LEARNING THROUGH CONDITIONING ON NATURAL LANGUAGE FEEDBACK

Anonymous authors
Paper under double-blind review

ABSTRACT

In this paper we explore the simple idea of teaching models by allowing them to condition their answers on natural language feedback. Motivated by the idea that natural language interactions provide a targeted, flexible, and level-appropriate reward signal, we study the ability of small instruction-tuned models to leverage feedback from a larger frontier model. We find while the frontier model provides generally high quality feedback, especially smaller models can struggle to use this due to noise in their generative output. After incorporating techniques like negative sampling, we find that models trained on these feedback-conditioned responses can perform similarly to those trained directly on teacher responses. We explore training using supervised finetuning and preference learning algorithms over a broad set of tasks including Big-Bench Hard. These findings are broadly applicable and our methods rely only on the ability of models to give and receive linguistic feedback. As such, they contribute to a growing body of work exploring how to best utilise the linguistic capabilities of language models for human-like instructive learning.

1 INTRODUCTION

One essential aspect of human learning is our ability to provide and leverage the feedback of others. This plays a pivotal role in developmental tasks such as language acquisition (Tomasello, 1992), and is a major paradigm in human instruction. Feedback similarly plays a key role in artificial intelligence in self-supervised losses, hand-labelled classification tasks, and preference learning algorithms that permeate the literature. In this work, we seek to investigate whether instruction-tuned language models that can provide and learn from feedback, enable us to teach language models in the same way that humans are able to: through natural language. Assuming that pretrained language models have these capabilities, our motivation is threefold: firstly we want to use the *specificity* of language to improve the informational content and sample efficiency of feedback; secondly we want to use the fact that language feedback can be *personalised* to the current understanding of the learner agent; finally natural language acts as a *flexible* interaction pattern, enabling learning directly from humans, or from the most powerful foundation models. Indeed we see this work as an initial step in establishing the viability of feedback conditioning as a learning mechanism for decision making agents more generally, given that it is modality agnostic provided that teacher and student possess sufficient linguistic capabilities.

Training Technique	Requires ...	Personalised	Precise
Learning from Rewards	Dense Reward Function	✗	?
Imitation Learning	Expert Trajectories	✗	✓
Feedback Conditioning	Natural Language Feedback	✓	✓

Table 1: Comparison of Training Techniques for Learning in Decision Making environments.

In general, there are two broad approaches to training models on sequential decision making tasks: Predetermined reward functions and imitation of teachers (Zhang et al., 2021). We summarise the differences between these techniques and feedback-conditioning in Table 1. Providing useful reward functions however is made difficult by the *credit assignment problem* – the problem of identifying the specific aspect of a decision or behaviour that is responsible for the ultimate outcome

or reward (Minsky, 1961). Indeed, when such signals are binary or preference-based, informational constraints can heavily limit their efficiency as learning signals; as is often the case in the alignment of language models (Wu et al., 2024). Given the ability of language models to understand feedback, by allowing a student model to retry a task conditioned on natural language feedback, we aim to produce trajectories that differ exactly where it matters. For instance in Figure 1 we demonstrate a 7B model receiving feedback that allows it to refine its answer additively, resulting in two responses that differ with regards to the level of detail, but not with regards to their content. During training then, this can provide a clearer signal of which aspect of the response can be improved.

With regards to teacher imitation, directly learning from the products of a teacher model may also be undesirable. In the first instance, the specific strategies and reasoning paths used by larger models may not be feasible for smaller ones. For instance, a large model may have more domain knowledge and be able to make reasoning leaps or inferences that are only available to a smaller model through the use of explicit reasoning. Indeed, data from weaker teachers often performs better (Bansal et al., 2024). Under a Vygotskian view of human development, children learn best in the so called *Zone of Proximal Development* – the space in which children can act "through problem solving under adult guidance or in collaboration with more capable peers" (Vygotsky, 1978). By analogy then, we aim to test whether the same holds true for language models: *do they learn best through problem solving under teacher guidance*. Consider for example learning to play tennis. If you tried to learn by directly copying a professional player, or just through playing, you likely wouldn't get very far. Instead you could use instructors to give feedback on specific movements to practice, and through practising, conditioned on this information, improve your performance over time.

Studying how language models can learn from natural language feedback has broad relevance due to its universality. Firstly, it is compatible with using teachers that are locked behind APIs, or unable to actually act in the decision making environments in question (due to costs or missing capabilities like the embodied physicality that would be required in our tennis example). More importantly, a better understanding of how well models can give and receive feedback is important for enabling models to learn from and teach humans, through a format that is widely accessible.

