Teaching Large Language Models to Handle the Composition of Multiple Problems Simultaneously

Anonymous ACL submission

Abstract

With the widespread application of large language models (LLMs), the problem of generating non-existing facts, which is so-called *hallucination*, has earned increasing attention. In previous research on LLM hallucination, many methods only focus on single problem setting, leaving the more practical multiple problem setting under-explored, which requires the model to accurately answer multiple problems simultaneously without causing knowledge confusion. Drawing inspiration from earlier approaches aimed at discovering the knowledge gap between LLMs' internal parameterized knowledge and instruction-tuning data, we introduce a novel method named **M**ultiple **A**nswers and **C**onfidence Stepwise **Tuning** (**MAC-Tuning**), which separate the learning process of ground-truth answer and confidence during fine-tuning. Extensive experiments demonstrate that our method outperforms strong baselines by up to 12% in this challenging setting.

1 Introduction

Large language models (LLMs) are widely used in knowledge-intensive scenarios, such as question answering (Gu et al., 2023), information retrieval (Ren et al., 2023) and recommendation systems (Liu et al., 2023). However, LLMs have tendency to generate non-existing facts when faced with questions that are out of their parametric knowledge (Maynez et al., 2020). Many efforts are dedicated to mitigating hallucinations in LLMs, such as retrieval-augmented generation (Gao et al., 2024,Peng et al., 2023), multi-agent debate (Du et al., 2023,Sun et al., 2023), and model confidence calibration (Zhang et al., 2024a, Hu et al., 2023).

However, these approaches mainly focus on the **single-problem** setting, where models only need to answer problems one by one. In broader real-world scenarios, answering multiple questions simultaneously is more common and practical, such as in reading comprehension, data analysis, or multi-turn dialogues, especially when the context is lengthy, as it reduces the number of model accesses and lowers API costs. As shown in Figure 1, we define the scenario where single input contains several distinct sub-questions as the **multi-problem** setting. This setting requires the model to extract, focus on, and address different sub-problems from a single input, fully demonstrating its ability to handle multiple tasks simultaneously. Meanwhile, the model must generate accurate, contextually appropriate answers in a coherent order, ensuring that each sub-problem is addressed without causing any knowledge confusion. Although the performance of LLMs in single-problem settings has been extensively studied, multi-problem settings are still under-explored. Wang et al. (2024) figures out that LLM can handle muliple problems of the same type at once but lack of true understanding towards this complex scenario. Laskar et al. (2023) analyzes the ability of GPT-series models to solve 5 shorts question at once, showing that only instruction fine-tuned GPT models can deal with multiple problems with varying performance drops. However, it is still

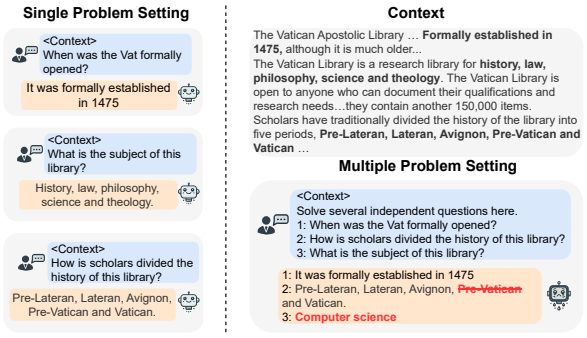

Figure 1: Examples of single-problem setting and multi-problem setting. Even if the model can accurately solve individual problems, it still faces challenges when dealing with multiple problems simultaneously. Due to space limitations, the context has been truncated. For the full case, please refer to Figure 3 in Appendix.

unknown for LLM hallucination under this multi-problem setting.

In this paper, we investigate teaching LLM to handle the composition of multiple problems simultaneously. Inspired by the work of Zhang et al. (2024a), which encourages the LLM to express confidence to reduce hallucinations, we introduce **M**ultiple **A**nswers and **C**onfidence Stepwise **Tuning** (**MAC-Tuning**) under multi-problem setting: (1) identify the knowledge gap between parametric knowledge and the multi-problem data we constructed, and obtain the uncertain questions. (2) automatically label the model's confidence for both certain and uncertain data, and then use this labeled data to develop multi-problem tuning data, and then enable the original model to undergo training on the multi-problem answering data and the corresponding multi-problem confidence data separately.

To summarize, we make following contributions:

- We are the first one to explore LLM hallucination under the more challenging context of multi-problem setting, where LLM need to handle multiple problems simultaneously without causing knowledge confusion.
- We propose MAC-Tuning that separate the learning process of ground-truth answers and confidence, achieving a higher accuracy up to 12%, compared to baseline methods.
- Extensive experimental results are conducted across different problem settings and diverse datasets, revealing meaningful trends under Multiple Problem setting.

