UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

Direct Preference Optimization

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon
Christopher D. Manning, Chelsea Finn

Sumuk Shashidhar

University of Illinois, Urbana Champaign

June 22, 2024

Overview

Motivation

Why Preference Learning Matters

- Many scenarios where we want to emphasize sections of training data during fine-tuning

Why Preference Learning Matters

- Many scenarios where we want to emphasize sections of training data during fine-tuning
- Example: Biasing the model towards producing good code, even when good code is rare in the training data

Why Preference Learning Matters

- Many scenarios where we want to emphasize sections of training data during fine-tuning
- Example: Biasing the model towards producing good code, even when good code is rare in the training data
- Preference learning is a crucial problem to address

The Future of Language Model Improvement

- Personal opinion: Preference learning is the last great frontier for LLM improvement

The Future of Language Model Improvement

- Personal opinion: Preference learning is the last great frontier for LLM improvement
- Focus most research efforts on preference learning

The Future of Language Model Improvement

- Personal opinion: Preference learning is the last great frontier for LLM improvement
- Focus most research efforts on preference learning
- GPT-4 class models are already highly capable and commoditized (e.g., Google Gemini, Claude 3 Opus, Mistral Next)

The Importance of Fine-Tuning

- Tasks are often easily accomplished by LLMs, with differences in performance being subtle

The Importance of Fine-Tuning

- Tasks are often easily accomplished by LLMs, with differences in performance being subtle
- Example: Using Claude 3 for various tasks due to its human-like reasoning

The Importance of Fine-Tuning

- Tasks are often easily accomplished by LLMs, with differences in performance being subtle
- Example: Using Claude 3 for various tasks due to its human-like reasoning
- GPT-4 likely has similar reasoning skills but is fine-tuned for a different audience

The Importance of Fine-Tuning

- Tasks are often easily accomplished by LLMs, with differences in performance being subtle
- Example: Using Claude 3 for various tasks due to its human-like reasoning
- GPT-4 likely has similar reasoning skills but is fine-tuned for a different audience
- All GPT-4 class LLMs generally succeed on tasks given sufficient information

Scope of the Presentation

- Focus on high-level concepts rather than deep mathematical details

Scope of the Presentation

- Focus on high-level concepts rather than deep mathematical details
- Aiming to provide a clear overview of the paper's significance and implications

Scope of the Presentation

- Focus on high-level concepts rather than deep mathematical details
- Aiming to provide a clear overview of the paper's significance and implications

Scope of the Presentation

- Focus on high-level concepts rather than deep mathematical details
- Aiming to provide a clear overview of the paper's significance and implications

Goal

Goal: Simplifying with Binary Cross-Entropy Loss

- Aim to simplify the optimization objective using Binary Cross-Entropy (BCE) loss

Goal: Simplifying with Binary Cross-Entropy Loss

- Aim to simplify the optimization objective using Binary Cross-Entropy (BCE) loss
- BCE loss measures the dissimilarity between the model's predictions and the target preferences

Goal: Simplifying with Binary Cross-Entropy Loss

- Aim to simplify the optimization objective using Binary Cross-Entropy (BCE) loss
- BCE loss measures the dissimilarity between the model's predictions and the target preferences
- Enables the model to directly learn from human preferences without complex reward modeling

Goal: Overcoming the Limitations of RLHF

- Reinforcement Learning from Human Feedback (RLHF) is expensive and resource-intensive

Goal: Overcoming the Limitations of RLHF

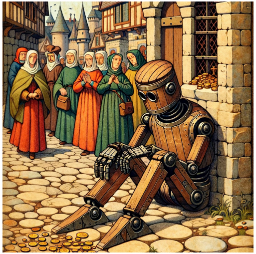
- Reinforcement Learning from Human Feedback (RLHF) is expensive and resource-intensive
- RLHF requires training multiple language models, extensive sampling, and iterative refinement [Raf+23]

Goal: Overcoming the Limitations of RLHF

- Reinforcement Learning from Human Feedback (RLHF) is expensive and resource-intensive
- RLHF requires training multiple language models, extensive sampling, and iterative refinement [Raf+23]

Goal: Overcoming the Limitations of RLHF

- Reinforcement Learning from Human Feedback (RLHF) is expensive and resource-intensive
- RLHF requires training multiple language models, extensive sampling, and iterative refinement [Raf+23]



Directly Adhering to Human Preferences

- Develop a method that directly incorporates human preferences into the model

Directly Adhering to Human Preferences

- Develop a method that directly incorporates human preferences into the model
- Avoid the need for explicit reward modeling or reinforcement learning

Directly Adhering to Human Preferences

- Develop a method that directly incorporates human preferences into the model
- Avoid the need for explicit reward modeling or reinforcement learning
- Aim to achieve performance at least as good as existing methods like RLHF

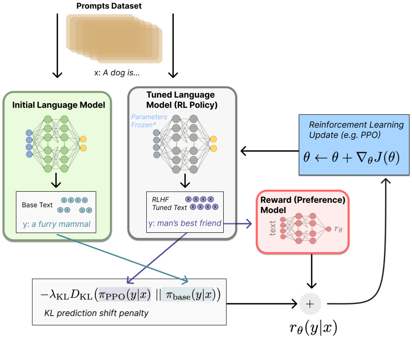
Directly Adhering to Human Preferences

- Develop a method that directly incorporates human preferences into the model
- Avoid the need for explicit reward modeling or reinforcement learning
- Aim to achieve performance at least as good as existing methods like RLHF
- Reduce the computational burden and complexity associated with existing methods

Prior Work

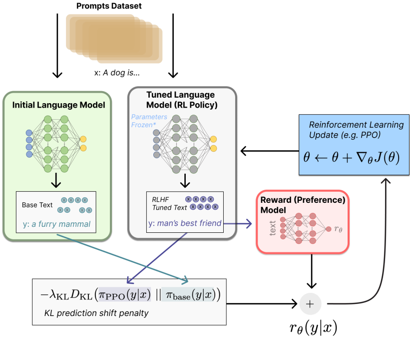
Reinforcement Learning from Human Feedback (RLHF)

- RLHF is a prominent approach for aligning language models with human preferences



