Dissecting Inaccuracies in Large Language Models: An Analysis on Reasoning-Error Causes on Large Language Models

Anonymous ACL submission

Abstract

While large language models (LLMs) have rapidly improved performance on a broad number of tasks, they still lag behind in abstract reasoning tasks. Wang et al. (2023) proposed *self-consistency*, finding that sampling multiple rationales before taking a majority vote stably improves performance in both mathematical and commonsense reasoning. This work augments self-consistency idea with a variety of clustering and mapping approaches to balance between diversity and accuracy, and additionally explore and evaluate sources of inaccuracies in reasoning performance more efficiently and concisely. We introduce two novel techniques: identifying consensus responses by clustering semantic embeddings of model outputs, and systematically varying temperature schedules during the course of generation. By doing so, we aim to capture a more comprehensive spectrum of reasoning paths employed by the model and increase confidence in coherent answers providing guidance about models wrong doings while improving accuracy on common benchmarks.

1 Introduction

In recent years, the development of large language models has witnessed remarkable strides, with significant advancements in their accuracy and expressive capabilities (Naveed et al., 2023). Despite these achievements, the computational demands associated with deploying such models, particularly during inference, pose challenges that necessitate innovative solutions (Sarker, 2021). This paper delves into the exploration of methodologies aimed at enhancing the accuracy of large language models while concurrently mitigating the computational resources required during the inference phase.

A fundamental aspect of our investigation builds upon the foundations laid by self-consistency training, a technique that leverages sampling to generate responses and subsequently combines them to refine model predictions and other augmentation methods (Mialon et al., 2023). In this pursuit, we introduce two techniques designed to augment the efficacy of self-consistency training and other new found reasoning techniques. First, we propose the application of semantic vector representations to cluster model outputs, facilitating the identification of alike responses to estimate an accurate representation about output sequences.

Second, we advocate for the systematic variation of temperature schedules throughout the training process and during the aggregation of responses (Holtzman et al., 2020). This dynamic modulation of temperature sampling not only introduces adaptability into the model's responses but also improves the decision process conducted by our verification method by potentially covering a more broad range of responses that therefore improve out marginalization techniques.

Our validation, conducted through comprehensive comparative analyses on benchmark datasets, substantiates the efficacy of the proposed techniques. In particular, our results on variation of temperatures reveal an improvement in accuracy when compared to baseline self-consistency training, all while utilizing an equivalent number of sampled sequences.

The innovations introduced in this research, namely, model-agnostic clustering and dynamic temperature sampling, present promising avenues for the advancement of large language models and their pretraining.

As computational efficiency becomes an increasingly vital consideration in the practical deployment of language models, the contributions presented herein provide valuable insights and strategies for the development of future-generation models and techniques that lead pathways to lower increasing performance while keeping a steady parameter count.

