ChatGPT meets Wireless:

What will ChatGPT bring to wireless communications?

Jun Zhang

香港科技大學
THE HONG KONG
UNIVERSITY OF SCIENCE
AND TECHNOLOGY

Special thanks to Jiawei Shao, Shenyuan Gao, Zehong Lin, Jingwen Tong for helping prepare the slides!

Thanks also go to ChatGPT!

Outline

- A primer on ChatGPT
- ChatGPT for Wireless
- Wireless for ChatGPT
- Takeaways

A primer on ChatGPT

What is ChatGPT? What can it do?

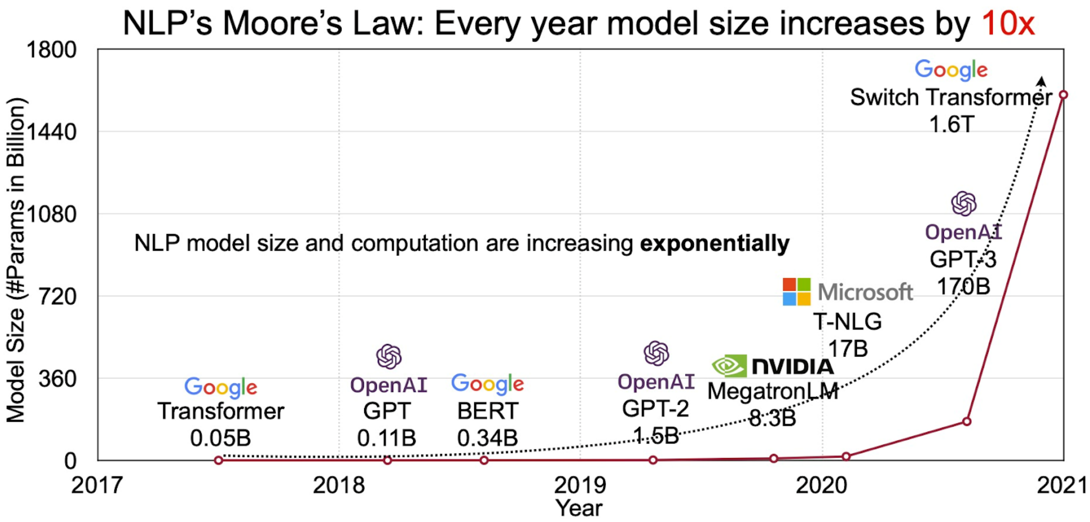
Terminology: What are LLM, GPT, AIGC, ChatGPT?

- **LLM** (Large Language Model)
  - Trained by large-scale data on the Internet (10~100TB)
  - A huge number of parameters (billion level)

- **GPT** (Generative Pre-trained Transformer)
  - 0.1 B (GPT-1) to ~200 B parameters (GPT-3)
  - In-context learning without fine-tuning

- **AIGC** (AI Generated Content)
  - Image (Stable Diffusion, DALL-E 2, etc.)
  - Text (ChatGPT)

- **ChatGPT**
  - A combination of cutting-edge NLP techniques:
    - Code pre-training + Instruction tuning + Reinforcement learning from human feedback (RLHF)



ChatGPT Examples

- Assist writing and editing.

User: help me write a short note to introduce myself to my neighbor

ChatGPT: Hi there,

My name is [Your Name] and I just moved in next door. I wanted to reach out and introduce myself. I'm looking forward to getting to know you and the neighborhood. If you ever need anything or just want to chat, please don't hesitate to stop by or give me a call.

Best,
[Your Name]

ChatGPT Examples

- Maintain the context of the on-going conversation.

User: can you make it more formal?

ChatGPT: Dear Neighbor,

I am writing to introduce myself. My name is [Your Name] and I have recently moved in next door. I wanted to take a moment to reach out and say hello. I am looking forward to getting to know you and the neighborhood.

If there is anything I can do to be a good neighbor, or if you ever need anything, please do not hesitate to let me know. I am happy to help in any way that I can.

Sincerely,
[Your Name]

ChatGPT Examples

- Aware of its own capability.

User: can you send it for me?

ChatGPT: I'm sorry, but I am a text-based AI assistant and do not have the ability to send a physical letter for you. I can only provide suggestions and advice on what to write. It is up to you to take that advice and send the letter yourself.