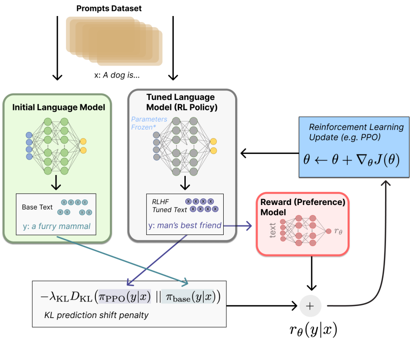
Reinforcement Learning from Human Feedback (RLHF)

- RLHF is a prominent approach for aligning language models with human preferences
- Involves training a reward model to estimate the quality of generated outputs



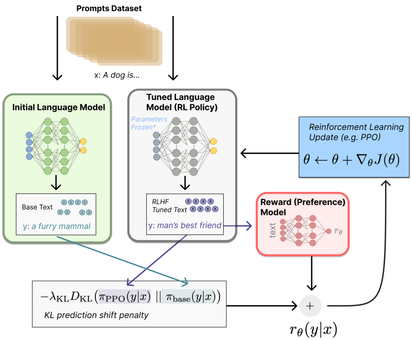
Reinforcement Learning from Human Feedback (RLHF)

- RLHF is a prominent approach for aligning language models with human preferences
- Involves training a reward model to estimate the quality of generated outputs
- Reinforcement learning is then used to fine-tune the language model based on the reward model



Reinforcement Learning from Human Feedback (RLHF)

- RLHF is a prominent approach for aligning language models with human preferences
- Involves training a reward model to estimate the quality of generated outputs
- Reinforcement learning is then used to fine-tune the language model based on the reward model
- Examples: InstructGPT [Ouy+22], Anthropic's Constitutional AI[Bai+22]



Reinforcement Learning with Human Feedback (RLHF)

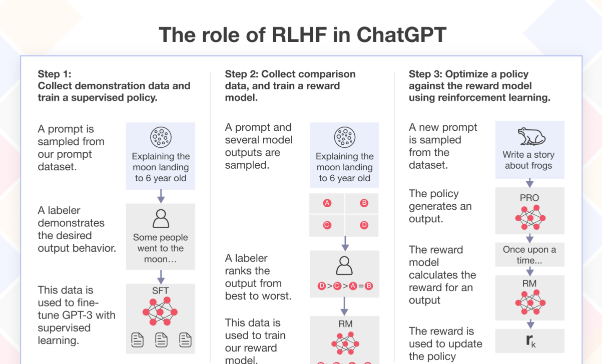
- RLHF is a method for fine-tuning language models using human preferences
- It involves a two-stage process:
  1. Collect human feedback on model outputs
  2. Use the feedback to fine-tune the model using reinforcement learning

Stage 1: Collecting Human Feedback

- Generate a set of prompts and multiple outputs from the base model for each prompt
- Ask human raters to compare the outputs and select the best one
- Collect a dataset of prompts, outputs, and human preferences

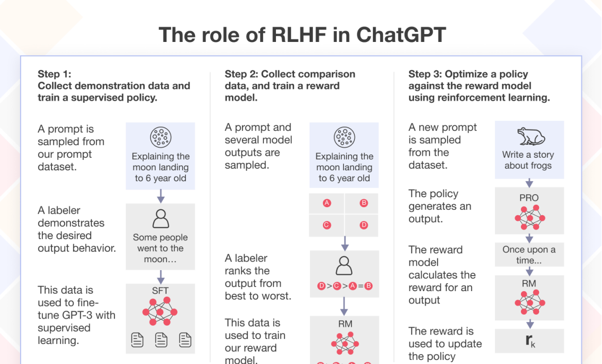
Stage 2: Fine-tuning with Reinforcement Learning

- Use the collected dataset to define a reward function based on human preferences



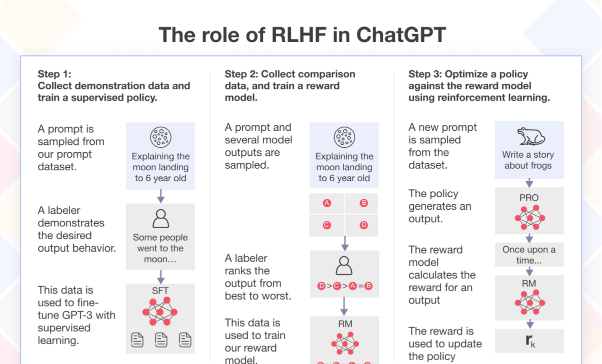
Stage 2: Fine-tuning with Reinforcement Learning

- Use the collected dataset to define a reward function based on human preferences
- Fine-tune the model using reinforcement learning to maximize the reward function



Stage 2: Fine-tuning with Reinforcement Learning

- Use the collected dataset to define a reward function based on human preferences
- Fine-tune the model using reinforcement learning to maximize the reward function
- The model learns to generate outputs that align with human preferences



Preference Learning in Language Models

- Various approaches have been proposed to incorporate human preferences into language models

Preference Learning in Language Models

- Various approaches have been proposed to incorporate human preferences into language models
- Reward modeling: Learning a reward function that captures human preferences [Sti+22]

Preference Learning in Language Models

- Various approaches have been proposed to incorporate human preferences into language models
- Reward modeling: Learning a reward function that captures human preferences [Sti+22]
- Preference-based reinforcement learning: Directly optimizing the language model based on human feedback [Chr+23]

Preference Learning in Language Models

- Various approaches have been proposed to incorporate human preferences into language models
- Reward modeling: Learning a reward function that captures human preferences [Sti+22]
- Preference-based reinforcement learning: Directly optimizing the language model based on human feedback [Chr+23]

Preference Learning in Language Models

- Various approaches have been proposed to incorporate human preferences into language models
- Reward modeling: Learning a reward function that captures human preferences [Sti+22]
- Preference-based reinforcement learning: Directly optimizing the language model based on human feedback [Chr+23]

These approaches often rely on explicit reward modeling or reinforcement learning, which can be computationally expensive and complex to implement. All of them are also multi stage, unlike DPO's single stage.

Binary Classification for Preference Learning

- Binary classification has been used in preference learning for other domains

Binary Classification for Preference Learning

- Binary classification has been used in preference learning for other domains
- Examples: Learning to rank [Joa02], collaborative filtering, etc.

