# 2024 AI - HW#6 Reinforcement Learning with Human Feedback

生醫電資所 碩一 **r12945040** 郭思言

## ♦ Provide a brief description and comparison of DPO and ORPO

### ✧ Direct Preference Optimization (DPO)

Direct Preference Optimization (DPO) offers a streamlined and effective approach to aligning large-scale language models with human preferences, bypassing the complexities of traditional RL methods while maintaining or improving performance. This method makes it easier to control and steer the behavior of LMs in a computationally efficient and stable manner.

DPO simplifies the preference learning process by simplified optimization. Instead of using RL, DPO uses a binary cross-entropy objective to directly optimize the LM to adhere to human preferences. This approach is derived from re-parameterizing the reward model in a way that the optimal policy can be extracted in closed form.

DPO also incorporates a dynamic, per-example importance weight that prevents model degeneration, which can occur with a naive probability ratio objective.

#### Mechanisms of DPO:

DPO leverages a theoretical preference model, such as the Bradley-Terry model, to measure how well a reward function aligns with empirical preference data. It reformulates the RL objective to directly optimize the policy. This reformulation avoids fitting an explicit reward model while still aligning with human preferences.

**Key Insight**: By mapping reward functions to optimal policies analytically, DPO transforms a loss function over reward functions into a loss function over policies, allowing for direct optimization using a simple binary cross-entropy objective.

### DPO Methodology:

1. Start with RL Objective: DPO begins with the standard RL objective under a general reward function.
2. Reparameterization: The reward function is re-parameterized in terms of the optimal policy, reference policy, and partition function.
3. Preference Model Application: Using models like Bradley-Terry, the probability of human preference data is expressed in terms of the optimal policy.
4. Maximum Likelihood Objective: A maximum likelihood objective is formulated for a parameterized policy, fitting an implicit reward using an alternative parameterization.

### Benefits of DPO:

1. Simplicity: DPO is straightforward to implement and does not require RL-specific components.
2. Effectiveness: It performs at least as well as existing methods in tasks like sentiment modulation, summarization, and dialogue.
3. Efficiency: DPO reduces computational costs and simplifies the training process by avoiding the complexity of RL-based methods.

## ✧ Odds Ratio Preference Optimization (ORPO)

ORPO eliminates the need for an additional preference alignment phase and a reference model, simplifying the process. Empirical and theoretical analyses demonstrate that ORPO effectively contrasts favored and disfavored styles during SFT. ORPO's efficacy is validated across various LM sizes, outperforming state-of-the-art models with larger parameters on several benchmarks. Code and model checkpoints are released to support reproducibility. Unlike previous methods, ORPO does not require an SFT warm-up stage or a reference model, making it resource-efficient.

### ORPO Mechanisms

ORPO introduces an odds ratio-based penalty to the conventional negative log-likelihood (NLL) loss, which differentiates between favored and disfavored responses. The odds ratio reflects the likelihood of generating a preferred response over a rejected one, guiding the model to align with desired preferences during SFT.

### Objective Function of ORPO:

The ORPO objective function consists of two components:

1. SFT Loss (LSFT): Conventional NLL loss to maximize the likelihood of generating reference tokens.
2. Relative Ratio Loss (LOR): Maximizes the odds ratio between the likelihood of generating preferred and disfavored responses.

### Gradient of ORPO:

The gradient of the ORPO loss comprises terms that penalize wrong predictions and contrast between chosen and rejected responses, ensuring efficient parameter updates.

### Comparison to Probability Ratio:

ORPO uses the odds ratio instead of the probability ratio, offering better stability and sensitivity to the model's preference understanding.

## ✧ Comparison

1. **Algorithm Basis:** DPO is based on probability ratios, while ORPO uses odds ratios.
2. **Efficiency:** ORPO is more efficient due to the elimination of the SFT warm-up stage and the reference model.
3. **Stability:** ORPO provides more stable preference alignment, avoiding extreme penalties for disfavored responses.
4. **Implementation Complexity:** ORPO simplifies the implementation by combining preference alignment directly within the SFT phase.

ORPO offers a more streamlined, stable, and resource-efficient approach to preference alignment compared to DPO.

