MARAGS: A Multi-Adapter System for Multi-Task Retrieval Augmented Generation Question Answering

Mitchell DeHaven
mdehaven@darkhive.com
Darkhive
San Antonio, Texas, USA

Abstract

In this paper we present a multi-adapter retrieval augmented generation system (MARAGS) for Meta's Comprehensive RAG (CRAG) competition for KDD CUP 2024. CRAG is a question answering dataset contains 3 different subtasks aimed at realistic question and answering RAG related tasks, with a diverse set of question topics, question types, time dynamic answers, and questions featuring entities of varying popularity.

Our system follows a standard setup for web based RAG, which uses processed web pages to provide context for an LLM to produce generations, while also querying API endpoints for additional information. MARAGS also utilizes multiple different adapters to solve the various requirements for these tasks with a standard cross-encoder model for ranking candidate passages relevant for answering the question. Our system achieved 2nd place for Task 1 as well as 3rd place on Task 2.

CCS Concepts

• **Computing methodologies → Information extraction**; **Multi-task learning**; **Natural language generation**.

Keywords

Natural Language Processing, Large Language Models, Information Retrieval

ACM Reference Format:
Mitchell DeHaven. 2018. MARAGS: A Multi-Adapter System for Multi-Task Retrieval Augmented Generation Question Answering. In *Proceedings of (KDD Cup 24).* ACM, New York, NY, USA, 6 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

Retrieval augmented generation (RAG) has been a popular approach for question answering systems for some time [6], although recently has become a very popular approach for a wide range of tasks due to the zero-shot capabilities of large language models (LLMs) with an appropriate prompt and access to the relevant context for the task. Despite the existence of numerous question answering benchmarks, many do accurately reflect the diverse usage of current RAG systems. Thus, tracking both the efficacy of certain RAG architectures as well as tracking process remains difficult. The CRAG [14] benchmark aims to resolve this with 3 different subtasks representing realistic RAG usage scenarios. The final key element of the CRAG benchmark is it's scoring metric which explicitly punishes hallucinations. With the rising capabilities of LLMs, increasingly their outputs are taken at face value, despite the known issue of hallucinations. This has lead to high profile incidents causing concern with their use [1]. The CRAG score aims to punish hallucinated answers and encourages returning missing answers, equivalent to returning "i don't know" from the model, by giving scores of 1, 0, and -1 to correct, missing, and hallucinated answers respectively. To address these various tasks, we train individual adapters for each of the various tasks, as well as API call generation required for accessing information in the mock API. This approach allows us to use a single LLama 3 [2] in memory while swapping out adapters based on the current needs.

2 Related Works

The initial approach of Lewis et al. [6] showed the benefits of providing additional text context for seq2seq models for NLP tasks that are knowledge entensive. Using BART, they were able to improve question answering tasks using dual biencoders for retrieval and training the model jointly, without the need for knowing which documents were relevant.

Adapters have become increasingly used since introduced by Houlsby et al. [4]. LoRa [5] has become a popular adapter approach, particularly for LLMs as they have grown substantially larger in recent years. The use of adapters allows modifying a models output without training the entire network, which substantially saves on VRAM memory when training. Hu et al. [5] discovered that when replacing a dot product between large vectors with an intermediate dot product of a much lower rank vector, the impacts on performance were minimal while further reducing the training parameters required.

Finally, Stickland and Murray [11] produced a multi-adapter model based on BERT, an approach that our system follows. In particular for the GLUE [12] benchmark, which is comprised of multiple datasets, they showed that simply training a task specific adapter per dataset, they could improve the average performance by 0.5 points for BERT, while only introducing 10% more parameters.