Efficient Fine-Tuning Methods

- Researchers have explored efficient methods for fine-tuning large language models.

Efficient Fine-Tuning Methods

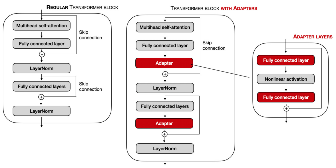
- Researchers have explored efficient methods for fine-tuning large language models.
- Examples: Adapter layers [Hou+19], LoRA [Hu+21], Prefix-tuning [LL21].

Efficient Fine-Tuning Methods

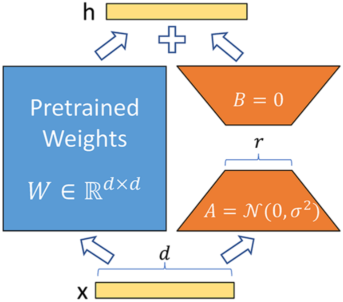
- Researchers have explored efficient methods for fine-tuning large language models.
- Examples: Adapter layers [Hou+19], LoRA [Hu+21], Prefix-tuning [LL21].

Efficient Fine-Tuning Methods

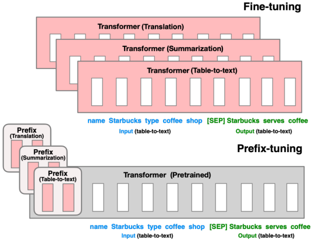
- Researchers have explored efficient methods for fine-tuning large language models.
- Examples: Adapter layers [Hou+19], LoRA [Hu+21], Prefix-tuning [LL21].



Adapter Layer



LoRA



Prefix-tuning

Fine-tuning Methods

Adapter Layers

- Add new layers between existing layers
- Only train the new layers

Prefix Tuning

- Prepend a learnable prefix to the input
- Only optimize the prefix during fine-tuning

LoRAs

- Add low-rank matrices to existing layers
- Only train the low-rank matrices

Method

Main Intuition

- Relative Preferences are easier to gather, compared to complex, expert demonstrations.

Main Intuition

- Relative Preferences are easier to gather, compared to complex, expert demonstrations.
- Instead of learning a reward, and then optimizing, it is easier to do this in one stage by **transforming a loss function over rewards to a loss function over policies**

Overview

- Direct Preference Optimization (DPO) aims to fine-tune language models directly based on human preferences

Overview

- Direct Preference Optimization (DPO) aims to fine-tune language models directly based on human preferences
- Formulates preference learning as a binary classification problem

Overview

- Direct Preference Optimization (DPO) aims to fine-tune language models directly based on human preferences
- Formulates preference learning as a binary classification problem
- Optimizes the model using Binary Cross-Entropy (BCE) loss

Method: Problem Formulation

- Given a pair of text sequences (x_1, x_2), the goal is to predict which sequence is preferred

Method: Problem Formulation

- Given a pair of text sequences (x_1, x_2), the goal is to predict which sequence is preferred
- Human preferences are represented as binary labels $y \in \{0, 1\}$

Method: Problem Formulation

- Given a pair of text sequences (x_1, x_2), the goal is to predict which sequence is preferred
- Human preferences are represented as binary labels $y \in \{0, 1\}$
- The language model f_θ assigns a score to each sequence, denoted as $s_1 = f_\theta(x_1)$ and $s_2 = f_\theta(x_2)$

Method: Binary Cross-Entropy Loss

$$\mathcal{L}(\theta) = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log(\sigma(s_1^i - s_2^i)) + (1 - y_i)\log(1 - \sigma(s_1^i - s_2^i))\right]$$

- $\mathcal{L}(\theta)$: Binary Cross-Entropy loss function

Method: Binary Cross-Entropy Loss

$$\mathcal{L}(\theta) = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log(\sigma(s_1^i - s_2^i)) + (1 - y_i)\log(1 - \sigma(s_1^i - s_2^i))\right]$$

- $\mathcal{L}(\theta)$: Binary Cross-Entropy loss function
- N: Number of preference pairs in the dataset

Method: Binary Cross-Entropy Loss

$$\mathcal{L}(\theta) = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log(\sigma(s_1^i - s_2^i)) + (1 - y_i)\log(1 - \sigma(s_1^i - s_2^i))\right]$$

- $\mathcal{L}(\theta)$: Binary Cross-Entropy loss function
- N: Number of preference pairs in the dataset
- y_i: Binary label for the i-th preference pair

Method: Binary Cross-Entropy Loss

$$\mathcal{L}(\theta) = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log(\sigma(s_1^i - s_2^i)) + (1 - y_i)\log(1 - \sigma(s_1^i - s_2^i))\right]$$

- $\mathcal{L}(\theta)$: Binary Cross-Entropy loss function
- N: Number of preference pairs in the dataset
- y_i: Binary label for the i-th preference pair
- s_1^i, s_2^i: Scores assigned by the model to the sequences in the i-th pair

Method: Binary Cross-Entropy Loss

$$\mathcal{L}(\theta) = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log(\sigma(s_1^i - s_2^i)) + (1 - y_i)\log(1 - \sigma(s_1^i - s_2^i))\right]$$

- $\mathcal{L}(\theta)$: Binary Cross-Entropy loss function
- N: Number of preference pairs in the dataset
- y_i: Binary label for the i-th preference pair
- s_1^i, s_2^i: Scores assigned by the model to the sequences in the i-th pair
- σ: Sigmoid function to map the score difference to a probability

Method: Sigmoid Function

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

- The sigmoid function maps the score difference to a probability between 0 and 1

Method: Sigmoid Function

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

- The sigmoid function maps the score difference to a probability between 0 and 1
- It allows the model to interpret the score difference as a preference probability

Method: Sigmoid Function

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

- The sigmoid function maps the score difference to a probability between 0 and 1
- It allows the model to interpret the score difference as a preference probability
- A higher probability indicates a stronger preference for the first sequence in the pair

Method: Optimization

- The model parameters θ are optimized using gradient descent to minimize the BCE loss

Method: Optimization

- The model parameters θ are optimized using gradient descent to minimize the BCE loss
- The optimization process adjusts the model's weights to align its predictions with human preferences