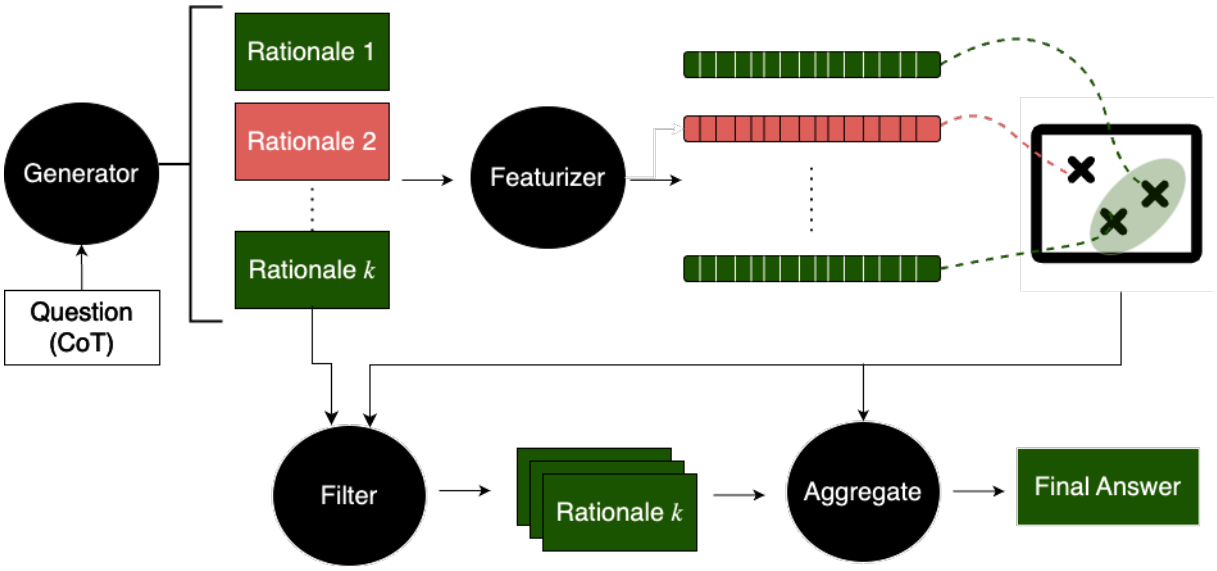

Figure 1: Default self-consistency comprises three steps: (1) Prompt a model with chain-of-thought reasoning; (2) Generate n sampled sequences, and (3) Marginalize results based on the most occurring numerical output.
Our proposed method samples results and marginalizes not only based on consistency in the output but also on the coherency of the employed reasoning path. Our assumption is that Language Models often apply the correct reasoning path but lack the ability to conduct mathematical operations correctly. Therefore, wrong numerical results cannot imply that a reasoning path is wrong. We utilize this concept to let correct reasoning paths improve the confidence in similar reasoning responses, assuming that the model's overall mathematical capabilities are high enough to incorporate correct arithmetic results in a majority of subsequent sequences.

2 Methodology

We utilize the premise that exposing the model to a spectrum of temperatures facilitates the model of more abstract decision-making processes. By using temperature, as a controlling parameter, introduces an element of stochasticity into the generation process, where the model combines higher temperature outputs to encourage a more explorative approach, leading to diverse and potentially more abstract responses and conversely, lower temperatures to emphasize more deterministic and focused outputs.
This Process can be harnessed to improve the introduced clustering mechanism and improve the out-turn based on the filtering mechanisms by providing more clear differentiation between the employed reasoning paths.

The underlying process can be described in a simple set of step by step instructions.

1. ***Generate n base responses:*** Given a query of few-shot examples, we aim to harness the model's capabilities of abstract thinking on different temperatures to generate diverse outputs.
2. ***Determine a filtering mechanism or clustering method:*** We introduce a variety of mechanisms to filter the responses based on the final numerical result, reasoning path and similarity to subsequent responses.
3. ***Marginalize the results based on the filtering system*** We marginalize and/or aggregate the results using one of the above-mentioned methods, to conclude to one final answer.

Our experiments are conducted on different sets of configurations. For top-k and top-p we use a default of 50; more detailed information on the configurations used can be found in Appendix A.

3 Experimental Setup

We conduct multiple experiments with varying setups in form of different benchmarks tested on each model to cover a broad range of possible outputs.

3.1 T-SNE configuration

We employ the t-Distributed Stochastic Neighbor Embedding (t-SNE) technique for the visualization of high-dimensional vector spaces. (van der Maaten and Hinton, 2008)

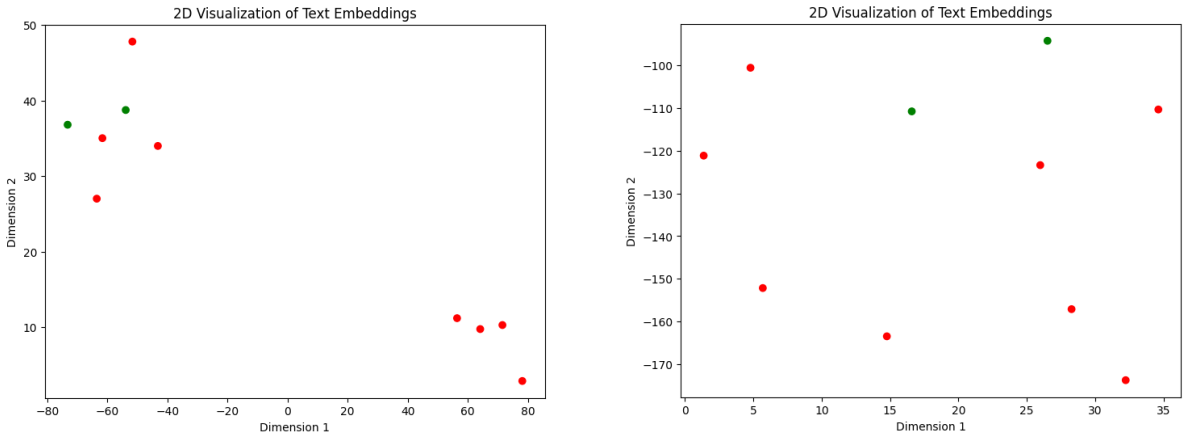

Figure 2:
We select a perplexity parameter of 2, grounded in the rationale that local distributions yield a more informative representation than global distributions.
This is attributed to the increased density of points in close proximity, enhancing the detail captured in the mapping. Based on a test on a subset of arithmetic reasoning examples, evaluated with Microsoft Phi1.5 on 10, 15 and 20 outputs based on baseline self-consistency with the in the Appendix provided n-Shot prompts.

