Easy-to-Hard Generalization in Math Problems

Zhirui Wang, Katie Guo, Alicia Sui
{zhiruiw, katieguo, jialusui}@andrew.cmu.edu

1 Introduction

LLMs are designed to answer general questions, but they often struggle immensely to answer domain-specific questions. To improve model accuracy, we can finetune the pre-trained model on a smaller, specialized dataset. But the question remains: do we *need* to use all this data to achieve decent accuracy? It turns out that LLMs can generalize remarkably well for general-knowledge trivia and STEM questions – that is, even when trained exclusively on "easy" data, they are able to answer "hard" questions. Using a pre-trained language model (Llama-2-7B and Llemma-7B) and finetuning it on the MATH dataset using LoRA, we hope to investigate easy-to-hard generalization in competition math problems by comparing accuracy after finetuning on the easy data versus the full data. We expect that finetuning on a subset will yield worse, but still comparable results to the full data, and that both will perform better than zero-shot QA.

2 Dataset/Task

The MATH dataset contains 12,500 challenging competition math problems (7,500 training and 5,000 test) and their associated answers, complete with a thorough, step-by–step explanation of the work needed to come up with the solution. Each question is also tagged with its subject (Prealgebra, Algebra, Number Theory, Counting and Probability, Geometry, Intermediate Algebra, and Precalculus) and a difficulty level (1 to 5). We will use this dataset to finetune a pre-trained language model to improve the base model's accuracy when answering math problems.

As proposed by Hendrycks et al. (2021), the original creators of MATH, we calculate accuracy based only on the output within \boxed{}, forcing the model to adhere to a specific format. We are interested in whether we can achieve accuracy comparable to fine-tuning using the entire dataset even when only using some subset of the data. Specifically, we ask the question: will fine-tuning exclusively on the "easy" questions (difficulty levels 1 to 3) will allow us to achieve comparable accuracy as fine-tuning on the entire dataset? Because training data is hard to curate, we want to guarantee that it is effective. Easier questions are usually easier to obtain, so if we can show that a model can generalize from easy to hard questions, we can focus exclusively on obtaining easy questions. Even if the accuracy is not as high when trained only on easy questions, the benefits of simpler curation of data, allowing us to train on a larger dataset may offset these costs.

We will use two metrics to measure the performance of our models. Firstly, we will have a raw test accuracy, the proportion of questions the model answers correctly. Secondly, we care about Supervision Gap Recovered (SGR) (Hase et al., 2024) defined as:

$$\frac{\text{Easy - Unsupervised}}{\text{Hard - Unsupervised}}$$

where Easy, Hard, and Unsupervised are test accuracy on hard test data when fine-tuned on easy data, easy and hard data, and no data (zero-shot), respectively.

3 Related Work

Curriculum Learning. Inspired by the way humans learn, Bengio et. al (2009) first formalized curriculum learning as an easy-to-hard learning strategy where the data samples used progressively increase in difficulty over multiple iterations. Curriculum learning strategies have been replicated in various other domains including computer vision (Chen and Gupta, 2015; Li et al., 2017), machine translation (Zhang et al., 2018; Wang et al., 2020), and even mathematics (Zhao et al., 2015). But whereas curriculum learning investigates the optimal ordering of data on model performance, generalization relates to how well models generalize on hard data given varying difficulty levels of the training data.

Generalization. Past work has shown that models struggle with compositional generalization, failing to combine reasoning steps in a novel way (Lake and Baroni, 2018; Bogin et al., 2022), though further research has been done to mitigate the issue (Xu et al., 2022; Zhou et al., 2023). In a similar vein, training with easy data as opposed to hard data usually results in worse performance (Swayamdipta et al., 2020). Furthermore, Fu et al. (2022) showed that prompting with easier examples yields worse results than prompting with complex examples.

Despite this, recently, Hase et al. (2024) have demonstrated that current language models can generalize surprisingly well from easy to hard data. Given training datasets with difficulty levels of College, High School, 8th Grade, and 3rd Grade, the model performed similarly on college-level STEM questions across all training datasets. We did not achieve comparable results on MATH, but we concluded that there is definitely potential for easy-to-hard generalization on mathematical reasoning.