Method: Optimization

- The model parameters θ are optimized using gradient descent to minimize the BCE loss
- The optimization process adjusts the model's weights to align its predictions with human preferences
- Stochastic gradient descent (SGD) or its variants (e.g., Adam) can be used for optimization

Method: Training Procedure

1. Collect a dataset of human preference pairs (x_1, x_2, y)

Method: Training Procedure

1. Collect a dataset of human preference pairs (x_1, x_2, y)
2. Initialize the language model f_θ with pre-trained weights

Method: Training Procedure

1. Collect a dataset of human preference pairs (x_1, x_2, y)
2. Initialize the language model f_θ with pre-trained weights
3. Iterate for a fixed number of epochs or until convergence:

Method: Training Procedure

1. Collect a dataset of human preference pairs (x_1, x_2, y)
2. Initialize the language model f_θ with pre-trained weights
3. Iterate for a fixed number of epochs or until convergence:
    - Sample a batch of preference pairs from the dataset

Method: Training Procedure

1. Collect a dataset of human preference pairs (x_1, x_2, y)
2. Initialize the language model f_θ with pre-trained weights
3. Iterate for a fixed number of epochs or until convergence:
    - Sample a batch of preference pairs from the dataset
    - Compute the scores s_1 and s_2 for each pair using f_θ

Method: Training Procedure

1. Collect a dataset of human preference pairs (x_1, x_2, y)
2. Initialize the language model f_θ with pre-trained weights
3. Iterate for a fixed number of epochs or until convergence:
   - Sample a batch of preference pairs from the dataset
   - Compute the scores s_1 and s_2 for each pair using f_θ
   - Calculate the BCE loss $\mathcal{L}(\theta)$ for the batch

Method: Training Procedure

1. Collect a dataset of human preference pairs (x_1, x_2, y)
2. Initialize the language model f_θ with pre-trained weights
3. Iterate for a fixed number of epochs or until convergence:
   - Sample a batch of preference pairs from the dataset
   - Compute the scores s_1 and s_2 for each pair using f_θ
   - Calculate the BCE loss $\mathcal{L}(\theta)$ for the batch
   - Update the model parameters θ using gradient descent to minimize the loss

Method: Training Procedure

1. Collect a dataset of human preference pairs (x_1, x_2, y)
2. Initialize the language model f_θ with pre-trained weights
3. Iterate for a fixed number of epochs or until convergence:
    - Sample a batch of preference pairs from the dataset
    - Compute the scores s_1 and s_2 for each pair using f_θ
    - Calculate the BCE loss $\mathcal{L}(\theta)$ for the batch
    - Update the model parameters θ using gradient descent to minimize the loss
4. Fine-tuned model f_θ is aligned with human preferences

Fitting a Loss Function over Policies

- Traditional approaches often learn a reward function and then optimize the policy based on the learned rewards

Fitting a Loss Function over Policies

- Traditional approaches often learn a reward function and then optimize the policy based on the learned rewards
- DPO directly optimizes the policy by fitting a loss function over policies instead of rewards

Fitting a Loss Function over Policies

- Traditional approaches often learn a reward function and then optimize the policy based on the learned rewards
- DPO directly optimizes the policy by fitting a loss function over policies instead of rewards
- This approach has several advantages:

Fitting a Loss Function over Policies

- Traditional approaches often learn a reward function and then optimize the policy based on the learned rewards
- DPO directly optimizes the policy by fitting a loss function over policies instead of rewards
- This approach has several advantages:
  - Avoids the need for explicit reward learning, which can be challenging

Fitting a Loss Function over Policies

- Traditional approaches often learn a reward function and then optimize the policy based on the learned rewards
- DPO directly optimizes the policy by fitting a loss function over policies instead of rewards
- This approach has several advantages:
  - Avoids the need for explicit reward learning, which can be challenging
  - Allows for more direct alignment with human preferences

Fitting a Loss Function over Policies

- Traditional approaches often learn a reward function and then optimize the policy based on the learned rewards
- DPO directly optimizes the policy by fitting a loss function over policies instead of rewards
- This approach has several advantages:
  - Avoids the need for explicit reward learning, which can be challenging
  - Allows for more direct alignment with human preferences
  - Enables the model to capture complex and nuanced preferences

Fitting a Loss Function over Policies

- Traditional approaches often learn a reward function and then optimize the policy based on the learned rewards
- DPO directly optimizes the policy by fitting a loss function over policies instead of rewards
- This approach has several advantages:
  - Avoids the need for explicit reward learning, which can be challenging
  - Allows for more direct alignment with human preferences
  - Enables the model to capture complex and nuanced preferences
- The BCE loss function is defined over the policy space, guiding the model towards preferred behaviors

Intuition: BCE over Policy Space I

- In DPO, the BCE loss is defined over the policy space instead of the reward space

Intuition: BCE over Policy Space II

Analogy: Sculpting a Statue: Reward Space

Intuition: BCE over Policy Space III

Analogy: Sculpting a Statue: Policy Space

DPO Update Explanation

- Each DPO update aims to improve the policy π_θ based on human preference data

DPO Update Explanation

- Each DPO update aims to improve the policy π_θ based on human preference data
- The update process can be broken down into the following steps:

DPO Update Explanation

- Each DPO update aims to improve the policy π_θ based on human preference data
- The update process can be broken down into the following steps:
  1. Sample a batch of preference pairs (x_1, x_2, y) from the dataset

DPO Update Explanation

- Each DPO update aims to improve the policy π_θ based on human preference data
- The update process can be broken down into the following steps:
  1. Sample a batch of preference pairs (x_1, x_2, y) from the dataset
  2. Compute the scores s_1 and s_2 for each pair using the language model f_θ