3 CRAG Dataset

CRAG is question answering dataset aimed at providing a realistic task to benchmark RAG systems as they are used in practice with a diverse set of questions, including 8 distinct question types and 5 distinct domains. Additionally, two sources of diversity which

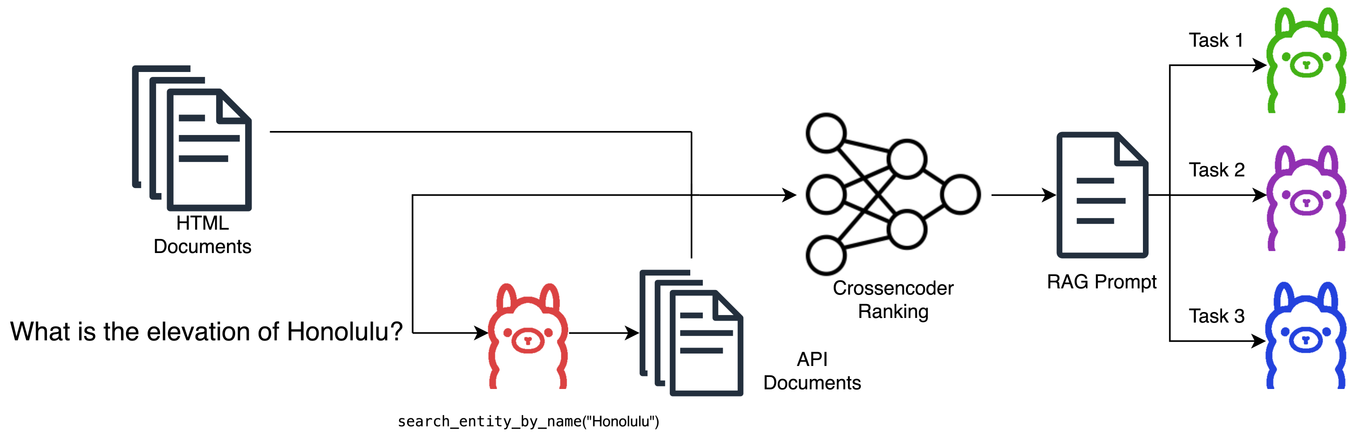

Figure 1: Pipeline for MARAGS. Each Llama in the figure represents a distinct LoRa model that can be rapidly swapped out during inference and is trained for its specific task. These tasks include the API call geneartion and the final question answering, for each task. Note, Task 1 does not include API documents in its final prompt.

pose difficulty for LLMs are how dynamic a question's answer is and the popularity of the topic of the question. As shown in the baseline systems, real-time answers pose a challenge for RAG systems and they similarly struggle when the topic of the question is less common (referred to as "torso" and "tail" questions).

3.1 Task 1

For the first task the system must process 5 candidate HTML documents for generating answers, reflecting a standard web-based RAG application. A caveat is that the 5 candidates are sampled from the top-10 relevant documents retrieved from web search. Thus, there is no guarantee that the relevant information for answering the question is actually found within the top 5 documents. This creates an interesting challenge for hallucinations, as in some cases the answer should be easily generated by the model without the context from retrieved documents.

3.2 Task 2

Task 2 reflects a more sophisticated RAG scenario, where the system is provided with the same 5 HTML documents from before, however it now has access to a knowledge graph, accessible via a REST API. The system must determine which API endpoint to call, with the correct arguments, to retrieve additional relevant context from the knowledge graph.

3.3 Task 3

Finally, Task 3 represents an extension of Task 2, where the system has access to both HTML and the knowledge graph API. However, in this task, the number of HTML documents that are to be processed is 50. This task is meant to measure both the computational efficiency of the approach as well as its ability to filter large amounts of potentially irrelevant information.

4 MARAGS Pipeline

4.1 Webpage Processing

Our webpage processing pipeline utilizes BeautifulSoup4 [10] for generating candidate segments from the HTML documents provided to the system. A common difficulty with RAG systems is determining a process for segmenting documents into smaller chunks to narrow the candidates to relevant sections and also reducing the length of the text sent to the model, given that a single document could exceed the context window of a model.

In our case, we utilize the structure of the HTML to provide the segmentation. We traverse the tree structure of the parse HTML with breath first search. Any time the length of the text contained within a node (which includes all of the text of its decedents) is less than 2000 characters, we treat that as a segment. If a leaf node is reached with greater than 2000 characters, the text is split on the white space and into as many segments are needed such that each segment is under the threshold. The segment length was determined via inspection of of HTML documents and their associated segmentation, thus future work could treat this as a hyperparameter and tune for performance.

4.2 API Call Generation

For Task 2 and 3, the knowledge graph mock API is available to be used for gathering additional information. The difficulty, however, is not only determining which API endpoint is the most appropriate, but also the proper arguments and their formatting for getting valid results from the API.