Through our experiments we find that even state of the art frontier models can struggle to identify errors (Section 5.1.1), and that especially smaller models introduce significant noise into the feedback-refined responses (Section 5.1.2). We find that incorporating negative sampling (Section 5.2) significantly boosts performance and finally that if we can reduce the noise in our responses, Learning from Feedback Conditioning is able to perform on par with traditional imitative and preference learning methods (Section 5.3).

2 Learning through Feedback Conditioning

We begin with a high-level overview of how feedback conditioning can work for decision making agents, before moving onto the specific case of natural language tasks that we are primarily concerned with in this paper.

A decision making environment is typically characterised as a Markov Decision Process $\langle \mathcal{S}, \mathcal{A}, \pi, \mathcal{R}, \gamma \rangle$ for state set S, action space A, policy π, discount factor γ and reward function $\mathcal{R}$; the goal of which is to maximise the expected cumulative discounted reward over trajectories. Instead of learning an optimal policy π through the feedback of $\mathcal{R}$ itself (Reinforcement Learning) or through copying a teacher policy (Imitation Learning), we suppose instead that there exists a teacher that can provide natural language feedback over trajectories.

In particular we suppose that our student and teachers have linguistic capabilities in the following ways: Firstly we suppose that the student model is a generalist agent that can be parameterised both by its parameters θ and a natural language instruction l. Secondly we characterise the teacher as a function ϕ_t that maps a given trajectory $\tau : \{\mathcal{S} \times \mathcal{A}\}^*$ to a natural language description of what went wrong, and how to improve upon it.

The process of feedback conditioning can be described in the following way:

- First generate the student trajectory $\tau_l \sim \pi_{\theta,i}$ using the base student model with parameters θ and instruction i.

Question

You are playing the game Taboo. Given the word 'president', generate a brief description that will help your team guess the word. You cannot use the following words: [president, vice, met, secretary, presidency, administration, asked, promised, former, announced] When providing your answer please make sure it matches the following regex: "Description: [a-zA-Z0-9,]*". (You can ignore the quotes)

Initial Answer

"Description: A leader in government."

Feedback

Good job! Your description is clear and concise, but try to add a bit more detail to help your team guess the word more easily. For example, you might mention their role in making decisions or leading a country.

Revised Answer

Here's an updated version of the description:
Description: Leading figure, makes big choices, runs a nation.

Figure 1: Example of the feedback given by GPT-4-o-mini to our Qwen2 7B student on the Taboo environment from Big-Bench Strategic Reasoning. The resulting revised response improves on the original one in a precise way alluded to in the feedback. A side effect, however, is the addition of the statement "Here's an updated version..." which may add noise to the training process.

- Then generate teacher feedback $f_t \sim \phi_t(\tau_l)$ according to the teacher's feedback policy ϕ_t
- Generate a feedback conditioned trajectory $\tau_f \sim \pi_{\theta,\text{COMBINE}(i,\tau_l,f)}$ where COMBINE integrates the feedback over the previous trajectory into the instructions.
- Finally we update θ to increase the probability of generating the feedback conditioned trajectory when *unconditioned by the feedback* i.e. $\prod_{(s,a)\in\tau_f} \pi_{\theta,i}(s,a)$
- This update step can either be through *imitation* or by means of comparison to the original trajectory through *preference learning*.

For this technique to work, we rely on the following conditions:

- Firstly, the teacher must be able to identify flaws in the student's reasoning or approach
- Secondly, the teacher must be able to provide useful feedback that enables the student to be more successful
- Finally, this must provide a useful learning signal: over time the student should be able to achieve the same quality of trajectory without conditioning on that feedback.

2.1 Learning through Feedback Conditioning for Language Tasks

Given these requirements, in this work we seek to establish whether and when these conditions hold for tasks in the language domain. For such tasks, the instructions and trajectories correspond to the prompts and responses (or series of responses) used in instruction tuning tasks (with the actions being tokens and states corresponding to the concatenation of those tokens). In this work, we explore the use of Supervised Finetuning (SFT) i.e. next-token prediction, and Direct Preference Optimization (DPO) (Rafailov et al., 2024) as our imitation and preference learning algorithms respectively.

3 Related Work

3.1 Reinforcement Learning from Human Feedback (RLHF)

We build on existing research looking to leverage feedback to improve models beyond the scope of just imitating textual data. LLMs have been increasingly used as chat-agents, designed not to model language, but instead to act as helpful assistants to humans. To achieve this, Ouyang et al. (2022) popularised the idea of training models on human feedback, through the use of reinforcement learning. In particular they proposed to train a reward model to match human ratings of completions, and then use this to optimize the base language model using PPO. Later, in DPO, Rafailov et al. (2024) introduced the idea of reparameterising the reward model in terms of the model's policy, enabling the model to be trained without requiring the use of an explicit, separately trained reward model. These have been followed up in turn by a variety of methods and proposed improvements; we encourage the interested reader to refer to a survey such as Wang et al. (2023).