2 Methodology

In this section, we introduce our MAC-Tuning approach, with automatically constructed data for LLM finetuning as illustrated in Figure 2.

2.1 Multi-Problem Dataset Construction

We first combine ***n*** number of independent single questions together to get the Multi-Problem dataset. Taking ***n*** as 3, as presented in Figure 1, the *Question* template is:

> *<Context>*
> Solve several independent questions here.
> 1: *<Question>*
> 2: *<Question>*
> 3: *<Question>*

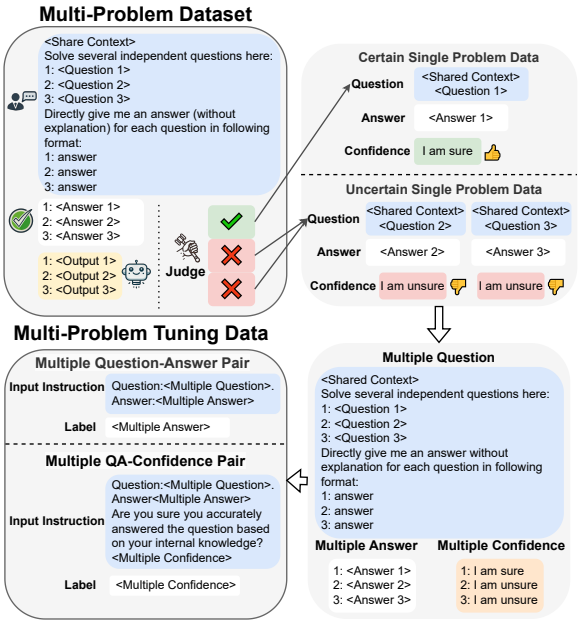

Figure 2: We first construct the Multi-Problem dataset and then use it to generate Multi-Problem Tuning data, for LLM finetuning.

The *Answers* component then follows the below template:

> 1.*<Answer>* 2.*<Answer>* 3.*<Answer>*.

2.2 Multi-Problem Tuning Data Construction

To better evaluate the knowledge gap between LLM parameters and instruction data, we utilize the Multi-Problem dataset to test model and compare these outputs with ground-truth answers. Based on the result, we categorize the training dataset of Single Problem $D = [(q_1, a_1)...(q_n, a_n)]$ into two dataset: certain dataset D_0 if model output aligns with the ground-truth answer; and uncertain dataset D_1 if model output does not. The two datasets are automatically assigned confidence labels. Then, as shown in Figure 2, we randomly shuffle the single-problem data, combine them together and construct Multi-Problem Tuning data with two different kind of data pairs:

Multiple Question-Answer pair: We directly combine the questions and answers together. The input-output pair consists of the *Question* instruction as input and *Answers* component as output label.

Multiple QA-Confidence pair: Confidence means the LLM's certainty towards the question. Specifically, it should be "*I am sure*" if the single question-answer pair is from certain data and "*I am unsure*" if the single question-answer pair is from uncertain

data. The form of QA-Confidence pair is:

> Here are some questions and corresponding answers:
> <*Multiple Question-Answer pairs*>
> Are you sure you accurately answered the question based on your internal knowledge?
> 1: <*Confidence*>
> 2: <*Confidence*>
> 3: <*Confidence*>

The whole pair should be the input instruction and only the *Confidence* part is the label.

2.3 Training and Inference

With the Multi-Problem Tuning data, we can proceed to fine-tune a large language model. The training objective is the cross-entropy loss $\mathcal{L}$:

$$\mathcal{L} = -\frac{1}{T}\sum_{i=1}^{T} \log P(t_i|t_1, t_2, \ldots, t_{i-1}). \quad (1)$$

Here, $t_1, t_2 \ldots t_n$ is the input instruction and $P(t_i|t_1, t_2, \ldots, t_{i-1})$ is the probability of generating i^{th} token given $t_1, t_2, \ldots t_{i-1}$. Note that we only calculate the loss for the label part when doing the fine-tuning, which means *Answer* component for Multiple Question-Answer pair and *Confidence* component for Multiple QA-Confidence pair. In this way, we let LLM first learn the ground-truth answer and then the confidence.

When performing inference, we first let the model generate the answers. Then, based on the question-answer pair, we prompt the model to further generate its confidence for each answer.