DPO Update Explanation

- Each DPO update aims to improve the policy π_θ based on human preference data
- The update process can be broken down into the following steps:
  1. Sample a batch of preference pairs (x_1, x_2, y) from the dataset
  2. Compute the scores s_1 and s_2 for each pair using the language model f_θ
  3. Calculate the probability of preferring x_1 over x_2 using the sigmoid function:

$$p = \sigma(s_1 - s_2)$$

DPO Update Explanation

- Each DPO update aims to improve the policy π_θ based on human preference data
- The update process can be broken down into the following steps:
  1. Sample a batch of preference pairs (x_1, x_2, y) from the dataset
  2. Compute the scores s_1 and s_2 for each pair using the language model f_θ
  3. Calculate the probability of preferring x_1 over x_2 using the sigmoid function:

$$p = \sigma(s_1 - s_2)$$

  4. Compute the BCE loss for the batch based on the predicted probabilities and true labels

DPO Update Explanation

- Each DPO update aims to improve the policy π_θ based on human preference data
- The update process can be broken down into the following steps:
  1. Sample a batch of preference pairs (x_1, x_2, y) from the dataset
  2. Compute the scores s_1 and s_2 for each pair using the language model f_θ
  3. Calculate the probability of preferring x_1 over x_2 using the sigmoid function:

$$p = \sigma(s_1 - s_2)$$

  4. Compute the BCE loss for the batch based on the predicted probabilities and true labels
  5. Calculate the gradients of the loss with respect to the model parameters θ

DPO Update Explanation

- Each DPO update aims to improve the policy π_θ based on human preference data
- The update process can be broken down into the following steps:
  1. Sample a batch of preference pairs (x_1, x_2, y) from the dataset
  2. Compute the scores s_1 and s_2 for each pair using the language model f_θ
  3. Calculate the probability of preferring x_1 over x_2 using the sigmoid function:

     $$p = \sigma(s_1 - s_2)$$

  4. Compute the BCE loss for the batch based on the predicted probabilities and true labels
  5. Calculate the gradients of the loss with respect to the model parameters θ
  6. Update the model parameters using gradient descent:

     $$\theta \leftarrow \theta - \alpha \nabla_\theta \mathcal{L}(\theta)$$

     where α is the learning rate

DPO Update Explanation

- Each DPO update aims to improve the policy π_θ based on human preference data
- The update process can be broken down into the following steps:
  1. Sample a batch of preference pairs (x_1, x_2, y) from the dataset
  2. Compute the scores s_1 and s_2 for each pair using the language model f_θ
  3. Calculate the probability of preferring x_1 over x_2 using the sigmoid function:

     $$p = \sigma(s_1 - s_2)$$

  4. Compute the BCE loss for the batch based on the predicted probabilities and true labels
  5. Calculate the gradients of the loss with respect to the model parameters θ
  6. Update the model parameters using gradient descent:

     $$\theta \leftarrow \theta - \alpha \nabla_\theta \mathcal{L}(\theta)$$

     where α is the learning rate
- Each update step minimizes the discrepancy between the model's predictions and human preferences, aligning the policy with the desired behaviors

Theoretical Analysis

Convergence

Theorem

Under mild assumptions, the DPO algorithm converges to a globally optimal solution at a rate of $O(\frac{1}{\sqrt{N}})$*, where* N *is the number of preference pairs.*

- The convergence rate depends on the square root of the number of preference pairs

Convergence

Theorem

Under mild assumptions, the DPO algorithm converges to a globally optimal solution at a rate of $O(\frac{1}{\sqrt{N}})$, where N is the number of preference pairs.

- The convergence rate depends on the square root of the number of preference pairs
- Increasing the size of the preference dataset leads to faster convergence

Convergence

Theorem

Under mild assumptions, the DPO algorithm converges to a globally optimal solution at a rate of $O(\frac{1}{\sqrt{N}})$*, where N is the number of preference pairs.*

- The convergence rate depends on the square root of the number of preference pairs
- Increasing the size of the preference dataset leads to faster convergence
- This result ensures the stability and efficiency of the DPO optimization process

Generalization Bounds

Theorem

With high probability, the generalization error of DPO is bounded by $O(\sqrt{\frac{\log(1/\delta)}{N}})$*, where* N *is the number of preference pairs and* δ *is the confidence parameter.*

- The generalization bound provides an upper limit on the expected performance of DPO on unseen preference pairs

Generalization Bounds

Theorem

With high probability, the generalization error of DPO is bounded by $O(\sqrt{\frac{\log(1/\delta)}{N}})$, *where N is the number of preference pairs and* δ *is the confidence parameter.*

- The generalization bound provides an upper limit on the expected performance of DPO on unseen preference pairs
- The bound decreases with the square root of the number of preference pairs

Generalization Bounds

Theorem

With high probability, the generalization error of DPO is bounded by $O(\sqrt{\frac{\log(1/\delta)}{N}})$, where N is the number of preference pairs and δ is the confidence parameter.

- The generalization bound provides an upper limit on the expected performance of DPO on unseen preference pairs
- The bound decreases with the square root of the number of preference pairs
- Factors such as model complexity and data distribution also affect the generalization performance

Connection to Ranking Problems

- DPO can be viewed as a special case of ranking problems with pairwise preferences

Connection to Ranking Problems

- DPO can be viewed as a special case of ranking problems with pairwise preferences
- The BCE loss in DPO is related to the pairwise ranking loss in learning to rank literature

Connection to Ranking Problems

- DPO can be viewed as a special case of ranking problems with pairwise preferences
- The BCE loss in DPO is related to the pairwise ranking loss in learning to rank literature
- This connection allows for the application of theoretical results and algorithms from ranking problems to DPO

Sample Complexity

Theorem

To achieve an error rate of ϵ with probability at least $1-\delta$, DPO requires $O(\frac{1}{\epsilon^2}\log(\frac{1}{\delta}))$ preference pairs.

- The sample complexity result provides an estimate of the number of preference pairs needed for effective learning

Sample Complexity

Theorem

To achieve an error rate of ϵ with probability at least $1 - \delta$, DPO requires $O(\frac{1}{\epsilon^2} \log(\frac{1}{\delta}))$ preference pairs.

- The sample complexity result provides an estimate of the number of preference pairs needed for effective learning
- The required number of pairs grows quadratically with the inverse of the desired error rate

Sample Complexity

Theorem

To achieve an error rate of ϵ with probability at least $1 - \delta$, DPO requires $O(\frac{1}{\epsilon^2} \log(\frac{1}{\delta}))$ preference pairs.

