Non-deterministic nature of llms

martes, 23 de julio de 2024 8:42

https://community.openai.com/t/a-question-on-determinism/8185

https://www.reddit.com/r/LocalLLaMA/comments/19crr3f/how_to_get_deterministic_output_across_100_gpus/

Why LLM's responses vary from time to time?

There are so many possibilities to explain this behaviour:

(1) In addition to the inherent non-deterministic underlying architecture built in the core ethos of modern AI work.

(2) At the end, models generate next tokens. We see the output as text but inside it is generating a string of numbers and probabilities from the tokens distribution it can pick. For example, GPT-4 has 100,000 tokens it can choose from, so each result is the 100,000 list of model's probabilities to each possible token, selecting the top-k inherent in the configuration metric used.

Then, these probabilities are passed to a SoftMax (https://www.baeldung.com/wp-content/uploads/sites/4/2023/05/softmax_animation.gif) where Temperature is applied, which smooths out the probability distribution, decreasing the likelihood of the most probable token choices and increasing the likelihood of the less likely tokens. After SoftMax and Temperature is applied, one token is picked in accordance with the metrics and that is how you get the token the user sees.

(Note: So, with too low temperature the results are more predictable in responses every time but setting it too high you increase the possibilities of receiving nonsense sentences or causing coherent but incorrect responses.)

Integrity and authenticity of hosted model using different GPUs:

Even if you set Temperature as Zero, it is still possible to get different answers because you are using GPUs.

In this case, the model will choose the token with the highest probability. However, when there are multiple words competing probabilities associated super close to each other, due to how GPUs calculate via floating points, it can be a coin toss.

(Note: Floating points are the representation used by computers by redefine real numbers within a limited precision. It is efficient but often provides minimal error due to rounding.)

When using models, there are speed trade-offs, so to make the endpoints, fast GPUs are used, which do parallel (non-deterministic) calculations. Any modern GPU neural net calculations will be subject to these. The CPUs compute sequentially the calculations, whereas the GPUs perform many calculations simultaneously (in parallel) gaining this time advantage.

When using GPUs, due to this parallel way of performing, the order in which calculations are performed might vary overtime, even for the same input. Leading to non-deterministic results, meaning the output can differ slightly on each run. So, A*B*C could be calculated (A*B) *C or A* (B*C) or ... but tiny differences can occur when performing floating point operations, which lead to slightly different results.

This variability is a byproduct of the trade-off between achieving fast processing speeds and maintaining exact reproducibility. (Personal opinion: the thinking about craving

deterministic results at all costs must go, because we are co-working with the model, and that approach gives the best results).

It is not possible to get deterministic outputs with a LLM, you can narrow the margin of error significantly, but it is not bullet proof. Basically, you can only get the same output with the exact same input with the same setup, that is consistent, but change in a variable just the slightest, and it will diverge.

You cannot score the deterministic value of the same model in different environments by comparing their outputs, there are just too many differing variables (outside of generation settings) between the systems, so it is difficult to define.

“LLM non-deterministic nature is more of a feature than a bug”

“If you are asking for deterministic results on an LLM you basically do not understand the basics of an LLM. But the good news is you can lower the chances of different answers by setting some parameters, however it still does not be deterministic”

In practice, achieving absolute determinism can be difficult due to the complex nature of deep learning models and the factors that introduce variability. However, in a controlled scenario, (for instance, when a model is finetuned extensively on a specific pattern) the output of that pattern can become deterministic.