3.2 Weighting

We propose several methods for sampling from different temperatures. Current work includes a majority vote system, which will give us a benchmark to compare against. Additionally, we employ a weighting system based on the inverse of the applied temperature:

$$\sum_{i=1}^{n} \frac{1}{t_i} \tag{1}$$

Furthermore, we conducted tests using weighted squared inverse weighting on a small subset. However, these tests did not yield substantial results due to the robustness of sampling towards numerical solutions:

$$\sum_{i=1}^{n} \left(\frac{1}{t_i}\right)^2 \tag{2}$$

3.3 Datasets

We use AQuA-rat and SVAMP to evaluate on more complex arithmetic tasks. (Ling et al., 2017; Patel et al., 2021) Additionally, we used GSM8K at different parts for cross section and different ablations to evaluate on datasets of lower-end difficulty.

3.4 Language Models

Our models are split up into *generators*, which provide the reasoning/result sequences of of which we build the solutions and *featurizers*, which convert the output sequences into a suitable vector representation.

3.4.1 Generators

- **Microsoft Phi 1.5:** The Phi1.5 model introduced by Gunasekar et al. (2023) has a smaller architecture with 1.3 billion Parameters, that was made to create a non-restricted small model to explore vital challenges and generate "basic/starting point" responses and outputs for text and code.

- **Microsoft Phi 2:** The Microsoft Phi2 model is an highly optimized 2.7 billion-parameter language model. That outperforms models up to 25 times larger showing promising results on common benchmarks.

- **GPT-3:** For our evaluation we use code-davinci-002 a descendant of the GPT-3 architecture which is a dense higher parameter large-scale language model with 175 billion parameters.(Brown et al., 2020)

- **Llama 2:** Llama 2 is a collection of transformer models presented by Meta, which are trained on large amounts of publicly available texts. With a focus on Llama 7B for our evaluation, Llama 2 performs well on

common benchmarks for its size and can be fine-tuned for specific areas. (Touvron et al., 2023)

- **Mistral 7B:** Mistral 7B is a strong front to back transformer Model developed for performance and efficiency, and renowned for its scalability and adaptability between different areas. It outperforms larger-parameter Models in processing large contextual information and can be fine-tuned[1] for specific tasks. (Jiang et al., 2023)

3.4.2 Featurizers

- **roBERTa:** roBERTa is an "robustly" fine-tuned model derived from the original BERT architecture introduced by Devlin et al. (2019). It is featuring enhancements that enabled roBERTa to outperform its predecessor in several natural language processing benchmarks. (Liu et al., 2019)

- **sciBERT:** sciBERT is a BERT-model fine-tuned on scientific language, comprising a multi-domain corpus of roughly 1.14M scientific publications. Making it particularly adept at understanding more complex terminology and structure in academic contexts. (Beltagy et al., 2019)

- **MathBERT** MathBERT is a 100M token BERT-model that is fine-tuned on mathematical language based on up to an college level Math curriculum, books and Math arXiv-paper-abstracts.(Shen et al., 2023)

4 Results

4.1 Finetuned featurizers

We tested the "featurization" process on multiple featurizer-models on differing levels of applicability. Due to RoBERTa general robust training it limitits it ability to distinguish and evaluate unique features in mathematical reasoning paths. Conversely MathBERT prioritizes accuracy in mathematical operations and results making it a valid method for grouping similar results, but beeing less effective in increasing the validity of a cluster solely by its reasoning path. SciBERT combines this focus with a comprehensive understanding of the reasoning process. This process makes sciBERT the most effective model for our evaluation, due to its high prominence on the produced sequence rather then its outcome.

BERT-Model	avg distance (↓)	
RoBERTa	48.697	
MathBERT	45.892	**(-2.8)**
SciBERT	45.281	**(-3.4)**

Table 1: Showcasing the comparison by averaging the unnormalized distance of each point in the vector space to its assigned Cluster centroid, reveales the importance of finetuning of the featurizer to recieve accurate output representations.

