Self Reward Scaling

Stanford CS224N {Custom} Project

Arjun Chandran
Department of Computer Science
Stanford University
arjuncha@stanford.edu

Abstract

With the advent of reasonably good open source LLM models there has been significant research into how to optimally fine tune these models to specific tasks and how we can develop training protocols that boost the performance of these models in cheap resource poor settings. Recently, self reward methods where a pretrained LLM augments its own dataset with synthetically generated data and rewards has shown promising results for improving model performance. These findings come at a time where high quality novel datasets have become increasingly difficult to find/expensive to procure so finding ways to generate high quality synthetic data has become more important now than ever. MetaAI has shown in recent reports that for very large language models, synthetic data can close gaps in the data space of a dataset thereby effectively augmenting it and improving the performance of models trained with said data. They also suggest that by using the same model for data generation and evaluation models can leverage the benefits of multi task learning thereby improving there performance on both response creation and reward generation tasks. In this paper we explore how these results scale to smaller models and show that for most small models self reward methods are likely not a good fit due to the model's inability to understand and generate appropriate rewards.

1 Key Information to include

- Mentor: Ryan Li
- External Collaborators (if you have any): None
- Sharing project: None

2 Introduction

Large language models serve as one of the primary backbones of modern AI. Their ability to converse in an all too human and seemingly coherent way has captured the human imagination and lead to the demand for increasingly useful agents that can be deployed at scale for cheap. Currently, large language model development is prohibitively expensive with many of the best models being created by big tech companies with enormous budgets. To improve the adoption of this technology in a wider array of settings companies like huggingface have emerged creating a market of open source models that people can then fine tune to specific tasks. This has lead to a host of fine tuning methods starting with supervised fine tuning where a model is directly trained on new data to reinforcement learning based methods like DPO, CPO, and ORPO that improve on a models ability to learn during the fine-tuning stage of training over standard SFT.

Improvements on fine tuning methods are however one component in generating production ready LLM systems. Another major component driving LLM development is developing new AI driven constitution and reward methods to allow LLMs to self regulate and learn on their own with limited

human intervention. This serves two primary purposes. First, human intervention whether for data generation or for red teaming LLM behavior for safety is usually time consuming and costly. Secondly, as A.I models improve human generated data limits the potential performance an A.I. model can achieve to within the bounds of what is achievable by a human so developing systems that can synthetically generate data, learn from that data, and improve their performance/ability to generate even better data is imperative to developing super human levels of artificial intelligence. In an attempt to accomplish this, constitutional AI and RLAIF methods have been developed to utilize LLMs as reward models capable of self evaluating [1]. The key finding in these methods is that as LLM performance improves their ability to judge and reward their own actions improves, however we still don't understand what the minimum level of competency required by an LLM is before it is eligible for training via constitutional AI and RLAIF methods. Identifying model eligibility is a imperative to allowing for the research, development and productionalization of LLMs in organizations that do not have the resources typically found at wealthier firms.

With this in mind, one of the fine tuning methods that has come out recently from MetaAI with promising results has been utilizing self reward as part of a model's training paradigm. Essentially, a LLM is initially fine-tuned using supervised fine-tuning before subsequently being asked to generate synthetic data which it ranks and then trains on. MetaAI utilized LLama 70B for this task and they saw preliminary model performance improvement. This method is particularly interesting when looking at how it applies to smaller models for two reasons. First, at the core of RLAIF/Constitutional AI methods is the ability for an LLM to understand reward/constitution prompting and provide appropriate feedback similar to how for self reward the same model that generates responses must be capable of understanding reward prompting and generate a parse-able reward that can be used to rank synthetically generated response data. Second, self reward methods offer an interesting perspective on when a LLM might have a good enough model of the world through language to understand what is being asked of it and improve itself. In an attempt to look into these questions, this paper looks at how self reward can be applied to models with 7B parameters or less and hopes to provide preliminary findings that show

- Smaller models show moderate improvement but may require additional fine tuning on real data before they are ready to use self reward
- A key metric for whether model is capable of using self reward is whether it is able to understand reward prompting and generate appropriate reward metrics.
- At the 7B parameter mark for Llama models LORA dimensionality reduction can be used to reduce computational requirements during training while still allowing for performance gains via self reward training.