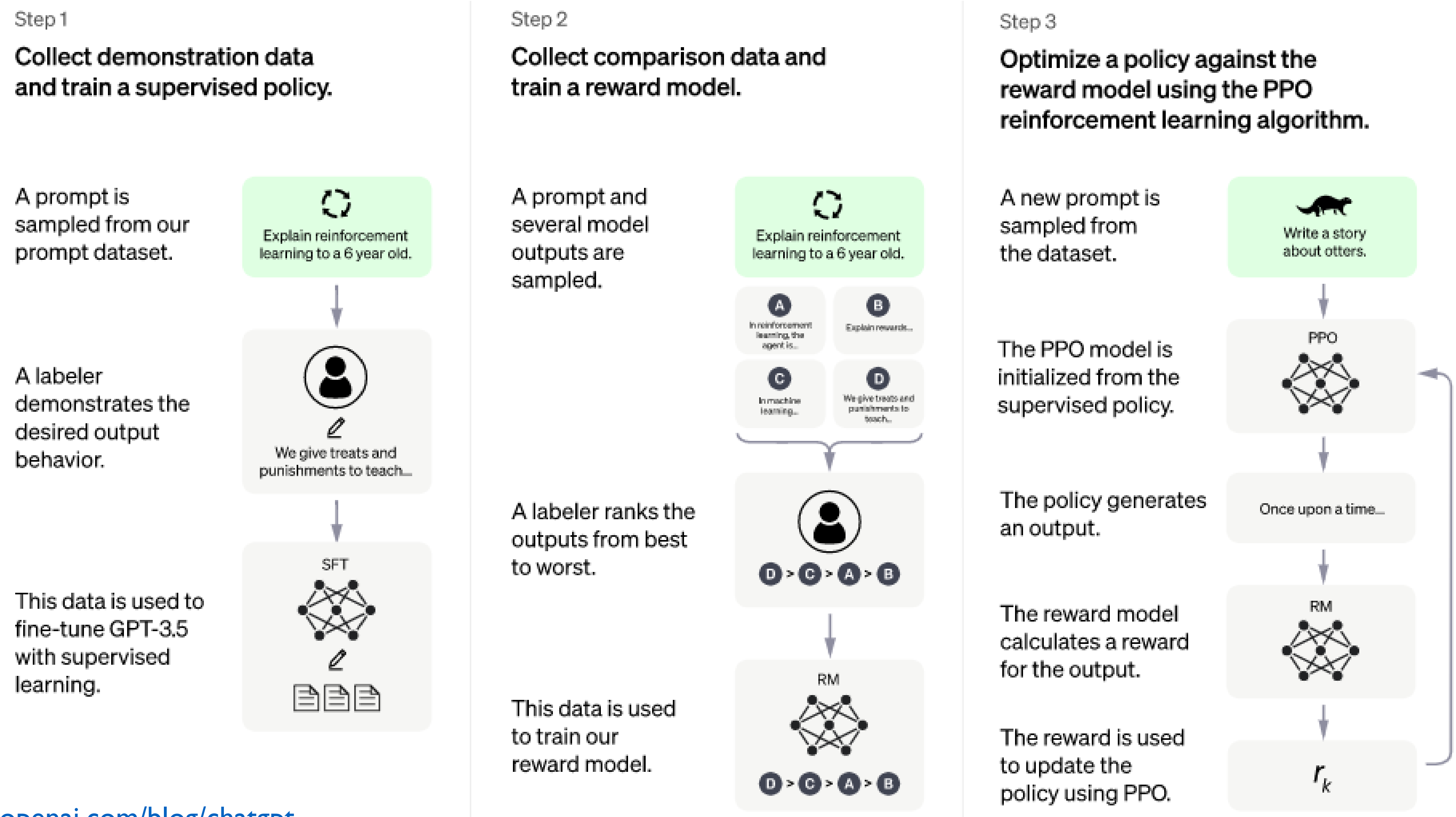
Training ChatGPT – InstructGPT

Training language models to follow instructions with human feedback

Step 1: supervised fine-tuning

Step 2: reward modeling

Step 3: reinforcement learning

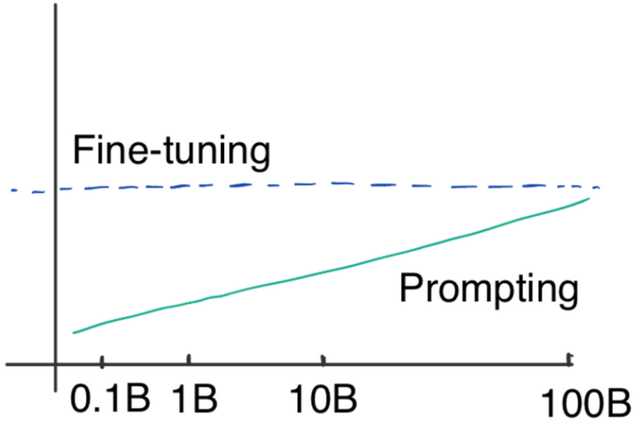


https://openai.com/blog/chatgpt

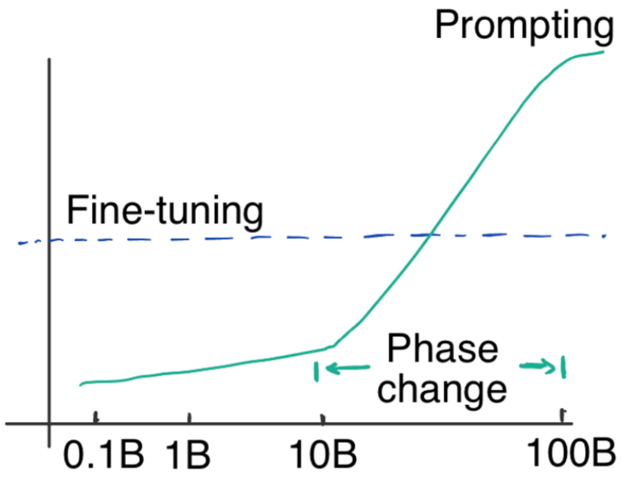
Key enablers of ChatGPT

- Super computing resource
  - Microsoft Azure

- Emergent ability of large language model breaks the scaling law
  - Automatically capture different kinds of context information in a forward pass



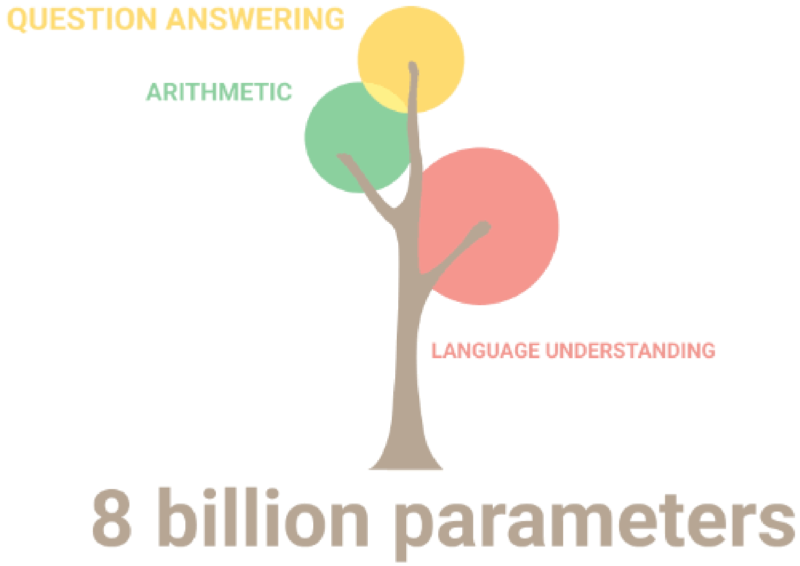
Scaling Law (~2020 - 2021)



Emergent Ability (from early 2022)

Emergent abilities of LLMs

As the scale of the model increases, the performance improves across tasks while also unlocking new capabilities.



https://ai.googleblog.com/2022/04/pathways-language-model-palm-scaling-to.html?ref=news-tutorials-ai-research

What ChatGPT can do?

Generalization

- Generate responses that match the user’s intent with multiple turns.

Creativity

- Strong performance in creative writing tasks, and can even polish its writing step by step.

Correction

- Can actively admit its own mistakes.

Safety

- Good at rejecting unsafe questions or generating safe responses with the consideration of ethical and political factors.

What ChatGPT cannot do?

Modality

- Limit to text generation.

Logic reasoning

- Cannot effectively deal with accurate logic problems (e.g., simple math).

Reliability

- May generate factually incorrect or biased responses.

Knowledge learning

- Cannot search from the website in real time to learn new knowledge and up-to-date information.

Robustness

- Vulnerable to attacks, including instruction attack (making the model follow a wrong instruction to do something illegal or unethical) and prompt injection.

ChatGPT is revolutionizing different application areas

- Question answerer
- Language translator
- Help search engine
- Polish writing
- Hints for brainstorming and creativity
- Education (as a human mentor)
- Entertainment (e.g., story generation)
- …

ChatGPT for Wireless Communication

ChatGPT-enabled techniques

What does ChatGPT say?

Prompts

SI can you summarize some of the challenges in wireless communications

Certainly! Some of the challenges in wireless communications include:

1. Limited spectrum availability: The radio frequency spectrum that wireless communication systems rely on is limited, which can cause congestion and make it difficult to allocate sufficient bandwidth to all users.
2. Interference: Wireless communication systems are susceptible to interference from other devices operating on the same frequency, as well as physical obstructions like buildings and trees.
3. Signal degradation: Wireless signals can be weakened by distance, atmospheric conditions, and other factors, which can result in dropped calls, slow data transfer rates, and other performance issues.
4. Security: Wireless communication systems are vulnerable to a range of security threats, including unauthorized access, eavesdropping, and denial of service attacks.
5. Battery life: Many wireless devices are battery-powered, and optimizing battery life can be a challenge, especially for devices that require constant connectivity or frequent data transmission.
6. Cost: Building and maintaining wireless communication infrastructure can be expensive, especially in rural or remote areas with limited resources.
7. Regulatory compliance: Wireless communication systems are subject to a range of regulations and standards, which can be complex and vary by jurisdiction.

These challenges require ongoing research and development efforts to improve wireless communication systems and ensure that they meet the needs of users in a variety of settings.

SI How can general AIGC help overcoming the above challenges

General artificial intelligence generated content (AIGC) can help overcome some of the challenges in wireless communications in the following ways:

1. Limited spectrum availability: AIGC can be used to optimize spectrum usage by analyzing network traffic and dynamically allocating bandwidth based on demand, which can help reduce congestion and improve overall network performance.
2. Interference: AIGC can be used to identify sources of interference and dynamically adjust wireless settings to avoid or mitigate interference, which can help improve signal quality and reduce dropped calls.
3. Signal degradation: AIGC can be used to optimize wireless signal strength by analyzing environmental factors and adjusting transmission parameters, which can help improve overall network performance and reduce the impact of signal degradation.
4. Security: AIGC can be used to detect and respond to security threats in real-time by analyzing network traffic for patterns that indicate unauthorized access or other security breaches.
5. Battery life: AIGC can be used to optimize device settings and network configurations to reduce power consumption and improve battery life, which can help extend the operating time of wireless devices.
6. Cost: AIGC can be used to optimize network design and infrastructure deployment by analyzing data on user demand, network topology, and other factors, which can help reduce costs and improve overall network efficiency.
7. Regulatory compliance: AIGC can be used to ensure compliance with regulatory requirements and industry standards by monitoring network performance and identifying areas where improvements are needed.

Overall, AIGC has the potential to significantly improve the performance, efficiency, and security of wireless communication systems, helping to overcome many of the challenges that currently exist in this field.

