1 Background

In this section, we will introduce standard speculative sampling [1, 2] to help readers unfamiliar with this field better understand EAGLE. We use t_i to denote the i-th token and $T_{a:b}$ to represent the token sequence $t_a, t_{a+1}, \cdots, t_b$. We use M_o to denote the original LLM and M_d to represent the draft model.

Modern LLMs generate text autoregressively, requiring the full model weights to be transferred from memory to cores for each token generation. The time cost of accessing weights far exceeds the computation cost. Therefore, LLM inference is memory-bound. In text generation tasks, the difficulty of generating different tokens varies, and a smaller model can generate some simple tokens. This observation has inspired a class of methods, represented by speculative sampling, which generate multiple tokens in one forward pass to accelerate LLM inference. The core idea of speculative sampling methods is to first draft and then verify: quickly generate a potentially correct draft and then check which tokens in the draft can be accepted.

Consider a prefix $T_{1:j}$, speculative sampling alternates between drafting and verification stages. In the drafting stage, speculative sampling invokes a draft model M_d (a smaller LLM than M_o) to generate a draft $\hat{T}_{j+1:j+k}$ with $T_{1:j}$ as the prefix. In the verification stage, speculative sampling calls the original LLM M_o to check the draft $\hat{T}_{j+1:j+k}$ and concatenates the correct parts into the prefix.

Drafting Stage. We use the draft model M_d to autoregressively generate k tokens while recording the corresponding distributions $\hat{p}$:

$$\hat{t}_{j+1} \sim \hat{p}_{j+1} = M_d(T_{1:j}),$$

$$\hat{t}_{j+i} \sim \hat{p}_{j+i} = M_d(\text{concat}(T_{1:j}, \hat{T}_{j:j+i-1})), i = 2, \cdots, k,$$

where concat$(\cdot, \cdot)$ denotes the concatenation of two sequences. M_d is a smaller LLM. The draft $\hat{T}_{j+1:j+k}$ generated by M_d has a lower computational cost while having a certain probability of being consistent with the generation results of M_o.

Verification Stage. The verification stage checks the draft $\hat{T}_{j+1:j+k}$ and keeps the parts consistent with M_o. We leverage the parallelism of LLMs. Given the input sequence concat$(T_{1:j}, \hat{T}_{j+1:j+k})$, one forward pass of the LLM can compute $k+1$ distributions:

$$p_{j+1} = M_o(T_{1:j}),$$

$$p_{j+i} = M_o(\text{concat}(T_{1:j}, \hat{T}_{j:j+i-1})), i = 2, \cdots, k+1.$$

Then, we decide whether to accept each token in the draft from front to back. For token $\hat{t}_{j+i}$, the probability of it being accepted is $\min(1, p_{j+i}(\hat{t}_{j+i})/\hat{p}_{j+i}(\hat{t}_{j+i}))$. If $\hat{t}_{j+i}$ is accepted, we continue checking the next token; otherwise, we sample a token from the distribution norm$(\max(0, p_{j+i} - \hat{p}_{j+i}))$ to replace $\hat{t}_{j+i}$ and discard the remaining tokens in the draft. The result of this sampling method is exactly consistent with directly sampling from the distribution p computed by M_o. The proof can be found in Appendix A.1 of [2].

We concatenate the accepted draft to $T_{1:j}$ to form a new prefix, and then start the next round of drafting and verification.

References

[1] Charlie Chen et al. "Accelerating large language model decoding with speculative sampling". In: *arXiv preprint arXiv:2302.01318* (2023).

[2] Yaniv Leviathan, Matan Kalman, and Yossi Matias. "Fast inference from transformers via speculative decoding". In: *International Conference on Machine Learning*. PMLR. 2023, pp. 19274–19286.