3 Related Work

In this section we aim to contextualize our approach by discussing related work in detail. The key areas of relevance here are self reward methods, constitutional AI methods, and LLM as a judge methods. To start we based the experiments of this paper off of MetaAI's self reward method. Their proposed method used a base pre-trained model (LLAMA 70B for this paper) along with a seed data set of human responses. The data was then split into two data types: instruction fine tuning data (IFT) containing instruction prompts and results and evaluation fine tuning data (EFT) which contains response evaluation prompts and and valid evaluations that score and justify why a response deserves a score. From here the authors had the model generate N new responses to a subset of prompts from their dataset. The model was then asked to rank it's N responses to generate a preference set. Once self prompting is complete a new AI feedback training data (AIFT) set is created by pairing the self generated prompt x_i with the the winning y_i^w and losing y_i^l responses to the self generated prompt. The model is then trained using iterative DPO with the training process and model generation sequence (model M_i generated on the ith training iteration) in the paper summarized below.

- M_0: Base pre-trained LLM with no additional training
- M_1: M_0 is initialized and fine tuned using human generated seed IFT and EFT data (paper used Open Assistant dataset data here).

[1] `https://arxiv.org/pdf/2212.08073`

- M_2: M_1 is fine tuned using the first iteration's generated AIFT(M_1) dataset + IFT data
- M_3: M_2 is fine tuned using the second iteration's generated AIFT(M_2) dataset + IFT data

The synthetic data creation process builds heavily off of LLM as a judge methods [2] which proposes three LLM as a judge techniques. These include pairwise comparison where an LLM is presented with two different responses and asked to pick the better option, single answer grading where an LLM is asked to score individual responses and reference guided grading where a reference solution is provided to help guide the LLMs judgement. The authors of this particular paper found that these techniques offer a higher degree of scalability and explain-ability since the judging model can provide an explanation for why it provides the score it does. Additionally, the three major limitations of these methods found by the authors included positional bias (i.e. always preferring the first answer presented over the second one), verbosity bias, and a bias towards answers generated by the judging model. In the context of self reward, verbosity bias is relatively easy to control for since you can define the max response length which was an approach taken for this study's experiments. However something to note which is particularly important is how model's prefer the responses they themselves generate. In the context of the previously described self reward method this is less of a concern since reward scores provide a relative ranking between two different responses both generated by the same model however it does make it more difficult to compare synthetically generated data to human generated data since models tend to prefer their own responses.

Additionally, for the general training process of the self reward method, constitutional AI methods were used as a template to help define the various fine tuning phases and reward prompting for refined responses. Specifically, constitutional AI methods prompt for improved helpfulness and harmlessness via chain of thought prompting. This prompt style was specifically used as part of the self reward method to both improve model scoring capabilities as well as provide visibility into how the model is thinking about the scoring process and what the rational for scoring is. This is of particular important because as training processes move from using human data/feedback to AI generated data and feedback visibility becomes more important to allow for explainable AI models.

4 Approach

The general approach taken was to re-implement self reward training to try out model's of different sizes. To accomplish this a training script was developed which utilized Lora configurations to reduce the computational overhead of training TinyStories-1M, Phi 1, TinyLlama 1.1, and Llama2 7b. The core approach for all experiments was to start with supervised fine tuning on the DPO data. Once this model is fine tuned the model was fed a set of prompts and responses were sampled twice from the model (per prompt) given this prompt data. The model was then prompted to assign a score/reward with each response and this new synthetic data is then appended to our dataset for a round of DPO training. One of the major challenges with working with smaller models was ensuring that they were actually capable of generating sensible rewards. Models that were found to generate nonsense reward responses more than ten times in a row (after considerable prompt tuning, Lora config adjustment, and additional supervised training) were automatically disqualified from proceeding further in the training process as they were deemed unfit to create constructive synthetic data. Finally, for models that were found to be capable of generating consistent rewards 3-5 rounds of training (supervised-finetuning, self-reward based DPO, self reward based DPO) for n epochs per round were performed before our final model is returned.

5 Experiments

5.1 Data

For supervised fine tuning and DPO baselining Intel/orca_dpo_pairs was used [3]. With approximately 12.8k entries this served as seed data from which subsequent training iterations were able to generate synthetic data.