Improving Credit Assignment. In particular a number of works have looked into trying to improve the informational content of RLHF addressing the credit assignment issue. One such technique has been to employ linguistic reward shaping, through the use of fine-grained reward models (Wu et al., 2024) or optimising the likelihood of positive/negative continuations on following turns (Zhang et al., 2024). Other works have addressed the credit assignment issue through process supervision (Lightman et al., 2023) i.e. trying to provide rewards at the level of sentences or tokens rather than whole completions. These have been found to provide significant benefit in tasks including mathematical reasoning (Shao et al., 2024) and planning (Jiao et al., 2024).

Instructive Learning. The extreme costs associated with training large foundation models mean that many researchers look to leverage the power of existing large models for improving the efficiency of training smaller models. Such methods have generally consisted in data distillation, by generating synthetic data with explanation traces (Mukherjee et al., 2023), rationales (Hsieh et al., 2023) or just their answers for instruction tuning (Peng et al., 2023). Generally, training on such synthetic data has been found to improve the sample efficiency of language models (Maini et al., 2024). We explore this form of data distillation as our baseline finetuning technique.

3.2 Learning from Natural Language Feedback

Prior work has explored applying natural language feedback to improve decision making agents in a number of ways. In particular a number of works have explored creating reward functions using LLMs (Yu et al., 2023) and for editing the reward function's code in response to the performance of the agent (Ma et al., 2024). Other works have explored using language feedback as a test-time refinement strategy of the output of models. For instance Madaan et al. (2024) find they can boost the performance of models by allowing them to repetitively adapt responses until they meet a stopping criterion. They find that this holds for models larger than 13 Billion parameters. In our work however we are interested in actually teaching the models, and furthermore explore the case in which the feedback is given by a stronger language model.

Most relevantly, Chen et al. (2024) propose a method of using imitation learning to learn from natural language feedback. Rather than resampling while conditioning on this feedback, they use a separate refinement policy to adapt the student model's outputs according to this feedback, and then perform imitation learning on the resultant outputs. Finally they employ rejection sampling while generating these refinements. The focus of their paper is on the ability of models to leverage *human feedback*, which they find to be significantly more informative than model feedback. Due to the costs and difficulties associated with gathering human feedback we primarily focus on leveraging model feedback, since it is more widely applicable, nevertheless we expect that our methods would perform even better if given access to human feedback in the same manner.

4 EXPERIMENTAL SETUP

4.1 DATASETS

To provide a more holistic understanding of learning from feedback conditioning, we run experiments on a wide range of difficult tasks. In particular, we use the following benchmarks:

- **Big-Bench Hard (BBH)** (Suzgun et al., 2022) is a subset of Big-Bench (bench authors, 2023)[1] picked out for being more challenging at the time. For each task, we use the first 100 samples as a train set. We identify regex for extracting the answer from each of these benchmark-tasks and in some cases change the presentation of the answer or add instructions around matching these regex to aid in the accuracy of our performance metrics. On average this leaves around 150 samples to use as the test set for each subtask.
- **Big-Bench Strategic Reasoning (BBSR)** is a re-implemented subset of several Big-Bench tasks that are either game-like or require strategic reasoning to solve. None of these tasks are part of Big-Bench Hard. These tasks are summarised in Table 7.
- **Mostly Basic Programming Problems (MBPP)** (Austin et al., 2021) is a set of python programming questions with accompanying tests. This was used by Chen et al. (2024), in the most relevant prior work.

We initially benchmark various student and teacher models on these environments, with our results displayed in Table 8. We use custom evaluation scripts, which can be found in our code base[2].

4.2 MODELS

For our teacher model we select GPT-4-o-mini, since it has a very high performance to cost ratio (see Appendix D.1 for our experiment on the performance of different teacher models). For our student model we use the Qwen 2 model series (Yang et al., 2024). This was motivated by the fact that it consisted in a series of strong instruction tuned model at a range of model scales.

4.3 EVALUATION AND EXPERIMENTAL CONFIGURATION

We use custom code for evaluating the performance of our models on each benchmark. In general, for the majority of tasks in BBH and BBSR we largely use regex-based methods for extracting model responses. For MBPP, we run the generated python code against the tests defined in the dataset, and treat the score as the percentage of tests passed.[3] For our finetuning experiments, we limit our datasets to 100 samples per iteration as we are interested in sample efficiency, and our benchmarks are small. We use off-the-shelf finetuners from the HuggingFace TRL package [4] and provide our training configurations in Appendix B. Additionally for all Feedback Conditioning Experiments our prompts are available in Appendix A.

5 EXPERIMENTS

We first experiment on the ability of models to give and receive feedback in Section 5.1. Then, based on these findings, we design modifications to our base method and experiment on how these affect feedback conditioning in Section 5.2. Finally, in Section 5.3 we explore the performance of learning from feedback conditioning as a finetuning method.

