Learning Dynamics of LLM Finetuning

fzeng 2025-05-13

APRIL
22
2025

Announcing the Outstanding Paper Awards at ICLR 2025

CARL VONDRICK / ICLR 2025

Outstanding Papers

- Safety Alignment Should be Made More Than Just a Few Tokens Deep.
  Xiangyu Qi, Ashwinee Panda, Kaifeng Lyu, Xiao Ma, Subhrajit Roy, Ahmad Beirami, Prateek Mittal, Peter Henderson.

- Learning Dynamics of LLM Finetuning.
  Yi Ren, Danica J. Sutherland.

- AlphaEdit: Null-Space Constrained Model Editing for Language Models.
  Junfeng Fang, Houcheng Jiang, Kun Wang, Yunshan Ma, Jie Shi, Xiang Wang, Xiangnan He, Tat-Seng Chua

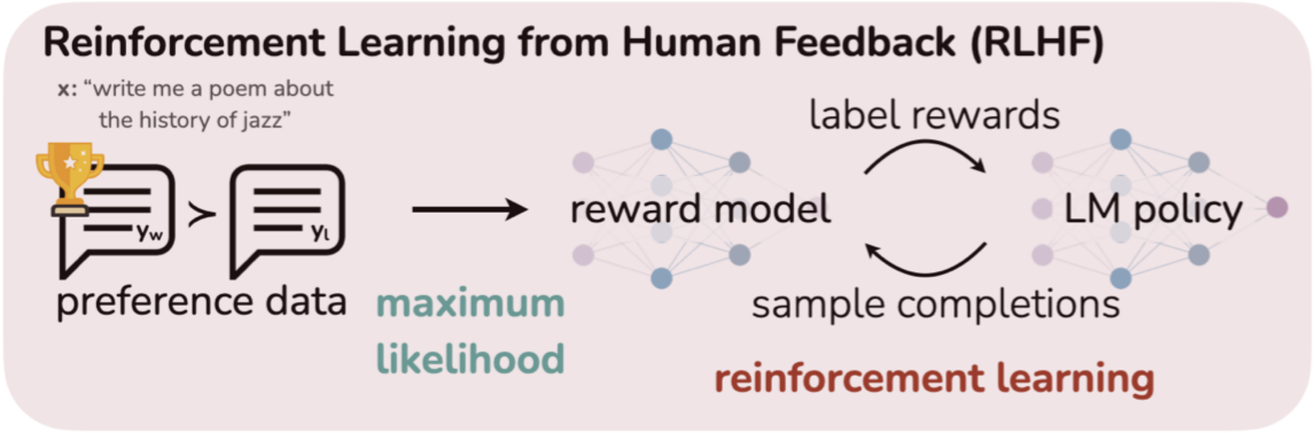
LLM post-training

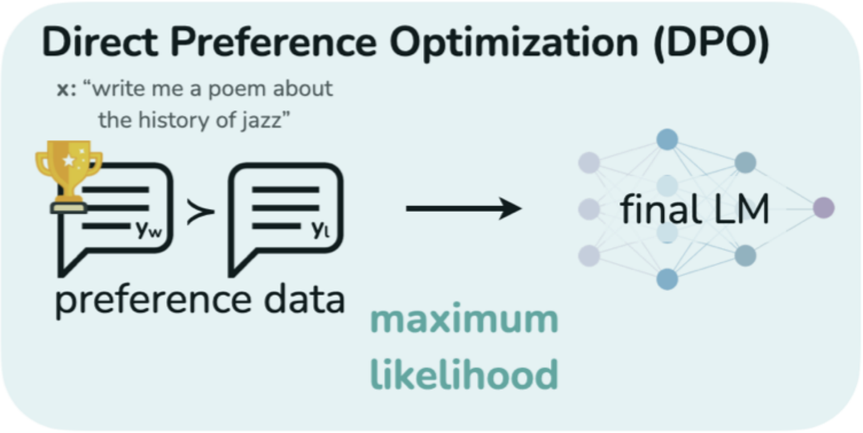
- 2 major stages: SFT and RL
- SFT: teaching format, preference

DPO

Direct preference optimization

- RL-free method for preference tuning
- Widely used by academics due to its simplicity
  - No need to train RM! And RL is hard to get right (training instability, requires expensive on-policy rollouts, reward hacking, sample efficiency, etc...)





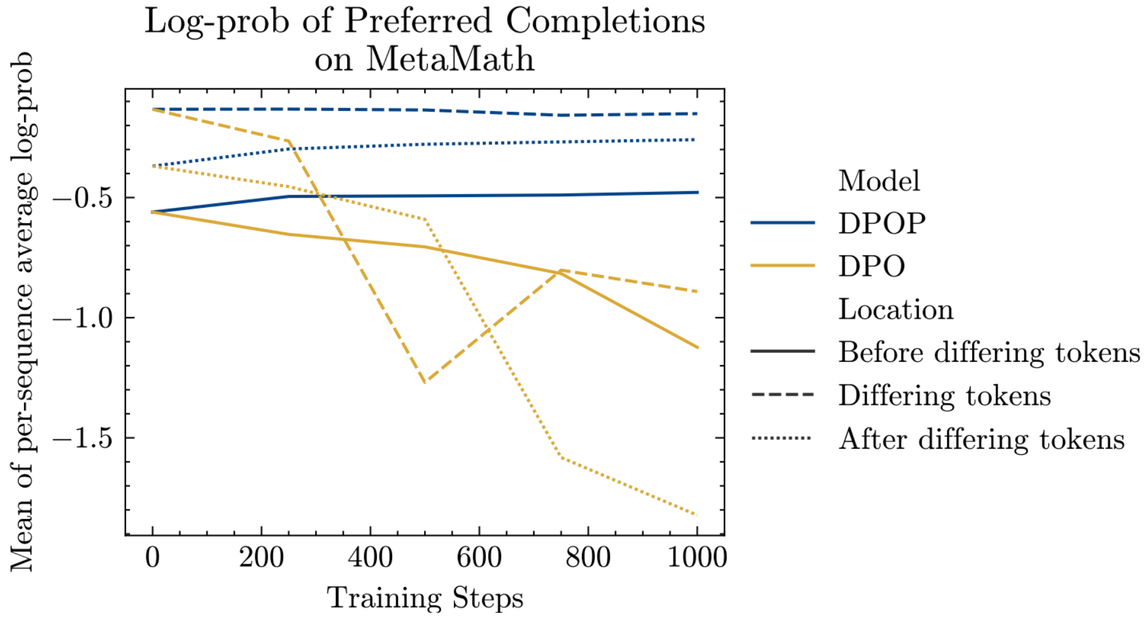
SFT can result in weird unwanted behavior

rithms for both instruction tuning and preference tuning. In particular, we propose a hypothetical explanation of why specific types of hallucination are strengthened after finetuning, e.g., the model might use phrases or facts in the response for question B to answer question A, or the model might keep repeating similar simple phrases when generating responses. We

DPO can also do weird things

5.4 Likelihoods should decrease when using DPO.

A surface level interpretation of DPO would lead one to believe it increases the likelihood of chosen responses, while decreasing the likelihood of rejected responses. This however, does not account for a well observed phenomena in which the likelihood of the chosen responses actually *decrease* over time (Pal et al., 2024). This is illustrated on the left half of fig. 3, which we show that when performing SFT before DPO, the implicit rewards of both the chosen and rejected response decline, though the margin between them increases. However, given a MaxEnt RL framing, this phenomena may be expected.



