3 MODEL TRAINING

Here we briefly describe the details of how the models we evaluate are trained. For a more detailed description of model training see Appendix E.

Pretrained Models. We use the LLaMa pretrained 7 billion parameter model (Touvron et al., 2023a). This is a standard decoder-only transformer-based causal language model trained on a large corpus of web text. This size of model has been shown to be effective in the tasks we investigate (Stiennon et al., 2022; Dubois et al., 2023).

In Appendix J we perform experiments with OPT (Zhang et al., 2022) models, using five model sizes. These models have worse performance in general, but the experiments allow us to see trends across several model scales.

We train **Reward Models**(RMs) following (Stiennon et al., 2022). We initialise the RM as the base model, and we add a scalar head before the unembedding layer. The model is then fine-tuned on inputs and pairs of outputs with one output labelled as preferred, and trained to output a scalar for each input-output pair representing which output is preferred.

Supervised Fine-Tuning (SFT) models are trained on reference input-output pairs using the cross-entropy loss on the output conditioned on the input.

Reinforcement Learning from Human Feedback (RLHF) models are trained using PPO (Schulman et al., 2017) as our base RL algorithm, following previous work (Stiennon et al., 2022). We initialise the model with the corresponding SFT model, use the language modelling head as the policy output, and learn a value function using a linear layer (an identical architecture to the RM), with the policy and value function sharing the same model backbone. We optimise the policy to maximise the RM described above, and use a KL divergence term as an auxiliary reward to ensure that the language model stays close to the SFT model, as in previous work (Jaques et al., 2017; Stiennon et al., 2022; Ziegler et al., 2020). The final reward for the policy is

$$R(x, y) = RM_{\theta_{RM}}(x, y) - \beta_{KL} D_{KL}(\pi_{\theta_{RL}}(y|x)||\pi_{\theta_{SFT}}(y|x)) \tag{1}$$

where RM denotes the reward model trained as described above; θ_{RL}, θ_{RM} and θ_{SFT} are the parameters of the policy, RM and SFT model respectively; x, y are the input and output; and β_{KL} is a hyperparameter that controls the weight of the KL penalty. We use $\beta_{KL} = 0.05$ throughout this work, following (Stiennon et al., 2022) and our own early experiments that found this choice struck a good balance between model performance and overoptimisation (Gao et al., 2022).

Best-of-N. The reward model can also be used to filter samples from another model; this is called Best-of-N (BoN) sampling and has been used in multiple previous works to achieve good performance (Menick et al., 2022; Nakano et al., 2022). N summaries are sampled from the SFT model, and then the RM is used to select the best one. We sample with temperature 0.7 and use $N = 16$, as that is what is used in previous works (Rafailov et al., 2023) and strikes a good balance between improved performance and computational cost. Evaluating BoN performance gives use a way to evaluate whether the differences between RLHF and SFT models are due to the use of a RM, or the type of optimisation applied. However, due to the increased inference time compute cost, this method is generally not used in practice, so we focus our analysis on RLHF and SFT policies.

4 DATASETS AND TASKS

We investigate the effects of RLHF in two tasks: text summarisation and instruction following.

Summarisation. We follow Stiennon et al. (2022); Ziegler et al. (2020) in evaluating LLM fine-tuning algorithms on a summarisation task. Models are trained to produce summaries of Reddit posts, given the post as input. We use the same dataset as Stiennon et al. (2022), which is a filtered version of the TL;DR dataset (Völske et al., 2017), consisting of approximately 120,000 Reddit posts with accompanying summaries. See Stiennon et al. (2022) for full details on the filtering procedure and the analysis of the resulting data distribution.

We use the preference data gathered by Stiennon et al. (2022) for reward model training. This consists of approximately 64,000 summary comparisons. These data consist of inputs (reddit posts) along with pairs of outputs (summaries), with one summary labelled as preferred. The preferences are from

human annotators contracted by Stiennon et al. (2022) to choose summaries according to a list of criteria designed to select high quality summaries.

Instruction Following (Chung et al., 2022; Dubois et al., 2023; Wang et al., 2022) is one of the main use cases for the LLM fine-tuning techniques we investigate, so we also evaluate models trained in this setting.

We use the SFT, RLHF, and RM models released by Dubois et al. (2023, AlpacaFarm). These models were trained in a very similar way to how we train our summarisation models, and all based on the LLaMa 7B base model. Models take a text instruction as input and are trained to output preferred answers to those instructions. The outputs used for the SFT model are from the `text-davinci-003` model from the OpenAI API, and the human preferences used to train the reward model are gathered by the authors of AlpacaFarm based on outputs of the SFT model. For precise details on how these models were trained, refer to Dubois et al. (2023).

5 Model Evaluation

We now describe how we evaluate both out-of-distribution generalisation and output diversity for the different fine-tuning techniques.

5.1 Generalisation Evaluation

To evaluate the performance of our trained models, we use GPT-4 (OpenAI, 2023) through the OpenAI API [1] as a *simulated human annotator*. While we ultimately care about human preferences, these are expensive, difficult to gather, and often noisy. Previous work has shown that GPT-4 accurately simulates human preferences in both summarisation (Rafailov et al., 2023) and instruction following (Dubois et al., 2023; Zheng et al., 2023), so we use it as the main performance metric in both tasks.

To evaluate a model's performance on a given dataset of inputs and reference outputs with GPT-4, we prompt GPT-4 with the input, the reference output, and the model output, and prompt it to decide which output is better. We use a variant of the prompts from (Rafailov et al., 2023) for summarisation, and alpaca_eval (Li et al., 2023) with the standard prompts for instruction-following. See Appendix D for the precise prompts and other details. This gives us a *percentage win rate of the model being evaluated versus the human-annotated reference output*, which we refer to as preference vs reference (**PvR**). In Appendix D.1 we validate that this evaluation is a good proxy for human preferences. We also perform *head-to-head comparisons between two policies*, by prompting GPT-4 in the same way to decide which of two model model outputs is better.

