Better Zero-Shot Reasoning with Self-Adaptive Prompting

Muning Wen

10.18

Better Zero-Shot Reasoning with Self-Adaptive Prompting

Xingchen Wan[*1,3], **Ruoxi Sun**[1], **Hanjun Dai**[2], **Sercan Ö. Arık**[1], **Tomas Pfister**[1]
[1]Google Cloud AI Research
[2]Google DeepMind
[3]Department of Engineering Science, University of Oxford
{xingchenw,ruoxis,hadai,soarik,tpfister}@google.com

https://arxiv.org/abs/2305.14106

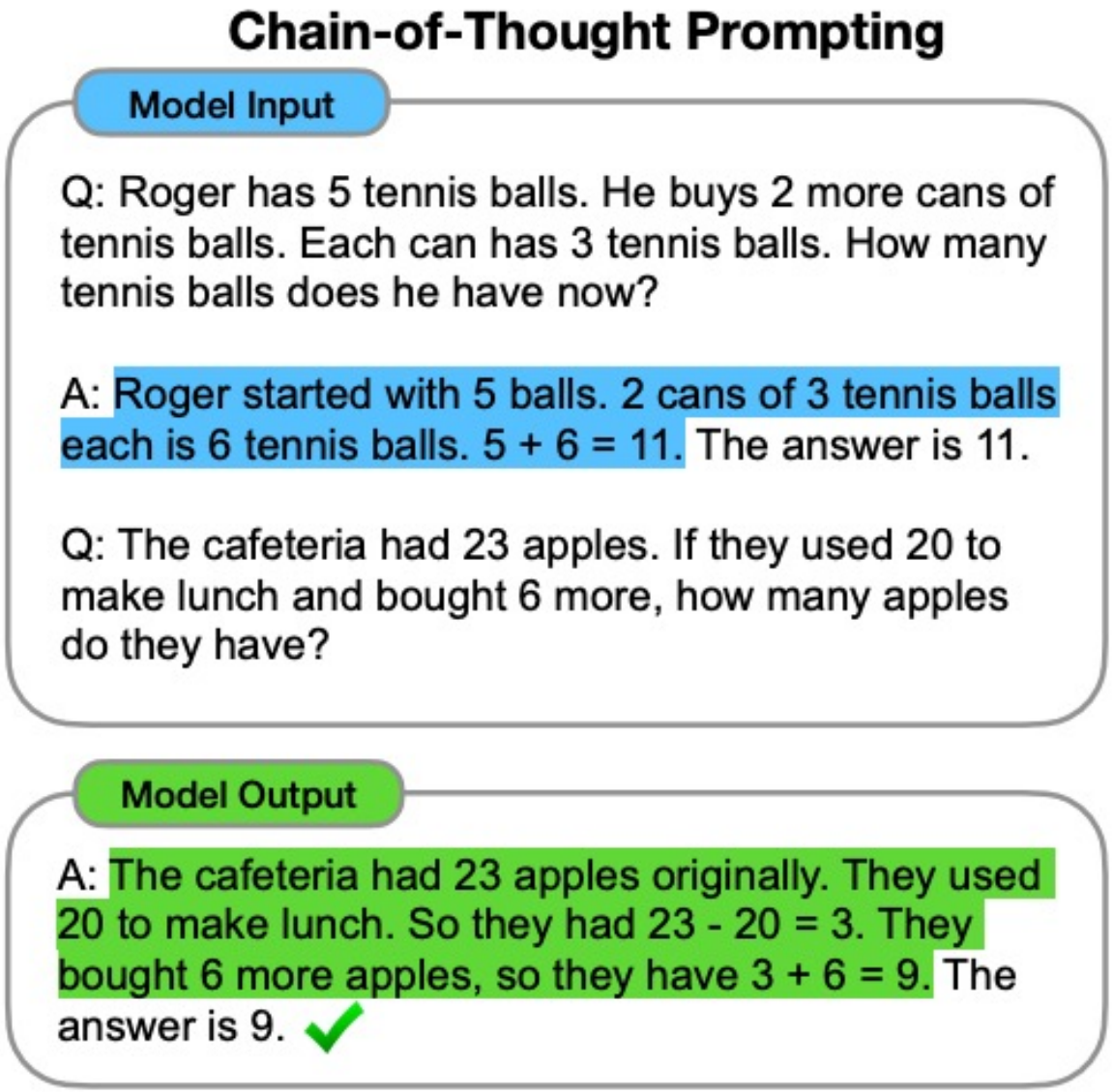
Motivation

Limitation of zero-shot reasoning:
Its performance is limited due to the lack of guidance to the LLMs.