Learning Dynamics

Learning dynamics is usually an umbrella term describing how the change of a specific factor influences the model's prediction. In this paper, we narrow down it to describe "how the change in model's parameter θ influences the corresponding change in f_θ", i.e., the relationship between $\Delta\theta$ and Δf_θ. When the model updates its parameters using gradient descent (GD), we have

$$\Delta\theta \triangleq \theta^{t+1} - \theta^t = -\eta \cdot \nabla \mathcal{L}\left(f_\theta(\mathbf{x}_u), \mathbf{y}_u\right); \quad \Delta f(\mathbf{x}_o) \triangleq f_{\theta^{t+1}}(\mathbf{x}_o) - f_{\theta^t}(\mathbf{x}_o), \tag{1}$$

where the update of θ during step $t \to t+1$ is given by one gradient update on the sample pair $(\mathbf{x}_u, \mathbf{y}_u)$ with learning rate η. In short, the learning dynamics in this paper address the question:

After an GD update on $\mathbf{x}_u$, how does the model's prediction on $\mathbf{x}_o$ change?

Warmup: MNIST

Dynamics decomposition (1 label case)

Proposition 1. *Let* $\pi = \mathrm{Softmax}(\mathbf{z})$ *and* $\mathbf{z} = h_\theta(\mathbf{x})$. *The one-step learning dynamics decompose as*

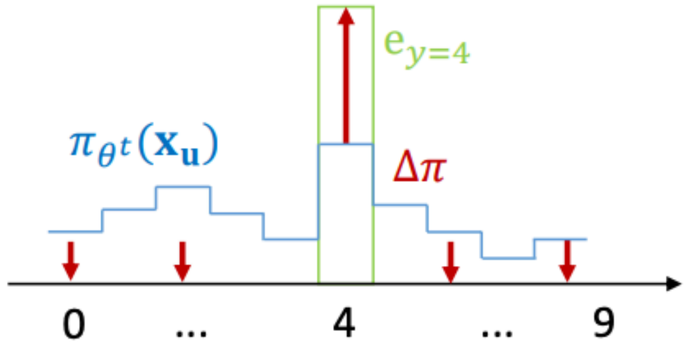
$$\underbrace{\Delta \log \pi^t(\mathbf{y} \mid \mathbf{x}_o)}_{V\times 1} = -\eta \underbrace{\mathcal{A}^t(\mathbf{x}_o)}_{V\times V} \underbrace{\mathcal{K}^t(\mathbf{x}_o, \mathbf{x}_u)}_{V\times V} \underbrace{\mathcal{G}^t(\mathbf{x}_u, \mathbf{y}_u)}_{V\times 1} + \mathcal{O}(\eta^2 \|\nabla_\theta \mathbf{z}(\mathbf{x}_u)\|_{\mathrm{op}}^2), \tag{3}$$

where $\mathcal{A}^t(\mathbf{x}_o) = \nabla_{\mathbf{z}} \log \pi_{\theta^t}(\mathbf{x}_o) = I - \mathbf{1}\pi_{\theta^t}^\top(\mathbf{x}_o)$, $\mathcal{K}^t(\mathbf{x}_o, \mathbf{x}_u) = (\nabla_\theta \mathbf{z}(\mathbf{x}_o)|_{\theta^t})(\nabla_\theta \mathbf{z}(\mathbf{x}_u)|_{\theta^t})^\top$ *is the empirical neural tangent kernel of the logit network* $\mathbf{z}$, *and* $\mathcal{G}^t(\mathbf{x}_u, \mathbf{y}_u) = \nabla_{\mathbf{z}} \mathcal{L}(\mathbf{x}_u, \mathbf{y}_u)|_{\mathbf{z}^t}$.

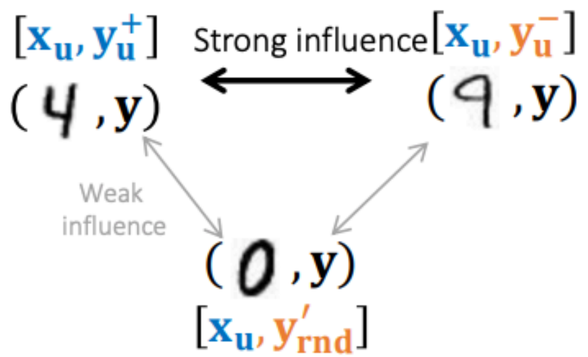
- Saying: after I do an update on (x_u, y_u), how does my output prediction on some other x_o change?
- $A^t(x_o)$: gradient of logprobs on input
- $K^t(x_o, x_u)$: measure of similarity between x_o, x_u
- $G^t(x_u, y_u)$: gradient of loss

We train on 4, get loss gradients G^t

Learn $(\mathbf{x_u} = \text{4}, \mathbf{y_u} = 4)$ using SGD

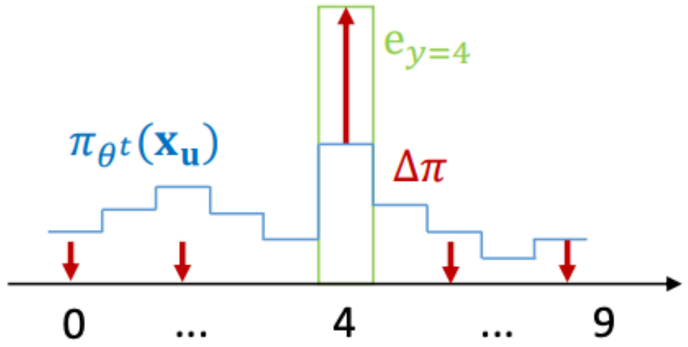


$$\Delta \log \pi^t(\mathbf{x_o}) = -\eta \mathcal{A}^t(\mathbf{x_o})\, \mathcal{K}^t(\mathbf{x_o}, \mathbf{x_u})\, \mathcal{G}^t(\mathbf{x_u}, \mathbf{y_u})$$

