

Ensembling Elicitation Strategies for Knowledge Base Construction

Vlad-Mădălin Ungureanu¹ Matei Neacsu^{2*} Ianis Belu^{2*}

¹National Institute for Research and Development in Informatics – ICI Bucharest

²International Computer High School of Bucharest (ICHB)

madalin.ungureanu@ici.ro

neacsu.matei@ichb.ro

belu.ianis@ichb.ro

Abstract

Large language models represent a promising alternative for automating structured knowledge bases construction, but their reliance on parametric knowledge makes them susceptible to hallucinations and incomplete information. The Automated Knowledge Base Construction competition investigates the use of large language models to construct structured knowledge bases from their parametric knowledge alone, without fine-tuning or external retrieval. Given a subject-relation pair, participants must predict the complete set of correct objects across several relation types. Our approach proposes an ensemble of four prompting strategies: question-style, masked-style, few-shot chain-of-thought, and least-to-most prompting. It aggregates their predictions using relation-aware majority voting. The resulting system achieves a Macro-F1 of 0.6651 on the test set, substantially outperforming the provided baseline of 0.294. The system implementation can be found at the GitHub repository referenced below.¹

1 Introduction

Structured knowledge bases store verified information, enabling LLMs to ground their outputs in factual data and thereby reduce hallucinations and improve reliability (Yang et al., 2025). Alternatively, some research explores constructing knowledge bases entirely from LLMs, though such systems still face challenges regarding precision (Hu et al., 2024).

Building on this line of work, the Automated Knowledge Base Construction shared task focuses specifically on leveraging Large Language Models to construct structured knowledge bases. The challenge lies in the requirement that predictions must be generated using the model’s parametric

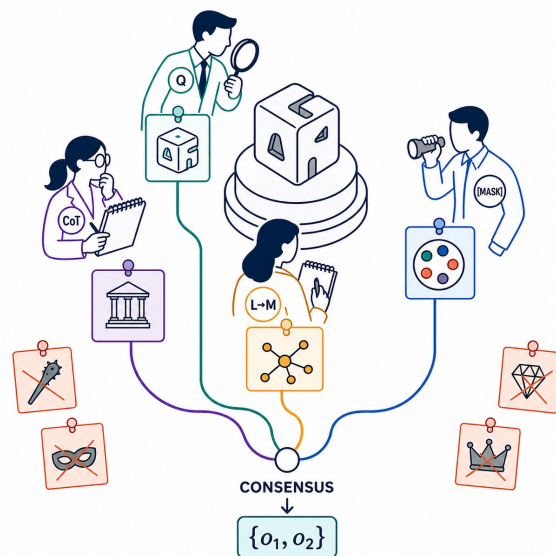


Figure 1: Multiple elicitation strategies examine the same fact from different perspectives. Their observations are combined through relation-aware aggregation to produce the final knowledge-base prediction.

memory alone, without fine-tuning or Retrieval-Augmented Generation (RAG) over external corpora. Additionally, although multiple LLMs may be combined within a single answer prediction system, the summed parameter count of the combined models must not exceed 32B. Given a subject s and a relation r , the task is to predict the complete set of correct object strings $\{o_1, o_2, \dots, o_k\}$, where there may be zero, one or multiple correct answers.

The dataset includes six distinct relation types, each capturing specific entity attributes or associations with varying cardinalities. First, `awardWonBy` identifies the recipients of a specified award, a relation that often exhibits high cardinality because many awards are conferred upon dozens or hundreds of entities over time. In geographic contexts, `countryLandBordersCountry` links a country to the nations sharing its land borders (yielding

* Work conducted during an internship at ICI Bucharest.

¹<https://github.com/madalinmdv/akbc-2026>

an empty set for borderless countries or multiple values otherwise), while `hasArea` specifies a geographic entity’s surface area in square kilometres. Commercial and venue-specific data are captured by `companyTradesAtStockExchange`, which maps a company to the stock exchange(s) where its shares are publicly traded (with unlisted subsidiaries yielding empty sets), and `hasCapacity`, which denotes the maximum spectator capacity of a venue. Finally, `personHasCityOfDeath` records the city where an individual passed away, remaining empty if the person is still living.