[1] Downloaded from `https://huggingface.co/datasets/tasksource/bigbench`

[2] To be released

[3] For the sake of comparison we note that this rate is typically $\sim 10\%$ higher than the metric of passing all tests used in Chen et al. (2024).

[4] `https://huggingface.co/docs/trl/en/index`

5.1 Feedback Conditioning

Before investigating how to finetune models with feedback we first perform analysis on the linguistic understanding of our teacher (*GPT-4-o-mini*) and student models (Qwen2 {0.5,1.5,7}B). We initially aim to answer two key questions:

- Can our teacher model identify errors in student answers? (See Section 5.1.1)
- Can our teacher model provide useful feedback on those answers *in the sense that the student performance is boosted by conditioning on that feedback* (See Section 5.1.2)

5.1.1 Error Identification

As depicted in Table 2 we explore both explicit and implicit error detection. Explicit error detection consists in asking the teacher model to judge whether or not the student's answer is correct (without access to the ground truth data). We find that the ability of the teacher to detect errors is inversely proportional to the ability of the student. In particular we find a marked drop in the recall of the error detection for our largest model size, which we believe corresponds to the fact that its incorrect reasoning paths are more likely to be misleading, while also producing more of them.

For the implicit error detection we give our teachers the option of not providing feedback if they think the student's answer is substantially correct, and instead reply with 'no feedback needed', which we can then automatically detect. We find however that the teacher rarely uses this capability, and will instead tend to praise the student model when it doesn't find anything wrong; as such in our experiments this technique has perfect recall but low precision.

As such we do not find strong confirmatory evidence for this identification ability being present in the teacher model. Nevertheless given the prevailing wisdom of using LLMs-as-judges (Zheng et al., 2023) we rely on downstream performance to establish the existence of this capacity.

Error Identification	Student Model Size	Recall	Precision	F1	Accuracy
Explicit	7B	0.65	0.83	0.66	0.73
	1.5B	0.82	0.89	0.85	0.76
	0.5B	0.85	0.93	0.89	0.81
Implicit	7B	1.0	0.71	0.83	0.71
	1.5B	1.0	0.84	0.91	0.84
	0.5B	1.0	0.92	0.96	0.92

Table 2: Explicit and implicit error identification ability of the teacher model. All metrics are based on macro-averages (i.e. across all samples rather than averaged across task-subsets). The accuracy is computed with respect to the environment's ground truth evaluation which uses Regex to extract an answer and match it against a target label.

5.1.2 Feedback Use

We next investigate the degree to which models can actually use this feedback. Our feedback conditioning technique in this section consists in providing the prompt, original answer, and feedback as the chat history with the feedback given to the model as in the Student Retry Prompt in Appendix A. We try out the following feedback conditions to elucidate information about the models' ability to leverage feedback.

- **Unconstrained**: in which the teacher directly gives feedback on the student answer with no additional information
- **Ground Truth**: in which the teacher gives feedback with access to the ground truth answers
- **Teacher Answer**: in which the teacher first answers the prompt, and then uses this answer to inform their feedback on the student's answer allowing them to directly compare different reasoning chains for the problem.

Student Size	Additional Feedback Data	Regex Match Rate	Downstream Performance	Correct if Matched
7B	Baseline	0.60	0.29	0.48
	Unconstrained	0.53	0.31	0.58
	Teacher Answer	0.51	0.32	0.63
	Ground Truth	0.56	0.38	0.68
	Lazy	0.89	0.65	0.73
	Regex Focused	0.61	0.31	0.51
1.5B	Baseline	0.53	0.17	0.32
	Unconstrained	0.45	0.21	0.46
	Teacher Answer	0.42	0.20	0.48
	Ground Truth	0.42	0.21	0.5
	Lazy	0.54	0.24	0.44
	Regex Focused	0.55	0.16	0.29
0.5B	Baseline	0.26	0.06	0.23
	Unconstrained	0.32	0.10	0.31
	Teacher Answer	0.32	0.09	0.28
	Ground Truth	0.31	0.10	0.32
	Lazy	0.40	0.12	0.3
	Regex Focused	0.36	0.10	0.28

Table 3: Results on Big-Bench Hard for different test-time feedback conditioning methods. For each trajectory, our implementation of Big-Bench Hard provides a binary score. The set of answers for which the model is correct is a strict subset of the set of answers for which the model provides an answer that can be extracted using regex (i.e. the Regex Match Rate). As such we can see that across model scales, following feedback tends to degrade the match rate. Nevertheless across conditions it improves the performance of the models, especially when we normalise for the model providing a valid answer at all (per the Correct if Matched column)

.