# ♦ Briefly describe LoRA

## ✧ Low Rank Adaptation (LoRA)

LoRA is a technique used in the context of fine-tuning large language models. It aims to reduce the computational cost and memory footprint of adapting pre-trained models to specific tasks. The core idea behind LoRA is to factorize the weight updates into low-rank matrices, which significantly

decreases the number of parameters that need to be learned during fine-tuning.

## Key Concepts:

1. **Low-Rank Decomposition:** In LoRA, the weight updates during fine-tuning are approximated by the product of two smaller matrices (low-rank matrices). This decomposition reduces the number of parameters and thus the computational resources required for training.
2. **Efficiency:** By focusing on low-rank adaptations, LoRA enables efficient fine-tuning of very large models without the need for extensive computational resources. This makes it feasible to fine-tune models on smaller hardware setups, such as GPUs with limited memory.
3. **Applications:** LoRA is particularly useful for scenarios where deploying large models is constrained by hardware limitations. It can be applied to various natural language processing tasks, such as text classification, question answering, and machine translation.

## How It Works:

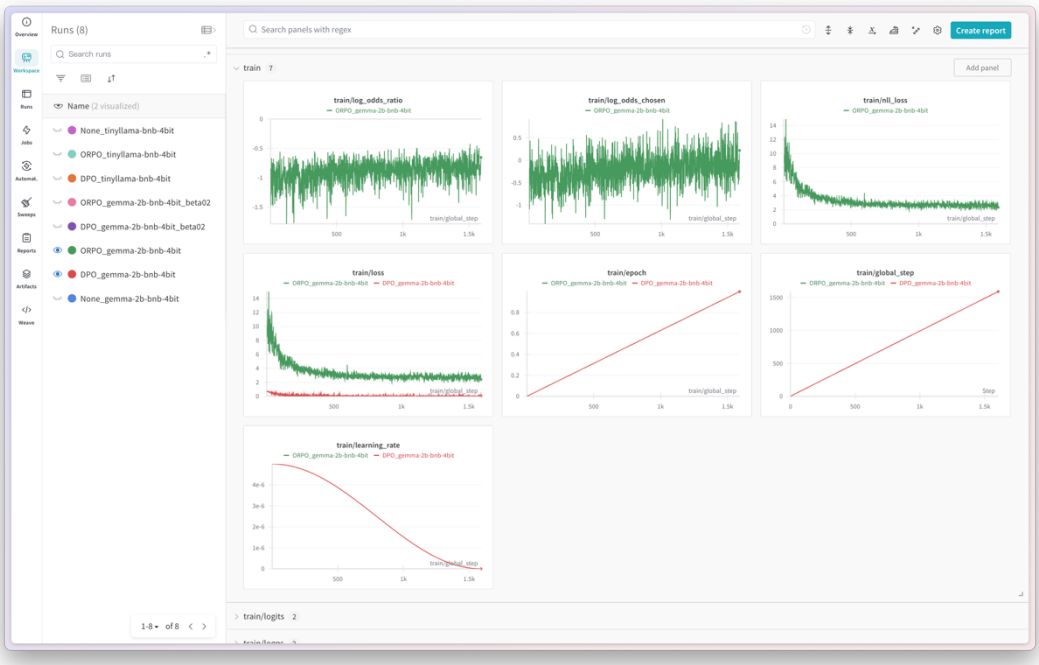
1. **Weight Adaptation:** Traditional fine-tuning adjusts the full set of weights in a pre-trained model. LoRA, on the other hand, introduces low-rank matrices to approximate these weight changes, significantly reducing the number of parameters that need to be updated.
2. **Implementation:** During the fine-tuning process, the original weights of the model are kept frozen. The low-rank matrices are inserted in such a way that their product represents the adaptation needed for the specific task.