The proposed system employs multiple prompting strategies and aggregates the resulting predictions according to the relation type in order to generate a final output. Specifically, multiple prediction sets produced by the same model under different prompting strategies are combined to distinguish stable parametric knowledge from prompt-induced hallucinations. The chosen model for this task is Meta’s 30B parameter Muse Glimmer.

This study aims to determine the extent to which different prompting strategies influence the quality and precision of model outputs, and to assess whether aggregating multiple responses generated by the same language model can help reduce hallucinations.

2 Related work

Considering the competition constraints in regards to fine-tuning the model or designing RAG implementations using external corpora, the workshop motivates a greater reliance on how effectively parametric knowledge can be elicited from the model. Under these circumstances, prompt-based strategies become one of the main pillars in the attempt to improve the reliability of the generated knowledge base. Prompting methods encompass a wide range of techniques, from foundational approaches such as few-shot and chain-of-thought prompting to more advanced methods (Sahoo et al., 2024).

Few-shot prompting refers to providing a language model with a small number of task-specific demonstrations at inference time, without updating the model’s parameters (Brown et al., 2020). Demonstrations consist of paired inputs and their corresponding desired outputs, allowing the model to infer the task format. The study by Min et al., “Rethinking the Role of Demonstrations,” shows that demonstrations derive their value not from

teaching the model new input-output mappings, but from exposing it to the label space, the distribution of the input text, and the overall format of the sequence (Min et al., 2022).

Chain of thought prompting consists of including a series of intermediate reasoning steps in the input. This method has been proven to significantly improve the ability of large language models to perform complex reasoning (Wei et al., 2022). Automatic Chain-of-Thought prompting utilises LLMs to automatically generate reasoning chains for selected examples, with diversity in the sampled questions helping mitigate errors in the generated chains (Zhang et al., 2022). Participants in last year’s knowledge base construction workshop, Edouard Albert-Roulhac and Amal Zouaq, utilized a large model to compose a comprehensive set of chain-of-thought style prompts (Albert-Roulhac and Zouaq, 2025). These examples were then used as few-shot demonstrations to produce multiple reasoning paths and answers during inference.

Self-refinement represents a method of improving LLM outputs through self-generated feedback. Madaan et al. showcased this method, drawing an analogy to how humans reflect on their own writing (Madaan et al., 2023). In their study, after an LLM produces an initial output, the same model evaluates it and, based on that feedback, refines its response. This process is repeated iteratively until the model’s own feedback indicates that no further improvement is needed, or a fixed number of iterations is reached. This workflow assigns the LLM with multiple roles, acting as both generator and critic.

Least-to-most prompting solves complex problems by decomposing them into simpler subproblems (Zhou et al., 2022). Unlike standard chain-of-thought prompting, which reasons in a single pass, this approach decouples planning from execution via a two-stage process: first, it decomposes the initial query into subquestions, and second, it sequentially solves each subquestion using the accumulated history of previous answers.

3 System

The system ensembles the elicitation techniques described below using a 30B parameter model.

The results are then aggregated by majority vote, except for the `awardWonBy` relation, where the large number of correct objects per subject makes voting across strategies too restrictive.

Technique	Description
Question style prompt	Zero-shot, natural question.
Masked style prompt	Zero-shot; the same fact phrased as a sentence with a [MASK] token.
Few-Shot CoT	Few-shot chain-of-thought; hand-written worked examples per relation, provided in a single turn.
Least-to-most prompting	A chain of per-relation subquestions, each posed as its own conversational turn, with answers accumulating before the final query.

Table 1: Prompting techniques used in the ensemble, with a description of each.

3.1 Question and masked style prompting

We adopt question-style and masked-style prompting using templates provided by the task organizers.² These are used as a baseline against which the gains from more elaborate prompting strategies can be assessed.