- **Lazy**: in which the feedback just consists in directly being told the correct answer. This should provide a ceiling on the models ability to integrate information from teachers since it can just be copied verbatim.
- **Regex Focused**: in which the teacher is instructed to focus feedback on matching the regex used for answer extraction

Our results are shown in Table 3. We find that all the models are able to leverage the feedback to some degree, with the registered performance increasing across the board. That said the increase is relatively minor, on the order of 2/3 (absolute) percentage points for the low-requirement techniques (i.e. excluding Ground Truth and Lazy conditions). In addition to this quantitative analysis we manually checking 150 samples of responses, feedbacks and refinements and combining these we hypothesise the following:

- **Failing to Match**: Generally we note that match rate falls across the board for methods incorporating feedback, and adjusting for this we see even more significant gains for test-time feedback conditioning. After manually checking we find that the teacher rarely picks up on the fact that the student has failed to match the Regex, and even when the student has initially matched it, the addition of the feedback condition reduces the ability of the student to follow this additional constraint.
- **Mismatch between Refinement and Response** Oftentimes if the models original reasoning is substantially correct, the model simply directly responds with the answer (correct or otherwise). More generally we note that there are often artefacts in the response due to the fact that the model is responding to the feedback, as demonstrated in Figure 1. While this isn't well reflected in the performance measured here, this would result in poor performance if such responses were used as targets for learning.

Method	Submethod	Regex Match Rate	Score	Correct if Matched
Negative Sampling	–	0.53	0.31	0.58
	Error Identification	0.57	0.34	0.60
	Ground Truth	0.63	0.45	0.71
Best-of-k (No Feedback)	–	0.60	0.29	0.48
	Student Logits	0.65	0.29	0.45
	Student Choice	0.61	0.30	0.49
	Teacher Choice	0.61	0.36	0.59
Best-of-k (/w Feedback)	–	0.54	0.30	0.56
	Student Logits	0.54	0.31	0.57
	Student Choice	0.55	0.32	0.58
	Teacher Choice	0.55	0.34	0.62
Feedback Following Finetuning	SFT	0.47	0.23	0.55
	SFT + Feedback	0.53	0.31	0.58
	DPO	0.62	0.26	0.55
	DPO + Feedback	0.52	0.31	0.60

Table 4: Results on Big-Bench Hard for our modifications to the basic Feedback Conditioning Algorithms. For Negative/Best-of-k sampling we compare using these to the baseline result (-) not using them. We use the 'unconstrained' feedback condition throughout.

- **Task Reinterpretation**: Instruction tuned models have a tendency to reinterpret certain tasks as coding tasks, which often misleads the teacher into instead providing feedback on the quality of t
- **Insufficient Ability**: Especially with the smaller models, oftentimes while the feedback does give enough information to arrive at the correct answer, the model still has to reason through and proves unable to integrate this. We see this most starkly with their inability to substantially benefit from being directly given the answer in the feedback condition

Collectively we refer to these effects as *noise* introduced by the student model. We design further experiments to help mitigate some of these issues in Experiment 5.2. Moreover in proceeding experiments we only use the 7B model, due to the ceiling on its benefit in the Lazy and Ground Truth conditions is higher than the other models.

5.2 Modifications

We introduce the following modifications to our language-based feedback conditioning algorithm (Described in Section 2):

- **Negative sampling**: given the noise introduced in the process, the risk of an already correct response being degraded by feedback conditioning means that it makes sense to utilise negative sampling: asking the teacher directly to identify correct answers and then removing these from the feedback conditioned training sets. We additionally explore using ground truth data instead to perform this task.
- **Feedback Tuning**: We explore finetuning the model on following the feedback instead. To do this we train a single model on 5 samples from each of our environments. As training data we use our refinement prompts (see Appendix A) as input, using the teacher and student models to provide targets for SFT/DPO.
- **Best-of-k Sampling**: Similarly to the effect of negative sampling, we can increase the chance that a feedback conditioned response improves upon the original one according to the feedback, by generating k trajectories and then selecting the best. In particular we explore methods in which the student/teacher models select the best response with respect to the feedback, as well as an additional techniques using the logits of the student model to determine which response it would most likely generate.

In Table 4 we demonstrate the results of these modifications on test-time feedback conditioning. We find that concordant with our expectations, allowing the teacher to explicitly identify samples in which a mistake is made, results in a significant performance increase, although not as high as allowing the teacher access to the ground truth. Using the ground truth data to negatively sample on the other hand results in a very large increase in performance of 50% above the original. We note however that this is not some special feature of feedback conditioned responses: negatively (re)sampling ordinary responses in this way leads to a proportional increase in the performance of the base model. The same can be seen in the case of the Best-of-k sampling technique, which boosts the performance of ordinary trajectories similarly to those of their negative counterparts.

