Training Day 11 Report:

Overview

When using AI to evaluate or generate code, parameters like **temperature**, **top-k**, **top-p**, and **token limits** play a crucial role in controlling the behavior of the model. These values directly influence how accurate, creative, or concise the output is.

Prompt Design with Control Parameters

A typical function may take user input (such as a block of code) and pass it along with an instruction prompt to a model. Example logic:

- Ask the model to identify and correct errors.
- Request an optimized version of the same code.
- Use control parameters to keep responses focused and relevant.

Understanding Key Parameters

- **Temperature (e.g., 0.3):** Lower values make the model more deterministic and focused—ideal for code correction.
- **Top-k (e.g., 40):** Limits the number of top probable words considered at each step.
- **Top-p (e.g., 0.7):** Chooses from a dynamic set of words whose total probability mass is ˜70%.
- **Max output tokens (e.g., 550):** Controls how much the model can respond—useful for ensuring complete code snippets.

Sample Use Case

A simple Python script collects code from the user and asks the AI model to:

1. Detect and name errors.
2. Provide a corrected version.
3. Suggest an optimized alternative.

The parameters are tuned to avoid overly creative responses while ensuring completeness and accuracy, especially useful for technical prompts.

Conclusion

Fine-tuning prompt parameters allows better control over AI-generated coding outputs. For tasks like code validation and improvement, a lower temperature and a defined token limit help produce reliable and focused results, especially when using advanced multimodal models capable of understanding complex instructions.