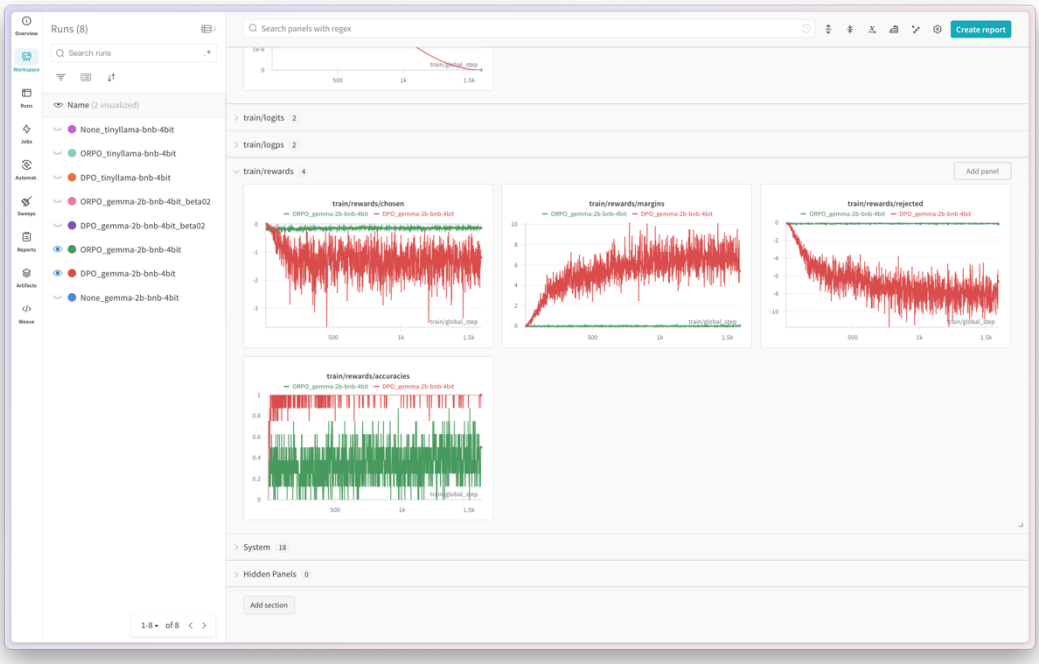
## Benefits:

1. **Reduced Computational Cost:** LoRA reduces the number of trainable parameters, leading to lower computational costs and faster training times.
2. **Memory Efficiency:** By using low-rank matrices, the memory footprint of the model during fine-tuning is minimized, allowing larger models to be fine-tuned on hardware with limited memory.
3. **Flexibility:** LoRA can be applied to various types of models and tasks, providing a flexible solution for model adaptation.

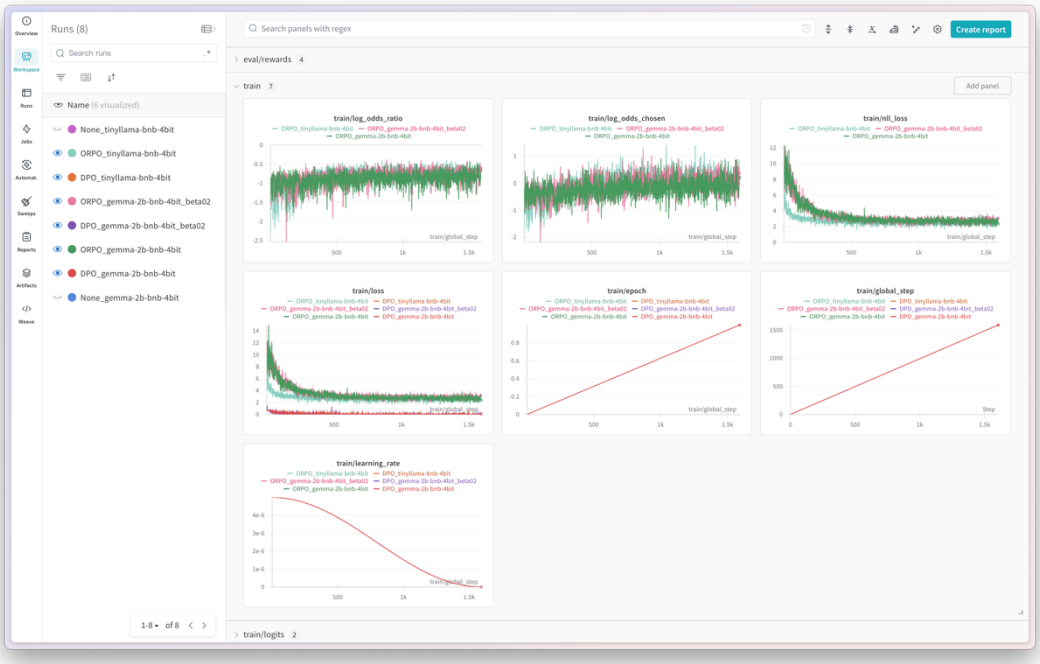
♦ **Plot your training curve by W&B, including both loss and rewards.**