4 Methods

We will use two models: Llama-2-7B, a general purpose language model, and Llemma-7B, a language model for mathematics. We treat the model's zero-shot performance on the MATH dataset as the baseline, and our main methods all involve fine-tuning the model using LoRA without an instruction fine-tuning prompt for 3 epochs on the 7,500 training examples in the MATH dataset. Because of time and computation constraints, testing is done on 10% of the test

examples in the test dataset. Of the 500 examples, 40 are level 1, 96 are level 2, 117 are level 3, 122 are level 4, and 125 are level 5. Each training epoch takes 0.5 hours to complete on one single A100 GPU.

To investigate the pre-trained model's easy-to-hard generalization capabilities on the MATH dataset, we compare the model's performance when fine-tuned exclusively on an easy subset of the training dataset (difficulty level 1-3), with the model's performance when fine-tuned on the full training data.

5 Experiments

Baseline Evaluations (Zero-Shot Accuracy)

Method	**4**	**5**	**Overall Hard**
Llama-2-7B	0	0	0
Llemma-7B	4.1	3.2	3.6

When directly evaluating the MATH test set with Llama-2-7B, the zero-shot accuracy on the hard problems is 0. There are a few reasons for this:

1. Mathematical reasoning is quite hard, especially since this is not a particularly large model to begin with;
2. We require the answer to be formatted using LaTeX. Specifically, the answer must be \boxed{}, and the model was not pre-trained on the formatting.

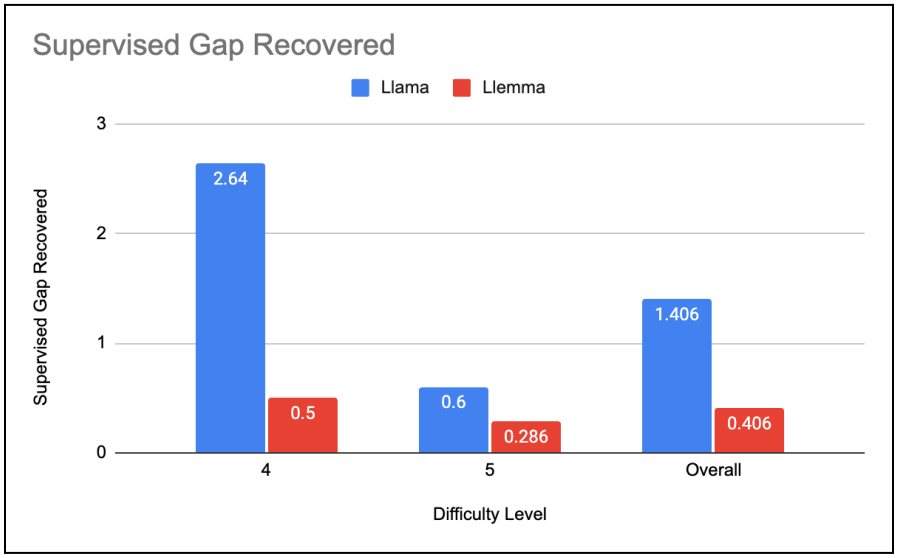
In contrast, Llemma-7B, a mathematical language model, was able to answer some of the hard questions zero-shot because it has been trained to reason mathematically and also output LaTeX. That being said, its performance is still quite poor.

Measurements (Fine-Tuned Accuracy)

Method (Model-Train)	**4**	**5**	**Overall Hard**
Llama-All	2.5	4	3.2
Llama-Easy	6.6	2.4	4.5
Llemma-All	10.7	8.8	9.7
Llemma-Easy	7.4	4.8	6.1

Unexpectedly, Llama-2-7B performed better overall on the hard questions when it was fine-tuned exclusively on the easy questions.