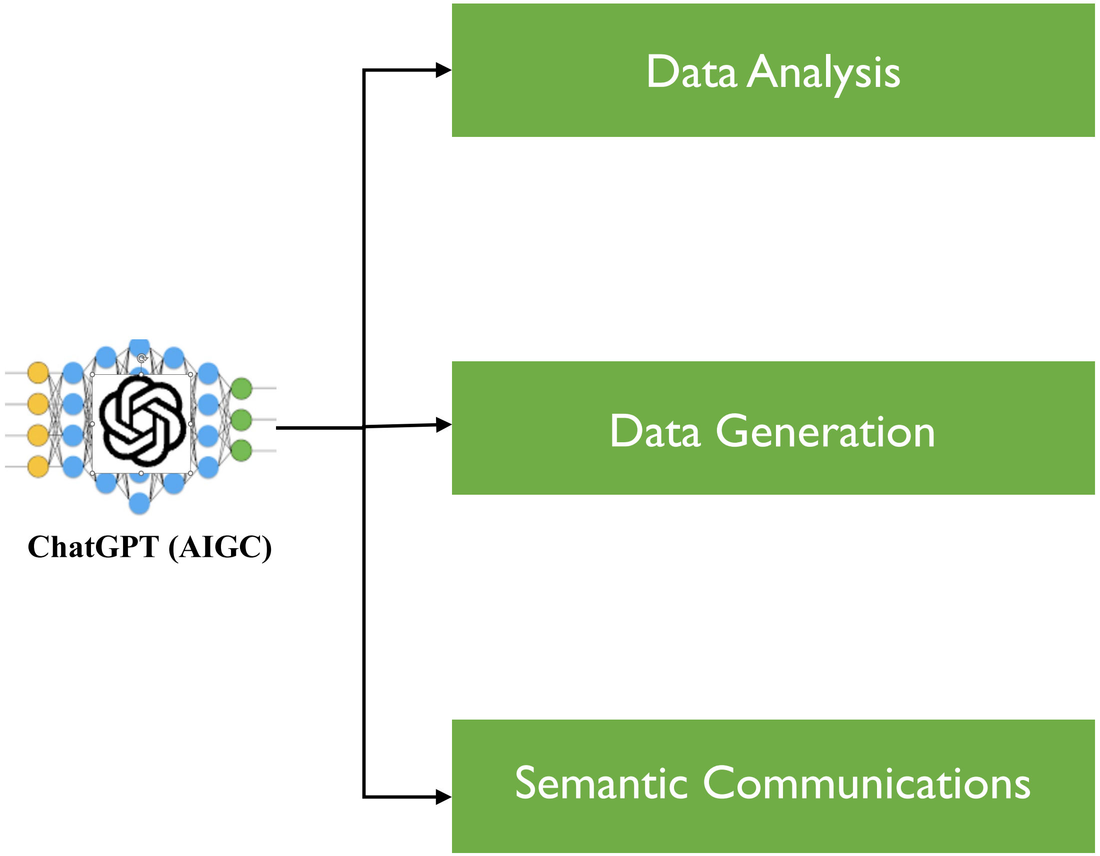
How can ChatGPT/AIGC help wireless?

SI How can general artificial intelligence generated content (AIGC) help wireless communications

General artificial intelligence generated content (AIGC) can help wireless communications in various ways:

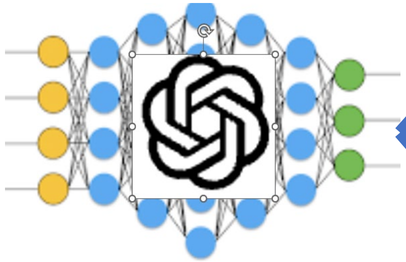
1. Enhance network performance: AIGC can be used to optimize the performance of wireless networks by analyzing data on network traffic, signal strength, and other factors. This can lead to faster data transfer rates, fewer dropped calls, and better overall network reliability.
2. Predictive maintenance: AIGC can be used to predict maintenance needs of wireless communication systems by analyzing performance data and identifying trends that may lead to future problems. This can help network operators proactively address issues before they result in downtime or other negative impacts.
3. Real-time network management: AIGC can be used to analyze real-time data from wireless communication systems and make decisions on network management, such as adjusting power levels or routing traffic to minimize congestion.
4. Enhanced security: AIGC can be used to detect and prevent security breaches in wireless communication systems by analyzing network traffic and identifying anomalies that may indicate unauthorized access or other security threats.
5. Intelligent antenna systems: AIGC can be used to optimize the performance of antenna systems by analyzing data on signal strength, interference, and other factors. This can lead to better coverage and stronger signals, especially in areas with high levels of interference.

Overall, AIGC can greatly improve the performance, efficiency, and security of wireless communication systems, making them an important tool for network operators and other stakeholders in the wireless industry.



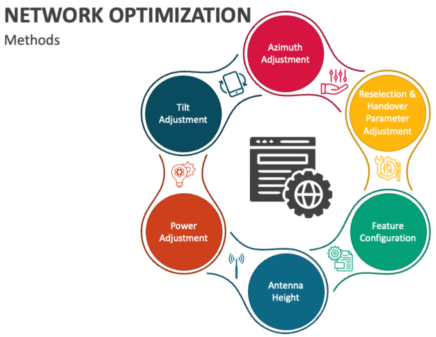
Use case: Data analysis

Mobile big data

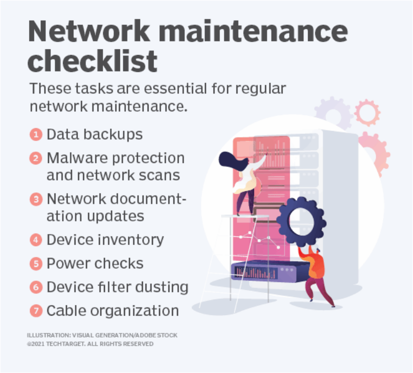


ChatGPT/AIGC

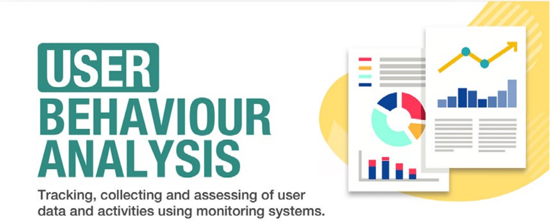
Performance evaluation



Network optimization



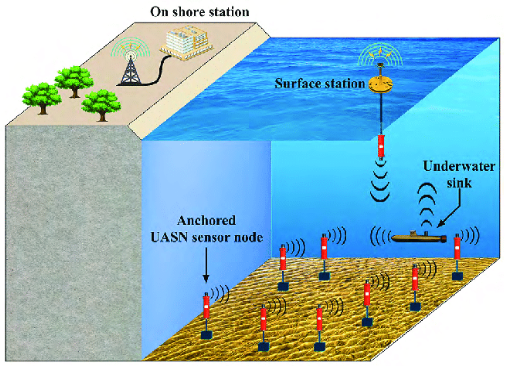
Predictive maintenance

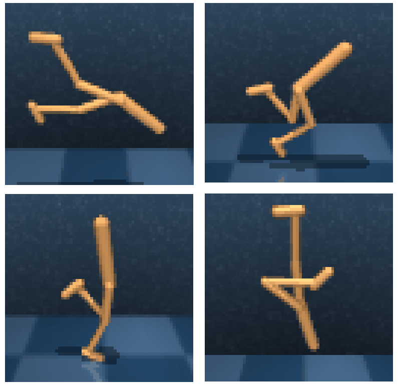


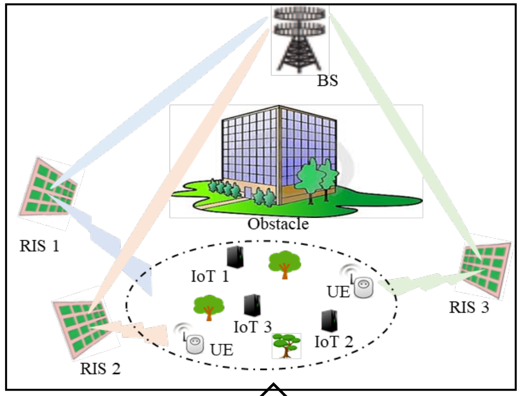
User behavior analysis

Use case: Data generation

- Use ChatGPT to generate various data for communication simulations







Channel generation

Simulated channels, e.g., underwater acoustic channel, indoor/outdoor channel

Experience generation

Interactive dataset for multi-agent reinforcement learning and bandit learning

Image/video generation

Image and video dataset for edge AI, e.g., FL, unsupervised learning

Scenario generation

Real-world environments for system and protocol design

https://arxiv.org/abs/2211.13350

Example: Code generation for simulations (QPSK evaluation)

➢ Use ChatGPT to generate the Matlab code for performance evaluation of QPSK modulation

SI

Imagine we are working on a communication simulation. The goal is to evaluate the QPSK encode and decode modulation. The SNR is (-10,2) dB; The message length is 10,000; The transmit power is 10 mW. The channel is Gaussian. Can you make use of these to write Matlab code to do this work?