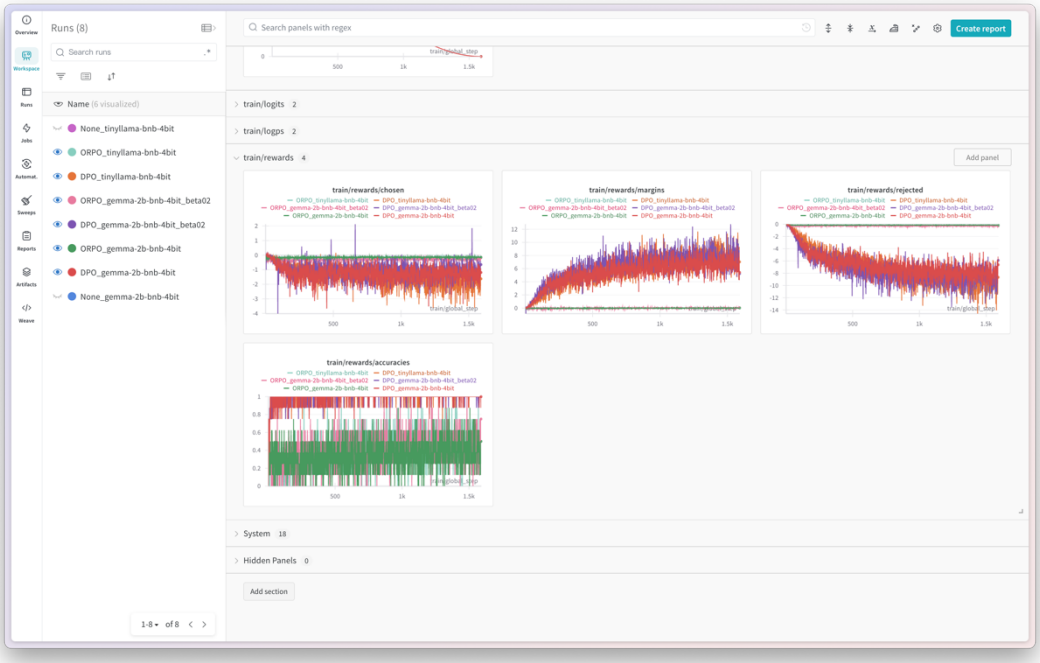
✧ **Base setting:**



## ✧ All Experiments:



## ♦ Comparison and analysis of results (before & after DPO & after ORPO)

### ✧ Initial Output (Before Fine-tuning)

1. **Repetition:** The initial responses tend to repeat the same answer multiple times, especially noticeable in the response to the question about the first planet in the solar system (Mercury).
2. **Conciseness:** The responses are generally concise but sometimes too brief, lacking detailed information or elaboration (e.g., the library card response).
3. **Correctness:** The answers provided are correct but often not sufficiently detailed.

### ✧ Fine-tuning with DPO

1. **Improved Detail:** Responses become more detailed. For example, the answer about the visible spectrum mentions the continuous range and divides the colors correctly.
2. **Consistency:** Responses to repeated questions (e.g., about Mercury) are consistent but still repetitive.
3. **Clarity:** The responses are clearer and provide more context (e.g., the electric vehicle response).

### ✧ Fine-tuning with ORPO

1. **Specificity:** The responses are more specific and directly address the questions. For instance, the colors in the rainbow are listed accurately.
2. **Detail and Depth:** Responses are more comprehensive and informative. For example, the photosynthesis response is well-detailed, explaining the process and its significance.
3. **Reduction in Repetition:** There is a noticeable reduction in repetitive answers compared to the initial output and DPO fine-tuning. The answers are more varied and context-specific.
4. **Practical Advice:** The responses, such as the one on why college students should get a library card, provide practical and useful advice, highlighting the benefits effectively.