Limitation of in-context reasoning:
1. Performance is sensitive to the choice of examples.
2. Designing examples requires significant human effort.
3. The diversity of downstream tasks of LLMs/novel test-time tasks unseen previously.

The foucus of this work:
Improving LLM reasoning ability in the general zero-shot setup with access to input queries but not labels.

Key idea:
Collect a pool of rationales and answers to a set of questions with Zero-shot CoT, then select the most suitable questions as in-context examples.



Problem Analysis - How impactful are the contextual examples?

Problem Settings:

1. Zero-shot reasoning (self-adaptive prompting)
2. A set of input queries are available (diverse).
3. Do not need labels.

The influence of in-context examples:

1. Zero-shot CoT with no demo: correct logic but wrong answer;
2. Correct demo and correct answer;
3. Correct but repetitive demo leads to repetitive outputs;
4. Erroneous demo leads to a wrong answer;
5. Combining erroneous and correct demo leads to a correct answer.

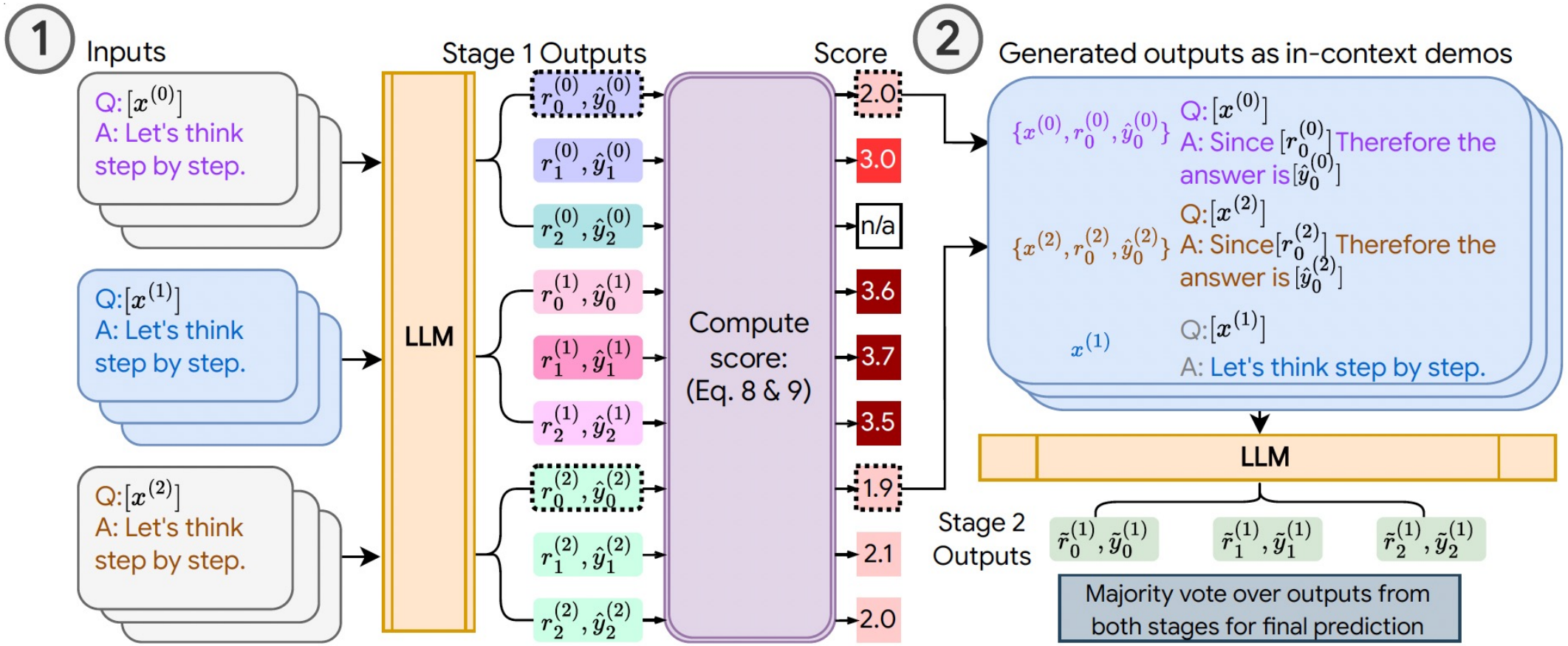
Thus:

In-context demos need carefully-designed selection procedure (key objective of this paper).