Prompts

Yes, we can use MATLAB to simulate QPSK modulation and demodulation with the given parameters. Here's an example code to get you started:

matlab Copy code

Generated M-code

```
% Simulation parameters
snr_dB = (-10:2);     % SNR range in dB
msg_len = 10000;      % Message length
tx_power = 10e-3;     % Transmit power in watts

% Generate random message bits
msg = randi([0 1], 1, msg_len);

% QPSK modulation
modulated = qammod(msg, 4);

% Loop over SNR range and simulate transmission over Gaussian channel
for i = 1:length(snr_dB)
    % Convert SNR from dB to linear scale
    snr_lin = 10^(snr_dB(i)/10);

    % Calculate noise variance
    noise_var = tx_power/snr_lin;
```

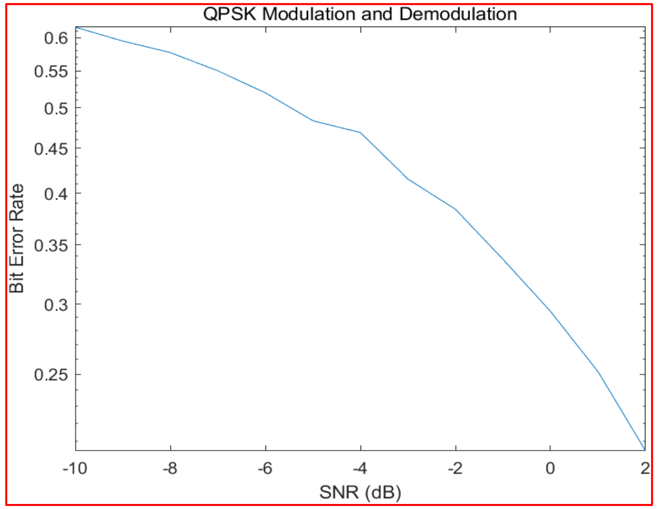
```
    % Add Gaussian noise to the modulated signal
    received = awgn(modulated, snr_dB(i), 'measured');

    % QPSK demodulation
    demodulated = qamdemod(received, 4);

    % Calculate bit error rate (BER)
    ber(i) = sum(msg ~= demodulated)/msg_len;
end

% Plot BER vs SNR curve
semilogy(snr_dB, ber);
xlabel('SNR (dB)');
ylabel('Bit Error Rate');
title('QPSK Modulation and Demodulation');
```

AWGN channel

Task explanation

This code generates a random message of length `msg_len`, modulates it using QPSK modulation, and simulates transmission over a Gaussian channel with different SNR values. For each SNR value, it calculates the noise variance based on the transmit power and SNR, adds Gaussian noise to the modulated signal, demodulates the received signal using QPSK demodulation, and calculates the bit error rate (BER) between the original message and the demodulated message. Finally, it plots the BER vs SNR curve using a logarithmic scale on the y-axis. Note that the transmit power in this simulation is higher than in the previous example, which may result in a better BER performance at higher SNR values.

- Result is reasonable



This result is obtained by directly running the above M-code in Matlab without any modification.

Example: Code generation for simulations (BPSK vs. QPSK)

➢ Use ChatGPT to generate the Matlab code to compare the performance of QPSK and BPSK modulations

SI compare it with BPSK

Prompts

To compare the performance of QPSK and BPSK modulations, we can modify the previous MATLAB code to use BPSK instead of QPSK. Here's an example code that generates a random message, modulates it using BPSK or QPSK modulation, simulates transmission over a Gaussian channel with different SNR values, demodulates the received signal using BPSK or QPSK demodulation, and calculates the bit error rate (BER) for both cases:

matlab   Copy code

Generated M-code

```matlab
% Simulation parameters
snr_dB = (-10:2);      % SNR range in dB
msg_len = 10000;       % Message length
tx_power = 10e-3;      % Transmit power in watts

% Generate random message bits
msg = randi([0 1], 1, msg_len);

% BPSK modulation
bpsk_modulated = pskmod(msg, 2);

% QPSK modulation
qpsk_modulated = qammod(msg, 4);

% Loop over SNR range and simulate transmission over Gaussian channel
for i = 1:length(snr_dB)
    % Convert SNR from dB to linear scale
    snr_lin = 10^(snr_dB(i)/10);

    % Calculate noise variance
    noise_var = tx_power/snr_lin;

    % Add Gaussian noise to the modulated signals
    % Add Gaussian noise to the modulated signals
    bpsk_received = awgn(bpsk_modulated, snr_dB(i), 'measured');
    qpsk_received = awgn(qpsk_modulated, snr_dB(i), 'measured');

    % BPSK demodulation
    bpsk_demodulated = pskdemod(bpsk_received, 2);

    % QPSK demodulation
    qpsk_demodulated = qamdemod(qpsk_received, 4);

    % Calculate bit error rate (BER) for BPSK and QPSK
    bpsk_ber(i) = sum(msg ~= bpsk_demodulated)/msg_len;
    qpsk_ber(i) = sum(msg ~= qpsk_demodulated)/msg_len;
end

% Plot BER vs SNR curves for BPSK and QPSK
semilogy(snr_dB, bpsk_ber, 'b', snr_dB, qpsk_ber, 'r');
xlabel('SNR (dB)');
ylabel('Bit Error Rate');
title('BPSK vs QPSK Modulation and Demodulation');
legend('BPSK', 'QPSK');
```

Task explanation

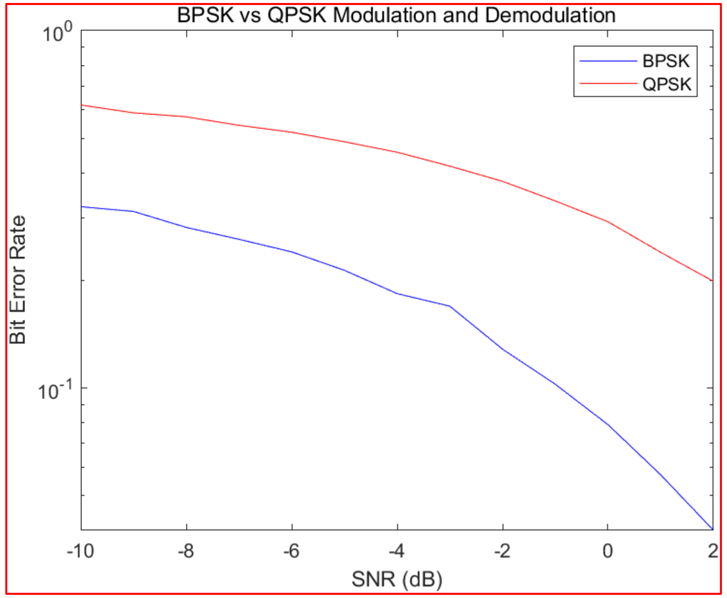
In this code, we first generate a random message of length `msg_len`, and then modulate it using both BPSK and QPSK. We then simulate transmission over a Gaussian channel with different SNR values, and for each SNR value, we calculate the noise variance, add Gaussian noise to the modulated signals, and demodulate the received signals using BPSK and QPSK demodulation. We finally calculate the bit error rate (BER) for both modulations and plot the BER vs SNR curves for comparison.

By running this code, we can see that QPSK modulation provides better BER performance than BPSK modulation at the same SNR value. This is because QPSK can transmit twice as many bits per symbol as BPSK, resulting in a higher spectral efficiency. However, QPSK also requires more complex modulation and demodulation schemes compared to BPSK, which may result in higher implementation complexity and cost.

- Prompt is based on the previous one.

