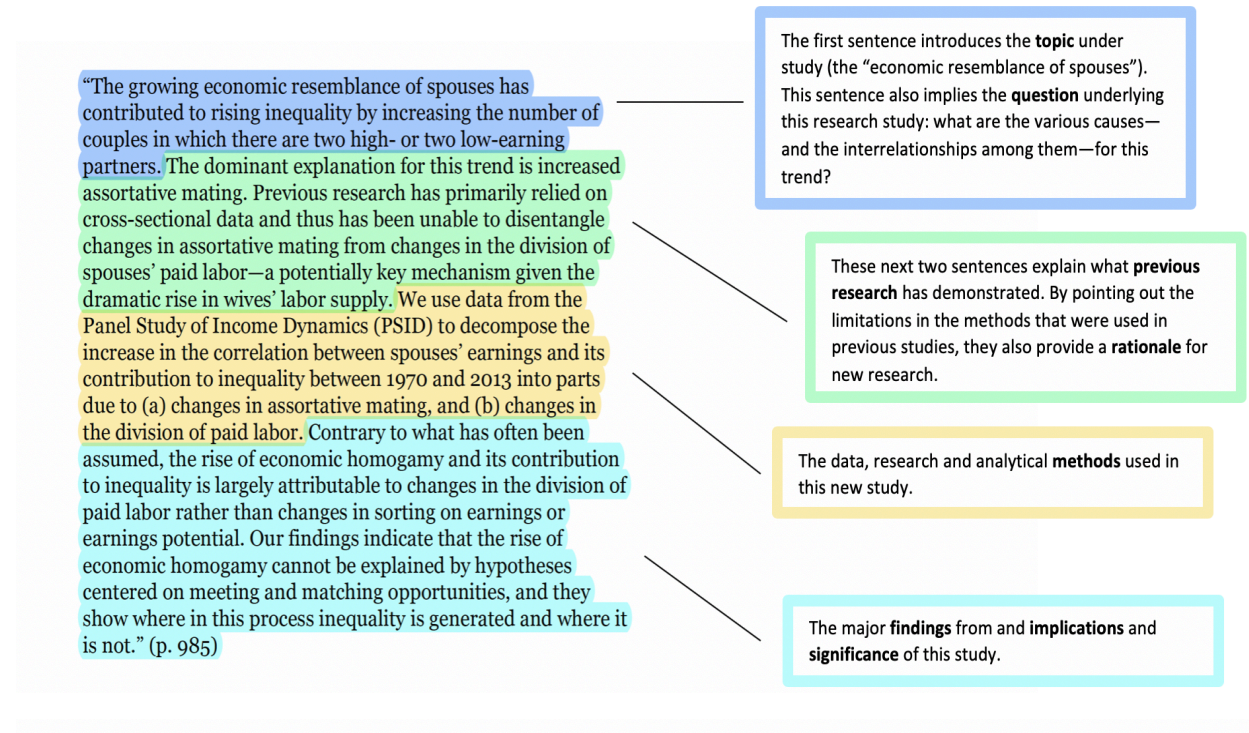
Education **B.Tech (Computer Science, Savitribai Phule Pune University, 2020);**
M.Tech (Artificial Intelligence, IIT Bombay, 2022)

Work **Backend Developer, Infosys (2 years); Cloud Engineer, TCS (1 year)**
Experience

Lab 4: Summarization and Text Transformation

Summarize news articles, research abstracts

Original paragraph



Output

The study examines how the growing economic similarity between spouses has increased inequality, as more couples consist of two high or two low earners. While past research linked this trend mainly to assortative mating, it could not separate this effect from changes in paid labor, especially the rise in wives’ employment. Using Panel Study of Income Dynamics (PSID) data (1970–2013), the authors decompose inequality into effects from assortative mating and from the division of labor. They find that inequality growth is driven more by changes in labor division than by partner selection based on earnings. Thus, rising economic homogamy stems from shifts in paid work rather than sorting by earning potential, clarifying where inequality is generated.

Rewrite emails in formal/informal tone

Sample email:

Subject: Seeking guidance on [topic]

Dear [Recipient's name],

I came across your profile on [platform] and noticed your expertise in [topic]. I am currently [briefly explain your situation or interest]. [Specific questions or requests for guidance.]