3 Experimental Settings

3.1 Dataset

We validate the effectiveness of our method across different problem settings and datasets. Specifically, for the *Question Answering* (QA) setting, where the model generates the corresponding answer given a question, we used the **CoQA** (Reddy et al., 2019), **ParaRel** (Elazar et al., 2021), and **GSM** (Cobbe et al., 2021) datasets. For the *Multiple Choice* (MC) setting, where the model selects an answer from given options, we used the **MMLU** (Hendrycks et al., 2021) and **GSM-MC** (Zhang et al., 2024b) datasets. We plot the distribution of certain and uncertain data across different datasets in Figure 6. More details can be found in Appendix A.2.

3.2 Baselines

We compare our MAC-Tuning with related methods and their variants in the new multi-problem setting. For **LLaMA3**, we evaluate the performance of the base model under Multiple Problem setting. In **Vanilla-Tuning**, we fine-tune the base LLaMA3 model directly using the question-answer pairs from the constructed certain Multi-Problem Dataset to evaluate the effectiveness of the traditional instruction tuning method under the Multiple Problem setting. In **R-Tuning-SP**, we evaluate the performance of original R-tuning method Zhang et al. (2024a) using Single-Question data to fine-tune, directly applied now to the Multiple Problem setting. Finally, in **R-Tuning-MP**, we evaluate the performance of original R-tuning method under Multiple Problem setting. Further details can be found in Appendix A.4.

3.3 Evaluation

We directly compare the LLM generation to the ground-truth answer for Question Answer setting. For Multiple Choice setting, we check both the choice (A, B, C, D) and the option in the LLM generation. When evaluating the final performance, we evaluate only the responses that LLMs confidently provide, indicating they are sure about the answers. Each single question-answer pair is counted as "1" in our calculation of accuracy, based on the following formula:

$$\text{accuracy} = \frac{\#\text{ of correct answers}}{\#\text{ of sure answers}} \quad (2)$$

3.4 Implementation

We choose LLaMA3-8B (Dubey et al., 2024) as the base model in our experiments. We use HuggingFace PEFT (Mangrulkar et al., 2022) to conduct LoRA fine-tuning (Hu et al., 2021). We set the training epoch to 3, learning rate to $1e^{-5}$, LoRa rank to 8, and LoRa scaling factor to 32. The batch size is 1 and the temperature is 0. All experiments are implemented on Nvidia A100-40GB GPUs.

4 Experimental Results

The main results is presented in Table 1, where we have two main observations: (1) MAC-Tuning has the best accuracy among all cases. It has an accuracy improvement up to 40% comparing with original LLaMA3 (and up to 12% comparing with baseline methods). In our method, we expect LLM

Model	QA			MC	
	CoQA	Pararel	GSM	MMLU	GSM-MC
LLaMA3	50.00%	42.24%	10.17%	52.07%	34.75%
Vanilla-Tuning	56.40%	64.66%	16.64%	53.90%	38.65%
R-Tuning-SP	53.40%	71.03%	20.00%	51.50%	34.81%
R-Tuning-MP	68.14%	75.33%	31.30%	54.32%	31.91%
MAC-Tuning	**77.13%**	**87.33%**	**37.76%**	**65.56%**	**41.00%**

Table 1: Accuracy across different methods when combining 3 single problems together. Bold font highlights the highest accuracy among all methods for this dataset.

to learn the correct answer and the confidence separately, as well as the pattern for Multiple Problem setting. After learning the ground-truth answer, LLM can better understand the confidence and during the learning process of confidence, LLM still remember the ability to extract, correctly response and address different problems simultaneously. (2) Both R-Tuning-SP and R-Tuning-MP increase the accuracy in most cases. Ideally, we would expect LLM to learn the correct answer with the confidence together in R-Tuning-SP and additionally learn the pattern to deal with Multiple Problem setting in R-Tuning-MP. The improvement is significantly higher in QA dataset than MC dataset, which might because the ground-truth answer is too short (only the choice) for MC dataset and the LLM might lose the attention towards the correct answer during fine-tuning. However, since R-Tuning-SP performs worse than R-Tuning-M in most cases, it is reasonable to say that solely relying on fine-tuning with Single-Question data is insufficient for LLM to handle more complex scenarios like Multiple Question setting.

Please refer to Appendix A.3 and Appendix A.5 for details on specific case study and answer rate.

4.1 Analysis on Combining Different Number of Problems

We explore different number of questions in the mult-problem setting to investigate how this varies the accuracy. The results are reported in Table 2. Ideally, we would expect the difficulty become higher as the question number ***n*** become larger and LLM should show the best performance when ***n*** = 1. However, the results show that the ability for original model to handle multiple problems simultaneously is even higher when compared with the traditional Single Problem setting, which is aligned with Wang et al. (2024)'s finding. A reasonable explanation is that LLM might leverage in-context learning and focus on relevant knowledge better, when simultaneously handling multiple questions from the same dataset.