Supervised Gap Recovered



Like in the original paper, we were expecting SGR to be about 0.7, but we find that the SGR for Llama is greater than one, meaning that Llama-Easy performed better on the hard questions than Llama-All. It is unexpected that the model with not only easier training data, but also less training data, would outperform the one that uses all the training data, but we speculate that using just 500 examples to evaluate accuracy was too few. When evaluated on more examples, we expect that the accuracy of Llama-All would increase while the accuracy of Llama-Easy would decrease. Preliminary experiments of testing on 293 exclusively hard examples seem to support this conclusion (see Appendix A).

The results for Llemma align much better with our expectations. However, the SGR is still noticeably lower than expected. There are a few potential reasons for this:

1. Like mentioned before, the test set is too small. Evaluating on a larger dataset could "stabilize" our accuracies. Evaluating on 293 exclusively hard examples seems to support this conclusion (see Appendix A).
2. There is not enough training data. 7,500 examples is not a large dataset to begin with, and because we don't use the hard data for training, the number of examples is further reduced. A more accurate comparison would be the train on 7,500 easy examples versus 7,500 mixed examples.
3. Llemma is a mathematical language model. We may already be approaching the upper limit of what this model is capable of in terms of mathematical reasoning. Because its zero-shot accuracy is already quite okay, SGR will naturally be lower.

6 Code Overview

1. math/modeling/tune_gpt.py: line 75 - 165
    a. Load previously fine-tuned model
    b. Load Llama or Llemma model and set up LoRA training
2. math/modeling/tune_gpt.py: line 420 - 423
    a. Configure various training settings
3. math/modeling/tune_gpt.py: line 327 - 335
    a. Get Llama or Llemma tokenizer
4. math/modeling/eval_math_gpt.py: line 102 - 110:
    a. Load Llama or Llemma tokenizer
5. math/modeling/eval_math_gpt.py: line 147 - 200:
    a. Load fine-tuned Llama or Llemma model
    b. Merge LoRA weights
6. math/modeling/eval_math_gpt.py: line 451 - 453:
    a. Configure evaluation setting
7. math/data/MATH/process_data.py
    a. Process and separate train and test data into easy/hard subset

7 Timeline

Task	**Time Spent (hours)**
Researching Datasets & Tasks	3
Reading/Annotating Papers	5
Code ● Understanding paper ● Compiling and run existing GPT-2 ● LoRA implementation ● GPT-2 experiments ● Llama Compiling/Merging with LoRA ● Llama/Llemma Experiments	50 ● 4 ● 1 ● 2 ● 10 ● 3 ● 30
Poster	4
Reports ● Project Proposal ● Midway Report ● Final Executive Summary	11.5 ● 3 ● 5.5 ● 3

8 Research Log

Choosing a task and dataset initially was unexpectedly difficult because we wanted to do something interesting, but also something that we were certain was actually feasible. Looking back at our project proposal, our main goal to investigate easy-to-hard generalization in mathematical reasoning stayed consistent, though our proposed experiments changed to reflect our time and computation constraints.

The main difficulty arose when a few days before the midway project report was due, our baseline accuracies using GPT-2-Large were zero for every single difficulty level. This was already slightly concerning, but we figured that if fine-tuning accuracies were non-zero, we would still have some results. However, the accuracies were zero all across the board even after fine-tuning for 5 epochs on the entire training set. After this, we considered a few options:

1. Choose a simpler dataset and continue studying generalization.
2. Pivot to a new topic completely.
3. Keep the zero accuracy and explain the shortcomings of the project.

We ended up going with none of the options, instead opting to use more powerful models (Llama-2-7B and Llemma-7B). This allowed us to achieve much better results, even including non-zero zero-shot accuracy on the easier questions.

However, we now had incredibly limited time to conduct all of our experiments. Evaluation also took much longer than expected, about 1 minute per example even when run on A100, so it was simply infeasible to test on all examples as we had initially planned to do. Instead, we tested our models on 10% of the examples (500 if testing on all, 293 if on hard only), which reduced our evaluation time significantly.

9 Conclusion

We study the problem of easy-to-hard generalization applied to mathematical reasoning and find that although our experimental results do not fully align with the previous literature, training on easy math problems has the potential to do as well as training on hard math problems. When hard data is costly to collect, we suggest that collecting and training on easy data is preferable because it yields reasonable results without the difficulty of curating difficult training examples.