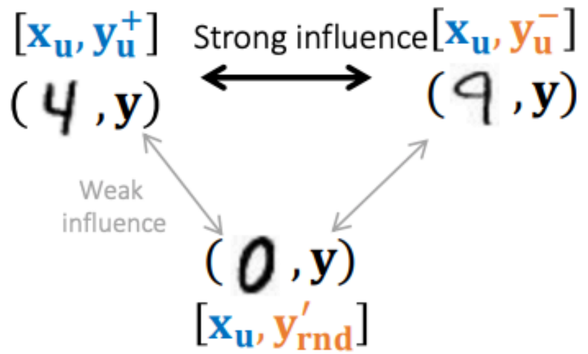


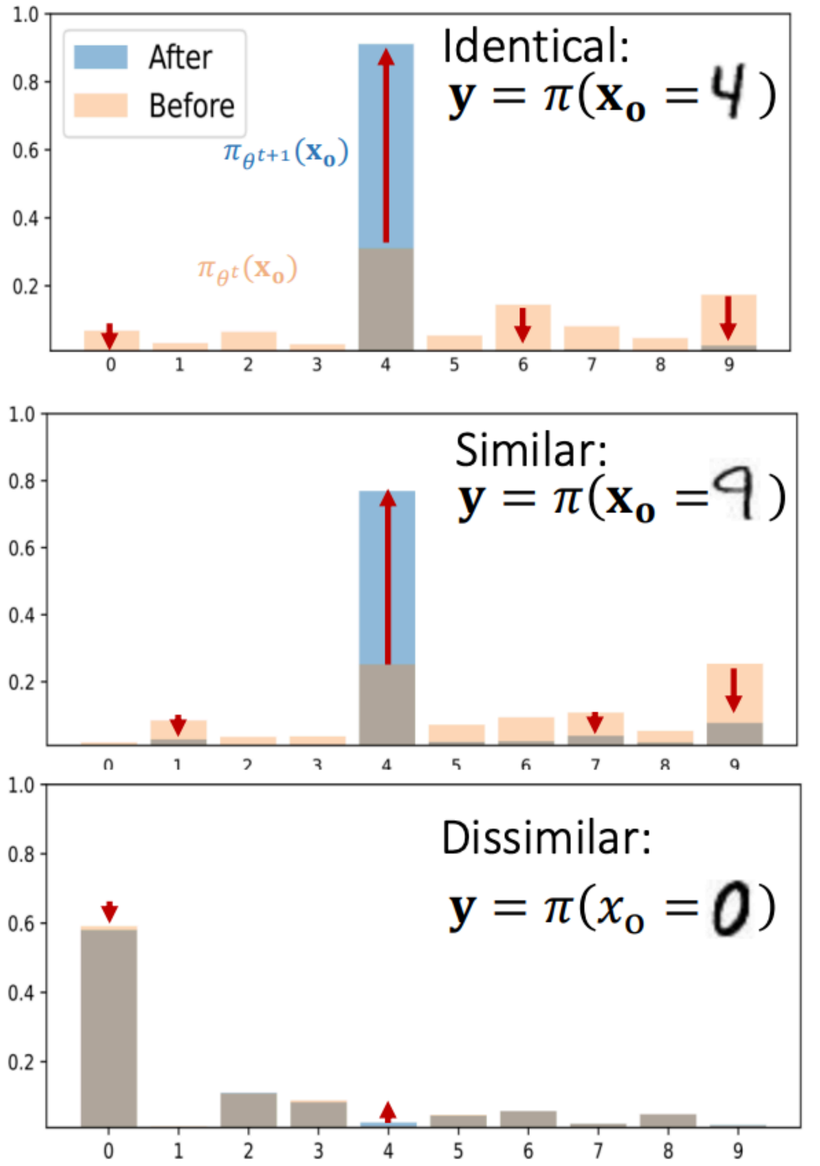
Predictions change!

Learn $(\mathbf{x_u} = 4, \mathbf{y_u} = 4)$ using SGD

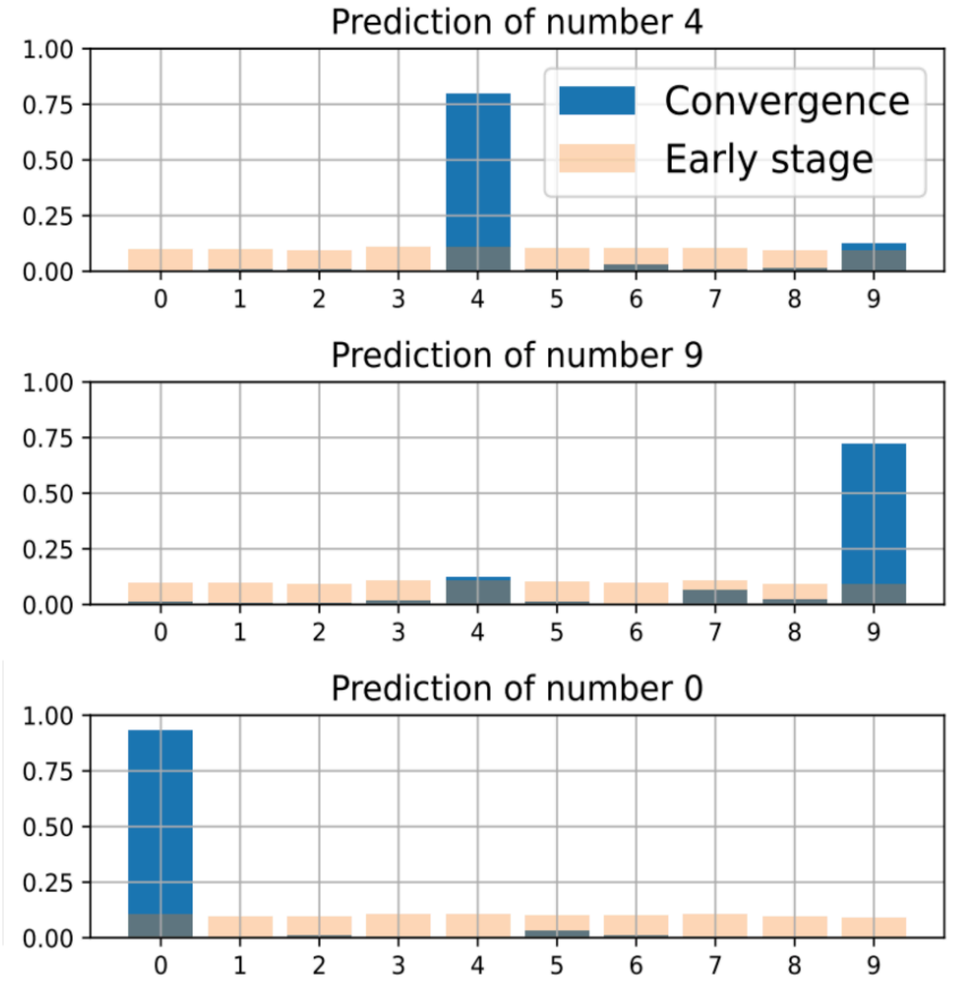


$$\Delta \log \pi^t(\mathbf{x_o}) = -\eta \mathcal{A}^t(\mathbf{x_o})\, \mathcal{K}^t(\mathbf{x_o}, \mathbf{x_u})\, \mathcal{G}^t(\mathbf{x_u}, \mathbf{y_u})$$