- The sample complexity result provides an estimate of the number of preference pairs needed for effective learning
- The required number of pairs grows quadratically with the inverse of the desired error rate
- This result helps in determining the size of the preference dataset for practical applications

Experimental Setup

Experiments Overview

- Evaluate DPO's ability to train policies directly from preferences

Experiments Overview

- Evaluate DPO's ability to train policies directly from preferences
- Compare efficiency of DPO to common preference learning algorithms (e.g. PPO)

Experiments Overview

- Evaluate DPO's ability to train policies directly from preferences
- Compare efficiency of DPO to common preference learning algorithms (e.g. PPO)
- Evaluate performance on larger models and more difficult RLHF tasks:

Experiments Overview

- Evaluate DPO's ability to train policies directly from preferences
- Compare efficiency of DPO to common preference learning algorithms (e.g. PPO)
- Evaluate performance on larger models and more difficult RLHF tasks:
    - Summarization

Experiments Overview

- Evaluate DPO's ability to train policies directly from preferences
- Compare efficiency of DPO to common preference learning algorithms (e.g. PPO)
- Evaluate performance on larger models and more difficult RLHF tasks:
  - Summarization
  - Dialogue

Experiments Overview

- Evaluate DPO's ability to train policies directly from preferences
- Compare efficiency of DPO to common preference learning algorithms (e.g. PPO)
- Evaluate performance on larger models and more difficult RLHF tasks:
  - Summarization
  - Dialogue
- Minimal hyperparameter tuning needed for DPO to match or outperform baselines

Experimental Setup

- Algorithms learn policy from preference dataset $D = (x^{(i)}, y_w^{(i)}, y_l^{(i)})_{i=1}^{N}$

Experimental Setup

- Algorithms learn policy from preference dataset $D = (x^{(i)}, y_w^{(i)}, y_l^{(i)})_{i=1}^N$
- Tasks:

Experimental Setup

- Algorithms learn policy from preference dataset $D = (x^{(i)}, y_w^{(i)}, y_l^{(i)})_{i=1}^{N}$
- Tasks:
  - Controlled sentiment generation (IMDb movie reviews)

Experimental Setup

- Algorithms learn policy from preference dataset $D = (x^{(i)}, y_w^{(i)}, y_l^{(i)})_{i=1}^{N}$
- Tasks:
  - Controlled sentiment generation (IMDb movie reviews)
  - Summarization (Reddit TL;DR)

Experimental Setup

- Algorithms learn policy from preference dataset $D = (x^{(i)}, y_w^{(i)}, y_l^{(i)})_{i=1}^{N}$
- Tasks:
  - Controlled sentiment generation (IMDb movie reviews)
  - Summarization (Reddit TL;DR)
  - Single-turn dialogue (Anthropic Helpful & Harmless)

Experimental Setup

- Algorithms learn policy from preference dataset $D = (x^{(i)}, y_w^{(i)}, y_l^{(i)})_{i=1}^N$
- Tasks:
  - Controlled sentiment generation (IMDb movie reviews)
  - Summarization (Reddit TL;DR)
  - Single-turn dialogue (Anthropic Helpful & Harmless)
- Evaluation:

Experimental Setup

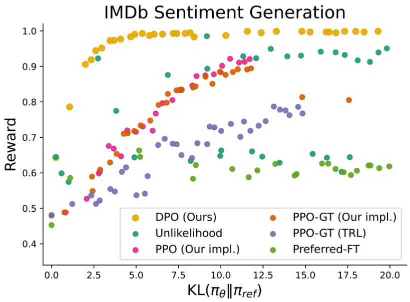
- Algorithms learn policy from preference dataset $D = (x^{(i)}, y_w^{(i)}, y_l^{(i)})_{i=1}^{N}$
- Tasks:
  - Controlled sentiment generation (IMDb movie reviews)
  - Summarization (Reddit TL;DR)
  - Single-turn dialogue (Anthropic Helpful & Harmless)
- Evaluation:
  - Controlled setting: Reward-KL frontier

Experimental Setup

- Algorithms learn policy from preference dataset $D = (x^{(i)}, y_w^{(i)}, y_l^{(i)})_{i=1}^{N}$
- Tasks:
  - Controlled sentiment generation (IMDb movie reviews)
  - Summarization (Reddit TL;DR)
  - Single-turn dialogue (Anthropic Helpful & Harmless)
- Evaluation:
  - Controlled setting: Reward-KL frontier
  - Real world: Win rate vs baseline using GPT-4 proxy

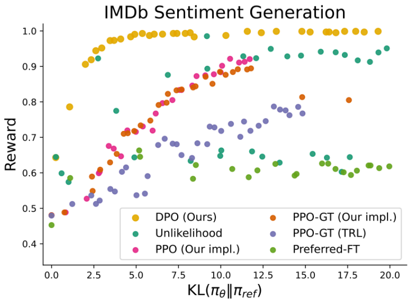
Sentiment Controlled Evaluation

- DPO produces most efficient reward-KL frontier



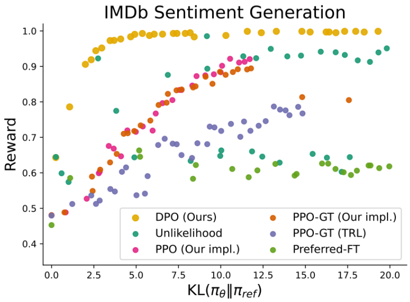
Sentiment Controlled Evaluation

- DPO produces most efficient reward-KL frontier
- Achieves highest reward with low KL divergence



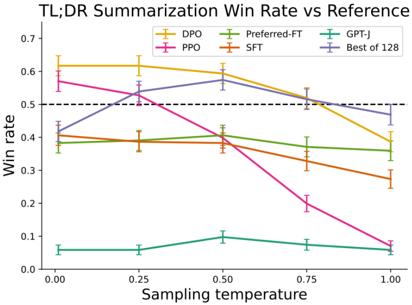
Sentiment Controlled Evaluation

- DPO produces most efficient reward-KL frontier
- Achieves highest reward with low KL divergence
- Strictly dominates PPO frontier, even with PPO accessing ground truth rewards