References

1. Yoshua Bengio, Jerome Louradour, Ronan Collobert, & Jason Weston. (2009). Curriculum learning. In Proceedings of the 26th Annual International Conference on Machine Learning (pp. 41-48).
2. Ben Bogin, Shivanshu Gupta, & Jonathan Berant. (2022). Unobserved Local Structures Make Compositional Generalization Hard.
3. Xinlei Chen, & Abhinav Gupta. (2015). Webly Supervised Learning of Convolutional Networks.
4. Yao Fu, Hao Peng, Ashish Sabharwal, Peter Clark, & Tushar Khot. (2023). Complexity-Based Prompting for Multi-Step Reasoning.
5. Peter Hase, Mohit Bansal, Peter Clark, & Sarah Wiegreffe. (2024). The Unreasonable Effectiveness of Easy Training Data for Hard Tasks.
6. Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, & Jacob Steinhardt. (2021). Measuring Mathematical Problem Solving With the MATH Dataset.
7. Siyang Li, Xiangxin Zhu, Qin Huang, Hao Xu, & C. -C. Jay Kuo. (2017). Multiple Instance Curriculum Learning for Weakly Supervised Object Detection.
8. Brenden M. Lake, & Marco Baroni. (2018). Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks.
9. Swabha Swayamdipta, Roy Schwartz, Nicholas Lourie, Yizhong Wang, Hannaneh Hajishirzi, Noah A. Smith, & Yejin Choi. (2020). Dataset Cartography: Mapping and Diagnosing Datasets with Training Dynamics.
10. Wei Wang, Ye Tian, Jiquan Ngiam, Yinfei Yang, Isaac Caswell, & Zarana Parekh. (2020). Learning a Multi-Domain Curriculum for Neural Machine Translation.
11. Zhenlin Xu, Marc Niethammer, & Colin Raffel. (2022). Compositional Generalization in Unsupervised Compositional Representation Learning: A Study on Disentanglement and Emergent Language.
12. Xuan Zhang, Gaurav Kumar, Huda Khayrallah, Kenton Murray, Jeremy Gwinnup, Marianna J Martindale, Paul McNamee, Kevin Duh, & Marine Carpuat. (2018). An Empirical Exploration of Curriculum Learning for Neural Machine Translation.
13. Qian Zhao, Deyu Meng, Lu Jiang, Qi Xie, Zongben Xu, & Alexander Hauptmann. (2015). Self-Paced Learning for Matrix Factorization. In Proceedings of the AAAI Conference on Artificial Intelligence, 29(1).
14. Xiang Zhou, Yichen Jiang, & Mohit Bansal. (2023). Data Factors for Better Compositional Generalization.

Appendix

A Full Experiment Results

We conducted more experiments, including evaluations on easy questions from the test dataset, but not all are completely relevant to the paper. The results are as follows:

Zero-Shot Accuracies

Model	1	2	3	4	5	Overall
Llama-2-7B	5.0	6.3	3.4	0	0	2.4
Llemma-7B	37.5	22.9	22.2	4.1	3.2	14.6

5 Epoch Fine-Tuning

Method (Model-Train-Test)	1	2	3	4	5	Overall
Llama-All-All	15.0	13.5	4.3	3.3	0.8	5.8

We were deciding between fine-tuning for 5 epochs versus 3 epochs and found that the accuracy was actually higher for 3 epochs, so for all later experiments, we fine-tuned for 3 epochs only.

Tested on Hard Levels

Method (Model-Train)	4	5	Overall
Llama-Easy	3.2	3.1	3.2
Llemma-Easy	9.7	6.2	7.9

These models were tested on 10% of the *hard* questions in the test dataset, for a total of 293 questions.

Tested on All Levels

Method (Model-Train)	1	2	3	4	5	Overall
Llama-All	15.0	13.5	4.3	3.3	0.8	6.4

Llama-Easy	10.0	12.5	6.8	2.5	4.0	7.0
Llemma-All	50.0	26.0	22.2	10.7	8.8	19.0
Llemma-Easy	40.0	28.1	25.6	7.4	4.8	17.6