Each API endpoint was transformed to a Python function with relevant documentation describing the purpose of the endpoint, the arguments, and what the endpoint returned. Each function also has an associated formatting function, which takes the returned JSON and converts it into segmented strings. The doc strings for each Python function are used to provide additional information to help guide the model on which one is the most appropriate to use.

Table 1: A comparison of the retrieval approaches on a 500 set sample of CRAG dev set against Accuracy and CRAG Score.

Retrieval Model	Accuracy	CRAG
TF-IDF	0.2740	**-0.110**
Biencoder	0.310	-0.132
Cross-encoder	**0.328**	-0.116
Ensemble (mean rank)	0.308	-0.128

For training a model to generate API calls with associated arguments, we use LoRa [5] to train one of the several adapters we use with Llama 3 8B. For generating the target string for training, we first use Llama 3 to generate an initial prediction for the API call. Any initial prediction that successfully calls a function is retained as a target, regardless of whether or not the relevant information for the question is contained in the returned JSON. Initial predictions that fail to make a successful call are inspected and manually corrected if the correct function call is clear from the initial prediction and the question. Again, the manually modified targets are evaluated only on successfully calling an endpoint, though not validating that the relevant information is returned by the API. Any question where the target cannot be quickly modified is changed to a target of "None".

We acknowledge that this approach to annotation is not optimal, as it likely results in successful, but incorrectly selected API endpoint calls. However, manually annotating each question to determine the correct API call and validating the returned information were indeed relevant would have been too time consuming give the size of the dataset.

4.3 Candidate Ranking

For candidate ranking, we attempted 4 different candidate ranking approaches. We utilized TF-IDF, a biencoder, cross-encoder, and an ensemble of mentioned approaches (using mean rank as the ranking metric). Our TF-IDF implementation is based on Scikit-Learn [8]. The biencoder and cross-encoder are from the SentenceTransformer [9] library, specifically the "multi-qa-MiniLM-L6-cos-v1" [1] and "ms-marco-MiniLM-L-6-v2" [2] respectively.

Evaluating candidate ranking in isolation is difficult, as relevant information is not labeled, so using the system accuracy and CRAG score is the most straight forward way to compare differences in each system. However, to test the various systems, we use the base Llama-3 8B model with not adapters for each retrieval approach and use the accuracy metric to determine the best performing approach. We use accuracy instead of CRAG at this stage for ranking, as we think this is a better representation of how often the relevant information retrieved. For a test set, we randomly select 500 samples from the Task 1 dataset.

The results of this experiment are shown in Table 1. From the results, the cross-encoder is the best performing system, thus we used it for our retriever. We suspect that with proper tuning TF-IDF would and ensembling would be much more performant overall, but as mentioned running extensive experimentation is difficult as it required LLM generation to get an overall accuracy score. Using an LLM to label passages as relevant or not is a possible approach to allow for tuning of just the retriever, however we did not explore this.

Despite the cross-encoder being the most computationally expensive approach, we found it to be fast enough processing the candidates in the required 30 seconds per sample. In the case of Task 3, we found it necessary to use Python's multiprocessing library to process multiple HTMLs simultaneously to meet the runtime requirement.

[1]https://huggingface.co/sentence-transformers/multi-qa-MiniLM-L6-cos-v1
[2]https://huggingface.co/cross-encoder/ms-marco-MiniLM-L-6-v2

Table 2: A comparison across inference models test. Training on relabeled "hittable" targets hurts accuracy, but provides the best CRAG Score overall.

Model	Accuracy	Hallucination	CRAG
Llama 3 8B	0.328	0.4440	-0.1160
- LoRa	**0.398**	0.602	-0.204
- LoRa (relabeled)	0.242	**0.056**	**0.186**

4.4 Retrieval Augmented Generation

Finally, with ranked candidates, we use Llama 3 8B to augment generation with the relevant context for the question. We ran experiments with 2 different prompt structures, the primary difference between them being the ordering of the question and context.