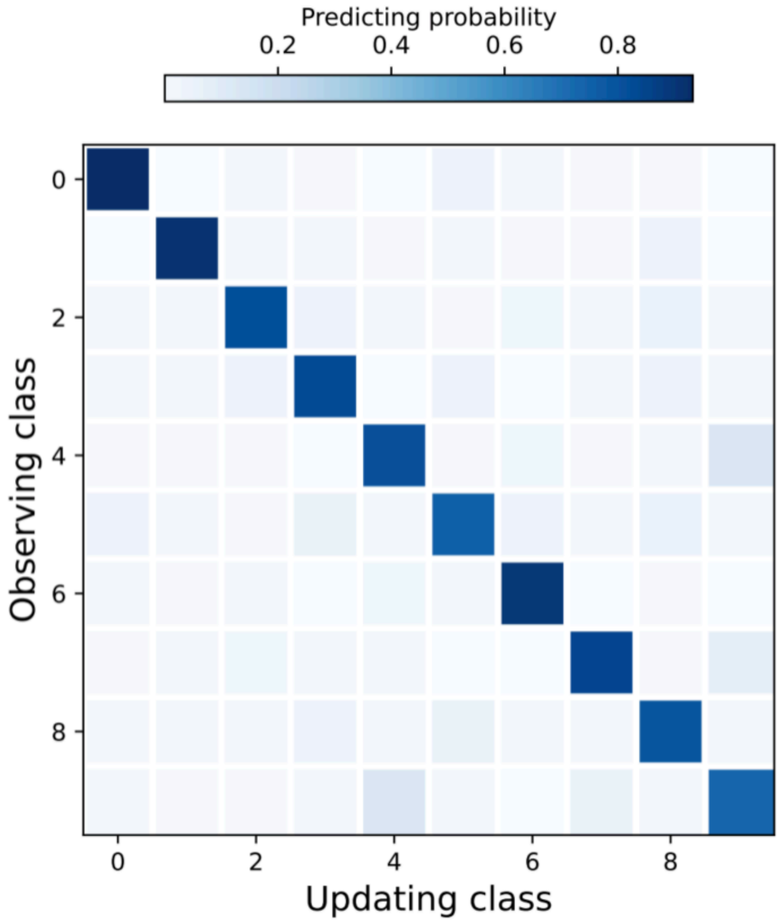




After training for a long time



(c) Accumulated change of epochs



(d) Correlation of the accumulated change

LLM case

- After some painful derivations we can show ~similar structure for decompositions for SFT prediction changes after training on a (prompt, response) pair

$$[\underbrace{\Delta \log \pi^t(\mathbf{y} \mid \boldsymbol{\chi}_o)}_{V\times M}]_m = -\sum_{l=1}^{L} \eta [\underbrace{\mathcal{A}^t(\boldsymbol{\chi}_o)}_{V\times V\times M}]_m [\underbrace{\mathcal{K}^t(\boldsymbol{\chi}_o, \boldsymbol{\chi}_u)}_{V\times V\times L}]_l [\underbrace{\mathcal{G}^t(\boldsymbol{\chi}_u)}_{V\times L}]_l + \mathcal{O}(\eta^2),$$

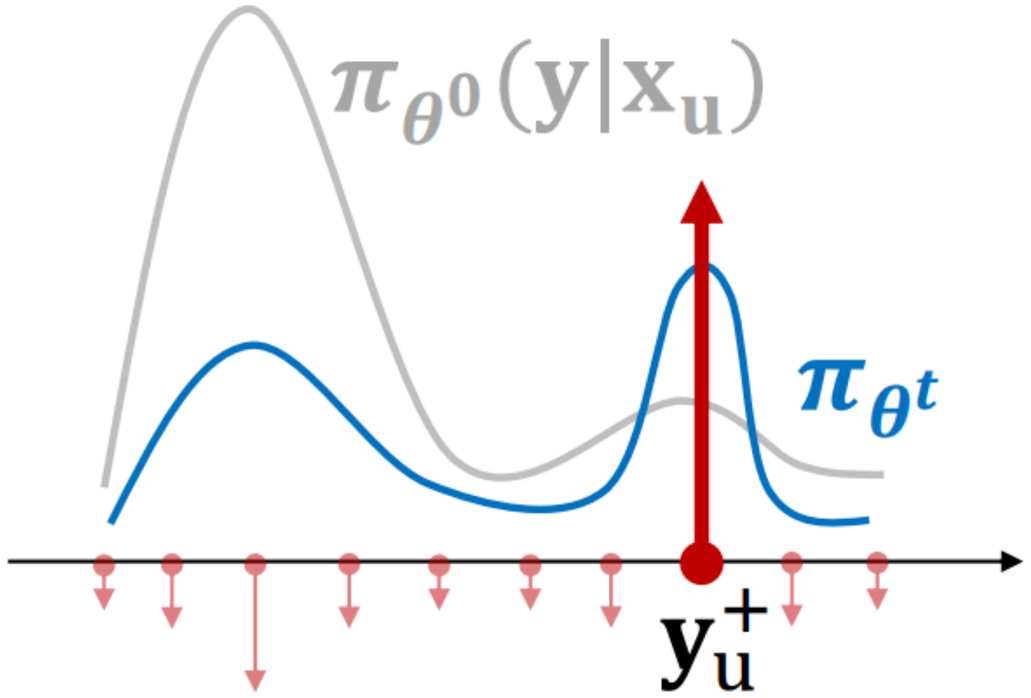
- DPO case slightly more complicated: update influenced by both (preferred, rejected) responses in opposing directions

$$[\Delta \log \pi^t(\mathbf{y} \mid \boldsymbol{\chi}_o)]_m = -\sum_{l-1}^{L} \eta [\mathcal{A}^t(\boldsymbol{\chi}_o)]_m \left([\mathcal{K}^t(\boldsymbol{\chi}_o, \boldsymbol{\chi}_u^+)]_l [\mathcal{G}^t_{\text{DPO+}}]_l - [\mathcal{K}^t(\boldsymbol{\chi}_o, \boldsymbol{\chi}_u^-)]_l [\mathcal{G}^t_{\text{DPO-}}]_l \right) + \mathcal{O}(\eta^2)$$

Training Dynamics of SFT

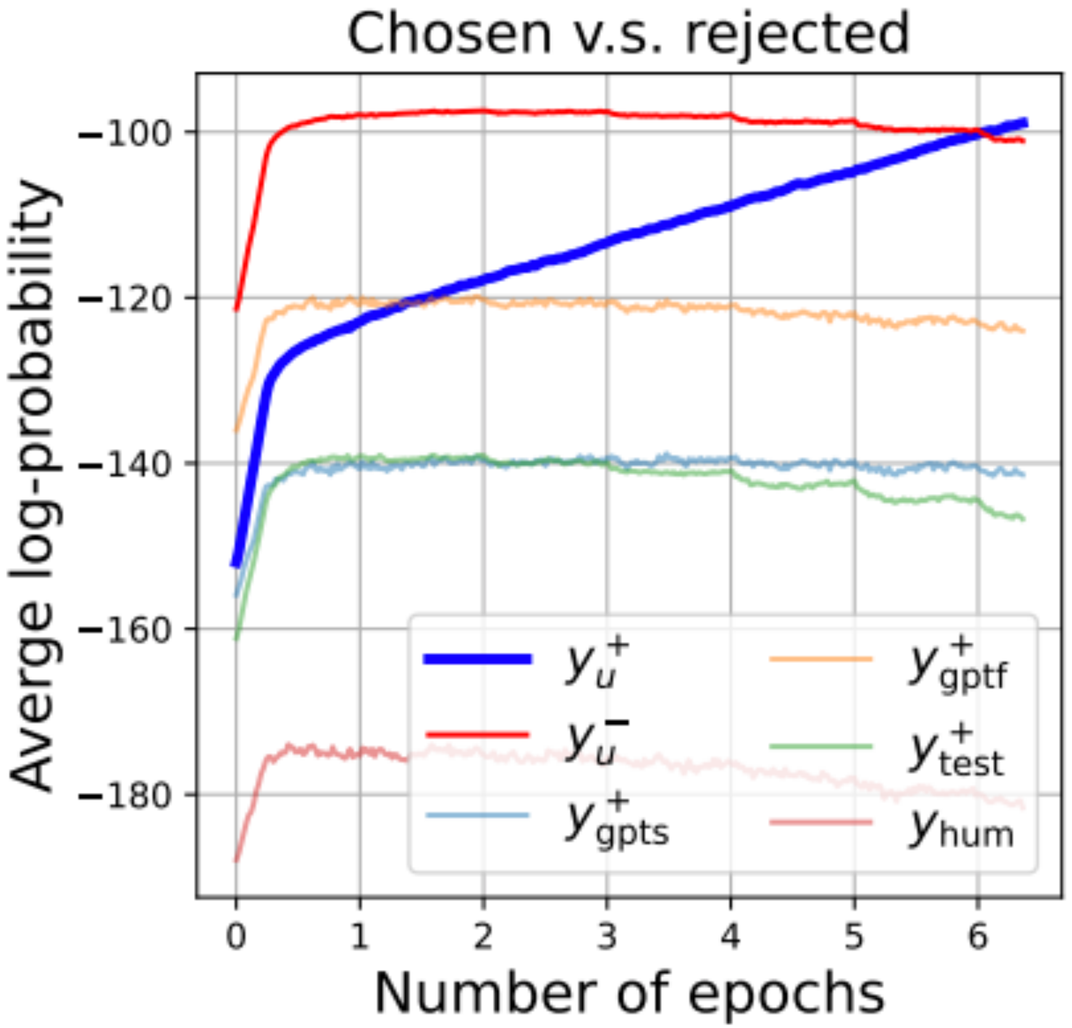
- Not too surprising
- Only 1 upward arrow, confidence of training datapoint goes up!
- Confidence of similar examples also goes up
- Other responses goes down