Dataset			LLaMA3	MAC-Tuning
QA	CoQA	n=1	46.81%	73.25%
		n=3	50.00%	**77.13%**
		n=5	**51.14%**	66.29%
	Pararel	n=1	42.94%	86.10%
		n=3	42.24%	**87.33%**
		n=5	**48.19%**	86.28%
	GSM	n=1	10.01%	**57.14%**
		n=3	**10.17%**	30.73%
		n=5	9.28%	19.05%
MC	MMLU	n=1	49.77%	**75.27%**
		n=3	52.07%	65.56%
		n=5	**53.30%**	63.00%
	GSM-MC	n=1	27.03%	**44.70%**
		n=3	**34.75%**	41.00%
		n=5	29.81%	31.88%

Table 2: Accuracy for combining different number (***n***) of single problem together. MC means "Multiple Choice" and QA means "Question Answer". Bold font highlights the highest accuracy for the dataset among different number (***n***).

After MAC-Tuning, the model have a better performance at ***n*** = 1 when facing difficult QA dataset (*e.g.,* GSM, GSM MC). When it comes to easy QA dataset like CoQA, the model have a better performance at ***n*** = 3. A reasonable explanation is that it may be out of an LLM's ability to learn several hard tasks together but within effective scope to learn several easy tasks at the same time.

5 Conclusion

In this paper, we introduce a novel method called MAC-Tuning, designed to enhance the performance of LLMs by mitigating hallucination in the emerging multi-problem scenario. This scenario, which involves answering multiple questions simultaneously, has been under explored by large in existing research. Our approach leverages the knowledge gap between parametric knowledge and instructional data, enabling LLMs to express uncertainty when encountering unfamiliar problems. Furthermore, our method separate the learning process of correct answer and the confidence, which not only increases accuracy on different dataset, but also yields significantly improved performance in cases where the original LLM struggles with.

Limitation

While our work provides valuable insight on the new Multiple Question setting and introduces an innovative fine-tuning method, there are several limitations to acknowledge. First, although we experimented with various prompts, as is typical in prompt-based LLM studies, we cannot ensure that slight changes in prompts would not significantly alter the results. Second, due to constraints of cost, time, and computational resources, we selected a subset of experiments that we believe to be informative and representative. However, additional experiments across a wider range of datasets and LLMs might provide further insights. Lastly, in this new setting, there may be other underlying reasons for the experimental results. Future work will aim to address these limitations by expanding datasets and conducting new experiments to explore other potential factors affecting performance.

References

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B. Tenenbaum, and Igor Mordatch. 2023. Improving factuality and reasoning in language models through multiagent debate.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaoqing Ellen Tan, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aaron Grattafiori, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alex Vaughan, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Franco, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Da-

mon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, Danny Wyatt, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Firat Ozgenel, Francesco Caggioni, Francisco Guzmán, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Govind Thattai, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Karthik Prasad, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kun Huang, Kunal Chawla, Kushal Lakhotia, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Maria Tsimpoukelli, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikolay Pavlovich Laptev, Ning Dong, Ning Zhang, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Rohan Maheswari, Russ Howes, Ruty Rinott, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Kohler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vítor Albiero, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaofang Wang, Xiaojian Wu, Xiaolan Wang, Xide Xia, Xilun Wu, Xinbo Gao, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yuchen Hao, Yundi Qian, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, and Zhiwei Zhao. 2024. The llama 3 herd of models.

Yanai Elazar, Nora Kassner, Shauli Ravfogel, Abhilasha Ravichander, Ed Hovy, Hinrich Schutze, and Yoav Goldberg. 2021. Measuring and improving consistency in pretrained language models. *ArXiv*, abs/2102.01017.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang. 2024. Retrieval-augmented generation for large language models: A survey.

Yu Gu, Xiang Deng, and Yu Su. 2023. Don't generate, discriminate: A proposal for grounding language models to real-world environments. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4928–4949, Toronto, Canada. Association for Computational Linguistics.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models.

Shengding Hu, Yifan Luo, Huadong Wang, Xingyi Cheng, Zhiyuan Liu, and Maosong Sun. 2023. Won't get fooled again: Answering questions with false premises. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5626–5643, Toronto, Canada. Association for Computational Linguistics.

Md Tahmid Rahman Laskar, M Saiful Bari, Mizanur Rahman, Md Amran Hossen Bhuiyan, Shafiq Joty, and Jimmy Huang. 2023. A systematic study and comprehensive evaluation of ChatGPT on benchmark datasets. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 431–469, Toronto, Canada. Association for Computational Linguistics.