### ✧ Analysis

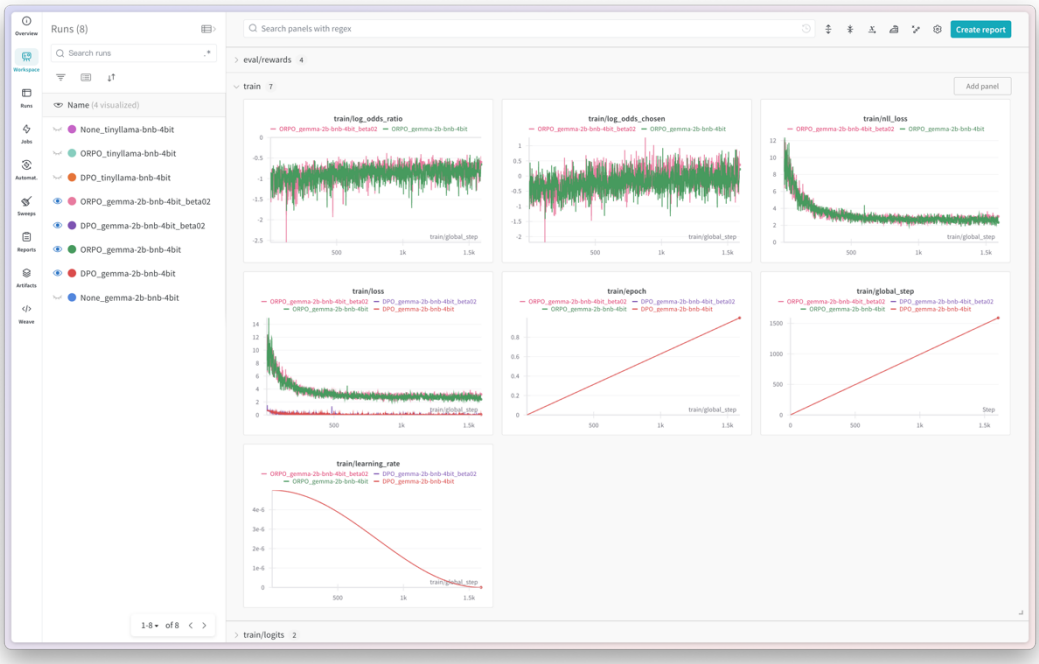
1. **Repetition:** The initial model struggles with repeating answers, particularly in straightforward factual questions. Fine-tuning with DPO and ORPO reduces this repetition significantly, with ORPO providing the best improvement.
2. **Detail:** Both fine-tuning methods improve the level of detail in responses, but ORPO fine-tuning adds more depth and context, making the answers more informative.
3. **Practicality:** ORPO fine-tuning responses are more practical and user-oriented. They not only provide the correct answers but also add value by explaining the reasons and benefits comprehensively.
4. **Clarity and Consistency:** Both fine-tuning methods improve clarity and consistency, but ORPO fine-tuning shows the most significant enhancement, providing clear, concise, and informative responses.