Given the substantial additional cost from Best-of-k sampling and poor performance of finetuning, we use negative sampling in our finetuning experiments.

5.3 FINETUNING EXPERIMENTS

Finally we employ our modified algorithm to the task of levaraging feedback. These results are shown in Table 5 and Table 6. We note that DPO does not perform well on Big-Bench Hard across the board for us, and the slight improvement from using the feedback conditioned variety is negligible. Furthermore we find that our basic feedback conditioned version of supervised finetuning *drastically underperforms* traditional teacher imitation, resulting in a drop in performance over the baseline.

As discussed in Section 5.1.2 the response to feedback often contains reference to the fact that that response is feedback conditioned. For instance in Figure 1 the model reply starts with "Here's an updated version of the description". Our hypothesis is that since we use the generation of the feedback conditioned response as a generation target, these relics of the feedback add excessive noise to the learning process. As such we employ an additional post processing step using the teacher to remove these noise aspects from the student's feedback-conditioned generative output as used in this learning process. These modified versions of our learning algorithm are denoted with an asterisk*. We observe that this leads to a significant performance increase for the model across Big-Bench Hard resulting in a 8% absolute increase in performance against the baseline, demonstrating the potential of feedback conditioning as a learning technique. This is significantly higher than that of the baseline Feedback Conditioning technique, but is nevertheless less performant than learning directly from the teacher.

Student Model	Training Method	Regex Match Rate	Score	Post-Match Score
Qwen2 7B	Imitative DPO	0.43	0.23	0.52
	Feedback DPO	0.54	0.27	0.47
	Imitative SFT	0.63	0.44	0.69
	Feedback SFT	0.51	0.24	0.49
	Feedback SFT*	0.70	0.37	0.54

Table 5: Results finetuning on Big-Bench Hard for each Qwen2 model size using the Imitative/Feedback variants of SFT/DPO. Imitative refers to using the teacher's trajectories as a target, whereas Feedback variants rely on the student's feedback conditioned trajectories. We give the teacher access to the ground truth, and use ground truth based negative sampling in constructing the feedback based preference learning dataset. Feedback SFT* additionally uses our 'response cleaning' technique.

For our experiments on Big-Bench Strategic Reasoning and MBPP, we do not assume access to ground truth data for the teacher model, and instead use the *teacher answer*-feedback conditioning strategy and *Error-Identification*-negative sampling strategy. We note that these have lower performances in our preliminary experiments vis-a-vis feedback conditioning, but are also more reflective of a more scalable application scenario for using learning from feedback. There are no clear best models, and each approach works better on some environments and worse than others. Additionally all of the finetuning techniques perform worse than the baseline on a majority of the tasks. These results are summarised in Table 6.

Model	Method	SPB	TAB	OPT	HILO	NEG	LFN	RFN	MBPP
Teacher	(GPT-4-o-mini)	0.029	1.7	0.75	0.55	0.74	0.63	0.87	0.63
Qwen2 7B	Baseline	0.022	1.20	0.25	0.33	0.08	0.15	0.65	0.39
	Imitative DPO	0.055	1.44	0.19	0.25	0.10	0.32	0.39	0.03
	Feedback DPO	0.021	1.34	0.22	0.25	0.14	0.11	0.46	0.34
	Imitative SFT	0.016	0.86	0.27	0.18	0.10	0.21	0.79	0.30
	Feedback SFT	0.013	1.22	0.15	0.2	0.24	0.12	0.68	0.39
	Feedback SFT*	0.006	1.46	0.31	0.2	0.15	0.11	0.75	0.30

Table 6: Results of Models on BBSR and MBPP. Our Feedback Conditioned techniques use the Unconstrained condition with Error-Identification based negative sampling, due to the lack of ground truth data.

6 Conclusion

Experimental Findings Overall we find that models *can learn from conditioning on feedback* in the language domain, although it requires accounting for the inability of teacher models to identify errors (through negative sampling) and removing the noise introduced by the refinement process. Nevertheless it is not as strong on most tasks as simply leveraging the trajectories of our strongest foundation models. On our strategic reasoning and coding tasks, none of our finetuning techniques perform well, indicating the challenge of finetuning in this few sample paradigm.

Extending Feedback Conditioning to Non-Linguistic Domains The promise of Feedback Conditioning lies mostly in its broad applicability to multimodal domains in which we can use the instructability of generalist, linguistically capable agents to perform a wide variety of tasks. Providing criticism and feedback in these domains does not necessarily require an agent to be able to act in them, and being able to leverage the reasoning of strong foundation models in these roles for training decision making agents has huge potential. However this will require addressing some of the drawbacks we discussed above, and the process of e.g. *cleaning trajectories* to enable their use as targets will be highly domain dependent.