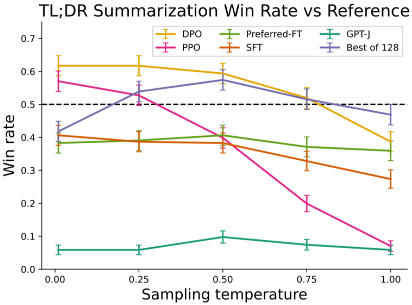
Summarization Results

- DPO exceeds PPO and Best of N baseline performance



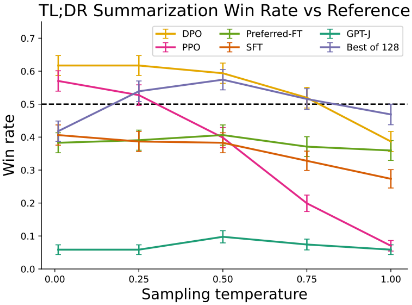
Summarization Results

- DPO exceeds PPO and Best of N baseline performance
- More robust to sampling temperature than PPO



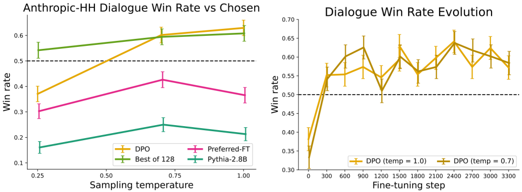
Summarization Results

- DPO exceeds PPO and Best of N baseline performance
- More robust to sampling temperature than PPO
- Preferred-FT does not improve over SFT model



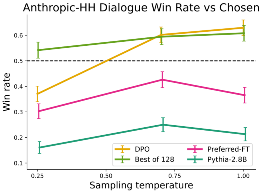
Dialogue Results

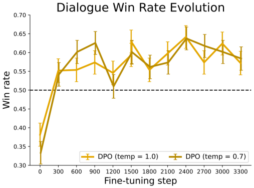
- DPO only method improving over dataset preferences



Dialogue Results

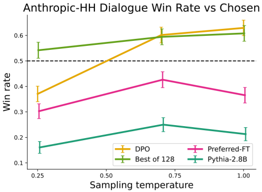
- DPO only method improving over dataset preferences
- Similar or better performance vs Best of 128 baseline

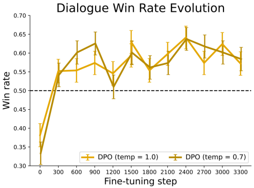




Dialogue Results

- DPO only method improving over dataset preferences
- Similar or better performance vs Best of 128 baseline
- DPO converges to best performance quickly





Qualitative Examples - Summarization

Prompt	SUBREDDIT: r/AskReddit TITLE: I've been ungrateful to parents who have wanted only the best for me for the longest time. I am so lucky to have the life I have but I squandered it, this isn't a humblebrag about my "wicked life" just tell me, what can I do to make it up to them after this. POST: I'm 17, leaving for University, and I've had epiphany after epiphany of how shitty i've beent to them, or how all the nice things they do I ruin with moods etc., I know there isn't a quick fix, but I want to do something nice. I've always thanked them and apologized where it is due (in earlier years not so much) but as they say, actions speak louder than words, so I was wondering what can I do now, before I go, to just make them happy. What can I keep doing, I know all the little things, doing chores, keeping my room clean, cleaning up after myself, helping around the house, but what's a big thing, for say christmas, or their birthdays, that I can do to make them happy?
DPO	TL;DR: I've been a shitty child and I want to make up for it, what can I do for my parents on their birthdays and christmas to show them how much I love them?
Baseline (PPO)	TL;DR: I've been a shitty person to my parents, what can I do to make it up to them, before I go, and what can I do after university to make them happy, and what can I do to make them happy after university (if that is the case)
GPT-4	Summary B [DPO] more effectively captures the main points of the post by focusing on making the parents happy through gestures on special occasions, while Summary A seems repetitive and less clear.

Qualitative Examples - Completion

Prompt	In a shocking turn of events, ...
DPO	In a shocking turn of events, the underdog team staged a stunning comeback in the final minutes of the game, securing a victory that will be remembered for years to come. The crowd erupted in cheers as the final buzzer sounded, and the players celebrated on the court, embracing each other and shedding tears of joy.
Baseline	In a shocking turn of events, the company announced massive lay-offs across all departments. The news came as a surprise to many employees who had been with the company for years. The CEO cited financial difficulties and a need to restructure as the main reasons behind the decision.

- DPO generates a more coherent and engaging continuation of the prompt

Qualitative Examples - Completion

Prompt	In a shocking turn of events, ...
DPO	In a shocking turn of events, the underdog team staged a stunning comeback in the final minutes of the game, securing a victory that will be remembered for years to come. The crowd erupted in cheers as the final buzzer sounded, and the players celebrated on the court, embracing each other and shedding tears of joy.
Baseline	In a shocking turn of events, the company announced massive lay-offs across all departments. The news came as a surprise to many employees who had been with the company for years. The CEO cited financial difficulties and a need to restructure as the main reasons behind the decision.

- DPO generates a more coherent and engaging continuation of the prompt
- The baseline output, while relevant, lacks the same level of coherence and narrative flow

Discussion

Summary of Main Findings

- DPO consistently outperforms baselines across text summarization, dialogue response generation, and text completion tasks

Summary of Main Findings

- DPO consistently outperforms baselines across text summarization, dialogue response generation, and text completion tasks
- The effectiveness of DPO is demonstrated through both automatic metrics and human evaluation

Summary of Main Findings

- DPO consistently outperforms baselines across text summarization, dialogue response generation, and text completion tasks
- The effectiveness of DPO is demonstrated through both automatic metrics and human evaluation
- DPO achieves state-of-the-art performance in aligning language models with human preferences

Advantages of Direct Preference Optimization

- DPO offers a simple and efficient approach to preference learning in language models

Advantages of Direct Preference Optimization

- DPO offers a simple and efficient approach to preference learning in language models
- It captures complex preferences without the need for explicit reward modeling or reinforcement learning

Advantages of Direct Preference Optimization

- DPO offers a simple and efficient approach to preference learning in language models
- It captures complex preferences without the need for explicit reward modeling or reinforcement learning
- DPO scales well to large language models and can be applied to a wide range of tasks