Relation	Prompt
hasArea	What is the area of {subject.entity} in square kilometers?
hasCapacity	What is the total capacity of {subject.entity}?
countryLandBordersCountry	Which countries share a land border with {subject.entity}?
personHasCityOfDeath	In which city did {subject.entity} die?
awardWonBy	Who won the {subject.entity}?
companyTradesAtStockExchange	Where do shares of {subject.entity} trade?

Table 2: Zero-shot question prompt templates used for each relation.

Relation	Prompt
hasArea	The area of {subject.entity} in square kilometers is {mask.token}.
hasCapacity	The total capacity of {subject.entity} is {mask.token}.
countryLandBordersCountry	{subject.entity} shares a land border with {mask.token}.
personHasCityOfDeath	{subject.entity} died in the city of {mask.token}.
awardWonBy	{subject.entity} was won by {mask.token}.
companyTradesAtStockExchange	{subject.entity} shares can be traded at {mask.token}.

Table 3: Masked-style prompt templates.

3.2 Few-shot chain-of-thought prompting

The system presented in this article combines few-shot demonstrations with human-authored chain-of-thought reasoning traces derived from examples in the training data.

Unlike zero-shot strategies, which elicit an answer directly, this approach provides the model with manually constructed worked examples, in

which each answer is preceded by an explicit reasoning trace, as illustrated in Figure 2.

```

Question: On which stock exchanges does Bangladesh
↪ Cement
Manufacturers Association trade?
Step 1 - The BCMA is a trade association
↪ representing cement producers in
Bangladesh, not a commercial corporation or
↪ operating company.
Step 2 - As a membership-based industry body, it
↪ does not issue equity shares
to the public and is therefore not publicly listed
↪ on any exchange.
Step 3 - Since it is not a publicly traded entity,
↪ it has no listings.
Answer: []

[...]

```

Figure 2: Chain-of-thought demonstration for the companyTradesAtStockExchange relation.

3.3 Least-to-most prompting

Least-to-most prompting decomposes each query into a sequence of simpler subquestions that are answered before the original question is posed. In contrast to the chain-of-thought strategy of Section 3.2, where the reasoning is produced in a single turn, each subquestion is here issued as its own conversational turn, and every subanswer is appended to the dialogue history before the next subquestion is asked.

The subquestion chains mirror the reasoning steps of the chain-of-thought schema. The chain for personHasCityOfDeath, shown in Figure 3, is representative, and is constructed so that the confusion between the city a person is famous for and the city in which they died is raised explicitly before the final answer is requested.

Each chain is preceded by worked dialogues demonstrating the expected form of the exchange. One such dialogue is shown, truncated, in Figure 4.

```

SUBQUESTIONS_DEATH = [
    "Who is or was {subject}, and has {subject} died
    ↪ ?",
    "Which city is {subject} most famous for, or most
    ↪ associated with through living or working
    ↪ there?",
    "If {subject} has died, in which city did the
    ↪ death actually take place, and is that the
    ↪ same city as the one just named or a different
    ↪ one?",
]

```

Figure 3: The least-to-most subquestion chain for the personHasCityOfDeath relation.

²<https://github.com/lm-kbc/dataset2026>

```

Problem: In which city did Sigmund Freud die?
Subquestion 1: Who is or was Sigmund Freud, and has
↳ Sigmund Freud died?
Subanswer 1: Sigmund Freud was the founder of
↳ psychoanalysis; he died in 1939
and is deceased.
Subquestion 2: Which city is Sigmund Freud most
↳ famous for [...]?
Subanswer 2: He is overwhelmingly associated with
↳ Vienna, where he lived and
worked for most of his life.
Subquestion 3: If Sigmund Freud has died, in which
↳ city did the death actually
take place, and is that the same city as the one
↳ just named or a different one?
Subanswer 3: He fled Vienna in 1938 after the Nazi
↳ annexation of Austria and
died the following year in London, so the death city
↳ differs from the famous
city.
Final question: In which city did Sigmund Freud die?
Answer: ["London"]

[...]

```

Figure 4: A worked least-to-most demonstration for the personHasCityOfDeath relation.