I'd love any insights or advice you might have whenever you have a free moment. Thanks for your help! If you'd rather, I'd be happy to do a quick virtual coffee chat—please let me know and I'll send you a calendar invitation.

Best regards,

[Your full name]

Output:

Subject: Request for Guidance on [Topic]

Dear [Recipient's Name],

I recently came across your profile on [platform] and was impressed by your expertise in [topic]. I am currently [briefly explain your situation or interest], and I would be grateful for your guidance. Specifically, I would appreciate your insights on [insert your questions or requests]. Your perspective would be extremely valuable in helping me better understand this area. If it is convenient, I would be glad to arrange a brief virtual meeting at a time that suits your schedule. Please let me know your availability, and I would be happy to share a calendar invitation.

Thank you very much for your time and consideration.

Sincerely,

[Your Full Name]

Lab 5: ChatGPT in Code Generation and Debugging

ChatGPT's code

```
def fibonacci(n):  
    fib_sequence = [0, 1] # starting values  
    while len(fib_sequence) < n:  
        fib_sequence.append(fib_sequence[-1] + fib_sequence[-2])  
    return fib_sequence
```

```
# Generate first 10 Fibonacci numbers  
print(fibonacci(10))
```

Expected Output:

[0, 1, 1, 2, 3, 5, 8, 13, 21, 34]

Output got after execution:

[0, 1, 1, 2, 3, 5, 8, 13, 21, 34]

Conclusion:

No errors in code.

Debug a given code snippet using ChatGPT.

```
def fibonacci(n):  
    fib_sequence = [0, 1]  
    for i in range(2, n+1):  
        next_num = fib_sequence[i-1] + fib_sequence[i-3]  
        fib_sequence.append(next_num)  
    return fib_sequence
```

Call the function

```
print(fibonacci(10))
```

Output:

```
def fibonacci(n):  
    fib_sequence = [0, 1]  
    for i in range(2, n): # ✓ Use n, not n+1  
        next_num = fib_sequence[i-1] + fib_sequence[i-2] # ✓ Correct indices  
        fib_sequence.append(next_num)  
    return fib_sequence
```

```
print(fibonacci(10))
```

Lab 6: Domain-Specific Applications

Educational Quiz Generation – Documentation

1. Prompt Design

Prompt used: “Build an interactive multiple-choice quiz application in HTML, CSS, and JavaScript. The quiz should display one question at a time, allow users to select answers, provide immediate feedback, track the score, and show the final results at the end.”

Design Reasoning:

- **Interactivity:** The quiz should update dynamically (question navigation, scoring).
- **Accessibility:** Mobile-friendly layout via CSS grid and media queries.
- **User Engagement:** Immediate feedback (✅ correct / ❌ incorrect) and motivational messages in final results.
- **Scalability:** Questions stored in a `quiz` array for easy modification or extension.
- **Gamification:** Progress bar, restart option, and percentage-based feedback for engagement.

2. Output (Quiz Implementation)

The generated output (your HTML/JS code) includes:

- **HTML Structure:** Container with title, question display, options grid, buttons for navigation, progress bar, and results section.
- **CSS Styling:** Clean design, responsive layout, hover effects, and color-coded feedback for selected, correct, and incorrect answers.
- **JavaScript Logic:**
 - Stores questions and answers in an array.
 - Tracks state: `currentQuestion`, `score`, `userAnswers`.
 - Handles answer selection (`selectAnswer`), progression (`nextQuestion`), and results display (`showFinalResults`).
 - Includes restart functionality for replay.

3. Output Evaluation

Criterion

Evaluation

Clarity	Questions and options are clearly displayed; interface is intuitive.
Correctness	Logic correctly checks answers, updates score, and shows correct answers on mistakes.
Interactivity	Next/Submit buttons, progress bar, and restart feature work smoothly.
Difficulty	Current sample quiz covers general knowledge (easy–moderate difficulty).
User Feedback	Immediate “Correct/Incorrect” messages and motivational final feedback improve learning.
Improvements	Could add randomized question order, timer for each question, and database/API for dynamic question loading.