- SFT



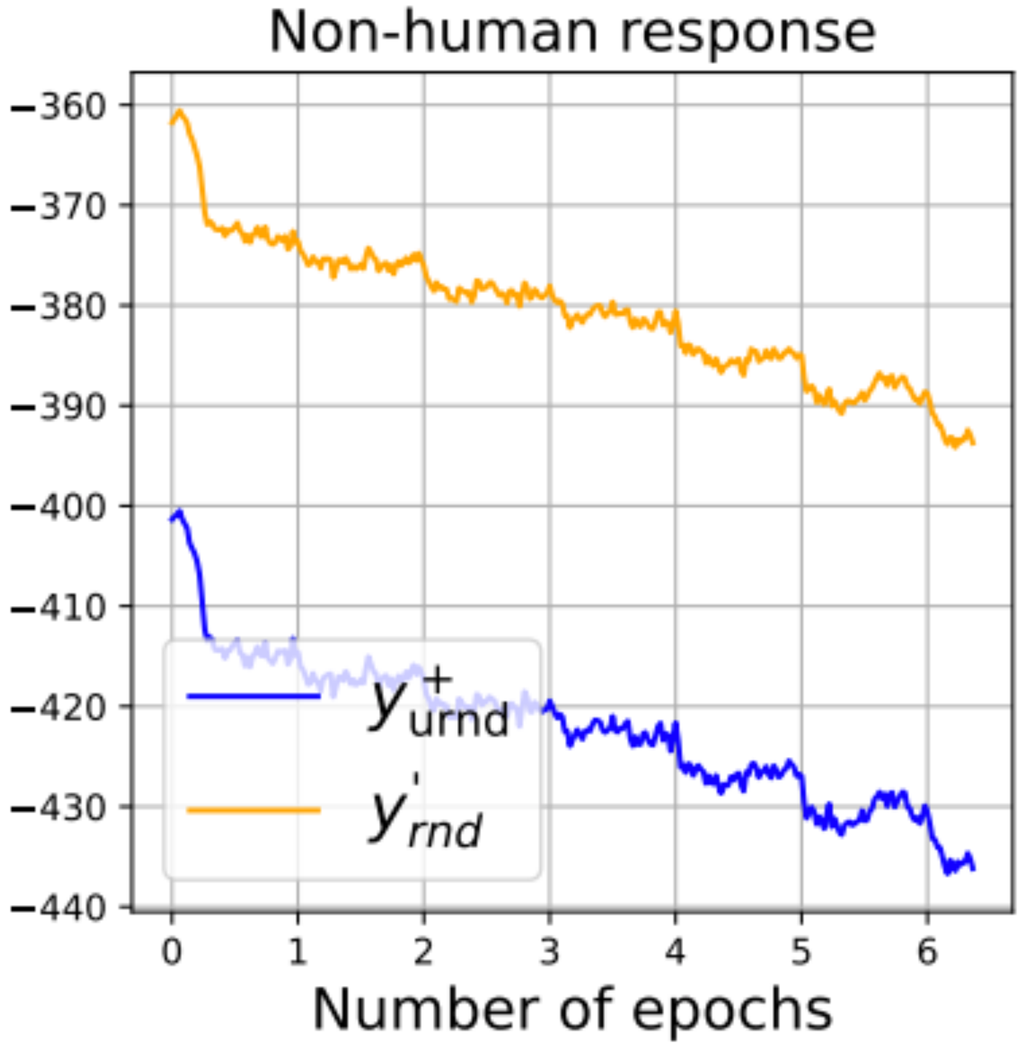
SFT Experiments

- Gpts, gptf: semantic-preserving and format-preserving rephrasing of inputs by GPT



SFT Experiments

- Same length as training inputs
- Random English words, permutations of each other
- No pull-up pressure: logprobs goes down



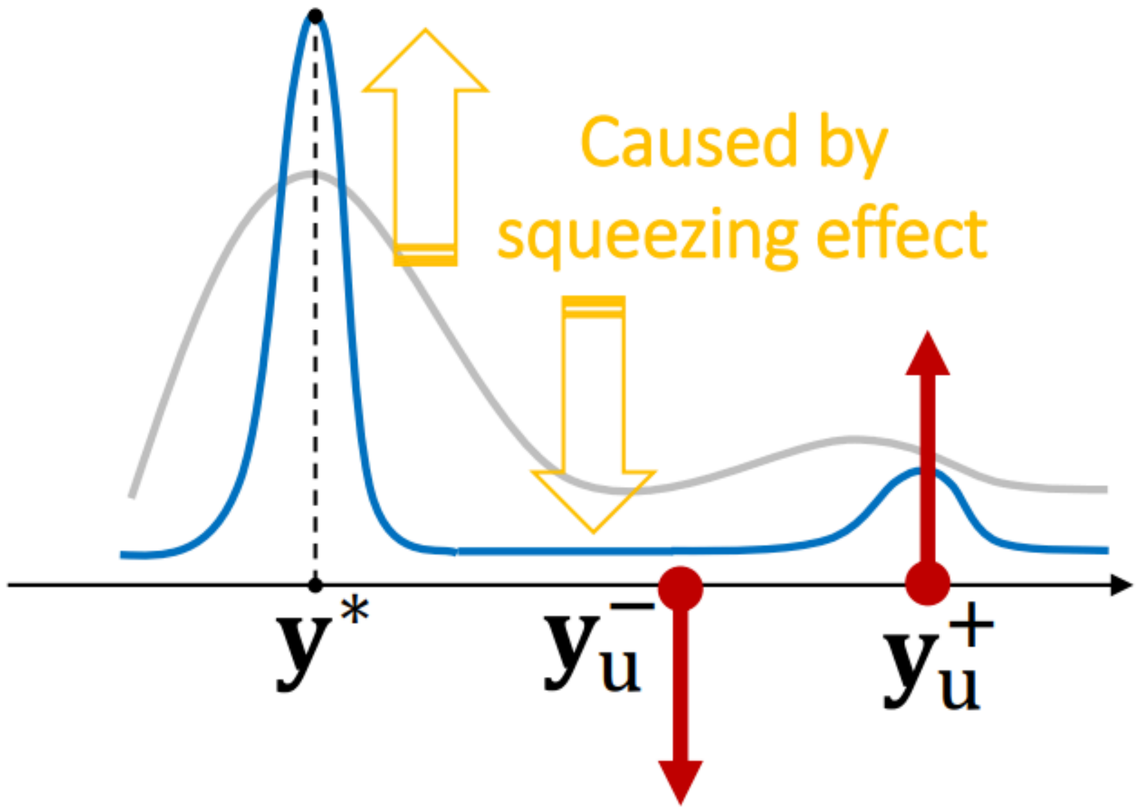
SFT Experiments (hallucination explanation)

- Give it another (unrelated) response that it has seen during training
- Pull-up pressure from two sources: if it's similar to other training examples, and when it's being trained on
- Strong enough to cause it to increase in probability!
- Possible explanation for hallucinations after SFT!!

