## **How to Reduce Hallucinations in LLMs**

### **🔍 What is “Hallucination” in LLMs?**

Before going into the strategies, let’s understand **what hallucination means** in the context of Large Language Models (LLMs) like ChatGPT.

**Hallucination** means when the AI gives an answer that sounds good or fluent but is **false, made-up, or inaccurate**.

For example:

* ❌ “The Eiffel Tower is in Berlin.” → Sounds confident but **completely wrong**.
* ❌ “The capital of Canada is Toronto.” → Also false (correct is **Ottawa**).

## **✅ How to Reduce Hallucinations in LLMs**

Now let’s explain each technique from your given text **step by step**:

### **1. Provide Clear and Specific Prompts**

#### **💬 What it means:**

If you give a **vague** or **general** prompt, the AI might guess or make up stuff. If your prompt is **detailed and specific**, the AI is more likely to stay **on topic** and give **accurate info**.

#### **✅ Example:**

* ❌ Bad prompt: “Tell me about World War Two.”  
   → This is too open-ended. The AI might go off-track or generalize.
* ✅ Better prompt:  
   “Summarize World War Two in 4 bullet points, including the main countries involved, causes, and the result.”  
   → This tells the model **exactly what you want**, reducing hallucination.

### **2. Active Mitigation Strategies (Like Adjusting Temperature)**

#### **🔧 What is “Temperature” in LLMs?**

This is a **setting that controls how random or creative** the model is.

* A **lower temperature** (like 0.2) means:  
  + The model plays it **safe**.
  + You get **more reliable and factual answers**.
  + Example: Good for **math, science, history**, etc.
* A **higher temperature** (like 0.9) means:  
  + The model becomes more **creative** or even **risky**.
  + It might **invent** new stories or say unusual things.
  + Example: Good for **writing poems, jokes**, or fiction.

#### **✅ Example:**

You’re asking the model:  
 “Write a sentence about the moon landing.”

* temperature = 0.2:  
   → “Neil Armstrong landed on the moon in 1969.” (✔ factual)
* temperature = 0.9:  
   → “The moon landing revealed an ancient alien base.” (❌ made-up)

So if your goal is **accuracy**, use a **lower temperature**.

### **3. Multi-shot Prompting**

#### **🧠 What is it?**

“**Shot**” here means an **example**.

* **One-shot prompting** = You give **one** example.
* **Few-shot prompting** = You give **a few** examples (usually 2–5).
* **Multi-shot prompting** = You give **several** examples to help the model **learn the pattern or style** you want.

This helps the model **stay consistent**, **follow instructions better**, and **reduce hallucination**.

#### **✅ Example: Asking the model to generate Python code**

**Prompt without example (zero-shot):** “Write a Python program to check even or odd.”

→ Model might do it, but could format the code differently every time.

**Prompt with multi-shot (multiple examples):**

# Example 1:

# Program to add two numbers

a = 3

b = 5

print("Sum is", a + b)

# Example 2:

# Program to check if number is positive

num = -4

if num > 0:

print("Positive")

else:

print("Not positive")

# Now write a program to check if a number is even or odd:

→ Model now **understands your style** and will follow the same format. This consistency helps prevent errors or hallucinations.

## **🔚 Summary (Simple Bullet Points)**

| **Method** | **How It Helps** | **Example** |
| --- | --- | --- |
| **Clear Prompts** | Removes confusion, gives model exact instruction | “Summarize WW2 in 4 points including causes” |
| **Temperature** | Lower = safer, Higher = more creative (but risky) | Use 0.2 for facts, 0.9 for poems |
| **Multi-shot Prompting** | Shows the model what output you expect | Give examples of code, answers, summaries |