[Question] Henry had 11 dollars. For his birthday he got 18 more dollars but spent 10 on a new game. How much money does he have now?
[Demo1] Q: John bought 21 games from a friend and bought 8 more at a garage sale. If 23 of the games didn't work, how many good games did he end up with? A: Let's think step by step. He bought 21 + 8 = 29 games in total. He has 29 - 23 = 6 good games. ✅
[Demo2] Q: John bought 21 games from a friend and bought 8 more at a garage sale. If 23 of the games didn't work, how many good games did he end up with? A: Let's think step by step. He bought 21 + 8 = 29 games in total. He bought 21 + 8 = 29 games in total. He bought 21 + 8 = 29 games in total. He has 29 - 23 = 6 good games. He has 29 - 23 = 6 good games. ✅
[Demo3] Q: Faye picked 23 and her mother picked 5 carrots from the garden. If only 12 of the carrots were good, how many carrots were bad? A: Let's think step by step. 23 + 5 = 28 28 - 12 = 16 16 / 2 = 8 8 / 2 = 4 4 / 2 = 2 2 / 2 = 1 1 / 2 = 0. ❌

Input	Output
Q: **[Question]** A: Let's think step by step.	How much money does he have now? Henry has 11 + 18 - 10 = 27. ❌
[Demo1] Q: **[Question]** A: Let's think step by step.	He had 11 + 18 = 29 dollars. He spent 10 dollars on a game, so he has 29 - 10 = 19 dollars. ✅
[Demo2] Q: **[Question]** A: Let's think step by step.	He had 11 dollars. He had 11 dollars. He had 11 dollars. He had 11 dollars.... ❌
[Demo3] Q: **[Question]** A: Let's think step by step.	11 + 18 = 29 29 - 10 = 19 19 / 2 = 9 9 / 2 = 4 4 / 2 = 2 2 / 2 = 1 1 / 2 = 0 ❌
[Demo3] **[Demo1]** Q: **[Question]** A: Let's think step by step.	11 + 18 = 29. 29 - 10 = 19. ✅

Framework - Consistency-based Self-adaptive Prompting (COSP)



Stage 1:

1. Collects the pool of rationales and answers to question set via Zero-shot CoT.
2. Compute the *score* of each question, a metric inspired by *self-consistency*.
3. Identify suitable question-reasoning pairs with majority vote on the score.

Stage 2:

1. Augment the target question with a number of selected in-context demonstrations.
2. The augmented question is used to query the LLM a second time.
3. A majority vote over outputs from both stages forms the final prediction.

Demonstration Selection - How to calculate the score?

Building the Candidate Pool:

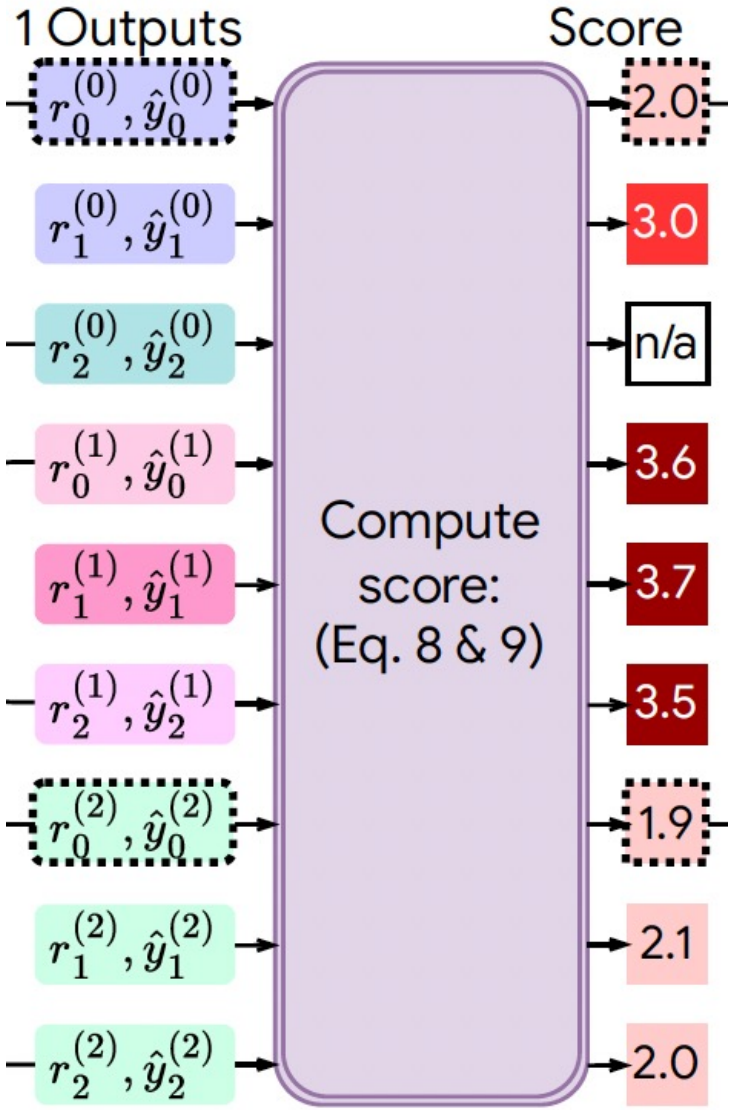
1. Run Zero-shot CoT over all questions.
2. Query the LLM m times with non-zero temperature for each question.
3. Extract reasoning paths $\left\{r_j^{(i)}\right\}_{j=1}^{m}$, and potential answers $\left\{y_j^{(i)}\right\}_{j=1}^{m}$.

Demonstration selection is the key objective of this paper:

1. In-context learning is sensitive to the choices of the demonstrations.
2. Select small K (typically ≤ 10) demos from a large set of candidates.
3. The candidate pool is imperfect (due to the absence of labels).

Criterias:

1. Consistency
2. Diversity
3. Repetition



Self-Consistency (Qualitative)

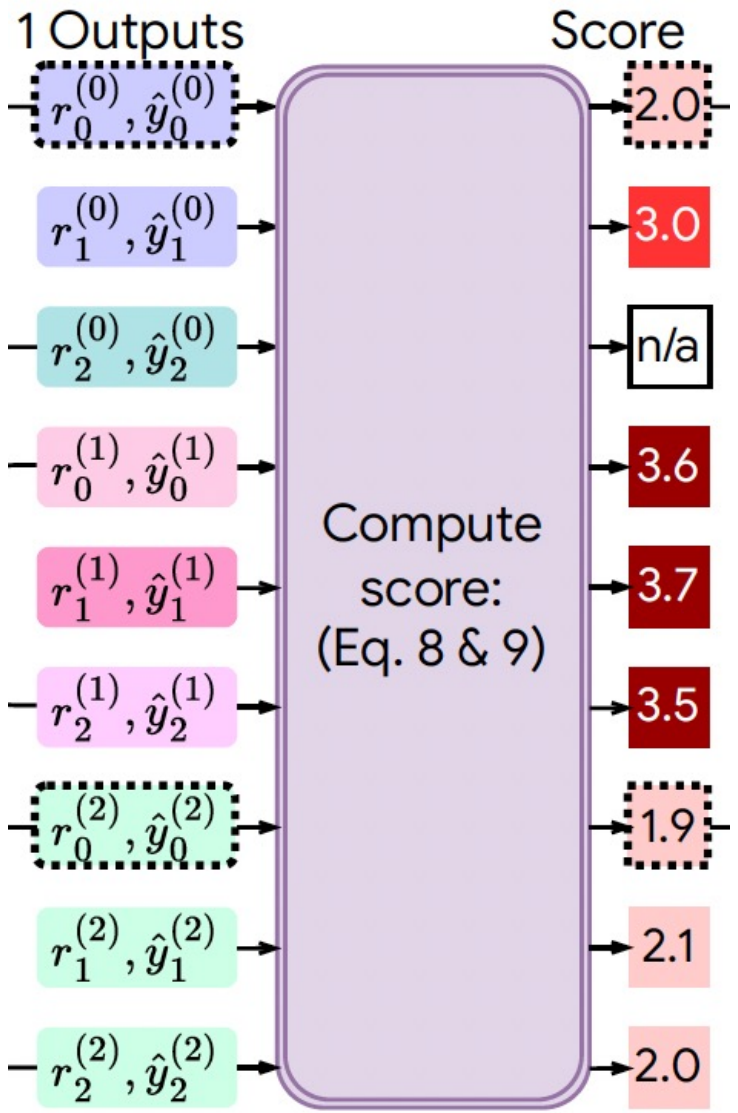
Reasons:
1. “Majority predictions are more likely to be correct”.
2. To prune the candidate pool.
3. To select the demonstrations in absence of ground-truths.