4.2 self-consistency with abstraction

To have a wide distribution of different reasoning paths, we sample from a variety of 5 different temperatures per produced output. In our example of k = 10, that equals 2 samples per temperature.

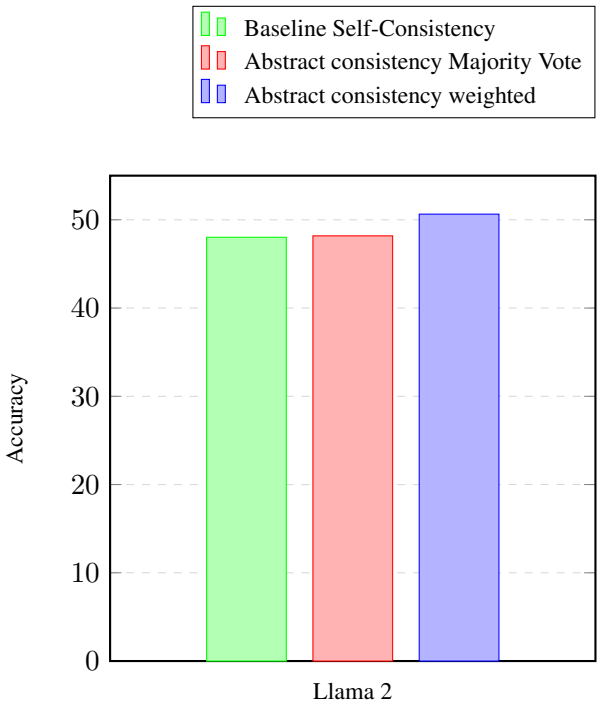

Figure 3: The above visible figure shows that self-consistency with varying levels of abstraction (here called abstract consistency) when used with inverse temperature weighting contains an increase in performance of evaluations of **2.5 %**.

[1]We apply Mistral without fine tuning on reasoning or pretraining on mathematical tasks to give a more accurate representation of effects introduced by our methods

Model	Method	AQuA-rat	SVAMP
Llama 2 7B	sc baseline	24.8	46.5
	inverse distance	24.6 **(-0.2)**	47.4 **(+0.9)**
	l1 inverse distance	24.9 **(+0.1)**	46.7 **(+0.2)**
Mistral 7B	sc baseline	25.6	68.5
	inverse distance	29.0 **(+3.4)**	69.8 **(+0.3)**
	l1 inverse distance	28.6 **(+3.0)**	69.8 **(+1.3)**

Table 2: This Table shows that with weighting models based on the inverse of the distance outputs we can improve overall self consisency by an average margin of 1.6 % for AQuA-rat and 0.6 % for SVAMP

4.3 Inverse-distance weighting

In a set of examples, it is common to observe that general answers exhibit similar operational patterns and behaviors. This observation underpins the application of inverse distance weighting, a technique where each vector in the set is assigned a weight based on its distance from a reference point or query. The essence of this approach lies in the principle that vectors closer to the query are more likely to be relevant and thus are given greater weight in the decision-making or reasoning process. The corresponding weights can be utilized in diagnosing reasoning error causes by since they are able to give more weight to better results to gain a more comprehensive overview of incorrect reasoning statements.

To perform these calculations, we first calculate the distances from each data point to the overall centroid. Then, we calculate the weights for each data point and normalize the weights so that they sum to 1. The process is shown below.

$$\text{centroid} = \frac{1}{N}\sum_{i=1}^{N}\text{data}[i]$$

$$\text{distances}[i] = \|\text{data}[i] - \text{centroid}\|$$

$$\text{weights}[i] = \frac{1}{\text{distances}[i]^p}$$

$$\text{weights}[i] = \frac{\text{weights}[i]}{\sum_{i=1}^{N}\text{weights}[i]}$$

In these equations, centroid represents the centroid of all data points, distances represents the distances of each data point to the centroid, and weights represents the weights assigned to each data point based on the distances also N is the number of data points, and data$[i]$ represents the i-th data point, for each i from 1 to N, where $\|\cdot\|$ represents the Euclidean norm.

4.4 Evaluation on multiple clusters

We applied the k-means clustering algorithm to various values of the parameter *k*, with a particular focus on $k = 2$, in order to highlight the phenomenon of favoring the cluster with the largest data set while discarding clusters with fewer samples.

This approach was employed to ensure the retention of the cluster exhibiting the highest degree of coherent reasoning paths. This method implies that the predictions associated with this cluster are the ones for which the model exhibits the greatest confidence.