Compare it with BPSK

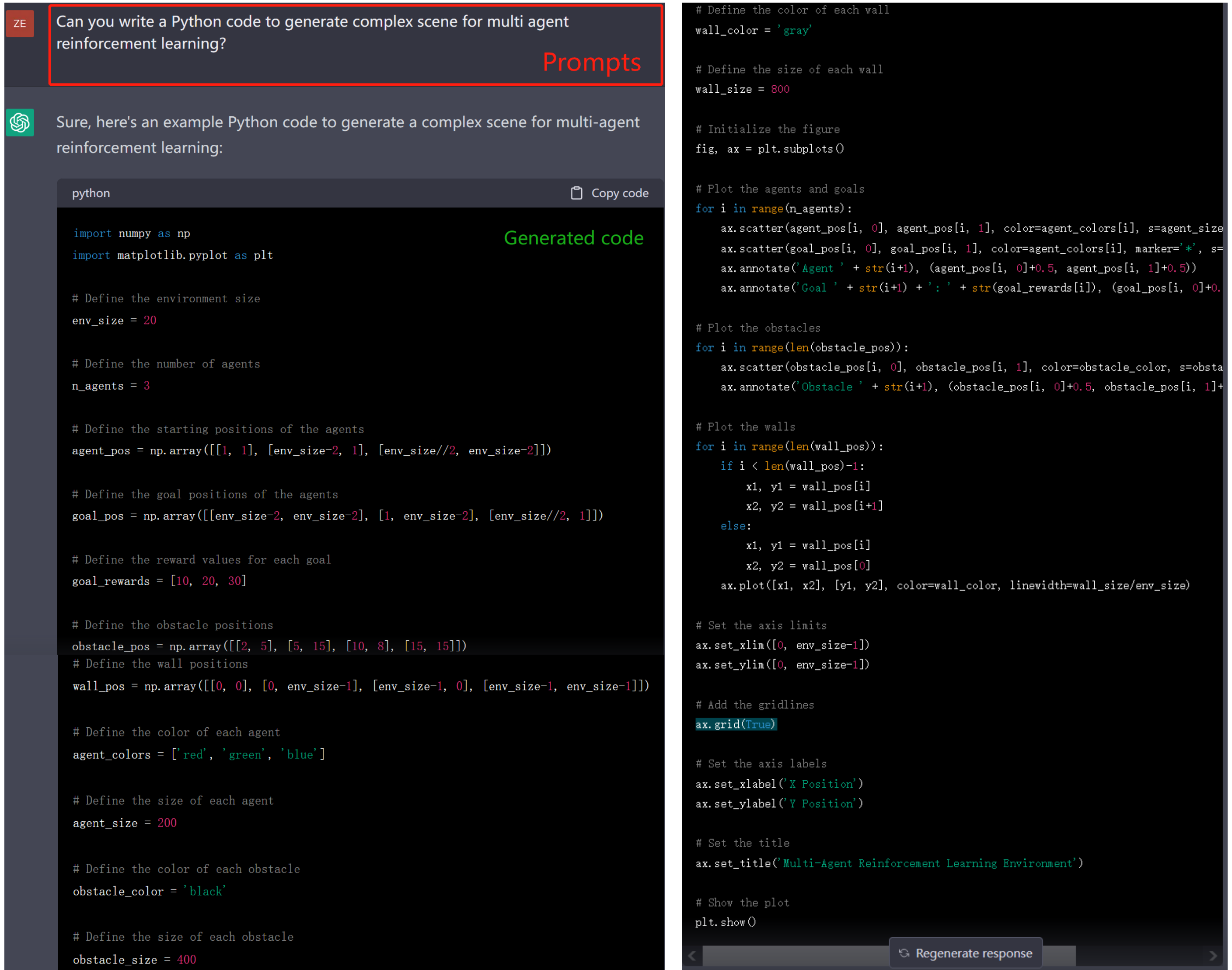
- Result is reasonable



The challenge is how to efficiently design the prompts for ChatGPT.

Example: data generation for multi-agent systems

- Use ChatGPT to generate a complex scene for multi-agent reinforcement learning



ZE

Can you write a Python code to generate complex scene for multi agent reinforcement learning?

Prompts

Sure, here's an example Python code to generate a complex scene for multi-agent reinforcement learning:

python Copy code

Generated code

```python
import numpy as np
import matplotlib.pyplot as plt

# Define the environment size
env_size = 20

# Define the number of agents
n_agents = 3

# Define the starting positions of the agents
agent_pos = np.array([[1, 1], [env_size-2, 1], [env_size//2, env_size-2]])

# Define the goal positions of the agents
goal_pos = np.array([[env_size-2, env_size-2], [1, env_size-2], [env_size//2, 1]])

# Define the reward values for each goal
goal_rewards = [10, 20, 30]

# Define the obstacle positions
obstacle_pos = np.array([[2, 5], [5, 15], [10, 8], [15, 15]])
# Define the wall positions
wall_pos = np.array([[0, 0], [0, env_size-1], [env_size-1, 0], [env_size-1, env_size-1]])

# Define the color of each agent
agent_colors = ['red', 'green', 'blue']

# Define the size of each agent
agent_size = 200

# Define the color of each obstacle
obstacle_color = 'black'

# Define the size of each obstacle
obstacle_size = 400
```

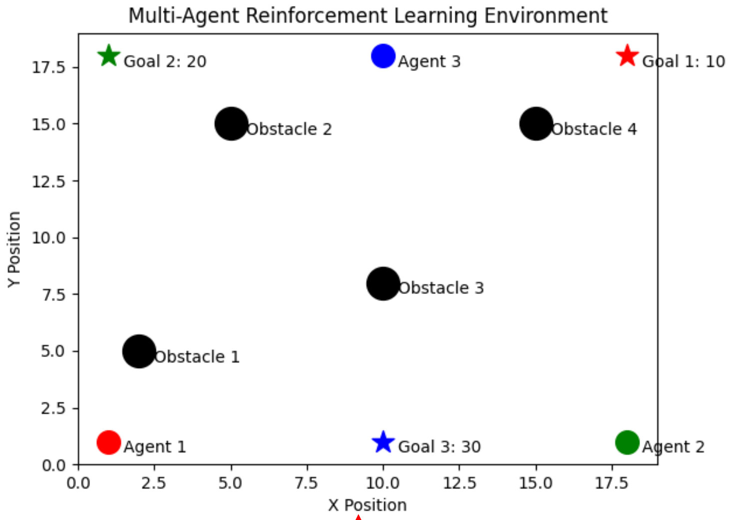
```python
# Define the color of each wall
wall_color = 'gray'

# Define the size of each wall
wall_size = 800

# Initialize the figure
fig, ax = plt.subplots()

# Plot the agents and goals
for i in range(n_agents):
    ax.scatter(agent_pos[i, 0], agent_pos[i, 1], color=agent_colors[i], s=agent_size
    ax.scatter(goal_pos[i, 0], goal_pos[i, 1], color=agent_colors[i], marker='*', s=
    ax.annotate('Agent ' + str(i+1), (agent_pos[i, 0]+0.5, agent_pos[i, 1]+0.5))
    ax.annotate('Goal ' + str(i+1) + ': ' + str(goal_rewards[i]), (goal_pos[i, 0]+0.

# Plot the obstacles
for i in range(len(obstacle_pos)):
    ax.scatter(obstacle_pos[i, 0], obstacle_pos[i, 1], color=obstacle_color, s=obsta
    ax.annotate('Obstacle ' + str(i+1), (obstacle_pos[i, 0]+0.5, obstacle_pos[i, 1]+

# Plot the walls
for i in range(len(wall_pos)):
    if i < len(wall_pos)-1:
        x1, y1 = wall_pos[i]
        x2, y2 = wall_pos[i+1]
    else:
        x1, y1 = wall_pos[i]
        x2, y2 = wall_pos[0]
    ax.plot([x1, x2], [y1, y2], color=wall_color, linewidth=wall_size/env_size)

# Set the axis limits
ax.set_xlim([0, env_size-1])
ax.set_ylim([0, env_size-1])

# Add the gridlines
ax.grid(True)

# Set the axis labels
ax.set_xlabel('X Position')
ax.set_ylabel('Y Position')

# Set the title
ax.set_title('Multi-Agent Reinforcement Learning Environment')

# Show the plot
plt.show()
```

Regenerate response

- Prompts

Can you write a Python code to generate complex scene for multi agent reinforcement learning?

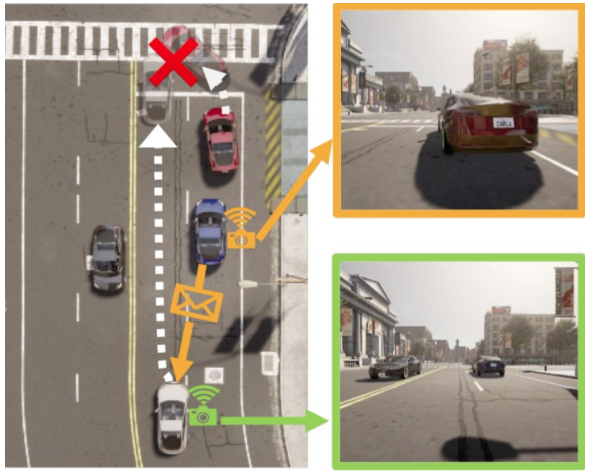
- Result



This result requires modifying the generated codes accordingly and manual fine-tuning

Example: data generation for cooperative perception

- Use ChatGPT to generate a 3D scene for cooperative perception



https://arxiv.org/abs/2209.12836

ZE: Can you write a Python code to generate such a scene for cooperative perception in safety-critical scenarios: the red car is before the blue car, and the white car is after the blue car but does not know the position of the red car. The white car and the red car may collide due to occlusion. This collision could be avoided when the blue car can share a message about the red car's position. Such a message is spatially sparse, yet perceptually critical. Considering the precious communication bandwidth, each agent needs to speak to the point!