Measurement:
1. Compute majority vote prediction(s) from all predictions with:

$$\{(r_j^{(i)}, \hat{y}_j^{(i)})\}_{j=1}^{m} \sim \mathbb{P}(r^{(i)}, \hat{y}^{(i)} | x^{(i)}, c, \theta), \quad (1)$$

$$\hat{y}^{(i)} = \arg\max_{\hat{y}_j^{(i)}} \sum_{k=1}^{m} \mathbb{I}(\hat{y}_j^{(i)} = \hat{y}_k^{(i)}), \quad (2)$$

2. Retain only the rationales that lead to the majority vote prediction.
3. Use further heuristics to remove obviously bad candidates (e.g. responses containing no numbers for arithmetic tasks, or overly short and/or fragmented responses).



Self-Consistency (Quantitative)

Reasons (cont):

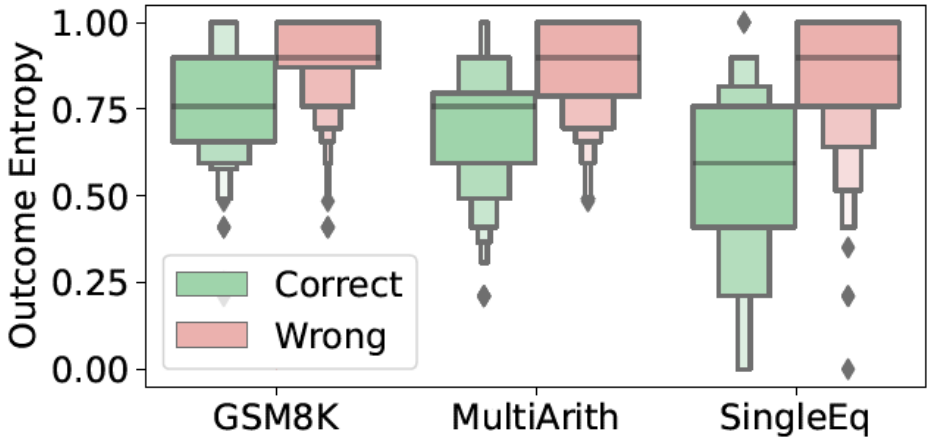
1. Self-consistency draws upon the insight that it approximates the amount of uncertainty (confidence) of the model for its prediction.

Measurement (cont):