Our initial prompt structure started with the question first, then all of the retrieved candidates prior to the Llama model response, however we noticed that often due to how much context would be provided, the model would occasionally forget the question being asked. For example a question like "What is the elevation of Honolulu?" would result in an answer of "Honolulu population is 343,421", indicating the model remember the subject of the question, but incorrectly answered with the population, rather than elevation. Our subsequent prompt structure placed the question after the context, which resolved the issue.

For training Llama 3, we trained LoRa models for each task individually. Given the penalization of hallucinations in the scoring metric, we try to take steps to mitigate further hallucinations due to finetuning, as it has been observed that finetuning LLMs can be a source of further hallucinations [3]. This likely applies to RAG systems in cases where the expected answer is not answerable given the context provided, i.e. no relevant information is given and the question is not answerable without further context, yet the model is trained to output the target answer regardless. Thus for our training setup, we first relabel the target for training samples in cases where our candidate retrieval system likely does not provide the correct information and Llama does not know the answer without that information. We use the provided dev set for training, with the 500 set sample used for retrieval comparison treated as our holdout set.

Our initial approach for determining which training samples need relabeling was has been explored previously [13]. A common and simple approach to filter/relabel incorrectly[3] labeled samples is to use a particular sample's training loss after training to the point of overfitting. High loss examples after overfitting likely indicate

[3]"Incorrectly" here simply means in the context of our retrieval system, not that the provided answer is not true.

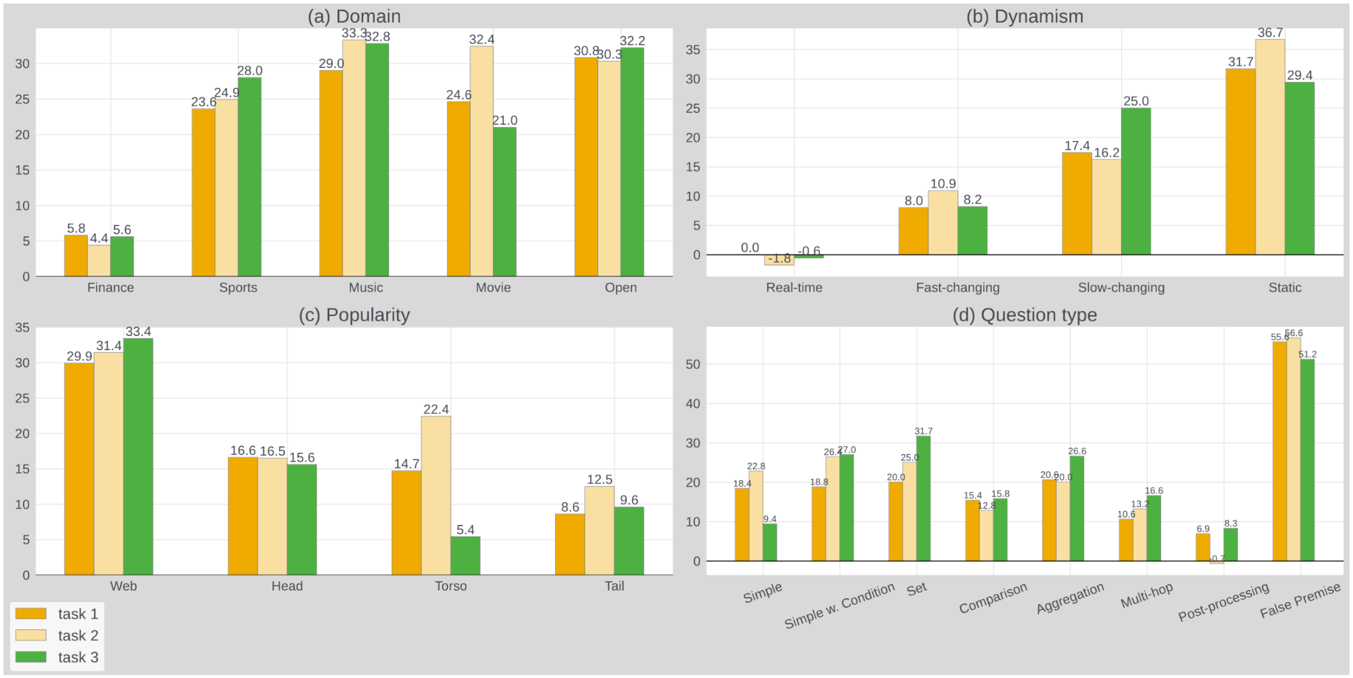

Figure 2: CRAG Score results on the test dataset calculated via manual assessment.

examples that are incorrectly labeled and thus can be filtered out. Not all samples with high loss will be incorrect labels, instead simply being hard examples, yet typically the benefit of disposing of incorrectly labeled samples outweighs ignoring difficult ones.