3.4 Self-refinement

Observing the results obtained for the awardWonBy relation under the other prompting strategies, it was concluded that a different approach is needed for this particular relation. In this version of the self-refinement approach, the model first iteratively expands its candidate list of award recipients across follow-up rounds to maximise recall, then self-reviews and prunes that list to remove low-confidence names, trading a modest drop in recall for a much larger gain in precision.

The first query asks the model to generate all the recipients it knows of for a given award. To improve recall, the generated answers are then given back to the model in the follow-up query shown in Figure 5, asking it to complete the list with any additional winners.

```

FOLLOW_UP_QUERY = (
  "Recipients of {award} identified so far:\n{
  ↳ listed}\n\n"
  "Who ELSE has received {award} that is NOT in the
  ↳ list above? "
  "Only add names you are genuinely confident about
  ↳ , and do not repeat any "
  "name already listed. If you cannot think of any
  ↳ more, reply with exactly: "
  "Answer: none"
)

```

Figure 5: Follow-up query used to expand the candidate list for the awardWonBy relation.

This process iterates for up to six additional rounds, continuing as long as the model generates

more than two new names in the previous round. To improve precision, a subsequent pruning step was introduced, in which the model uses the system prompt shown in Figure 6 to review its own previously generated predictions and remove any it is confident are incorrect.

```

SYSTEM_PROMPT_PRUNE = (
  "You are a strict fact-checker for award
  ↳ recipients. You are shown a "
  "candidate list and you name only the ones you
  ↳ are EXTREMELY confident "
  "never received the award -- not the ones you
  ↳ merely cannot confirm or "
  "are unsure about, which you leave alone. Provide
  ↳ only the requested "
  "information without explanations, uncertainty
  ↳ statements, or additional "
  "context. "
)

```

Figure 6: System prompt used to prune low-confidence candidates for the awardWonBy relation.

3.5 Post-processing: answer extraction

A single answer-extraction module is applied uniformly across all elicitation methods, ensuring that score differences arise solely from the chosen prompting strategies rather than from differences in post-processing. Each strategy’s final query includes explicit formatting instructions, determined by the relation type:

1. Answer: <number> for numeric relations, such as hasArea or hasCapacity;
2. Answer: [...] — a JSON array — for list relations;
3. Answer: followed by one name per line, specifically for the awardWonBy relation.

The parsing stage reads only the text following the final Answer: marker. For numeric relations, it first checks for a comma-grouped number and falls back to a plain number only if none is found, so that a value such as 51,286 is parsed in full rather than truncated at the first comma. For list relations, the segment is parsed as a JSON array, falling back to comma-splitting when the model’s array is malformed. Replies indicating no objects — “none,” “n/a,” and similar — are mapped to an empty set, which for many subject–relation pairs is the correct answer rather than a failure to respond.

3.6 Answer aggregation

Result aggregation is performed through majority voting, subject to the following relation-specific rules:

1. **Numeric relations:** Responses within a 5% tolerance of one another are grouped into clusters, and the mean value of the largest cluster containing at least two items is selected as the final prediction. When responses are too scattered for a cluster to form, a fallback mechanism selects the prediction from one of the four prompting strategies.
2. **List relations:** A quorum of two is required for a value to be included in the final prediction. If no consensus is reached, the predicted set is left empty. This is a deliberate design choice, since an empty set is itself a valid answer for a considerable share of the dataset: approximately 20% of the validation triplets have an empty set as the correct answer.
3. **awardWonBy:** This relation is excluded from the ensemble entirely, as aggregating results across strategies would significantly reduce recall.

4 Results

The evaluations presented in this section exclude the `awardWonBy` relation. This is because the prompting strategies described for the rest of the system did not yield satisfactory results for this relation, and including it in the aggregated metrics led to a considerable drop in recall. The evaluation of `awardWonBy` is therefore presented separately.

Table 4 compares the two zero-shot strategies. Overall, question-style prompting yields a marginally higher Macro-F1 (0.667) than masked-style prompting (0.658). Therefore, in a zero-shot setting, eliciting facts through direct questions is slightly more effective.

