SELF-CONSISTENCY IMPROVES CHAIN OF THOUGHT REASONING IN LANGUAGE MODELS

LLM Competitions

Yalei Yu

April 19, 2025

- Self-consistency leverages the intuition that a complex reasoning problem typically admits multiple different ways of thinking, leading to its unique correct answer

- a new decoding strategy, self-consistency, to replace the naive greedy decoding used in chain-of-thought prompting, that further improves language models' reasoning performance by a significant margin

Approaches

- propose a "sample-and-marginalize" decoding procedure
- first samples a diverse set of reasoning paths instead of only taking the greedy one
- selects the most consistent answer by marginalizing out the sampled reasoning paths
- majority vote

Examples

Self-consistency over diverse reasoning paths

Definitions

- $(\mathbf{r}_i, \mathbf{a}_i)$, where $\mathbf{a}_i$ a generated answer, $\mathbf{r}_i$ a sequence of tokens

a marginalization over $\mathbf{r}_i$ by taking majority vote over $\mathbf{a}_i$:

$$\arg\max_a \sum_{i=1}^{m} \mathbb{1}(\mathbf{a}_i = a) \tag{1}$$

Majority vote is a simple decision rule in many fields, like machine learning, ensemble methods, and voting systems. It means: **Choose the option that more than half of the voters (or models, or data points) agree on**.

Interpretation: This expression computes the value of a that appears most frequently in the sequence $a_1, a_2, \ldots, a_m$. In other words, it returns the **mode** of the list $\{a_i\}$.

Example:
If

$$\mathbb{A} = [2, 3, 2, 4, 2, 3],$$

then:

$$\sum_i \mathbf{1}(a_i = 2) = 3$$

$$\sum_i \mathbf{1}(a_i = 3) = 2$$

$$\sum_i \mathbf{1}(a_i = 4) = 1$$

So,

$$\arg\max_a \sum \mathbf{1}(a_i = a) = 2$$

Normalization

Normalize the conditional probability $P(\mathbf{r}_i, \mathbf{a}_i|\text{prompt, question})$ by the output length, "normalized" weighted sum as follows:

$$P(\mathbf{r}_i, \mathbf{a}_i|\text{prompt, question}) = \exp^{\frac{1}{K}\sum_{k=1}^{K} \log P(t_k|\text{prompt, question}, t_1, \cdots, t_{k-1})} \quad (2)$$

where $\log P(t_k|\text{prompt, question}, t_1, \cdots, t_{k-1})$ is the $\log$ probability of generating the $k-th$ token t_k in $(\mathbf{r}_i, \mathbf{a}_i)$ conditional previous tokens, and K is the total number of tokens.