Initial experiments, however, indicated that this method did not work well on finance based questions. Further analysis would be required for a more definitive answer, though we suspect that this is due to the fact that the loss in hallucinated answers when dealing with numeric outputs is likely less than typical string outputs. For example, with a question "What was Apple's closing price today?", with a hypothetical correct answer "$203.51", a prediction of "$204.52" would likely not result in filtering via this method. Compare that with a question such as "Which movie won Best Picture in 1990?" with an answer of "Driving Miss Daisy" and a prediction of "Dancing with Wolves", the loss will be comparatively much higher.

We instead determine these samples by first running the system with the base Llama 3 model, with a prompt indicating to always produce a prediction for each of the 4 candidate retrieval approaches mentioned previously. We use GPT-4o to evaluate whether any of the generated answers are correct. If any are correct, the original label is retained for the training, otherwise "i don't know" is used as the target label. In the case of false premise questions, we always retain the original label as the target label. We repeat this process for each task, given that each has access to a different data sources, to generate a training dataset for each LoRa adapter.

We use the llama-recipes repository [7] for training the LoRa adapters, utilizing the default LoRa configuration. The only modifications were changing the LoRa rank from 8 to 256 and increasing the weight decay from 0.01 to 1.0.

We demonstrate the effectiveness of relabeling in Table 2. We ran 3 different answer generation setups for the 500 sample Task 1 hold out set we created. The first is an unmodified Llama 3 8B model, the second is a LoRa model using the original targets, and the final a LoRa model with relabeled targets. As shown, using the original targets provide the best accuracy, but also worsens hallucinations over the base model. While using the relabeled targets hurts accuracy, it also substantially reduces hallucinations, providing the best CRAG Score amongst the three.

5 Results

As part of the competition, a manual evaluation was conducted on user submissions. The automatic evaluation throughout was dependent on scoring via GPT-3.5 Turbo, given that correct user submissions may not have been exact matches to the expected answer. However, issues such as prompt injection still pose problems for automatic evaluation via LLMs. The results of our system across the various aspects of the dataset are shown in Figure 2. As can be seen by the results, our system suffers many of the problems the dataset is meant to expose with most RAG systems.

Similar to the baseline systems for CRAG, finance was the most challenging domain. The exact reason warrants further analysis, though contributing factors likely include LLMs known issues with number processing and simple mathematics and the fact that much online finance data is often not stored in plain text, but rather visualizations such as graphs.

Dynamism proves to be the most challenging question categorization, with model performance steadily decreasing as a question becomes more dynamic. Real-time questions prove to be the most challenging question category of any of the breakouts. Our prompt structure did not include any meta-data provided by the HTML documents, such as publish data or access date, which likely would have improved performance on dynamic questions, although likely not significantly.

The performance difference between head, torso, and tail questions appeared less substantial than our original expectations, though clearly performance drops off as popularity falls. Interestingly, Task 3 here under performs the other tasks in head, torso, and tail. We suspect that including substantially more search results includes overlapping entities / subjects, at which point conflicting information would be difficult to resolve.

Finally, the most interesting results in the question type results are the false premise category. Our system was able to achieve scores similar to the SOTA systems featured in the CRAG paper, despite obviously being a much smaller system overall. Interestingly, the false premise questions were the only type where our training setup always kept the original target label, rather than mapping the target to "i don't know".

6 Future Work

Observations we had during the competition were instances of catastrophic forgetting due to our attempts to reduce hallucinations. For instance, the question "Who has had a longer musical career, Sharika or Machine Gun Kelly?" is answerable by Llama without context, simply based on the knowledge it has of the two artists. However, after LoRa training, questions like this and others were often answered with "i don't know" in cases where the answer was not discoverable in the retrieved information. Methods to prevent this is something we are interested in pursuing in future work.