4.5 Self-consistency with outlier detection

In our study, we conducted an extensive analysis using various anomaly detection techniques, including k-nearest neighbors (KNN) and isolation forest (ISF). To ensure the robustness of our results, we experimented with both dimensionality reduction techniques and without them. The obtained results exhibited slight deviations between the different configurations. However, to provide a more stable and representative assessment, we adopted an approach of averaging the results across all variations. Which lead to the conclusion that outlier detection didn't improve self consistency by a noticeable margin.

Although results aren't increased nor decreased one might use outlier detection techniques to marginalize out irrelevant results to get a cleaner analysis on actual deviation of relevant reasoning paths to gain a more comprehensive and meaningful distribution of results.

4.6 Results of systematically augmenting results

To enhance the quality of our embeddings and ensure they are not clustered solely based on output results, we implemented a process of result augmentation. This involved removing end results before generating embedding vectors, which were then used to form clusters. Our findings demonstrate that this approach not only mitigates the influence of inconclusive answers but also enhances the overall reasoning quality.

Model	with None	without None
Mistral	41.36	39.05
Llama-2	64.24	61.01

Table 3: Accuracy representation with and without incorporating results from None numerical solutions.

5 Additional studies

5.1 Detecting anomalies with Support vector machines

In classification and outlier detection tasks, support vector machines have frequently served as a prevalent tool. Given the inherent high-dimensional nature of our embedding vectors, we tried to gain insights into it.

Despite the capacity of SVM to effectively marginalize outcomes, it encounters a limitation when it excessively marginalizes a substantial number of results distributed in a non-predictable pattern. This distortion in the overall distribution of results has caused SVM to decrease overall performance by an average of 2.87%, making it an invalid solution for diagnosing and improving performance in reasoning.

6 Related Work

Reasoning has been identified as an ubiquitous issue, across many domains in Large Language Models (Creswell et al., 2022). After Rae et al. (2021) highlighted the challenges in reasoning across various domains in Large Language Models, subsequent research has increasingly focused on enhancing these models reasoning capabilities.

One general method applied in many of those studies, is **few-shot learning** which shown positive results in guiding a model into a more contextually aware and accurate direction. By training with a small but highly fitting set of examples, these models demonstrate an enhanced ability to infer and apply knowledge. (Brown et al., 2020)

Furthermore **fine-tuning** has shown positive results on specialized data in a broad amount of areas. Research by Radford and Narasimhan (2018) shows that targeted fine-tuning can notably enhance the model's performance in certain areas.

One other significant advancement in the area that has synergized with few shot has been the development of the '**chain of thought**' prompting, which guides LLM's to mimic human-like step-by-step reasoning processes (Wei et al., 2022). This method has proven effective in improving the accuracy and reliability of responses from LLMs in complex rational thought processes.

Building on these developments, our research extends the concept of self-consistency, as introduced by Wang et al. (2023) and harnesses the positive results delivered by chain-of-thought prompting.

7 Limitations

This study, aimed at enhancing the diagnostic tools of sampled reasoning paths using a novel approach that combines results from different temperature settings and leverages semantic vector clustering, encounters several limitations worth noting.

7.1 Sampling Quality Dependence

Our study's efficacy hinges on the quality of samples generated from different temperature settings. As described, we harness the model's capabilities to think abstractly at various temperatures to create a diverse range of outputs (Gunasekar et al., 2023). This approach, while innovative, also introduces a potential limitation. The diversity and representativeness of these samples are critical; if the samples at various temperatures are not sufficiently varied or if they are skewed towards certain types of responses, it could limit the accuracy and applicability of our findings. Moreover, the choice of temperature settings and their impact on the model's output diversity is a delicate balance. Too much diversity could lead to irrelevant or off-topic responses, while too little could stifle the innovative aspect of the approach. This dependence on the nuanced selection of temperature settings and the inherent variability in the model's responses at these settings underscores a significant limitation of our methodology.

7.2 Complexities in Semantic Clustering

Our study proposes the application of semantic vector representations to cluster model outputs, which is designed to facilitate the identification of consensus responses (Wang et al., 2023). While this technique is innovative in increasing the efficacy of self-consistency training, it also introduces significant complexities as a potential limitation. Semantic vectors must capture the subtle variations in meaning and context, which is particularly hard in abstract reasoning tasks without a sufficient amount of context making prompting techniques to enhance the models output structure an important factor. The process of clustering based on semantic vectors can be challenging due to the nuanced and abstract nature of reasoning processes. This limitation underscores the need for advanced featurization models in semantic analysis and clustering to ensure that the model outputs are grouped in a way that truly reflects their underlying meaning and relevance.