Qijiong Liu, Nuo Chen, Tetsuya Sakai, and Xiao-Ming Wu. 2023. A first look at llm-powered generative news recommendation. *ArXiv*, abs/2305.06566.

Sourab Mangrulkar, Sylvain Gugger, Lysandre Debut, Younes Belkada, Sayak Paul, and Benjamin Bossan. 2022. Peft: State-of-the-art parameter-efficient fine-tuning methods. https://github.com/huggingface/peft.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919, Online. Association for Computational Linguistics.

Baolin Peng, Michel Galley, Pengcheng He, Hao Cheng, Yujia Xie, Yu Hu, Qiuyuan Huang, Lars Liden, Zhou Yu, Weizhu Chen, et al. 2023. Check your facts and try again: Improving large language models with external knowledge and automated feedback. *arXiv preprint arXiv:2302.12813*.

Siva Reddy, Danqi Chen, and Christopher D. Manning. 2019. CoQA: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266.

Ruiyang Ren, Wayne Xin Zhao, Jing Liu, Hua Wu, Ji-Rong Wen, and Haifeng Wang. 2023. Tome: A two-stage approach for model-based retrieval.

Zhiqing Sun, Yikang Shen, Qinhong Zhou, Hongxin Zhang, Zhenfang Chen, David Cox, Yiming Yang, and Chuang Gan. 2023. Principle-driven self-alignment of language models from scratch with minimal human supervision.

Zhengxiang Wang, Jordan Kodner, and Owen Rambow. 2024. Evaluating llms with multiple problems at once: A new paradigm for probing llm capabilities.

Hanning Zhang, Shizhe Diao, Yong Lin, Yi Fung, Qing Lian, Xingyao Wang, Yangyi Chen, Heng Ji, and Tong Zhang. 2024a. R-tuning: Instructing large language models to say 'I don't know'. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 7113–7139, Mexico City, Mexico. Association for Computational Linguistics.

Ziyin Zhang, Zhaokun Jiang, Lizhen Xu, Hongkun Hao, and Rui Wang. 2024b. Multiple-choice questions are efficient and robust llm evaluators.

A Appendix

A.1 Full Case for Examples of Introduction

Full case for the examples in introduction can be found in Figure 3.

A.2 Dataset Details

We carry out our experiments across give datasets, described as follows.

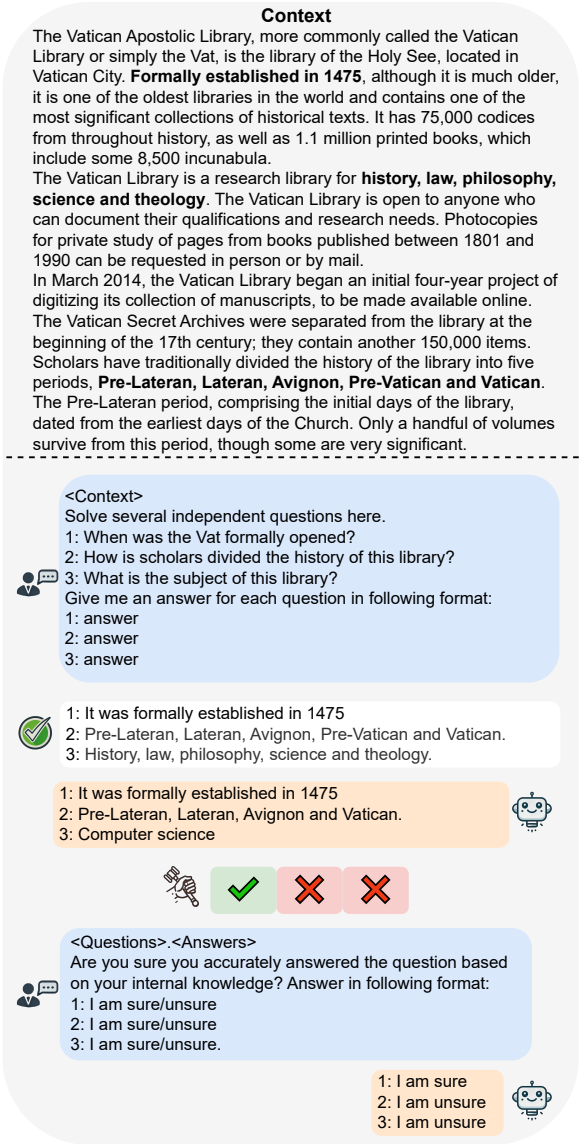

Figure 3: The full case of examples in introduction in Multiple Problem setting.