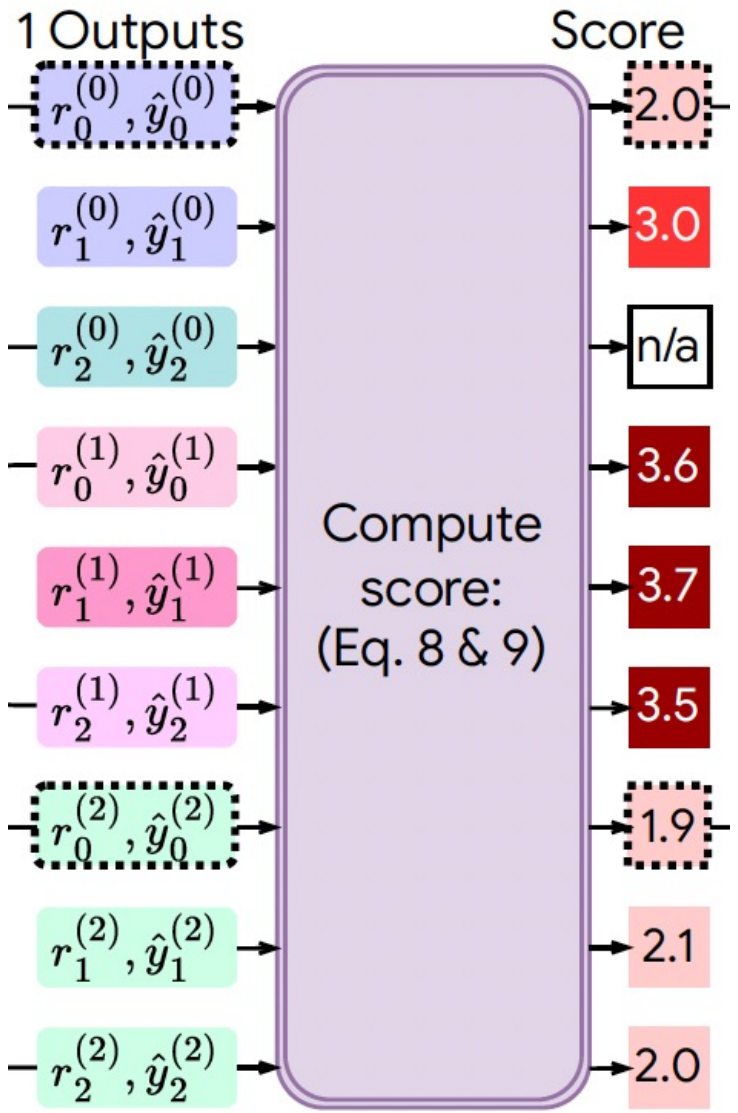
4. Compute the normalized entropy as:

$$\mathbb{H}(x^{(i)}|\{\hat{y}_j^{(i)}\}_{j=1}^m) = \frac{\sum_{\alpha=1}^{n} \tilde{\mathrm{p}}(\hat{y}_\alpha^{(i)}) \log \tilde{\mathrm{p}}(\hat{y}_\alpha^{(i)})}{\log m}, \tag{6}$$

where p is the empirical frequency of unique answer y_alpha.



The **normalized entropy** is a good proxy over a number of different tasks where low entropy is positively correlated with correctness.



Penalizing Repetitions

Reasons:

1. “Repetitive demonstrations often lead to worse performance”. (Strong but spurious pattern)
2. Should capture semantic-level repetitions.

Measurement:

1. Split demonstrations into phrases delimited by punctuation marks (“[.,?!]”).
2. Assuming with Q phrases, compute repetitiveness as:

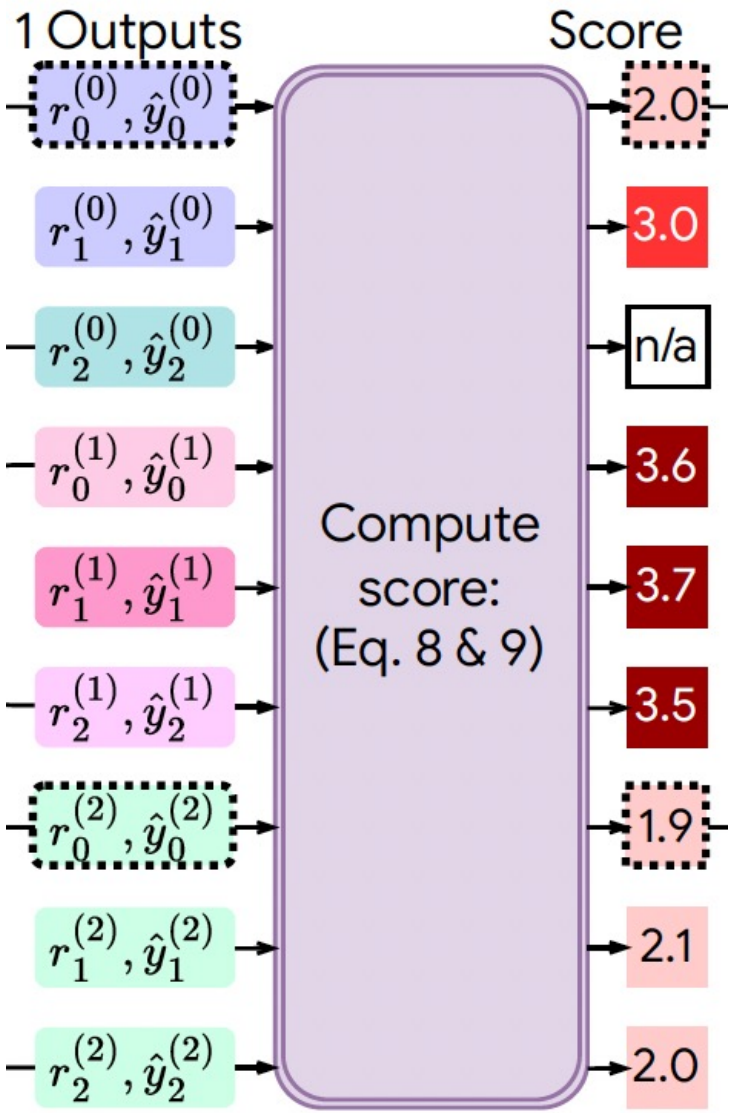
$$R_r(r_j^{(i)}) = \frac{2}{Q(Q-1)}\Big(\sum_{a=1}^{Q}\sum_{b=a+1}^{Q} W_{ab}\Big), \quad (7)$$