To evaluate out-of-distribution (OOD) generalisation, we specify an in-distribution (ID) test set and one or more out-of-distribution (OOD) test sets for each task, which have inputs drawn from a different distribution. In each of these sets we have evaluation inputs and corresponding reference outputs (produced either by humans in summarisation or `text-davinci-003` in instruction-following).

For summarisation, *the ID test set is the original TL;DR* test set from (Stiennon et al., 2022), and *the OOD test set is the CNN/DailyMail* test set, a dataset of news articles and corresponding summaries (Nallapati et al., 2016). This tests the ability of the model to have learnt the more general skill of summarisation and to apply it in a very different domain.

For instruction following, the *ID test set is a new test set generated in the same way as the training set was for AlpacaFarm, using the AlpacaFarm variant of Self-Instruct* (Dubois et al., 2023; Wang et al., 2023). Regenerating the test set ensures that it was not seen during training or model selection for AlpacaFarm models. For *the first OOD test sets, we use the AlpacaEval evaluation test set proposed in the original paper*. This is a set of inputs taken from a variety of open-source instruction following and dialogue training and evaluation datasets (Bai et al., 2022; Ganguli et al., 2022; Gudibande, 2023; Köpf et al., 2023; Wang et al., 2023; Zheng et al., 2023), curated by Dubois et al. (2023). For an *additional OOD test set, we generate a set of Sequential Instructions using an adjusted Self-Instruct protocol*. These instructions contain multiple steps in a single input, often building on each other, and require the model to ensure that they complete all the steps to produce a satisfactory outcome. For more details on this dataset see Appendix H.

[1] `https://platform.openai.com/docs/guides/gpt`

We also report the *generalisation gap* (the difference between in-distribution and out-of-distribution performance) which provides a measure of generalisation specifically. Lower generalisation gaps imply the model generalises better, as model performance does not drop as much when the model is evaluated out of distribution. For the head-to-head comparisons between models, we look at the change in head-to-head winrate to give different evaluation of generalisation; the model whose winrate increases generalises better.

5.2 Diversity Evaluation

To evaluate the output diversity of trained policies, we use several diversity measures which are well-supported by prior work, namely **distinct N-grams** (Li et al., 2016), **Sentence-BERT embedding cosine similarity** (Reimers & Gurevych, 2019) and **NLI diversity** (Stasaski & Hearst, 2022). All of these metrics have been shown to align well with human diversity evaluations and with underlying diversity generators by Tevet & Berant (2021).

Each diversity metric D takes a set of model outputs O, and produces a scalar score representing how diverse the set is. *Distinct N-grams* counts the number of distinct N-grams (averaging over $n = 1 \ldots_5$) in the set of outputs, and following (Liu et al., 2022) we use the expectation-adjusted distinct N-grams (EAD) formula to remove the bias towards shorter outputs. The *Sentence-BERT* metric embeds each element of the output set using a sentence transformer, and then measures the average cosine similarity between the embeddings. The metric is then 1 minus the average similarity.

The *NLI diversity* metric measures the number of entailments and contradictions between pairs of elements in the output set using a natural language inference (NLI) model and rates an output set as more diverse if it produces more contradictions and fewer entailments. We pass sentences sampled from elements of the output set (rather than complete elements) to the NLI model so that the inputs are closer to the model's training distribution.

For each policy π we produce a set of K outputs from the model for each of a subset of N inputs from the test set, sampling with temperature 1:

$$\text{for } i = 1 \ldots N : \mathcal{O}^i_\pi := \{y_j \sim \pi(y|x_i) | j = 1 \ldots K\}.$$

In this work we use $K = 16$, $N = 500$. For a diversity metric D we then evaluate the **per-input diversity** which is defined as the average diversity of the output sets over inputs (i.e. the diversity of $\pi(y|x)$), as well as **across-input diversity** defined as the diversity of outputs across inputs (i.e. the diversity of $\pi(y)$):

$$\text{PerInputDiversity}_D(\pi) := \frac{1}{N}\sum_{i=1}^{N} D(\mathcal{O}^i_\pi) \quad (2) \qquad \text{CrossInputDiversity}_D(\pi) := D\left(\bigcup_{i=1}^{N} \mathcal{O}^i_\pi[1]\right) \quad (3)$$

Here $\mathcal{O}^i_\pi[1]$ is the first element of the set $\mathcal{O}^i_\pi$. For conciseness, we refer to expectation-adjusted distinct N-grams, sentence-BERT average cosine similarity and NLI diversity as EAD, Sent BERT and NLI respectively. We can view them as measuring *syntactic, semantic and logical diversity*, and hence using all of them ensures that we are evaluating diversity in a wide range of ways.

6 Experimental Results

In this section we present the results of our experiments on generalisation and output diversity. In general we find that RLHF performs better than SFT in an absolute sense both ID and OOD, but generalisation gap and head-to-head metrics tell a more nuanced story. However, RLHF does reduce output diversity substantially in the per-input setting, and still reduces it to a lesser extent in the cross-input setting. These conclusions are supported by similar results for OPT models across a range of model scales in the summarisation task, presented in Appendix J.

6.1 Generalisation

Summarisation. Fig. 2 shows the *GPT4-PvR* for SFT, Bo16 and RLHF policies trained on the summarisation task, showing the ID (TL;DR) and OOD (CNN/DailyMail) performance and the