- **GSM** (Cobbe et al., 2021): a dataset containing high-quality grade school math problems created by the OpenAI group. These problems require between 2 and 8 steps to solve, primarily involving a sequence of elementary calculations with basic arithmetic operations such as addition, subtraction, multiplication, and division to arrive at the final answer. We directly use 7.5k training dataset and 1k testing dataset in our Question Answer setting.

- **GSM-MC** (Zhang et al., 2024b) a multiple-choice dataset constructed by collecting answers and incorrect predictions on GSM8k from 60 open-source models. We directly use 7.5k training dataset and 1k testing dataset in

our Multiple Choice setting.

- **Pararel** (Elazar et al., 2021): a dataset containing factual knowledge with a variety of prompts and relationships, originally created for mask prediction. In Question Answer setting, we employ the modified dataset from Zhang et al. (2024a).

- **MMLU** (Hendrycks et al., 2021) a dataset covering different subjects and difficulty. It tests both world knowledge and problem solving ability, which has good granularity and breath. We directly use this dataset in our Multiple Choice setting.

- **CoQA** (Reddy et al., 2019) a dataset designed to evaluate the ability of models to understand and generate answers in a conversational setting. We randomly pick 5k training dataset from theirs. In Question Answer setting, we combine multiple questions together under the same "story" category in the dataset.

A.3 Case Study

We show two specific cases for MAC-Tuning in Figure 4.

A.4 Detailed information for Baseline Methods

The detailed example for different baseline methods is shown in Figure 5.

A.5 Answer Rate

We also look at the answer rate of different method under Multiple Problem setting for different datasets, which is shown in Table 3.

Model	QA			MC	
	CoQA	Pararel	GSM	MMLU	GSM-MC
Vanilla-Tuning	**99.30%**	**99.30%**	**83.98%**	**53.90%**	**98.03%**
R-Tuning-SP	0.37%	13.20%	0.38%	51.50%	77.52%
R-Tuning-MP	63.19%	54.47%	8.73%	54.32%	64.92%
MAC-Tuning	31.04%	33.51%	7.44%	65.56%	34.85%

Table 3: Answer rate for combining 3 single problems together with different methods. Bold font highlights the highest answer rate among all methods for this dataset.

A.6 Cross Task Transfer Study

We fine-tune the model with question number $\boldsymbol{n}$ = 3 and let it response to question number $\boldsymbol{n}$ = 1 (which is Single Problem setting) and question number $\boldsymbol{n}$ = 5. For the former one, we want to test if the model understands the single problem. For the latter one, we want to test if MAC-Tuning can generalize across different question number $\boldsymbol{n}$. The results are reported in Table 4.

Dataset		n=1	n=5
QA	CoQA	78.78%	79.12%
	Pararel	84.24%	86.16%
	GSM	54.55%	31.15%
MC	MMLU	54.60%	63.69%
	GSM-MC	39.61%	39.36%

Table 4: Accuracy for MAC-Tuning with question number $\boldsymbol{n}$ = 3 transferring to question number $\boldsymbol{n}$ = 1 (which is Single Problem setting) and question number $\boldsymbol{n}$ = 5.

From the result of $\boldsymbol{n}$ = 1, we observe that accuracy improves in easy QA dataset like CoQA but decreases in MC dataset like MMLU, comparing with using question number $\boldsymbol{n}$ = 1 to fine-tune. This indicates that LLM learn knowledge during fine-tuning rather than just memorizing the patterns for Multiple Problem setting. With $\boldsymbol{n}$ = 5, the model performs about the same or even better than using question number $\boldsymbol{n}$ = 5 to fine-tune. This strengthen the statement we make in Section 4.1 that LLMs are more efficient at learning multiple easy tasks but struggle with several difficult tasks simultaneously.

A.7 Certainty Distribution of the Training Dataset

We demonstrate the certainty distribution of the training dataset under Multiple Problem setting with question number n = 3 in Figure 6:

A.8 Average Precision Score

We demonstrate the Average Precision score (**AP score**) in Table 5. The formula for confidence is:

$$\frac{P_{next_token}(sure)}{P_{next_token}(sure)+P_{next_token}(unsure)}$$

Model	QA			MC	
	CoQA	Pararel	GSM	MMLU	GSM-MC
LLaMA3	50.13%	40.77%	10.13%	50.34%	32.62%
MAC-Tuning	67.33%	66.53%	26.41%	58.12%	41.10%

Table 5: AP Scores for MAC-Tuning and original LLaMA3 with question number $\boldsymbol{n}$ = 3.