Relation	P	R	F1	Empty
<i>Question-style prompting</i>				
<code>companyTradesAtStockExchange</code>	0.950	0.758	0.759	54
<code>countryLandBordersCountry</code>	0.998	0.987	0.991	18
<code>hasArea</code>	0.820	0.820	0.820	0
<code>hasCapacity</code>	0.289	0.289	0.289	0
<code>personHasCityOfDeath</code>	0.880	0.610	0.570	66
All	–	–	0.667	138
<i>Masked-style prompting</i>				
<code>companyTradesAtStockExchange</code>	0.910	0.732	0.712	53
<code>countryLandBordersCountry</code>	0.996	0.993	0.994	18
<code>hasArea</code>	0.800	0.800	0.800	0
<code>hasCapacity</code>	0.361	0.361	0.361	0
<code>personHasCityOfDeath</code>	0.840	0.590	0.520	64
All	–	–	0.658	135

Table 4: Evaluation results for the two zero-shot strategies on the validation set, over the five relations handled by the ensemble. `awardWonBy` is excluded, as it is generated by the dedicated strategy of Section 3.4.

Few-shot chain-of-thought prompting achieves an F1 score of 0.680, outperforming both zero-shot strategies. The improvement is most pronounced for `companyTradesAtStockExchange`, whose F1 score rises from 0.759 with question-style prompting to 0.847 with few-shot chain-of-thought prompting. This suggests that the reasoning traces provided in the demonstrations are particularly effective at guiding the model when the extraction task requires disambiguating between multiple candidate entities.

However, this improvement is not uniform across all relations. `hasCapacity` performs worse under chain-of-thought prompting than under either zero-shot strategy.

Relation	P	R	F1	Empty
<code>companyTradesAtStockExchange</code>	0.937	0.852	0.847	45
<code>countryLandBordersCountry</code>	1.000	0.992	0.995	18
<code>hasArea</code>	0.810	0.810	0.810	0
<code>hasCapacity</code>	0.278	0.278	0.278	0
<code>personHasCityOfDeath</code>	0.880	0.600	0.560	67
All	–	–	0.680	130

Table 5: Evaluation results for few-shot chain-of-thought prompting on the validation set.

Least-to-most prompting is the weakest of the four strategies, with a Macro-F1 of 0.647. The comparison with few-shot chain-of-thought prompting is controlled, in that the two strategies employ similar reasoning decomposition and differ only in whether it is delivered in a single turn or across several. Least-to-most is inferior on every one of the five relations; for example, it reaches 0.812 against 0.847 on `companyTradesAtStockExchange`.

Relation	P	R	F1	Empty
<code>companyTradesAtStockExchange</code>	0.913	0.835	0.812	45
<code>countryLandBordersCountry</code>	1.000	0.988	0.993	18
<code>hasArea</code>	0.800	0.800	0.800	0
<code>hasCapacity</code>	0.237	0.237	0.237	0
<code>personHasCityOfDeath</code>	0.720	0.600	0.490	51
All	–	–	0.647	114

Table 6: Evaluation results for least-to-most prompting.

The aggregated results achieve a Macro-F1 of 0.689, marginally outperforming the best individual strategy observed previously: 0.680 with few-shot chain-of-thought prompting. This suggests that the overall gain in Macro-F1 stems from more balanced performance across relations rather than from consistently matching the best individual result for each relation.

Relation	P	R	F1	Empty
companyTradesAtStockExchange	0.950	0.838	0.839	46
countryLandBordersCountry	1.000	0.992	0.995	18
hasArea	0.810	0.810	0.810	0
hasCapacity	0.330	0.330	0.330	0
personHasCityOfDeath	0.850	0.620	0.560	62
All relations	0.776	0.701	0.689	126

Table 7: Validation results after relation-aware aggregation.

Turning to the `awardWonBy` relation, the self-refinement approach achieves a Macro-F1 of 0.209 on the validation set, more than double the 0.094 F1, which was the score obtained by the best-performing of the four other prompting strategies described earlier.

Relation	P	R	F1	Empty
awardWonBy	0.329	0.196	0.209	–

Table 8: Validation results for the separately evaluated `awardWonBy` relation.