$$W_{ab} = S_c(\phi(q_a), \phi(q_b)),$$

where S_c(·, ·) computes the cosine similarity and φ(q_a) and φ(q_b) denote the vector embedding of a-th and b-th phrases.

For now, the score is:

$$\mathcal{F}(p|x^{(i)}, r^{(i)}, \{\hat{y}_j^{(i)}\}_{j=1}^{m}) = \mathbb{H}(x^{(i)}) + \lambda R_r(r_j^{(i)}), \quad (8)$$



Diversity

Reasons:

1. To select a single in-context demonstration (K = 1), we utilize the minimizer of the scoring function $p^* = \arg\min_{p \in \mathcal{P}} \mathcal{F}(p)$
2. To select multiple demonstrations, we should penalize demonstrations that are similar to previous ones.

Measurement:

1. Greedy forward selection with modified objective function:

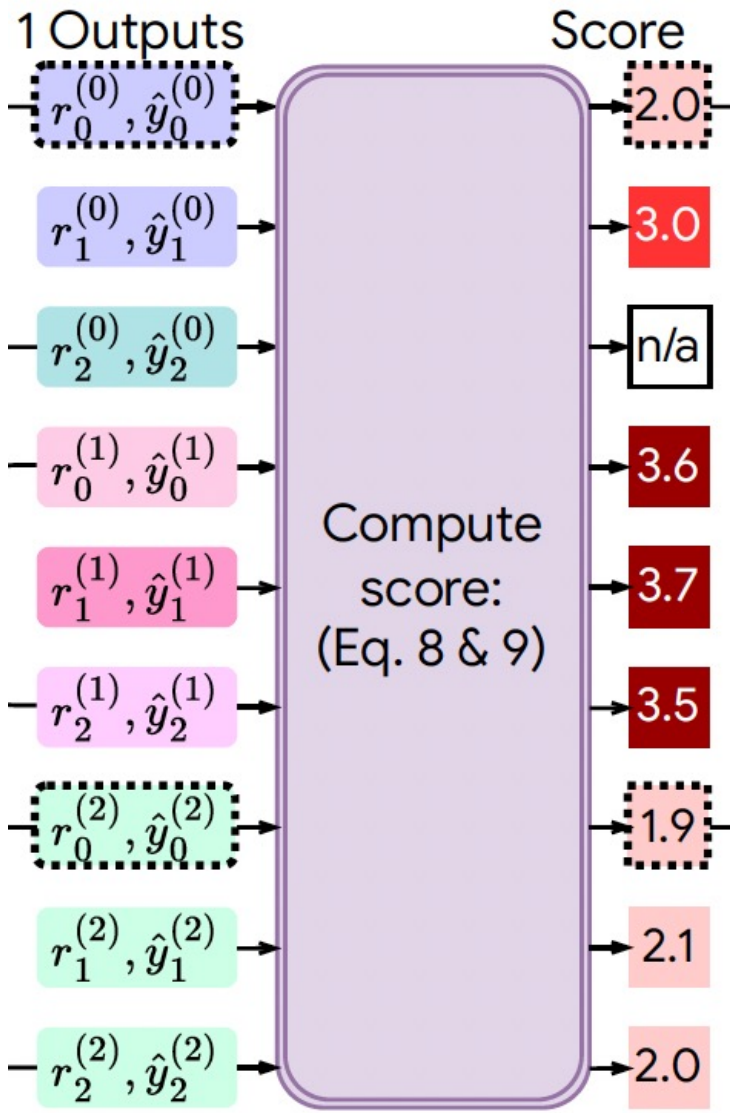
$$\mathcal{G}_k(p) = \mathcal{F}(p) + \lambda R_q(p, \mathcal{S}_{k-1}) \; \forall k \in [2, K], \quad (9)$$