8 Conclusion and discussion

In this study, we demonstrate that employing various clustering algorithms offers a straightforward yet effective method for diagnosing the causes of errors in large language models. Through the application of clustering techniques, we are able to discern whether inaccuracies in a model's outputs are attributable to the reasoning processes it employs or to its arithmetic capabilities. Furthermore, our findings suggest that clustering not only serves as a diagnostic tool but also contributes to enhancing the overall data quality and cross-validation processes within these models. The utilization of clusters as benchmarks for model calibration and error correction for improving the reliability and accuracy of large-scale language models. Our findings suggest that clustering serves as a valuable tool in the refinement and enhancement of large language models, contributing to improvements in their reliability and accuracy in other relevant reasoning methods. Future developments may use this method to increase performance on commonsense reasoning,

9 Reproducibility Statement

Our experiments include a variety of models with different sizes: Microsoft Phi1.5B is publicly available at https://huggingface.co/microsoft/phi-1_5/tree/main and can be used under the Microsoft Research License.
GPT-3 has an API that is open for public use https://openai.com/blog/openai-api.
Mistral 7B is available for unrestricted use under the Apache 2.0 license, while its model architecture and setup are open source https://github.com/mistralai/mistral-src.
Llama 2 is a model with restricted access, made available by Meta. You can gain access to it by requesting permission through the provided Meta license. You can find more information about it at https://ai.meta.com/llama/.
The new released Phi 2 model isnt realsed publicly yet, but is estimated to be released publicly in the upcoming weeks under the Microsoft Research License.
All of our BERT models are built upon the BERT-base model developed by google-research, which is accessible under the Apache 2.0 license. This applies to all the BERT models we use, including MathBERT and sciBert, except for roBERTa, which can be used under the MIT license.

Our Datasets as well as used configuration for our language Models, are accessible throughout this paper and in the Appendix to aid the reproducibilty of our experiments.
A majority of our experiments were done using huggingface to access datasets, models and general data. The used algorithms were implemented with scikit-learn (Pedregosa et al., 2011) and the sklearn api (Buitinck et al., 2013).

9.1 GPU usage

approx. Hours	GPU	Model	Memory
90 h	NVIDIA	T4	15GB
45 h	NVIDIA	V100	16GB
35 h	NVIDIA	A100	40GB

10 Ethical Considerations & Risks

Language Models can produce factual incorrect information and might induce biases based on user prompts.
Mistral 7B and Microsoft Phi1.5 do not include content moderation. Also Microsoft Phi is purely intended for research applications is not tested on production level applications.
We encourage anyone to use produced results and capabilities of Language Models in a responsible manner.

References

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. Scibert: A pretrained language model for scientific text.

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.

Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, Robert Layton, Jake VanderPlas, Arnaud Joly, Brian Holt, and Gaël Varoquaux. 2013. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pages 108–122.

Antonia Creswell, Murray Shanahan, and Irina Higgins. 2022. Selection-inference: Exploiting large language models for interpretable logical reasoning. *ArXiv*, abs/2205.09712.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Suriya Gunasekar, Yi Zhang, Jyoti Aneja, Caio César Teodoro Mendes, Allie Del Giorno, Sivakanth Gopi, Mojan Javaheripi, Piero Kauffmann, Gustavo de Rosa, Olli Saarikivi, Adil Salim, Shital Shah, Harkirat Singh Behl, Xin Wang, Sébastien Bubeck, Ronen Eldan, Adam Tauman Kalai, Yin Tat Lee, and Yuanzhi Li. 2023. Textbooks are all you need.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation : Learning to solve and explain algebraic word problems.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Grégoire Mialon, Roberto Dessì, Maria Lomeli, Christoforos Nalmpantis, Ram Pasunuru, Roberta Raileanu, Baptiste Rozière, Timo Schick, Jane Dwivedi-Yu, Asli Celikyilmaz, Edouard Grave, Yann LeCun, and Thomas Scialom. 2023. Augmented language models: a survey.

Humza Naveed, Asad Ullah Khan, Shi Qiu, Muhammad Saqib, Saeed Anwar, Muhammad Usman, Naveed Akhtar, Nick Barnes, and Ajmal Mian. 2023. A comprehensive overview of large language models.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are nlp models really able to solve simple math word problems?