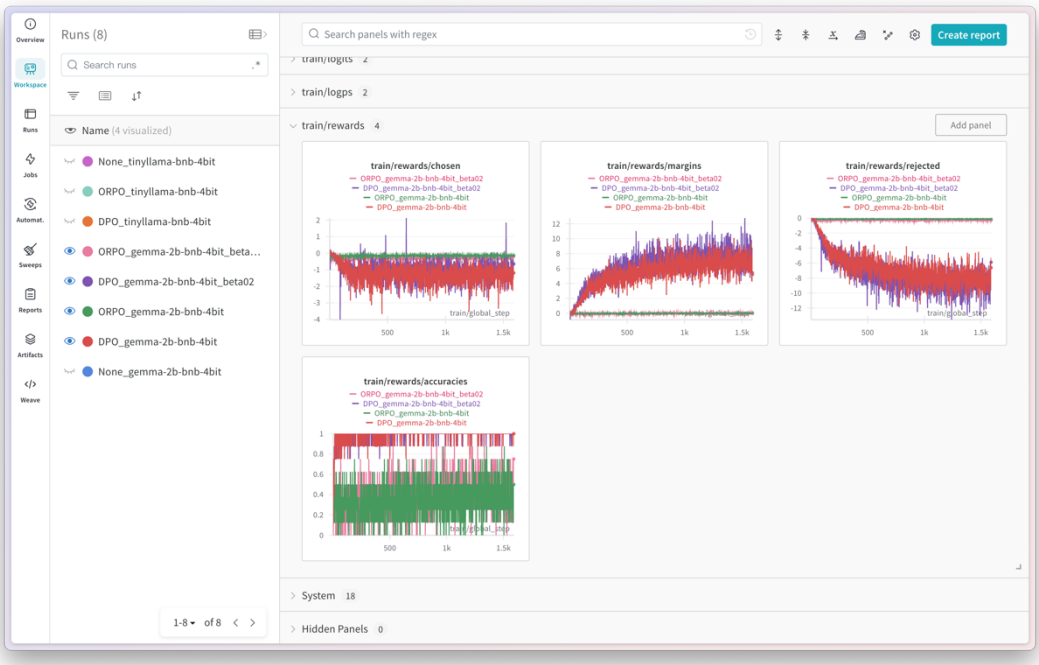
## ♦ Extra Experiments

### ✧ Base settings:

- **Environment:** Since there are no hardware resources that meet the requirements for me to create environment for unsloth, I used “Google Colab Pro”.
- **GPU:** L4 GPU
- **Selected model:** “unsloth/gemma-2b-bnb-4bit”
- **Batch size:** 2
- **Eval batch size:** 2
- **Gradient accumulation steps:** 4
- **Learning rate:** 5e-6
- **Max steps:** 0
- **Num epoch:** 1
- **Optimizer:** paged_adamw_8bit
- **Weight decay:** 0
- **Max grad norm:** 0
- **Warm up ratio:** 0
- **Beta:** 0.1
- **Max length:** 512
- **Max prompt length: 256**
- **Seed:** 2024