[2] https://arxiv.org/pdf/2306.05685

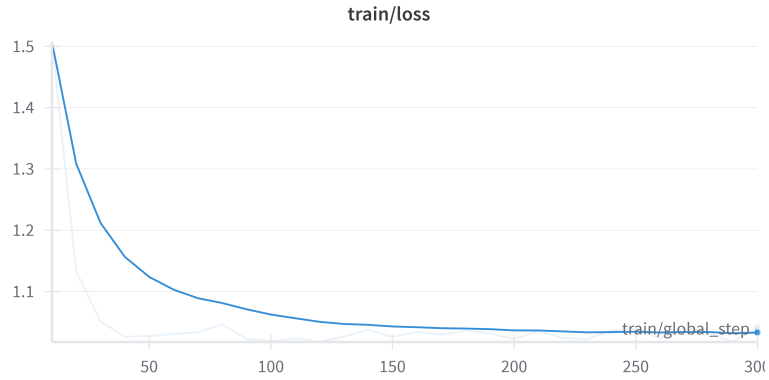
[3] https://huggingface.co/datasets/Intel/orca_dpo_pairs

5.2 Evaluation method

Models were trained and evaluated based on training performance as well as sampling synthetically generated data. Finally, eligible models capable of running the full self reward training protocol were run on MT-Bench with GPT 4 as a judge to view performance.

5.3 Experimental details

Initially, TinyStories-1M, Phi 1, TinyLlama 1.1, and Llama2 7b were each individually trained for 300 update steps in a supervised fashion. SFT training started out with a loss of $2e-4$ with a cosine learning rate scheduler. Training loss was manually inspected to ensure convergence as is exemplified by the below figure for Llama 7B's SFT training.



Once training was complete synthetic data was generated and manually inspected for both the quality of the generated responses to prompting as well as the quality of the reward that was generated. Regex filtering was used to extract score data from reward evaluations and model's that failed were manually evaluated to identify whether their failure to complete the synthetic data generation process was due to poor filtering or if the reward responses generated were nonsense. For data generation the following prompt was used for all models

```
Review the user's question and the corresponding response using the
additive 5-point scoring system described below.
Points are accumulated based on the satisfaction
of each criterion:
- Add 1 point for relevancy.
- Add another point if the response is comprehensive.
- Award a third point if the response is useful.
- Grant a fourth point if the response is clearly written.
- Bestow a fifth point for a response that is high-quality, engaging,
and insightful.
User: {{ prompt }}
Response: {{ response }}
After examining the user's instruction and the response:
- Briefly justify your total score, up to 100 words.
- Conclude with the score using the format: ''Score: <total points>''
```

All models initially failed to adequately generate synthetic data with each model failing several times more than 10 consecutive times so an additional DPO training phase with the Orca data set was performed for 6 epochs. Data generation turned out to be quite time intensive so only 20 synthetic data points per DPO iteration (5 DPO iterations total with three epochs per iteration) were generated. This prompt was fine tuned to reduce the length and make it more palatable for smaller models. Finally the SFT baseline model, DPO finetuned on orca data model, and self reward model were evaluated via MT-Bench with Gpt 4 as judge. All DPO training used a sigmoid loss function and both with a learning rate of $5e-5$. Additionally during both SFT and DPO training phases (regardless of

whether self reward generated data or real data was being used) all models were rank decomposed on the k_proj and q_proj layers to a rank of 16.

5.4 Results

Model	MT-Bench Score
SFT Baseline Llama 7B	3.5
DPO FineTuned Llama 7B (3 iterations with Orca data)	6.32
Self Reward Trained Llama 7B (3 Self Reward Training iterations)	6.36
Self Reward Trained Llama 7B (5 Self Reward Training iterations)	6.39

Overall we see that self reward training for Llama models at the 7B parameter mark does improve performance even with a relatively trivial synthetic dataset size of 20 synthetically generated preference pairs. The rate of improvement is relatively surprising given the small amount of synthetic data created and this may have to do with a certain level of synergy between learning to be a good reward model and learning to respond well.

