Diving Deep: How Do LLMs Learn On The Fly?

Charlie Kerfoot*, **Rashfiqur Rahman[1,2]*, Amy Wu[1,3]*,** Andrew Tang[1], Vishal Misra[1]
*Equal Contribution, [1]Columbia University, [2]Boston University, [3]University of Florida

Introduction

Large Language Models (LLMs) have significantly advanced Natural Language Processing (NLP), showcasing capabilities in text generation, and comprehension. One of the many phenomena in LLMs is in-context learning, where task-specific responses are generated by providing the model with task-specific prompts

Goal: This study investigates the behavior of LLMs through the lens of **Bayesian learning**, that is how the model is constantly learning based on new evidence. To further explore this topic, we propose a practical **open-source tool.** Using a BitsandBytes-4-bit quantized version of Meta Llama 3, it visualizes token probabilities during text generation, providing empirical insight into decision-making processes. This lets us analyze how the model is learning in **real-time**.

Methods

Prompt Token Probabilities

- The prompt (p) is tokenized:

$$T(p) = \{t_1, t_2, \ldots, t_n\}$$

- We compute the Logits Matrix $L \in R^{n\times|V|}$ through a single forward pass on $T(p)$. $L[i, j]$ represents the logit for the j-th token in the vocabulary at the i-th position.
- Iteratively, we extract the logits for each token, and apply the softmax function to convert logits to probabilities:

$$P(t = v_j \mid t_1, \ldots, t_{i-1}) = \frac{\exp(L[i, j])}{\sum_k \exp(L[i, k])}$$

- The top k probabilities are then retrieved from each iteration:

$$K = \{t_i \mid P(t_i) \in \text{top } k \text{ values of } P\}$$

- We then create a dictionary mapping the current token to predicted tokens in that same position, and its corresponding probabilities. This dictionary is what we display when hovering over each token in the prompt in our User Interface.

Completion Probabilities

- Using the tokenized prompt, the next step is generating output, or predicting the next tokens:

$$t_{n+1} = \arg\max(P(t \mid t_1, \ldots, t_n))$$

- We use `model.generate` with parameters like beam search to explore multiple potential sequences. Beam search keeps track of multiple potential sequences at each step and selects the one with the highest overall probability.
- Other parameters included `max_new_tokens`, which limits the maximum number of tokens generated to avoid overly long outputs. The `temperature` parameter controls the randomness of predictions by scaling the logits before applying softmax.
- Iteration process for generating new tokens until the desired length is reached or stopping criteria (newline or end of sentence token) is hit:

Open-source Tool

User Input

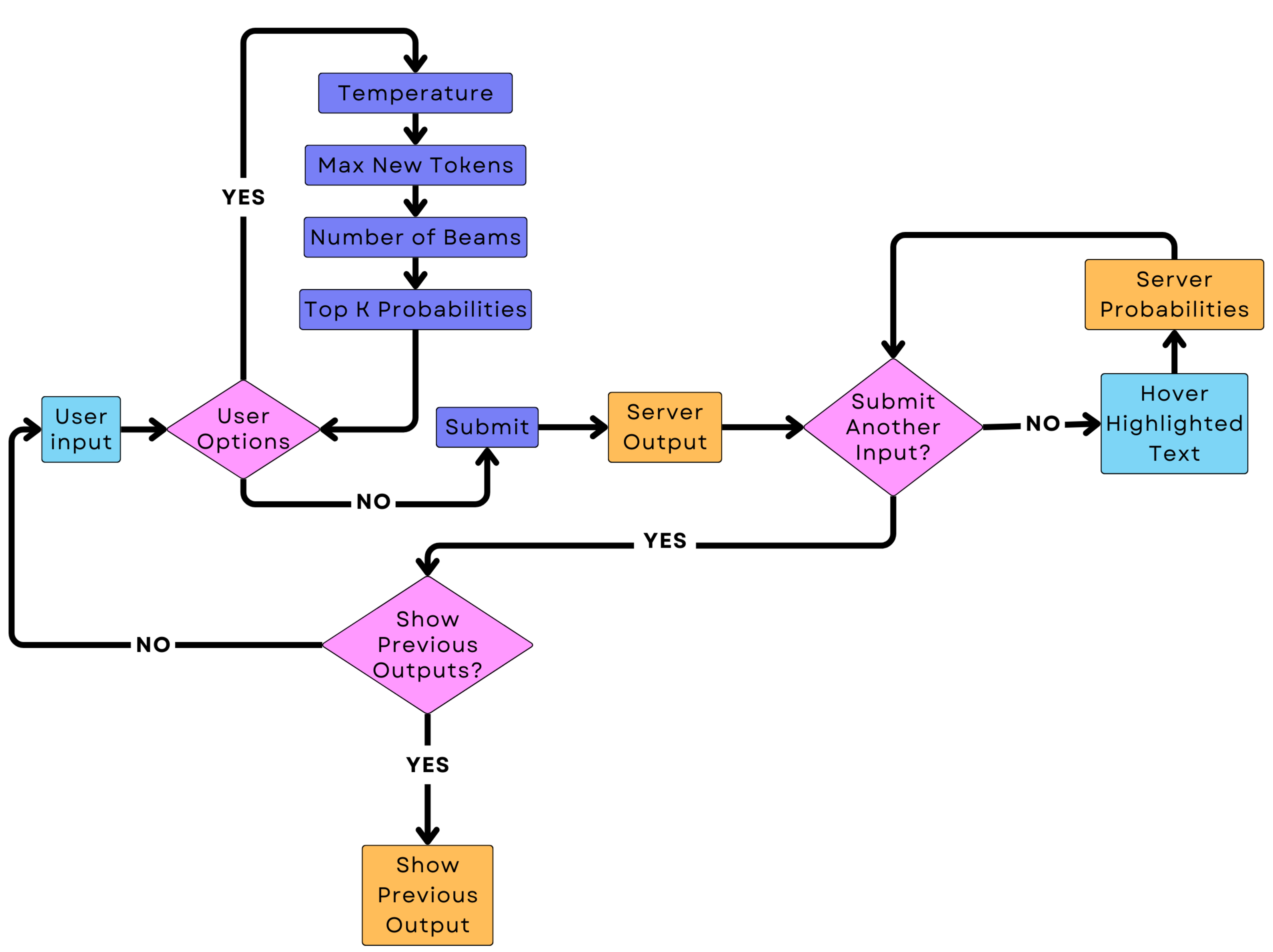
The server is currently generating an output based on user input.

Server Output

Server completion shows probabilities via text hover..

Server Workflow

The workflow displays how the user engages with the open-source tool.



Preliminary Results

- Created an **open-source tool** for token probabilities visualization that is extendible to all Huggingface models.
- **Temperature:** ↓ Deterministic ↑ Random
- **Max New Tokens:** ↓ Concise ↑ Verbose
- **Number of Beams:** ↑ Related ↓ Unrelated
- **Top K Probabilities:** Balance of the top probabilities for clarity
- The model becomes more confident after seeing repeated prompt tasks, as evident from the example in the "Server Output" section. After seeing multiple query-response pairs, the model is now able to learn on the fly and come up with its own answers to queries.

Future Works

- Make our repository **accessible** to community
- Develop **visualization tools** for LLM learning
- Explore **Bayesian framework** for in-context learning through our tool

References

[1]S. Dalal and V. Misra, “The Matrix: A Bayesian Learning Model for LLMs.” Accessed: Jun. 07, 2024. [Online]. Available: https://arxiv.org/pdf/2402.03175
[2]H. Touvron *et al.*, “LLaMA: Open and Efficient Foundation Language Models,” *arXiv:2302.13971 [cs]*, Feb. 2023, Available: https://arxiv.org/abs/2302.13971

Acknowledgements

Distributed Networks Analysis (DNA) Lab and Columbia SURE.

Questions/Comments: vishal.misra@columbia.edu, at3456@columbia.edu

BOSTON UNIVERSITY UF UNIVERSITY of FLORIDA COLUMBIA | ENGINEERING The Fu Foundation School of Engineering and Applied Science