Training Dynamics of DPO

- Upward for positive pair, downward for negative pair
- But what on earth is going on ???

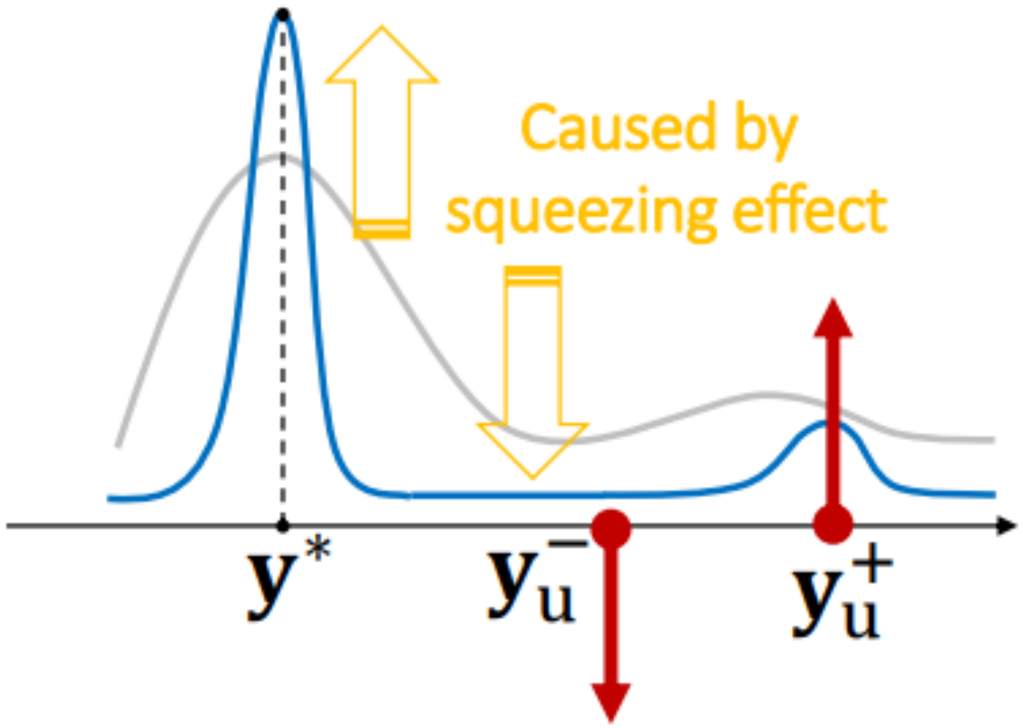
- Off-policy DPO, IPO



The squeezing effect argument

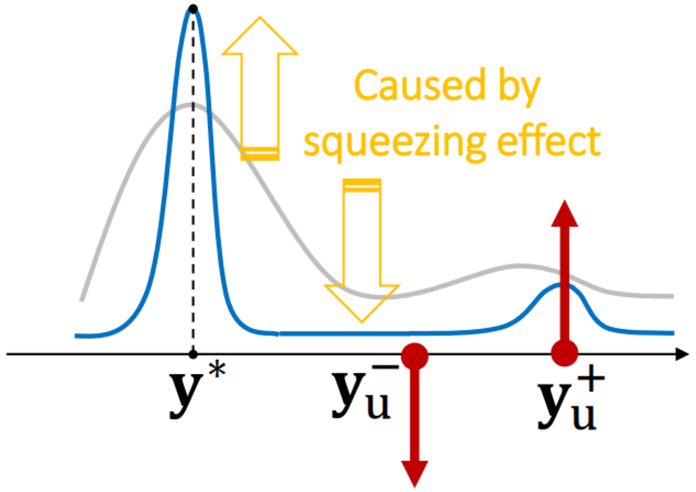
- Pull-up pressure from preferred response y_u^+ not as strong as in SFT
- Pull-down pressure from rejected response y_u^- drags low probability responses even further down
- Proportionately most likely response y^* becomes amplified