Prompts

Sure, here's a Python code that generates a scene with three cars (red, blue, and white) in a 3D space:

csharp — Copy code

Generated code

```
import numpy as np
from mpl_toolkits.mplot3d.art3d import Poly3DCollection
import matplotlib.pyplot as plt

# define the vertices and faces for a simple cube
cube_vertices = np.array([
    [-0.5, -0.5, -0.5],
    [-0.5, 0.5, -0.5],
    [0.5, 0.5, -0.5],
    [0.5, -0.5, -0.5],
    [-0.5, -0.5, 0.5],
    [-0.5, 0.5, 0.5],
    [0.5, 0.5, 0.5],
    [0.5, -0.5, 0.5]
])
cube_faces = np.array([
    [0, 1, 2, 3],
    [4, 5, 6, 7],
    [0, 1, 5, 4],
    [1, 2, 6, 5],
    [2, 3, 7, 6],
    [3, 0, 4, 7]
])

# define the positions and colors of the cars
red_pos = np.array([-2, 0, 0])
green_pos = np.array([2, 0, 0])
blue_pos = np.array([0, 0, 0])
white_pos = np.array([0, 0, -10])
red_color = 'red'
green_color = 'green'
blue_color = 'blue'
white_color = 'white'

# define the vertices and faces for the road
lane_width = 2
road_width = 4
road_length = 30
road_vertices = np.array([
    [-road_width/2, -road_length, 0],
    [-road_width/2, 0, 0],
    [-lane_width/2, 0, 0],
    [-lane_width/2, -road_length, 0],
    [lane_width/2, -road_length, 0],
    [lane_width/2, 0, 0],
    [road_width/2, 0, 0],
    [road_width/2, -road_length, 0],
])
road_faces = np.array([
    [0, 1, 2, 3],
    [3, 2, 5, 4],
    [4, 5, 6, 7]
])

# create a 3D figure
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.set_xlim(-5, 5)
ax.set_ylim(-15, 5)
ax.set_zlim(-15, 5)
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')

# create the cars as Poly3DCollection objects
red_car = Poly3DCollection([cube_vertices[cube_face] + red_pos for cube_face in cube_faces]
blue_car = Poly3DCollection([cube_vertices[cube_face] + blue_pos for cube_face in cube_face
white_car = Poly3DCollection([cub ... ite_pos for cube_face in cube_fa
```

Regenerate response

Code is incomplete

ZE: continue

Generate the remaining part

add the cars to the plot

ax.add_collection3d(red_car)
ax.add_collection3d(blue_car)
ax.add_collection3d(white_car)

add the road to the plot

ax.add_collection3d(road)

add a title

plt.title('Cooperative Perception in Safety-Critical Scenarios')

show the plot

plt.show()

The remaining part is a bit informal

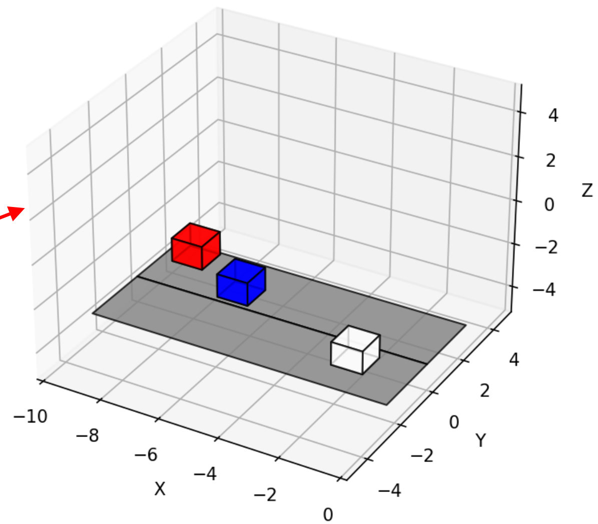
This result requires modifying the generated codes accordingly and manual fine-tuning

- Prompts

Can you write a Python code to generate such a scene for cooperative perception in safety-critical scenarios: the red car is before the blue car, and the white car is after the blue car but does not know the position of the red car. The white car and the red car may collide due to occlusion ….

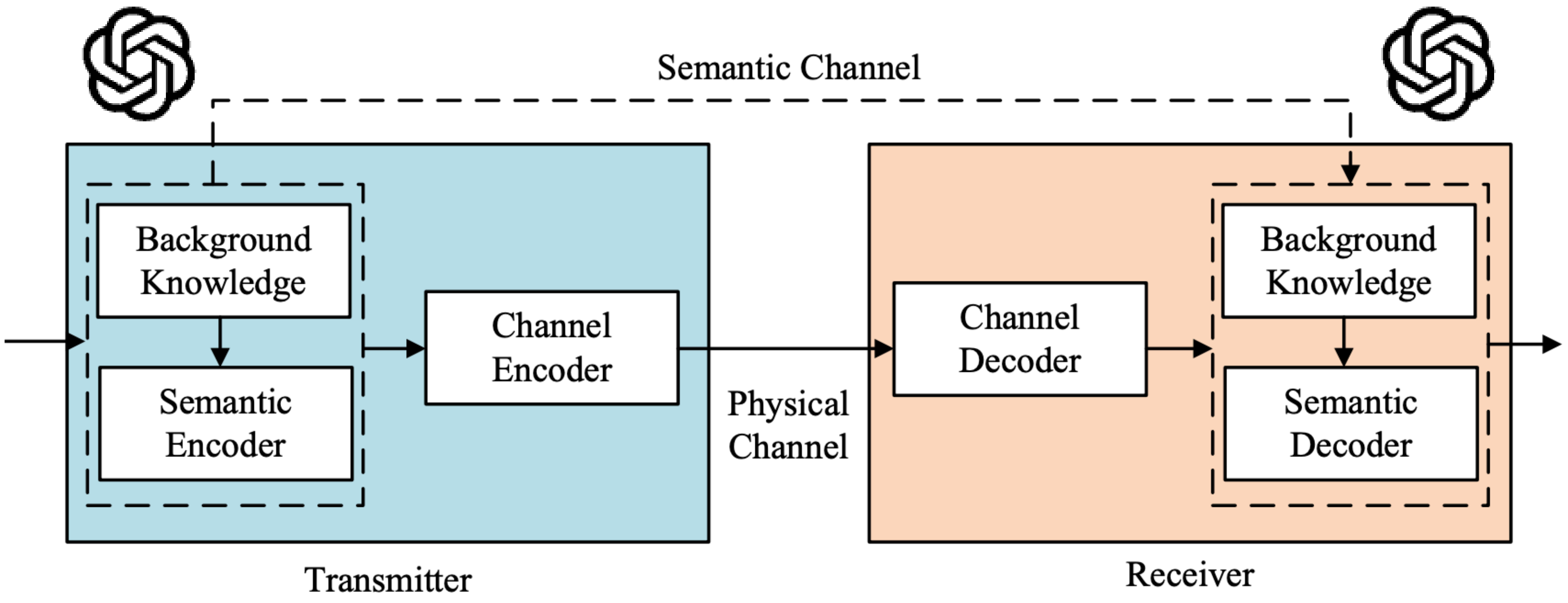
- Result

Cooperative Perception in Safety-Critical Scenarios



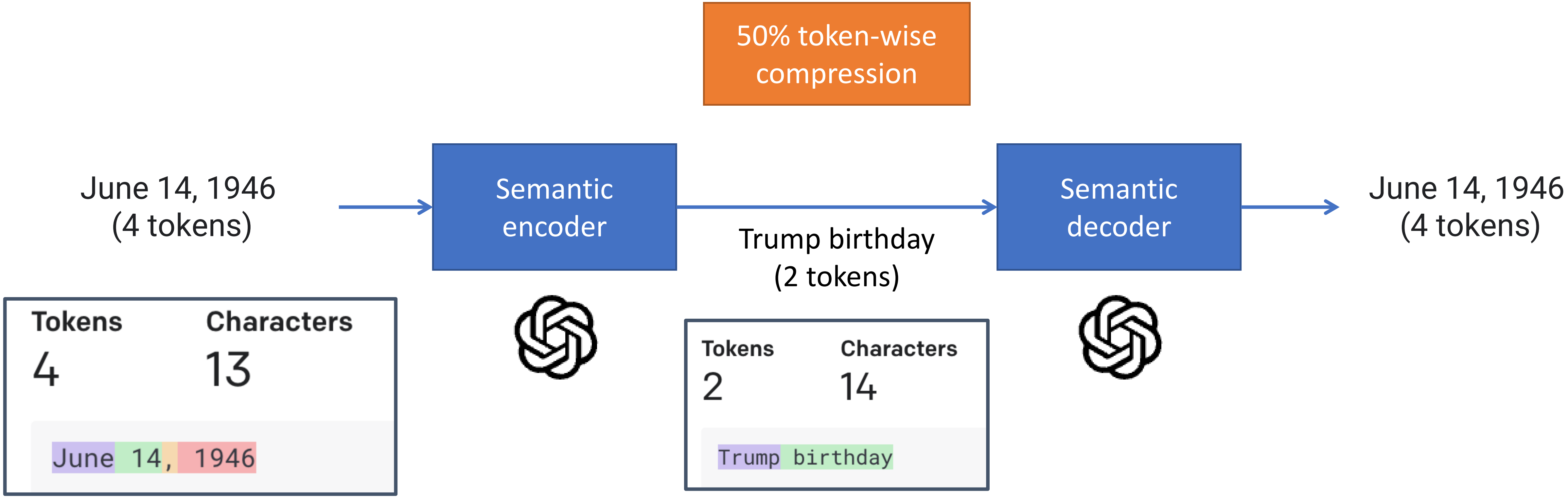
Use case: Semantic communication

- Use ChatPGT as a knowledge base in semantic communication.