$$R_q(p, \mathcal{S}_{k-1}) = \max\left(\{S_c(\phi(p), \phi(s_{k'}))\}_{k'=1}^{k-1}\right) \quad (10)$$

where S_{k−1} is the partially built demonstration set S with k − 1 elements already selected.

Algorithm 2 Building $\mathcal{S}$ for $K \geq 2$.

1: Initialize $\mathcal{S}$ with the minimizer of Eq. (8): $S \leftarrow \{p_0^* = \arg\min_{p \in \mathcal{P}} \mathcal{F}(p)\}$
2: **for** $k \in [2, K]$ **do**
3: Find the minimizer of the modified objective (Eq. (9)): $p_k^* = \arg\min_{p \in \mathcal{P}} \mathcal{G}_k(p)$.
4: Add p_k^* to $\mathcal{S}$ and remove p_k^* from candidate pool $\mathcal{P}$.
5: **end for**



Experiments

Model	PaLM-62B					PaLM-540B				
Setting	0-shot			5-shot	*Prev*	0-shot			5-shot	*Prev*
Method	0-shot CoT	Auto-CoT	COSP *(Ours)*	5-shot CoT	*8-shot CoT*	0-shot CoT	Auto-CoT	COSP *(Ours)*	5-shot CoT	*8-shot CoT*
# Paths	14	14	7+7	14		14	14	7+7	14	
MultiArith	67.2	9.4	**85.0**	**81.0**	-	95.2	**99.0**	**98.8**	96.0	*99.3*[b]
AddSub	69.1	**73.2**	**78.9**	72.4	-	88.9	**89.1**	**89.9**	86.6	*93.7*[b]
SingleEq	74.4	77.8	**78.7**	**79.8**	-	88.6	85.6	**90.4**	**89.2**	-
GSM-8K	20.9	9.2	**30.2**	**30.3**	*27.4*[a]	68.5	71.4	**71.9**	64.3	*74.4*[b]
CSQA	46.5	**68.2**	60.2	**66.8**	-	74.2	**79.4**	76.4	**80.7**	*80.7*[b]
StrategyQA	57.2	59.4	**64.7**	**67.9**	-	66.0	**75.7**	75.2	**81.4**	*81.6*[b]
(Average)	55.88	49.55	**66.28**	**66.37**	-	80.25	**83.37**	**83.77**	83.03	-

[a]Madaan and Yazdanbakhsh (2022). [b]Wang et al. (2022a): Significantly more (**40**) paths sampled.

Model	GPT-3 (`code-davinci-001`)				
Setting	0-shot			5-shot	*Prev*
Method	0-shot CoT	Auto-CoT	COSP (*Ours*)	5-shot CoT	*8-shot CoT*
# Paths	14	14	7+7	14	
MultiArith	50.3	**78.5**	**80.7**	60.7	*82.7*[b]
AddSub	43.5	31.9	**61.5**	**63.3**	*67.8*[b]
SingleEq	48.8	58.1	**64.8**	**65.9**	-
GSM-8K	**10.2**	6.6	8.7	**16.7**	*23.4*[b]
CSQA	29.2	50.9	**55.4**	**53.0**	*54.9*[b]
StrategyQA	47.8	**55.8**	52.8	**55.4**	*61.7*[b]
(Average)	38.32	46.96	**53.98**	**52.60**	-

[b]Wang et al. (2022a): Significantly more (**40**) paths sampled.

Notes:

1. Unpretentious experimental design.
2. The outcome entropy is also a natural gauge of difficulty of questions to the LLM, as a higher entropy (thus a higher uncertainty) implies that the LLM may require additional demonstrations for this question.
3. Can further feature an adaptively allocated number of in-context demonstrations that is proportional to its zero-shot entropy in Stage 1, with higher-entropy questions given more demonstrations.

Thank You!