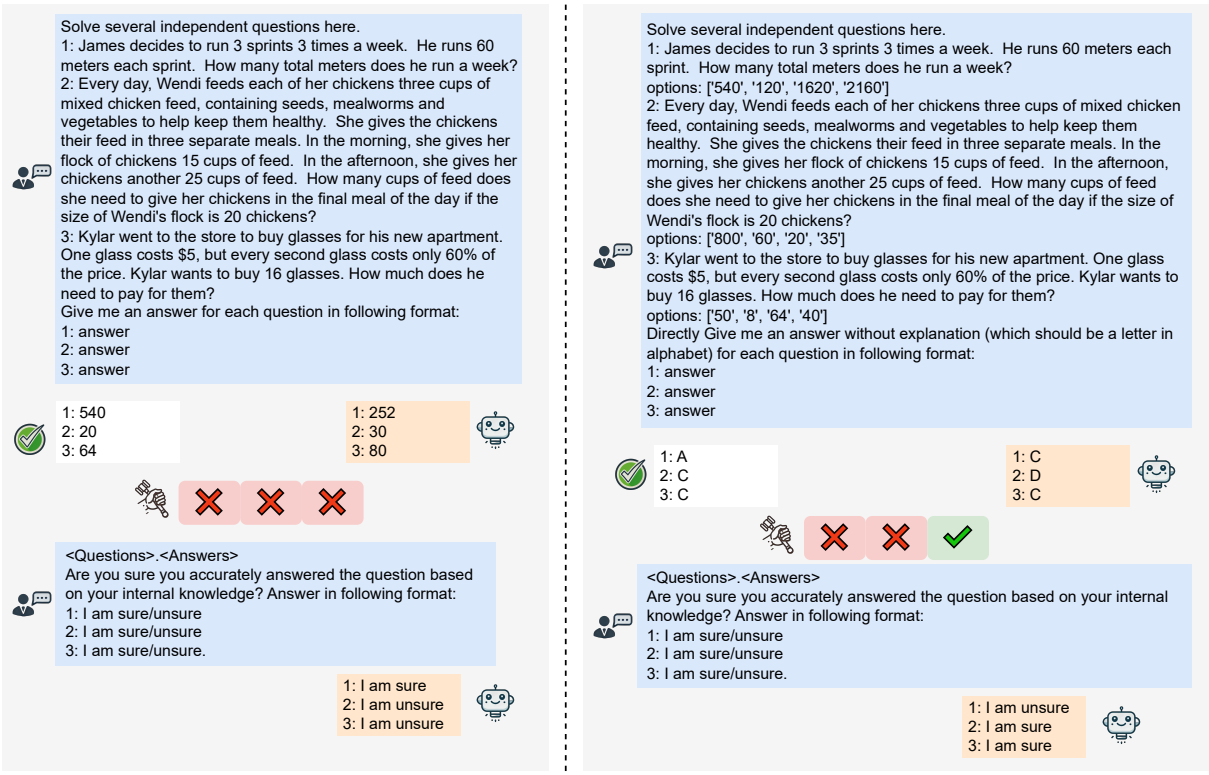

Figure 4: The left one is an example of GSM and the right one is an example of GSM-MC. We demonstrate the multiple problem setting with question number $n = 3$ in this picture. Taking the left side as example, since LLM only says "*I am sure*" to first question, only this problem will be counted as negative when calculating the accuracy. Also, it is good to see LLM says "*I am unsure*" for the other two questions that LLM gives wrong answers.

Vanilla

Input Instruction

<Shared Context>
Question: Solve several independent questions here.
1: Did Jane think she could control Lassiter?
2: Who is Bells?
3: How did he get his name?
Directly Give me an answer without explanation for each question in following format:
1: answer
2: answer
3: answer
Answer:
1: No
2: One of her safe racers
3: because of the way he struck his iron shoes on the stones.

Label

Answer:
1: No
2: One of her safe racers
3: because of the way he struck his iron shoes on the stones.

R-Tuning-M

Input Instruction

<Shared Context>
Question: Solve several independent questions here.
1: Did Jane think she could control Lassiter?
2: Who is Bells?
3: How did he get his name?
Directly Give me an answer without explanation for each question in following format:
1: answer
2: answer
3: answer
Answer:
1: No
2: One of her safe racers
3: because of the way he struck his iron shoes on the stones.
Are you accurately answered the question based on your internal knowledge?
1: I am sure
2: I am unsure
3: I am sure

Label

Answer:
1: No
2: One of her safe racers
3: because of the way he struck his iron shoes on the stones.
Are you accurately answered the question based on your internal knowledge?
1: I am sure
2: I am unsure
3: I am sure