Furthermore the language tasks we have used do not have the long horizon nature or stochasticity of many such real world tasks, which may impact the relevance of feedback conditioned trajectories used in the way we have here. We leave these extensions to future works but believe that our work establishes the conditions for leveraging this form of flexible, personalised, and precise feedback across a wide range of settings.

References

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

Hritik Bansal, Arian Hosseini, Rishabh Agarwal, Vinh Q Tran, and Mehran Kazemi. Smaller, weaker, yet better: Training llm reasoners via compute-optimal sampling. *arXiv preprint arXiv:2408.16737*, 2024.

BIG bench authors. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=uyTL5Bvosj.

Angelica Chen, Jérémy Scheurer, Jon Ander Campos, Tomasz Korbak, Jun Shern Chan, Samuel R. Bowman, Kyunghyun Cho, and Ethan Perez. Learning from natural language feedback. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=xo3hI5MwvU.

Cheng-Yu Hsieh, Chun-Liang Li, Chih-Kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alexander Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. Distilling step-by-step! outperforming larger

language models with less training data and smaller model sizes. *arXiv preprint arXiv:2305.02301*, 2023.

Fangkai Jiao, Chengwei Qin, Zhengyuan Liu, Nancy F Chen, and Shafiq Joty. Learning planning-based reasoning by trajectories collection and process reward synthesizing. *arXiv preprint arXiv:2402.00658*, 2024.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023.

Yecheng Jason Ma, William Liang, Guanzhi Wang, De-An Huang, Osbert Bastani, Dinesh Jayaraman, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Eureka: Human-level reward design via coding large language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=IEduRUO55F.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems*, 36, 2024.

Pratyush Maini, Skyler Seto, He Bai, David Grangier, Yizhe Zhang, and Navdeep Jaitly. Rephrasing the web: A recipe for compute and data-efficient language modeling. *arXiv preprint arXiv:2401.16380*, 2024.

Marvin Minsky. Steps toward artificial intelligence. *Proceedings of the IRE*, 49(1):8–30, 1961.

Subhabrata Mukherjee, Arindam Mitra, Ganesh Jawahar, Sahaj Agarwal, Hamid Palangi, and Ahmed Awadallah. Orca: Progressive learning from complex explanation traces of gpt-4, 2023.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. Instruction tuning with gpt-4. *arXiv preprint arXiv:2304.03277*, 2023.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36, 2024.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, YK Li, Yu Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V Le, Ed H Chi, Denny Zhou, , and Jason Wei. Challenging big-bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*, 2022.

Michael Tomasello. The social bases of language acquisition. *Social development*, 1(1):67–87, 1992.

Lev Semenovich Vygotsky. *Mind in society: The development of higher psychological processes*, volume 86. Harvard university press, 1978.

Yufei Wang, Wanjun Zhong, Liangyou Li, Fei Mi, Xingshan Zeng, Wenyong Huang, Lifeng Shang, Xin Jiang, and Qun Liu. Aligning large language models with human: A survey. *arXiv preprint arXiv:2307.12966*, 2023.

Zeqiu Wu, Yushi Hu, Weijia Shi, Nouha Dziri, Alane Suhr, Prithviraj Ammanabrolu, Noah A Smith, Mari Ostendorf, and Hannaneh Hajishirzi. Fine-grained human feedback gives better rewards for language model training. *Advances in Neural Information Processing Systems*, 36, 2024.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024.

Wenhao Yu, Nimrod Gileadi, Chuyuan Fu, Sean Kirmani, Kuang-Huei Lee, Montserrat Gonzalez Arenas, Hao-Tien Lewis Chiang, Tom Erez, Leonard Hasenclever, Jan Humplik, brian ichter, Ted Xiao, Peng Xu, Andy Zeng, Tingnan Zhang, Nicolas Heess, Dorsa Sadigh, Jie Tan, Yuval Tassa, and Fei Xia. Language to rewards for robotic skill synthesis. In *7th Annual Conference on Robot Learning*, 2023. URL https://openreview.net/forum?id=SgTPdyehXMA.

Chen Zhang, Dading Chong, Feng Jiang, Chengguang Tang, Anningzhe Gao, Guohua Tang, and Haizhou Li. Aligning language models using follow-up likelihood as reward signal. *arXiv preprint arXiv:2409.13948*, 2024.

Ruohan Zhang, Faraz Torabi, Garrett Warnell, and Peter Stone. Recent advances in leveraging human guidance for sequential decision-making tasks. *Autonomous Agents and Multi-Agent Systems*, 35 (2):31, 2021.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36:46595–46623, 2023.

A PROMPTS

Error Identification

I asked my student to follow the instruction below: {prompt}

They answered: {answer}

Was their answer correct in terms of both answering the question and matching the regex? Please finish your responce with Correctness: correct/incorrect according to whether it was correct"""

Unconstrained Feedback

Help give feedback on my student's answers They were given the instruction: {prompt}

They answered: {answer}

Directly give the feedback to them as I will pass it on. Format your feedback as either "Feedback: ..." or "No feedback needed" if you think their answer is correct already.