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Alec Radford and Karthik Narasimhan. 2018. Improving language understanding by generative pre-training.

Jack W. Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, Eliza Rutherford, Tom Hennigan, Jacob Menick, Albin Cassirer, Richard Powell, George van den Driessche, Lisa Anne Hendricks, Maribeth Rauh, Po-Sen Huang, Amelia Glaese, Johannes Welbl, Sumanth Dathathri, Saffron Huang, Jonathan Uesato, John F. J. Mellor, Irina Higgins, Antonia Creswell, Nathan McAleese, Amy Wu, Erich Elsen, Siddhant M. Jayakumar, Elena Buchatskaya, David Budden, Esme Sutherland, Karen Simonyan, Michela Paganini, L. Sifre, Lena Martens, Xiang Lorraine Li, Adhiguna Kuncoro, Aida Nematzadeh, Elena Gribovskaya, Domenic Donato, Angeliki Lazaridou, Arthur Mensch, Jean-Baptiste Lespiau, Maria Tsimpoukelli, N. K. Grigorev, Doug Fritz, Thibault Sottiaux, Mantas Pajarskas, Tobias Pohlen, Zhitao Gong, Daniel Toyama, Cyprien de Masson d'Autume, Yujia Li, Tayfun Terzi, Vladimir Mikulik, Igor Babuschkin, Aidan Clark, Diego de Las Casas, Aurelia Guy, Chris Jones, James Bradbury, Matthew G. Johnson, Blake A. Hechtman, Laura Weidinger, Iason Gabriel, William S. Isaac, Edward Lockhart, Simon Osindero, Laura Rimell, Chris Dyer, Oriol Vinyals, Kareem W. Ayoub, Jeff Stanway, L. L. Bennett, Demis Hassabis, Koray Kavukcuoglu, and Geoffrey Irving. 2021. Scaling language models: Methods, analysis & insights from training gopher. *ArXiv*, abs/2112.11446.

Iqbal H Sarker. 2021. Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions. *SN Comput Sci*, 2(6):420.

Jia Tracy Shen, Michiharu Yamashita, Ethan Prihar, Neil Heffernan, Xintao Wu, Ben Graff, and Dongwon Lee.

2023. Mathbert: A pre-trained language model for general nlp tasks in mathematics education.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9:2579–2605.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed H. Chi, Quoc Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. *CoRR*, abs/2201.11903.

11 Appendices

A Model configurations

- top-k: 50
- top-p: 50
- sampling: true
- max_new_tokens: 150
- temperature: see Appendix C

B Used k-shot prompts

The used **8-Shot prompt** for mathematical reasoning follows the example provided in pg. 43 and use the on pg. 36 referenced set for AQuA on the AQuA-rat dataset with of the original Chain of thought paper.

Additionally we follow the **6-Shot prompt** proposed on pg. 39 of the chain of thought paper for commonsense reasoning.

Proposed set of 8-shot Math examples

Q: There are 15 trees in the grove. Grove workers will plant trees in the grove today. After they are done, there will be 21 trees. How many trees did the grove workers plant today?
A: We start with 15 trees. Later we have 21 trees. The difference must be the number of trees they planted. So, they must have planted 21 - 15 = 6 trees. The answer is 6.
Q: If there are 3 cars in the parking lot and 2 more cars arrive, how many cars are in the parking lot?
A: There are 3 cars in the parking lot already. 2 more arrive. Now there are 3 + 2 = 5 cars. The answer is 5.
Q: Leah had 32 chocolates and her sister had 42. If they ate 35, how many pieces do they have left in total?
A: Leah had 32 chocolates and Leah's sister had 42. That means there were originally 32 + 42 = 74 chocolates. 35 have been eaten. So in total they still have 74 - 35 = 39 chocolates. The answer is 39. Q: Jason had 20 lollipops. He gave Denny some lollipops. Now Jason has 12 lollipops. How many lollipops did Jason give to Denny?
A: Jason had 20 lollipops. Since he only has 12 now, he must have given the rest to Denny. The number of lollipops he has given to Denny must have been 20 - 12 = 8 lollipops. The answer is 8.
Q: Shawn has five toys. For Christmas, he got two toys each from his mom and dad. How many toys does he have now?
A: He has 5 toys. He got 2 from mom, so after that he has 5 + 2 = 7 toys. Then he got 2 more from dad, so in total he has 7 + 2 = 9 toys. The answer is 9.
Q: There were nine computers in the server room. Five more computers were installed each day, from monday to thursday. How many computers are now in the server room?
A: There are 4 days from monday to thursday. 5 computers were added each day. That means in total 4 * 5 = 20 computers were added. There were 9 computers in the beginning, so now there are 9 + 20 = 29 computers. The answer is 29.
Q: Michael had 58 golf balls. On tuesday, he lost 23 golf balls. On wednesday, he lost 2 more. How many golf balls did he have at the end of wednesday?
A: Michael initially had 58 balls. He lost 23 on