- Off-policy DPO, IPO



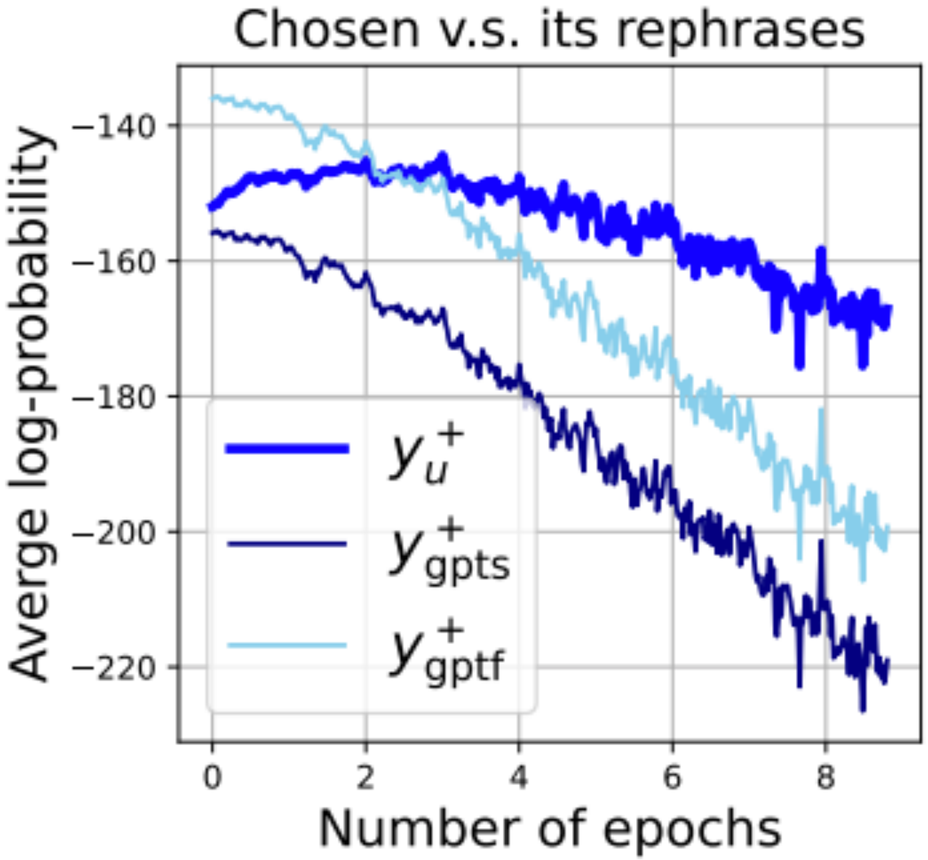
The Squeezing Effect

- *Guarantee:* the confidence of $\mathbf{y}_u^-$, i.e., $\pi_{\theta^{t+1}}(\mathbf{y}_u^-)$ will decrease.
- *Guarantee:* the decreased probability mass is largely "squeezed" into the output which was most confident before the update: if $\mathbf{y}^* = \text{argmax}_{i \in [V] \setminus \{\mathbf{y}_u^-\}} \pi_{\theta^t}(\mathbf{y} = i)$, then $\pi_{\theta^{t+1}}(\mathbf{y} = \mathbf{y}^*)$ will increase.
- *Trend:* the rich get richer and the poor get poorer: generally, dimensions with high π_{θ^t} tend to increase, and those with low π_{θ^t} tend to decrease.
- *Trend:* peakier π_{θ^t} squeezes more. If the probability mass concentrates on few dimensions in π_{θ^t}, which is common for a pretrained model, all $\pi_{\theta^{t+1}}(\mathbf{y} \neq \mathbf{y}^*)$ decrease (only $\mathbf{y}^*$ is "rich").
- *Trend:* smaller $\pi_{\theta^t}(\mathbf{y}_u^-)$ exacerbate the squeezing effect: if $\mathbf{y}_u^-$ is unlikely under π_{θ^t}, the probability mass of all other $\pi_{\theta^{t+1}}(\mathbf{y} \neq \mathbf{y}^*)$ will be more seriously decreased, and the $\pi_{\theta^{t+1}}(\mathbf{y} = \mathbf{y}^*)$ increases more.