Ground Truth Feedback

Help give feedback on my student's answers They were given the instruction: {prompt}

They answered: {answer}

The correct answer is: {correct_answer}

Directly give the feedback to them as I will pass it on. Format your feedback as either "Feedback: ..." or "No feedback needed" if you think their answer is correct already.

Teacher Answer Feedback

Help give feedback on my student's answers I gave my student the same instruction as you.
They answered: {answer}

Directly give the feedback to them as I will pass it on. Format your feedback as either "Feedback: ..." or "No feedback needed" if you think their answer is correct already.

Student Retry Prompt

Here is some feedback on your previous answer: {feedback}

Please follow this instruction again, incorporating the feedback: {prompt}

Figure 2: Various Prompts Used in This Work

B HYPERPARAMETERS

SFTConfig

```
gradient_checkpointing:True,
gradient_accumulation_steps:4,
remove_unused_columns:True,
learning_rate:5e-4,
lr_scheduler_type:"cosine",
max_steps:num_steps,
save_strategy:"no",
optim:"paged_adamw_32bit",
bf16:use_bfloat16,
fp16:use_float16,
warmup_ratio:0.1,
```

DPOConfig

```
beta=0.1,
max_prompt_length=1024,
gradient_checkpointing=True,
remove_unused_columns=False,
learning_rate=5e-5,
lr_scheduler_type="cosine",
max_steps=num_steps,
save_strategy="no",
optim="paged_adamw_32bit",
bf16=use_bfloat16,
fp16=use_float16,
warmup_ratio=0.1,
gradient_accumulation_steps=4,
```

Figure 3: Comparison of SFT and DPO configurations

C BIG-BENCH STRATEGIC REASONING TASKS

Task	Task Description	Modification from Original
Spelling Bee (SPB)	Tests ability to manipulate letters to spell out as many valid words as possible.	
Taboo (TAB)	A two-turn game that requires following taboo rules while coming up with a description of a concept, and then subsequently guessing what the concept is	We give 1 point for a valid description (i.e. not including a taboo word) and an additional point if the model can guess the word.
Optimization (OPT)	A mathematical reasoning task in which the model must reason about finding parameters that minimize a given simple function.	This is actually one of set of games around root finding[a]
Hi-Lo (HILO)	A game consisting of trying to guess a hidden number in the range of 1-100 in seven tries with feedback (with an obvious optimal strategy of performing a binary search)	The original version[b] was a two player guessing game that required answers to exactly match a guess format. We pose the game as a single player task, with the high/low feedback coming from the environment instead.
Nash Equilibria (NEG)	Consists in performing calculations to determine whether a given game has a pure-strategy nash equilibrium, and identifying one of them	The original task came from the same set as OPT and consisted in finding the optimal action. We adapt the task to consist in finding a Nash Equilibria (if one exists) for simple 2x2 games.
List Functions (LFN)	(Modified from the original variant) Consists in taking a description of a function over integer lists and a single input-output pair example before applying that function to unseen inputs and outputs.	The original consisted in trying to guess what the function was given a set of inputs and outputs and then apply it to unseen inputs.
Root-Finding (RFN)	Tests numerical methods and root-finding algorithms.	This also comes from the same set of games as OPT and is closely adapted.

Table 7: Summary of 'Game' Tasks we choose for our Big-Bench Strategic Reasoning subset

[a] `https://github.com/google/BIG-bench/tree/main/bigbench/benchmark_tasks/roots_optimization_and_games`. In particular this is about finding a minimizer for a sum of absolute values.

[b] `https://github.com/google/BIG-bench/blob/main/bigbench/benchmark_tasks/high_low_game/task.py`

D ADDITIONAL EXPERIMENTS

D.1 TEACHER MODEL BENCHMARKING

Model	BBH	BBSR	MBPP	Cost per Million Tokens ($)[a]	BBH Feedback
Teachers:					
GPT-4-o-mini	0.65	0.90	0.63	0.15	0.31
Claude-3-Haiku	0.64	0.75	0.65	0.25	0.28
Gemini-Flash	0.36	0.85	0.62	0.15	0.26

Table 8: Baseline Results on our Benchmarks for Different API-based models. Note that the evaluation scores here do not use the exact same evaluation as elsewhere in the paper, due to bugs we found in it later; nevertheless it formed the basis of our teacher selection. The final metric of BBH Feedback corresponds to using the models for test-time feedback in the unconstrained feedback condition described in Section 5.1

[a] As available on OpenRouter: `https://openrouter.ai/models/google/gemini-flash-1.5`