Shared Context

CHAPTER VII. THE DAUGHTER OF WITHERSTEEN
"Lassiter, will you be my rider?" Jane had asked him.
"I reckon so," he had replied.
Few as the words were, Jane knew how infinitely much they implied. She wanted him to take charge of her cattle and horse and ranges, and save them if that were possible. Yet, though she could not have spoken aloud all she meant, she was perfectly honest with herself. Whatever the price to be paid, she must keep Lassiter close to her; she must shield from him the man who had led Milly Erne to Cottonwoods. In her fear she so controlled her mind that she did not whisper this Mormon's name to her own soul, she did not even think it. Besides, beyond this thing she regarded as a sacred obligation thrust upon her, was the need of a helper, of a friend, of a champion in this critical time. If she could rule this gun-man, as Venters had called him, if she could even keep him from shedding blood, what strategy to play his flame and his presence against the game of oppression her churchmen were waging against her? Never would she forget the effect on Tull and his men when Venters shouted Lassiter's name. If she could not wholly control Lassiter, then what she could do might put off the fatal day.
One of her safe racers was a dark bay, and she called him Bells because of the way he struck his iron shoes on the stones.
When Jerd led out this slender, beautifully built horse Lassiter suddenly became all eyes. A rider's love of a thoroughbred shone in them. Round and round Bells he walked, plainly weakening all the time in his determination not to take one of Jane's favorite racers.

R-Tuning-S

Input Instruction

<Shared Context>
Question: Solve several independent questions here.
1: Did Jane think she could control Lassiter?
2: Who is Bells?
3: How did he get his name?
Directly Give me an answer without explanation for each question in following format:
1: answer
Answer:
1: No
Are you accurately answered the question based on your internal knowledge?
1: I am sure

Label

Answer:
1: No
Are you accurately answered the question based on your internal knowledge?
1: I am sure

Figure 5: A specific case to show how baseline methods are doing the fine-tuning.

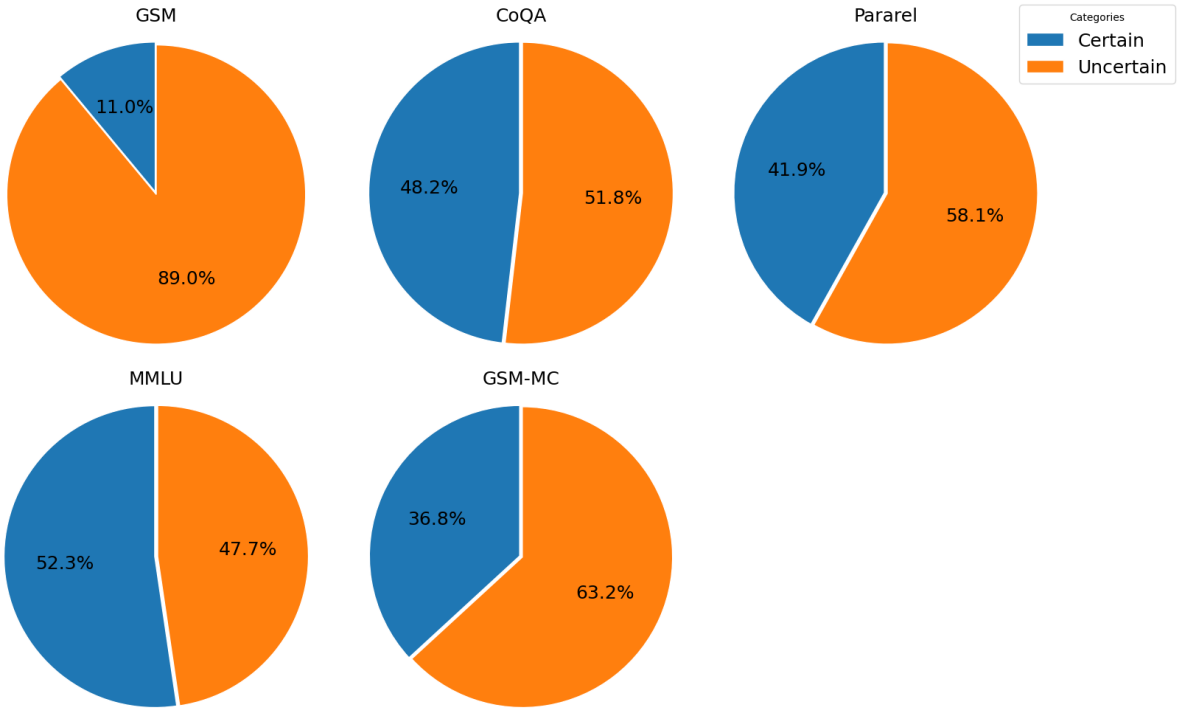

Figure 6: Certainty distribution of the training dataset under Multiple Problem setting with $\boldsymbol{n} = 3$