6 Analysis

Through the training process one of the most important observations was that TinyStories-1M, Phi 1, and TinyLlama 1.1 were unable to generate appropriate self evaluation responses and ended up failing out of the training process due to this. All model's had difficulty with this post fine tuning with the aforementioned models outputting reward responses like the below text quite frequently. The below example specifically being for Phi 1.

```
Reward Response Where Response Score is Expected:
 restaurant_tips += "Stay focused and
 open to new experiences. Don't miss out on any good
```

Responses like this were often times correlated with nonsensical responses to the prompt with longer nonsensical responses tending to confuse most model's. Getting appropriate rewards became more manageable once Llama 7B was used so it seems like model's around the 7B parameter mark may be good starting candidates. For this particular monitor relaxing the failure limit 3 to 10 for how many consecutive reward generation failures were tolerable along with truncating the responses to a max size and fine tuning via additional DPO iterations served to allow the model to generate reasonable rewards. While Llama 7B was training via self reward it was also observed that as the model continued training, its ability to identify good and bad responses improved perceptibly which provides some evidence that improved reward modeling and improved performance are correlated. This can be seen by comparing the response to the prompt "Give three tips for staying healthy" from self reward training iteration 1 to the response from iteration 4.

```
 Response 1 From Self Reward Training Iteration 1 for Llama 7B:
hopefully you will find them helpful.
1. Stay Hydrated: Drinking enough water is essential for
maintaining good health. Aim to drink at least eight glasses
of water a day, and more if you are physically active or live
in a hot climate. Dehydration can lead to a range of health problems,
including headaches, fatigue, and digestive issues.
2. Eat a Balanced Diet: A healthy diet provides the nutrients
and energy your body needs to function properly.
Aim to eat a variety of fruits, vegetables,


 Response 1 From Self Reward Training Iteration 4 for Llama 7B:
hopefully, you will find these tips helpful in maintaining your health
and well-being. 1. Drink plenty of water: Staying hydrated is crucial
for overall health, as water helps to flush toxins out of your body,
regulate body temperature, and support digestive health. Aim to drink
```

```
at least eight glasses of water per day, and consider increasing this
amount if you are physically active or live in a hot climate. 2. Eat
a balanced diet of fruits, vegetables: A healthy diet provides your
body with the nutrients it needs to function properly
```

Both responses are reasonably good but the second one seems to be slightly better formatted and worded.

7 Conclusion

Overall, these results show that self reward could improve the performance of smaller models (even when using computational reduction techniques like LORA dimensionality reduction) once sufficient training on real data is done by those models. Additionally, through this papers experimentation it has been identified that a good test for whether a model is eligible for self reward is its ability to generate reward without failing more then ten consecutive times. Looking at the future it would be interesting to see if smaller models could be used by increasing the amount of fine tuning on real data before self reward is applied.

8 Ethics Statement

Utilizing synthetic data to create AI model's offers an attractive alternative to relying on costly, time consuming, and potentially low quality synthetic data however it does pose some interesting ethical questions. One such ethical question that needs to be evaluated is how intrinsic biases in initial fine tuning data/pretrained models can affect synthetic data creation and learning. Because synthetic data does not allow a training model to access new information about the world it may serve as a way for models to reinforce their own biases magnifying previously small potentially undetectable biases intrinsic to our models/data. Additionally, identifying the specifics of how self rewarding models may fail is more important than ever because new training paradigms like this may have failure modes that are atypical of other training methods which can pose a risk when model's developed using these methods are put into critical/high impact positions. Finally, developing ways to reduce the barrier of entry to use methods like this is also ethically imperative. AI is a powerful tool in any ones hands and limiting its use to only the most wealthy and powerful labs/companies allows for the concentration of power within the hands of a few, potentially at the detriment of the many. Developing cost effective methods that allow for a healthy open source community with strong ethical guidelines is imperative to both driving new innovation as well as distributing the power intrinsic to this high impact technology.

References

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Sainbayar Sukhbaatar, Jing Xu, Jason Weston. (2024). *Self-Rewarding Language Models*. arXiv:2401.10020 [cs.CL]. Retrieved from `https://arxiv.org/abs/2401.10020`

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, Chelsea Finn. (2023). *Direct Preference Optimization: Your Language Model is Secretly a Reward Model*. arXiv:2305.18290 [cs.CL]. Retrieved from `https://arxiv.org/abs/2305.18290`

Haeyong Kang, Jaehong Yoon, Sultan Rizky Madjid, Sung Ju Hwang, Chang D. Yoo. (2023). *Forget-free Continual Learning with Soft-Winning SubNetworks*. arXiv:2303.14962 [cs.LG]. Retrieved from `https://arxiv.org/abs/2303.14962`

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez and Ion Stoica. Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena, 2023; arXiv:2306.05685.

Amirhossein Farzam, Shashank Shekhar, Isaac Mehlhaff and Marco Morucci. Multi-Task Learning Improves Performance In Deep Argument Mining Models, 2023; arXiv:2307.01401.