Tuesday, so after that he has 58 - 23 = 35 balls. On Wednesday he lost 2 more so now he has 35 - 2 = 33 balls. The answer is 33.
Q: Olivia has $23. She bought five bagels for $3 each. How much money does she have left?
A: She bought 5 bagels for $3 each. This means she spent 5 * $3 = $15 on the bagels. She had $23 in beginning, so now she has $23 - $15 = $8. The answer is 8

Proposed set of 6-shot Commonsense reasoning examples

Q: Do hamsters provide food for any animals?
A: Hamsters are prey animals. Prey are food for predators. Thus, hamsters provide food for some animals. So the answer is yes.
Q: Could Brooke Shields succeed at University of Pennsylvania?
A: Brooke Shields went to Princeton University. Princeton University is about as academically rigorous as the University of Pennsylvania. Thus, Brooke Shields could also succeed at the University of Pennsylvania. So the answer is yes.
Q: Yes or no: Hydrogen's atomic number squared exceeds number of Spice Girls?
A: Hydrogen has an atomic number of 1. 1 squared is 1. There are 5 Spice Girls. Thus, Hydrogen's atomic number squared is less than 5. So the answer is no.
Q: Yes or no: Is it common to see frost during some college commencements?
A: College commencement ceremonies can happen in December, May, and June. December is in the winter, so there can be frost. Thus, there could be frost at some commencements. So the answer is yes.
Q: Yes or no: Could a llama birth twice during War in Vietnam (1945-46)?
A: The War in Vietnam was 6 months. The gestation period for a llama is 11 months, which is more than 6months. Thus, a llama could not give birth twice during the War in Vietnam. So the answer is no.
Q: Yes or no: Would a pear sink in water?
A: The density of a pear is about 0.6g/cm³, which is less than water. Objects less dense than water float. Thus,a pear would float. So the answer is no.

Proposed 4-shot on AQuA:

Q: John found that the average of 15 numbers is 40. If 10 is added to each number then the mean of the numbers is?
Answer Choices: (a) 50 (b) 45 (c) 65 (d) 78 (e) 64
A: If 10 is added to each number, then the mean of the numbers also increases by 10. So the new mean would be
50. The answer is (a).
Q: If a / b = 3/4 and 8a + 5b = 22,then find the value of a.
Answer Choices: (a) 1/2 (b) 3/2 (c) 5/2 (d) 4/2 (e) 7/2
A: If a / b = 3/4, then b = 4a / 3. So 8a + 5(4a / 3) = 22. This simplifies to 8a + 20a / 3 = 22, which means 44a / 3 = 22. So a is equal to 3/2. The answer is (b).
Q: A person is traveling at 20 km/hr and reached his destiny in 2.5 hr then find the distance?
Answer Choices: (a) 53 km (b) 55 km (c) 52 km (d) 60 km (e) 50 km
A: The distance that the person traveled would have been 20 km/hr * 2.5 hrs = 50 km. The answer is (e).
Q: How many keystrokes are needed to type the numbers from 1 to 500?
Answer Choices: (a) 1156 (b) 1392 (c) 1480 (d) 1562 (e) 1788
A: There are 9 one-digit numbers from 1 to 9. There are 90 two-digit numbers from 10 to 99. There are 401 three-digit numbers from 100 to 500. 9 + 90(2) + 401(3) = 1392. The answer is (b).

C Temperature sets

We tested our theory of abstraction on a variety of temperature sets and found that *set 1* exhibits the best balance between diversity and correctness in our examples. Therefore, it outperforms the other proposed sets. When testing wiuth baseline self-consistency

Set 1 (t)	Set 2 (t)	Set 3 (t)
0.9	0.7	0.5
0.8	0.6	0.4
0.7	0.5	0.3
0.6	0.4	0.2
0.5	0.3	0.1

Table 4: Each Temperature is tested on 1/5 of the samples per generation, to ensure an even distribution.

D Datasets

We use the configuration splits for testing as suggested by default. We employ a test split of 1000

samples on SVAMP and GSM8K. For AQuA-rat, our test includes 254 examples.