## ✧ Experiment 1: beta = 0.1 / 0.2



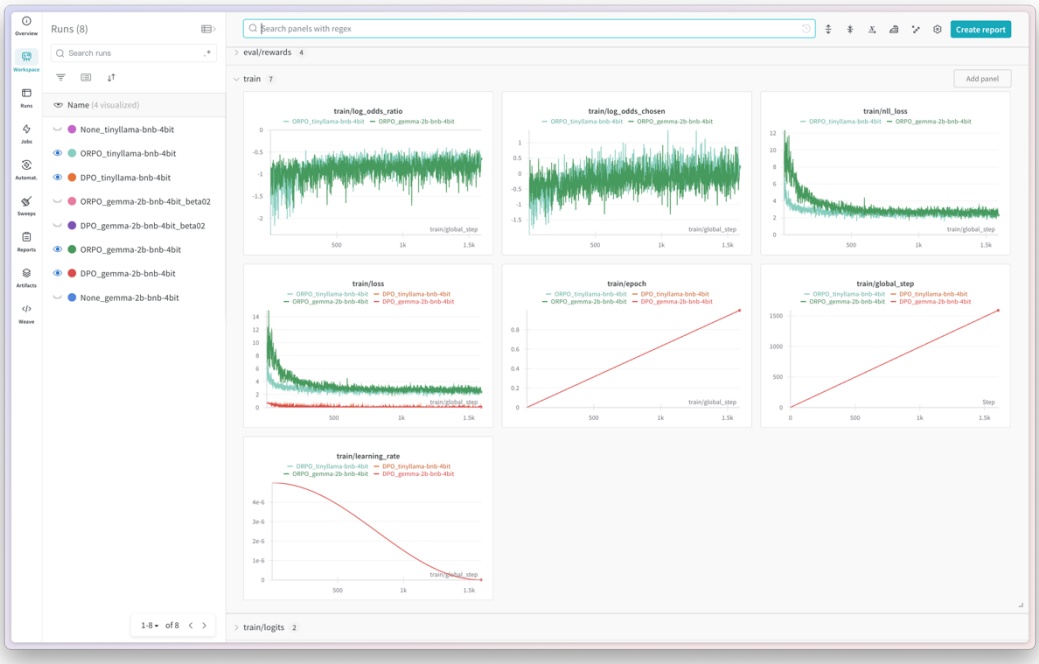



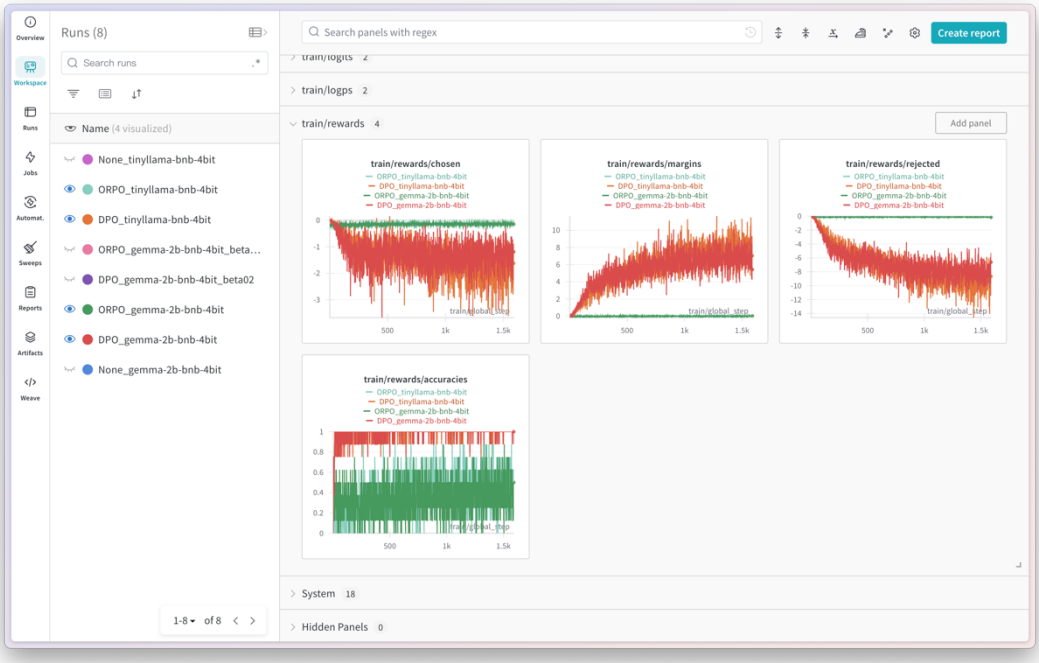

### Comparison and Analysis:

1. With different beta values, from the training curve, we can see that in almost all the training curves, there are no noticeable differences; the training curves are almost overlapping.
2. In the train/rewards/margins chart, we can observe that fine-tuning with DPO with beta=0.2 slightly has higher values than the beta=0.1 model.

3. In the train/rewards/rejected chart, we can observe that fine-tuning with DPO with beta=0.2 slightly has lower values than the beta=0.1 model.
4. The generated texts from the given prompts appear quite similar in their results.

✧ **Experiment 2: model =** “unsloth/gemma-2b-bnb-4bit” / “unsloth/tinyllama-bnb-4bit”



## Comparison and Analysis of the generated texts from the provided prompts before DPO & ORPO finetuning:

**Gemma:**

- **Strengths:** Provides specific and accurate responses to prompts.
- **Weaknesses:** Tends to repeat responses without much variation, especially noticeable in simple fact-based questions.

**TinyLLama:**

- **Strengths:** Attempts to handle more diverse queries.
- **Weaknesses:** Frequently provides repetitive or incorrect information, struggles with producing coherent and varied outputs.

## Comparison and Analysis of the training curve after finetuning with DPO & ORPO:

1. From the train/log_odds_ratio and train/log_odds_chosen plots, it can be observed that for the ORPO model, the Gemma curve remains relatively flat, while the TinyLlama curve starts low and gradually increases to the same level as Gemma.
2. In the train/nll_loss and train/loss plots, it is evident that for the ORPO model, Gemma has a relatively high initial loss that gradually converges, whereas TinyLlama starts with a lower initial loss and converges to the same level.
3. From the train/rewards/chosen plot, it can be seen that for the DPO model, Gemma's values remain stable throughout the training process, while TinyLlama's values gradually decrease.

## Comparison and Analysis of the generated texts from the provided prompts after DPO finetuning:

- **Gemma**: Provides more accurate, detailed, and contextually appropriate responses, though it occasionally struggles with numerical accuracy and tends to repeat responses verbatim.
- **Tinyllama**: Offers less detailed and sometimes inaccurate responses, frequently repeating answers without additional context or variety, and occasionally providing incomplete information.

**Observation:** While both models show repetitive patterns, Gemma generally offers higher quality, more detailed, and contextually accurate

responses compared to Tinyllama. Tinyllama, on the other hand, demonstrates significant limitations in accuracy and detail, making Gemma the better performer overall.

## Comparison and Analysis of the generated texts from the provided prompts after ORPO finetuning:

- **Gemma Model:** Provides more accurate and coherent responses but has some issues with redundancy and occasional format deviations.
- **TinyLlama Model:** Struggles with repetition, accuracy, and coherence, often failing to deliver the required information effectively.

**Observation:** while both models have areas for improvement, the Gemma model demonstrates better overall performance in terms of accuracy and coherence compared to the TinyLlama model.