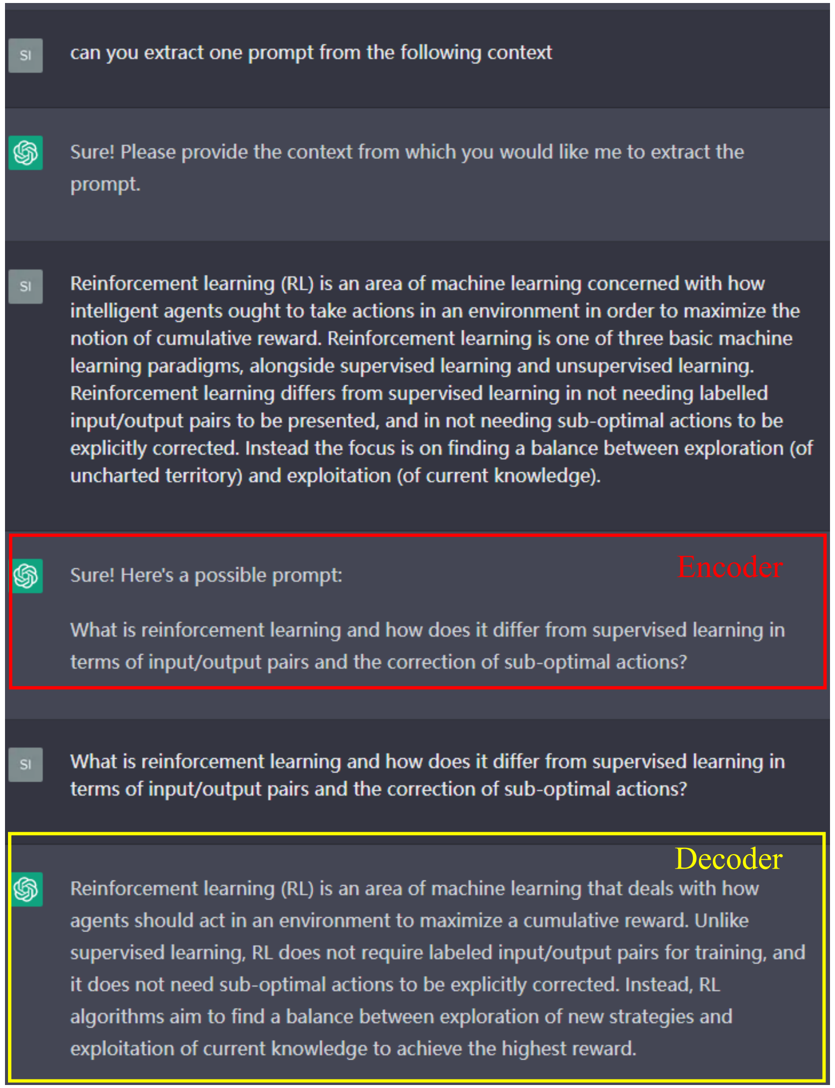
Use case: Semantic communication

- Semantic coding & decoding by using ChatGPT (and large language models).



Use case: Semantic communication

The challenge is how to quantify the transmission error rate.



Transmitter:

- Ask ChatGPT to extract the prompt or meaningful information from a context
- A context copy from Wikipedia about the definition of reinforcement learning (RL)
- Extracted meaningful information or prompt of the above context

Receiver:

- Feed this prompt to the ChatGPT
- Generated content about RL

Wireless Communication for ChatGPT

How to support ChatGPT-enabled mobile applications?

What does ChatGPT say?

SI can you list some challenges on chatgpt in wireless communications?

As an AI language model, ChatGPT does not directly interact with wireless communications systems, but there are a few challenges related to language processing and communication in wireless communications that could be relevant:

1. Limited bandwidth: Wireless communications often have limited bandwidth, which can pose challenges for transmitting large amounts of data, including language models like ChatGPT.
2. Interference: Interference from other wireless devices and sources can disrupt communications, which can affect the accuracy and reliability of language models like ChatGPT.
3. Latency: Latency is the delay between sending and receiving data, and it can be a challenge in wireless communications. High latency can affect the performance of real-time applications like chatbots.
4. Security: Security is always a concern in wireless communications, and language models like ChatGPT that transmit sensitive information can be vulnerable to interception and attack.
5. Power consumption: Wireless devices are often powered by batteries, which can limit their processing power and transmission range. This can pose a challenge for language models like ChatGPT, which require significant computing resources to function effectively.

➡

SI Can you provide some solutions to help overcoming the above challenges

Certainly! Here are some potential solutions to help overcome the challenges mentioned earlier:

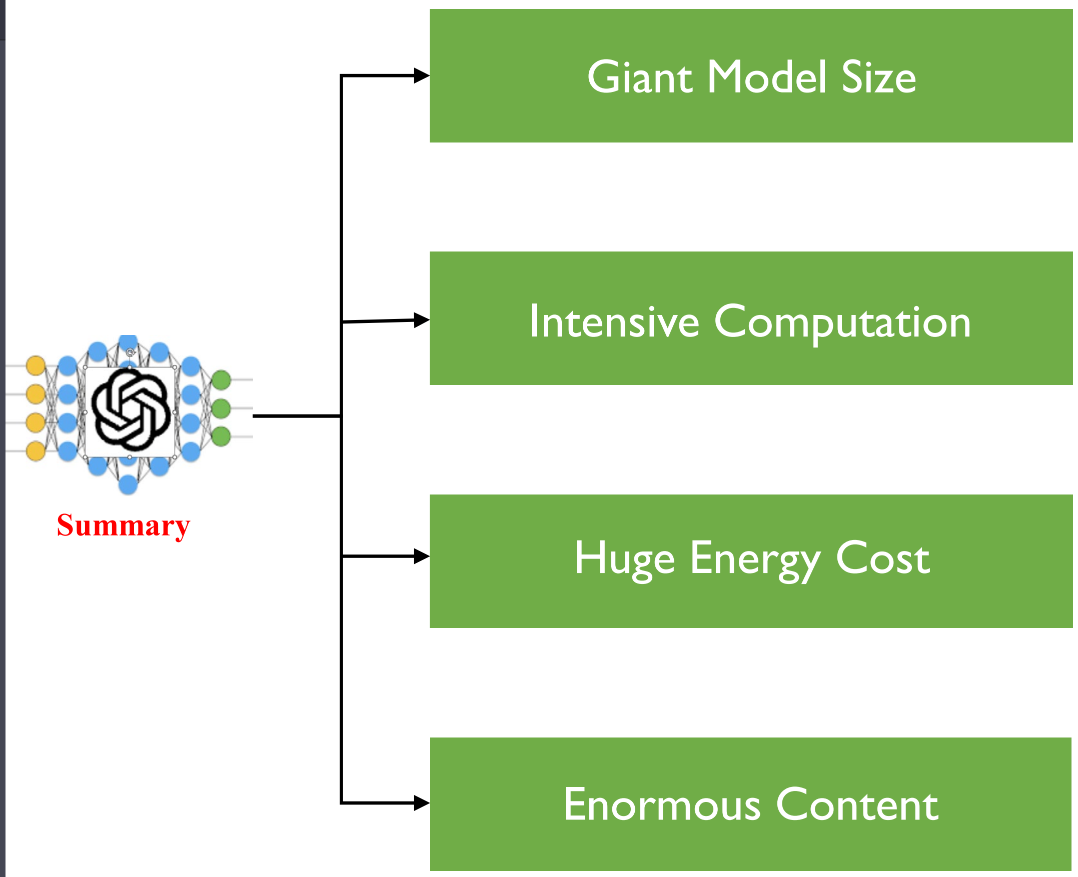
1. Limited bandwidth: To overcome the challenge of limited bandwidth in wireless communications, compression techniques can be used to reduce the amount of data that needs to be transmitted. Additionally, techniques like data prioritization and adaptive modulation can help optimize the use of available bandwidth.
2. Interference: Interference can be reduced by using frequency hopping, which allows wireless devices to switch between different frequencies to avoid interference. Signal processing techniques like interference cancellation can also help reduce the impact of interference.
3. Latency: To reduce latency, techniques like data caching and local processing can be used to minimize the amount of data that needs to be transmitted. Additionally, real-time data compression and transmission protocols like WebSockets can help improve the speed of data transmission.
4. Security: To ensure the security of wireless communications, encryption techniques like SSL and TLS can be used to encrypt data in transit. Additionally, authentication and authorization mechanisms can be used to ensure that only authorized users can access data.
5. Power consumption: To reduce power consumption in wireless devices, energy-efficient protocols like ZigBee and Bluetooth Low Energy can be used. Additionally, techniques like adaptive power management and duty cycling can help reduce power consumption without compromising performance.