Additionally, we hope to explore larger Llama models, 70B+, in the future for this task. We were unable to get the 70B model running in the competition compute environments, so did not spend much timer looking at larger models. However, it is very likely moving to larger models would provide a substantial improvement over the 8B model.

7 Conclusion

In this work we presented MARAGS, a multi-adapter solution to the CRAG dataset. We demonstrated the effectiveness of training individual LoRa adapters for the 4 tasks in the pipeline, specifically API call generation and Task 1,2, and 3 answer generation. CRAG presents a variety of different tasks and questions to allow the tracking of progress of various methods used to build RAG systems. The penalization of hallucinations is a unique and important feature as future AI systems become increasingly common throughout society, as hallucinations hurt user trust in these systems. We discussed our methods for reducing these hallucinations, but they are not without cost, as in some cases the model fails to output previously known knowledge. Clearly the importance of balancing these two factors is a key to leveraging LLMs to their full potential, while also improving user trust.

Acknowledgments

We are grateful to the KDD Cup organizers, Meta, and AIcrowd for all the work that goes into hosting a successful competition.

References

[1] [n. d.]. Lawyer Used ChatGPT In Court—And Cited Fake Cases. A Judge Is Considering Sanctions. https://www.forbes.com/sites/mollybohannon/2023/06/08/lawyer-used-chatgpt-in-court-and-cited-fake-cases-a-judge-is-considering-sanctions/. Accessed: 2024-08-08.

[2] 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, Damon 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, 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, 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. arXiv:2407.21783 [cs.AI] https://arxiv.org/abs/2407.21783

[3] Zorik Gekhman, Gal Yona, Roee Aharoni, Matan Eyal, Amir Feder, Roi Reichart, and Jonathan Herzig. 2024. Does Fine-Tuning LLMs on New Knowledge Encourage Hallucinations? arXiv:2405.05904 [cs.CL] https://arxiv.org/abs/2405.05904

[4] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-Efficient Transfer Learning for NLP. *CoRR* abs/1902.00751 (2019). arXiv:1902.00751 http://arxiv.org/abs/1902.00751

[5] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-Rank Adaptation of Large Language Models. In *International Conference on Learning Representations*. https://openreview.net/forum?id=nZeVKeeFYf9

[6] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Proceedings of the 34th International Conference on Neural Information Processing Systems* (Vancouver, BC, Canada) *(NIPS '20)*. Curran Associates Inc., Red Hook, NY, USA, Article 793, 16 pages.

[7] Meta. 2024. llama-recipes. https://github.com/meta-llama/llama-recipes.

[8] 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 (2011), 2825–2830.

[9] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics. http://arxiv.org/abs/1908.10084

[10] Leonard Richardson. 2007. Beautiful soup documentation. *April* (2007). https://beautiful-soup-4.readthedocs.io/en/latest/

[11] Asa Cooper Stickland and Iain Murray. 2019. BERT and PALs: Projected Attention Layers for Efficient Adaptation in Multi-Task Learning. *CoRR* abs/1902.02671 (2019). arXiv:1902.02671 http://arxiv.org/abs/1902.02671

[12] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. *CoRR* abs/1804.07461 (2018). arXiv:1804.07461 http://arxiv.org/abs/1804.07461

[13] Yisen Wang, Weiyang Liu, Xingjun Ma, James Bailey, Hongyuan Zha, Le Song, and Shu-Tao Xia. 2018. Iterative Learning with Open-set Noisy Labels. In *CVPR*.

[14] Xiao Yang, Kai Sun, Hao Xin, Yushi Sun, Nikita Bhalla, Xiangsen Chen, Sajal Choudhary, Rongze Daniel Gui, Ziran Will Jiang, Ziyu Jiang, Lingkun Kong, Brian Moran, Jiaqi Wang, Yifan Ethan Xu, An Yan, Chenyu Yang, Eting Yuan, Hanwen Zha, Nan Tang, Lei Chen, Nicolas Scheffer, Yue Liu, Nirav Shah, Rakesh Wanga, Anuj Kumar, Wen tau Yih, and Xin Luna Dong. 2024. CRAG – Comprehensive RAG Benchmark. arXiv:2406.04744 [cs.CL] https://arxiv.org/abs/2406.04744