On the hidden test set, the proposed implementation obtained a Macro-F1 score of 0.6651, as shown in Table 9. Among the evaluated relations, `hasCapacity` achieved the lowest F1 score; this limitation may arise because the relevant information was not included in the model’s training data. In contrast, the highest score was obtained for `countryLandBordersCountry`.

Relation	P	R	F1
awardWonBy	0.5179	0.2850	0.3357
companyTradesAtStockExchange	0.9433	0.8327	0.8105
countryLandBordersCountry	0.9861	0.9702	0.9680
hasArea	0.8200	0.8200	0.8200
hasCapacity	0.2857	0.2857	0.2857
personHasCityOfDeath	0.9150	0.6000	0.5667
All relations	0.7728	0.6760	0.6651

Table 9: Per-relation results on the hidden test set.

4.1 Observations

The ensemble strategy was also applied to prediction sets generated by a combination of smaller models: Mistral-Nemo:12b, Llama3.1:8b, and Gemma2:9B. Although the single-model system presented in this paper achieved a higher overall Macro-F1, aggregating predictions across these smaller, heterogeneous models yielded a proportionally larger improvement in Macro-F1 than aggregating predictions from a single model alone.

This leads to a second observation: given the constraints against adjusting model parameters or drawing on external information, the correctness

of the generated knowledge base is fundamentally bounded by whether the sought-after fact was present in the large language model’s training data.

Additionally, it should be noted that the competition baseline was produced using Qwen3.5-9B while our results were obtained with the considerably larger Muse Glimmer 30B, which limits the extent to which the results can be interpreted as a direct comparison of the proposed approach against the baseline.

5 Conclusions

This paper investigated how different elicitation strategies affect parametric-only knowledge-base predictions and whether aggregating single-model responses reduces hallucinations. Across the four evaluated strategies, few-shot chain-of-thought achieved the highest overall Macro-F1, narrowly outperforming two zero-shot approaches, whereas multi-turn least-to-most prompting consistently scored lowest. Finally, the high-cardinality `awardWonBy` relation proved unsuited to majority voting, requiring a dedicated expansion-and-verification procedure instead.

Acknowledgements

The authors acknowledge the support of project PN 23 38 01 01, “Contributions to the consolidation of emerging technologies specific to the Internet of Things and complex systems,” which provided technical resources essential to this research.

References

- Wenli Yang, Lilian Some, Michael Bain, and Byeong Kang. 2025. A Comprehensive Survey on Integrating Large Language Models with Knowledge-Based Methods. *arXiv preprint arXiv:2501.13947*.
- Yujia Hu, Tuan-Phong Nguyen, Shrestha Ghosh, and Simon Razniewski. 2024. Enabling LLM Knowledge Analysis via Extensive Materialization. *arXiv preprint arXiv:2411.04920*.
- Pranab Sahoo, Ayush Kumar Singh, Sriparna Saha, Vinija Jain, Samrat Mondal, and Aman Chadha. 2024. A Systematic Survey of Prompt Engineering in Large Language Models: Techniques and Applications. *arXiv preprint arXiv:2402.07927*.
- Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu,

- Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. *arXiv preprint arXiv:2005.14165*.
- Sewon Min, Xinxu Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Rethinking the Role of Demonstrations: What Makes In-Context Learning Work? *arXiv preprint arXiv:2202.12837*.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. *arXiv preprint arXiv:2201.11903*.
- Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. 2022. Automatic Chain of Thought Prompting in Large Language Models. *arXiv preprint arXiv:2210.03493*.
- Edouard Albert-Roulhac and Amal Zouaq. 2025. ReWiSe: Relation-Wise Self-consistency for LLM Probing. In *Proceedings of the KBC-LM Workshop and the LM-KBC Challenge 2025*, CEUR Workshop Proceedings, Vol-4041.
- Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegrefe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. 2023. Self-Refine: Iterative Refinement with Self-Feedback. *arXiv preprint arXiv:2303.17651*.
- Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, and Ed Chi. 2022. Least-to-Most Prompting Enables Complex Reasoning in Large Language Models. *arXiv preprint arXiv:2205.10625*.