New challenges brought by ChatGPT-enabled applications?

SI can you list some challenges on chatgpt in wireless communications?

As an AI language model, ChatGPT does not directly interact with wireless communications systems, but there are a few challenges related to language processing and communication in wireless communications that could be relevant:

1. Limited bandwidth: Wireless communications often have limited bandwidth, which can pose challenges for transmitting large amounts of data, including language models like ChatGPT.
2. Interference: Interference from other wireless devices and sources can disrupt communications, which can affect the accuracy and reliability of language models like ChatGPT.
3. Latency: Latency is the delay between sending and receiving data, and it can be a challenge in wireless communications. High latency can affect the performance of real-time applications like chatbots.
4. Security: Security is always a concern in wireless communications, and language models like ChatGPT that transmit sensitive information can be vulnerable to interception and attack.
5. Power consumption: Wireless devices are often powered by batteries, which can limit their processing power and transmission range. This can pose a challenge for language models like ChatGPT, which require significant computing resources to function effectively.



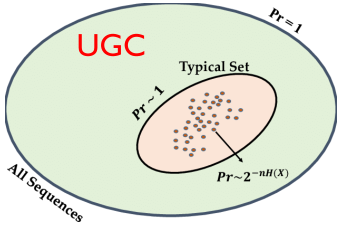
Implications to wireless communications

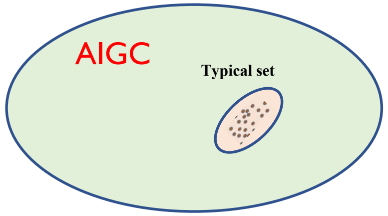
- Challenge 1: Giant model size →
  - Model transmission is not possible
  - May transmit part of/distilled model
- Challenge 2: Intensive computation →
  - Cloud-edge based solutions
  - Limited on-device fine-tuning
- Challenge 3: Huge energy cost →
  - More efficient training
  - Energy-efficient fine tuning
- Challenge 4: Enormous content →
  - Higher network capacity
  - New distortion metric of AIGC

Case study: Semantic measures for AIGC

➢ What is the difference between UGC and AIGC?

	UGC	AIGC
Creator	Individuals	Computers
Control	Fully controlled by users	Algorithms or model parameters
Purpose	A range of reasons	Specific, task-oriented purpose
Legitimacy and Authenticity	more authentic and trustworthy	more skepticism, error content generated randomly
Quality and Consistency	Informal, vary widely in quality, style, and tone	Consistent in quality and style, fewer variations in tone





Hypothesis: The AIGC typical set may be much smaller than the UGC typical set due to quality and consistency considerations

➢ What types of **distortion metrics** are more suitable for source coding?

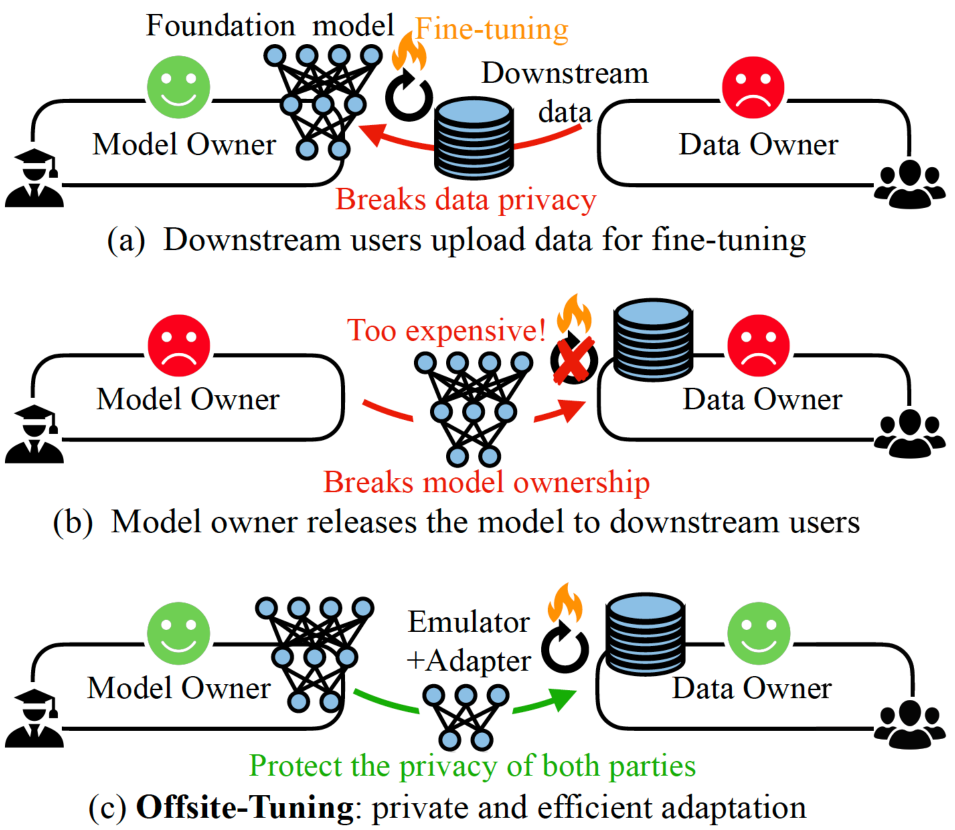
- Mean Square Error (MSE)
- Peak Signal-to-Noise Ratio (PSNR)
- Structural Similarity Index (SSI)
- Wasserstein Distance Metric?

➢ What types of **errors** are more detrimental in channel coding?

- Burst Error: two or more consecutive bits are corrupted during transmission
- Random error: single bit is corrupted randomly during transmission
- Other types of errors?

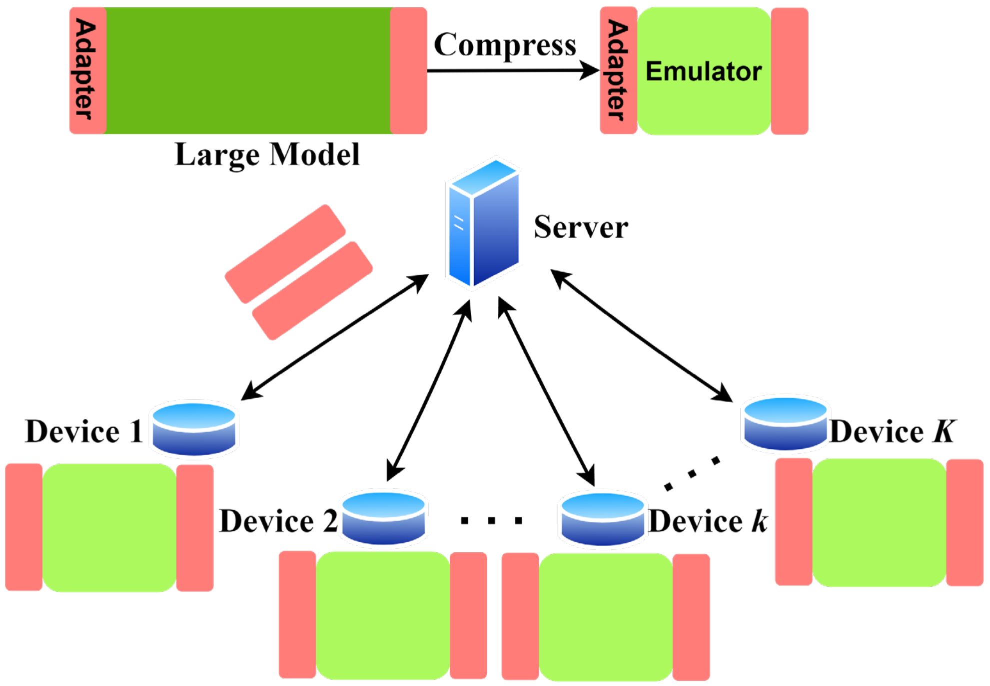
Case study: Privacy-preserving on-device fine tuning

Giant Model Size



https://arxiv.org/abs/2302.04870

- One solution
  - The model owner shares a lossy compressed emulator and an adapter
  - The data owner trains the adapter



Takeaways

Takeaways

- AI becomes real!
  - It is bringing a revolution
- New opportunities for wireless
  - AIGC-enabled techniques
- New challenges for wireless
  - Giant model size, intensive computation, huge energy cost, enormous content
- Everyone needs to learn prompting engineering!

Thank you!

- For more details

  https://eejzhang.people.ust.hk/