Advantages of Direct Preference Optimization

- DPO offers a simple and efficient approach to preference learning in language models
- It captures complex preferences without the need for explicit reward modeling or reinforcement learning
- DPO scales well to large language models and can be applied to a wide range of tasks
- The direct optimization of preferences leads to more aligned and user-centric language generation

Limitations and Challenges

- The performance of DPO depends on the quality and quantity of preference data

Limitations and Challenges

- The performance of DPO depends on the quality and quantity of preference data
- Biases introduced during the preference collection process can affect the learned preferences

Limitations and Challenges

- The performance of DPO depends on the quality and quantity of preference data
- Biases introduced during the preference collection process can affect the learned preferences
- Extending DPO to more complex and open-ended tasks may require additional techniques and considerations

Limitations and Challenges

- The performance of DPO depends on the quality and quantity of preference data
- Biases introduced during the preference collection process can affect the learned preferences
- Extending DPO to more complex and open-ended tasks may require additional techniques and considerations
- Balancing the trade-off between specificity and generalizability of learned preferences remains a challenge

Implications for Preference Learning in Language Models

- Drives development of more aligned and user-centric language models

Implications for Preference Learning in Language Models

- Drives development of more aligned and user-centric language models
- Enables the incorporation of personalized and context-aware preferences into language generation

Implications for Preference Learning in Language Models

- Drives development of more aligned and user-centric language models
- Enables the incorporation of personalized and context-aware preferences into language generation
- DPO can facilitate the easier integration of ethical and social considerations into language models

Implications for Preference Learning in Language Models

- Drives development of more aligned and user-centric language models
- Enables the incorporation of personalized and context-aware preferences into language generation
- DPO can facilitate the easier integration of ethical and social considerations into language models
- Success of DPO highlights the importance of preference learning in advancing language model capabilities

Future Research

- Exploring alternative preference elicitation methods, such as active learning or interactive feedback

Future Research

- Exploring alternative preference elicitation methods, such as active learning or interactive feedback
- Investigating the integration of DPO with other language model training techniques, such as pre-training or fine-tuning.

Future Research

- Exploring alternative preference elicitation methods, such as active learning or interactive feedback
- Investigating the integration of DPO with other language model training techniques, such as pre-training or fine-tuning.
  - Are there things that DPO does that SFT does not, that PPO does not and vice versa?

Future Research

- Exploring alternative preference elicitation methods, such as active learning or interactive feedback
- Investigating the integration of DPO with other language model training techniques, such as pre-training or fine-tuning.
    - Are there things that DPO does that SFT does not, that PPO does not and vice versa?
    - Can we combine them?

Future Research

- Exploring alternative preference elicitation methods, such as active learning or interactive feedback
- Investigating the integration of DPO with other language model training techniques, such as pre-training or fine-tuning.
  - Are there things that DPO does that SFT does not, that PPO does not and vice versa?
  - Can we combine them?
- Addressing the challenges of preference aggregation and conflicting preferences in real-world applications

Future Research

- Exploring alternative preference elicitation methods, such as active learning or interactive feedback
- Investigating the integration of DPO with other language model training techniques, such as pre-training or fine-tuning.
  - Are there things that DPO does that SFT does not, that PPO does not and vice versa?
  - Can we combine them?
- Addressing the challenges of preference aggregation and conflicting preferences in real-world applications
- Developing techniques to ensure the robustness and fairness of learned preferences

Conclusion

Main Contributions

- Theoretical analysis of DPO, including convergence guarantees and generalization bounds

Main Contributions

- Theoretical analysis of DPO, including convergence guarantees and generalization bounds
- Empirical evaluation demonstrating the effectiveness of DPO compared to existing methods

Main Contributions

- Theoretical analysis of DPO, including convergence guarantees and generalization bounds
- Empirical evaluation demonstrating the effectiveness of DPO compared to existing methods
- Advancements in preference learning for language models, enabling more aligned and user-centric generation

Impact on Language Model Development

- DPO paves the way for developing language models that better align with user preferences and values

Impact on Language Model Development

- DPO paves the way for developing language models that better align with user preferences and values
- It enables the incorporation of personalized and context-aware preferences into language generation

Impact on Language Model Development

- DPO paves the way for developing language models that better align with user preferences and values
- It enables the incorporation of personalized and context-aware preferences into language generation
- DPO has the potential to facilitate the development of language models that are more ethical, unbiased, and socially responsible

Citations and References

References I

[Bai+22] Yuntao Bai et al. *Constitutional AI: Harmlessness from AI Feedback*. 2022. arXiv: 2212.08073 [cs.CL].

[Chr+23] Paul Christiano et al. *Deep reinforcement learning from human preferences*. 2023. arXiv: 1706.03741 [stat.ML].

[Hou+19] Neil Houlsby et al. *Parameter-Efficient Transfer Learning for NLP*. 2019. arXiv: 1902.00751 [cs.LG].

[Hu+21] Edward J. Hu et al. *LoRA: Low-Rank Adaptation of Large Language Models*. 2021. arXiv: 2106.09685 [cs.CL].

References II

[Joa02] Thorsten Joachims. "Optimizing search engines using clickthrough data". In: *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '02. Edmonton, Alberta, Canada: Association for Computing Machinery, 2002, pp. 133–142. ISBN: 158113567X. DOI: 10.1145/775047.775067. URL: https://doi.org/10.1145/775047.775067.

References III

[LL21] Xiang Lisa Li and Percy Liang. "Prefix-Tuning: Optimizing Continuous Prompts for Generation". In: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Ed. by Chengqing Zong et al. Online: Association for Computational Linguistics, Aug. 2021, pp. 4582–4597. DOI: 10.18653/v1/2021.acl-long.353. URL: https://aclanthology.org/2021.acl-long.353.

[Ouy+22] Long Ouyang et al. *Training language models to follow instructions with human feedback*. 2022. arXiv: 2203.02155 [cs.CL].

[Raf+23] Rafael Rafailov et al. *Direct Preference Optimization: Your Language Model is Secretly a Reward Model*. 2023. arXiv: 2305.18290 [cs.LG].

References IV

[Sti+22] Nisan Stiennon et al. *Learning to summarize from human feedback*. 2022. arXiv: 2009.01325 [cs.CL].

Thank you

Sumuk Shashidhar

University of Illinois, Urbana Champaign

June 22, 2024