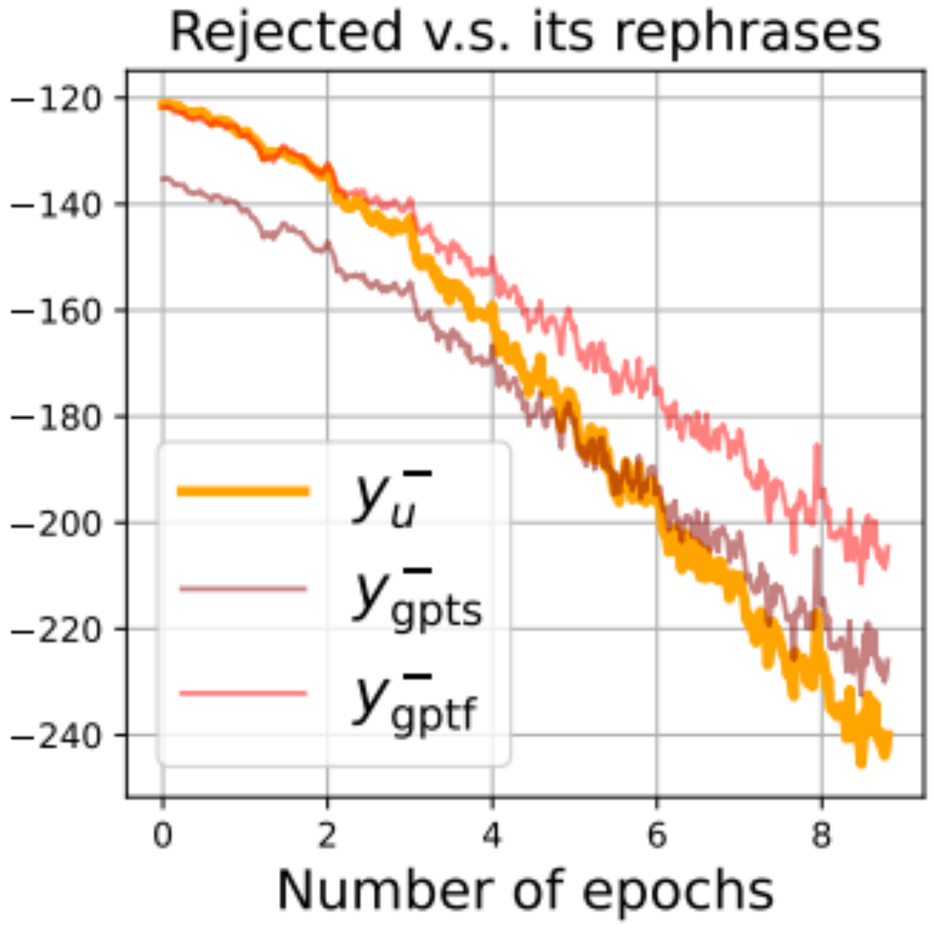
Off-policy DPO Experiments

Even preferred response decays, though slower than rephrases



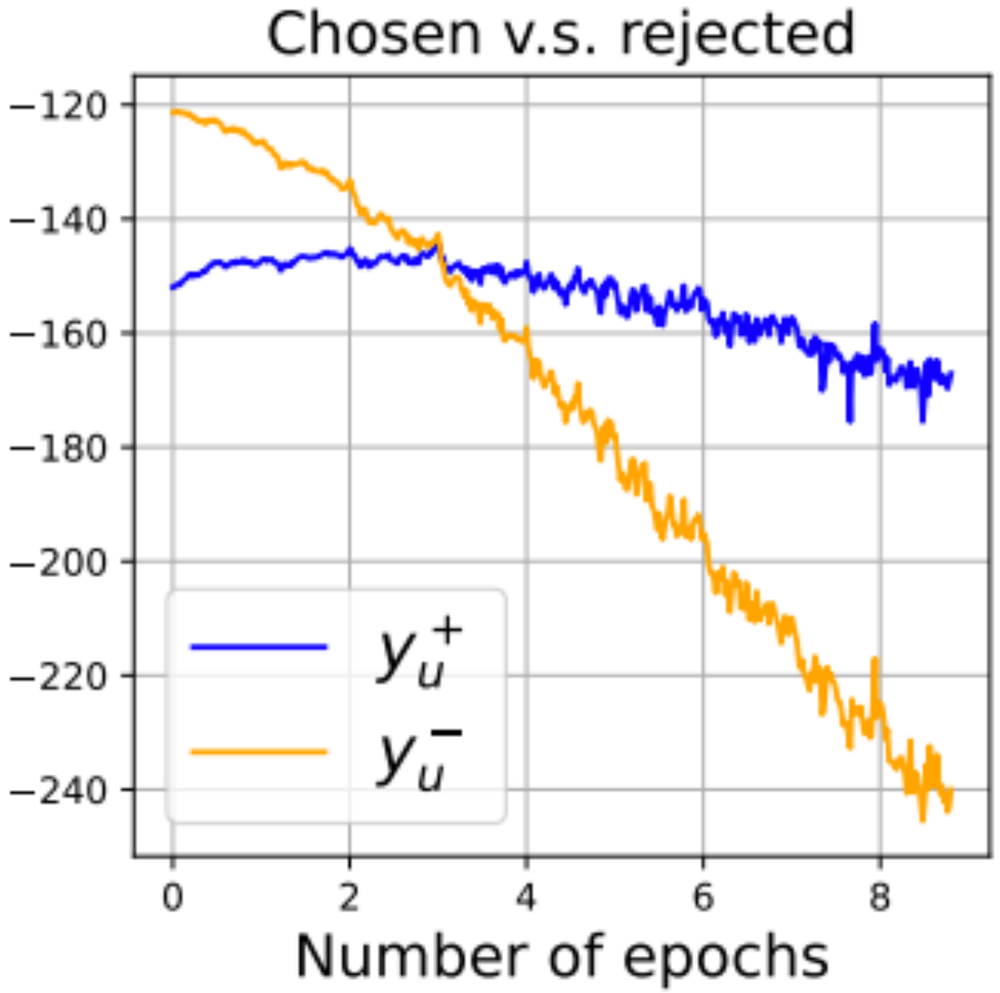
Off-policy DPO Experiments

Rejected response decays faster than similar paraphrases



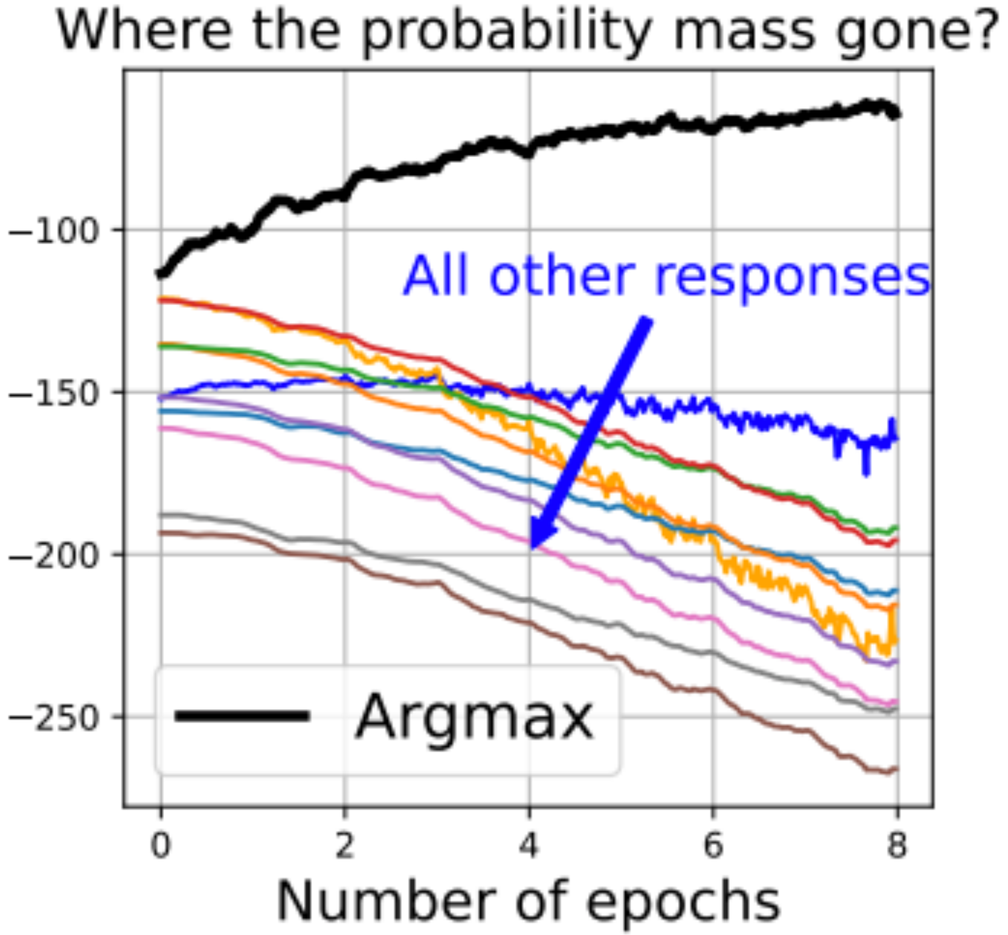
Off-policy DPO Experiments

Rejected response decays faster than preferred response



Off-policy DPO Experiments

Where did probability mass go? All accumulated into most likely response!

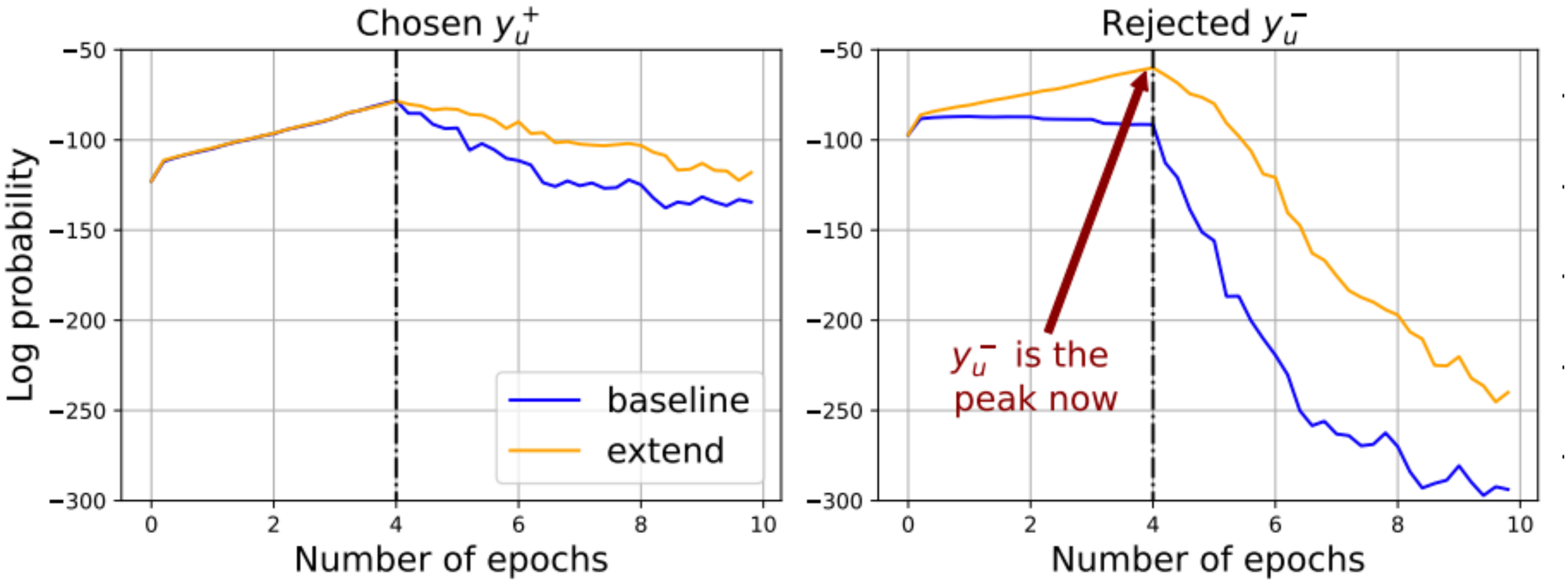


Remedy

- How to fix DPO squeezing effect issues? (Sorry no proposed fixes for SFT)
- Augment training dataset during SFT stage: train on both (prompt, accepted response) and (prompt, rejected response) pairs from DPO dataset during SFT to “pull up” even the rejected response
  - Why? Even rejected response can be reasonable responses to begin with…
- Rejected response will now still be squeezed during DPO, but proportionately less

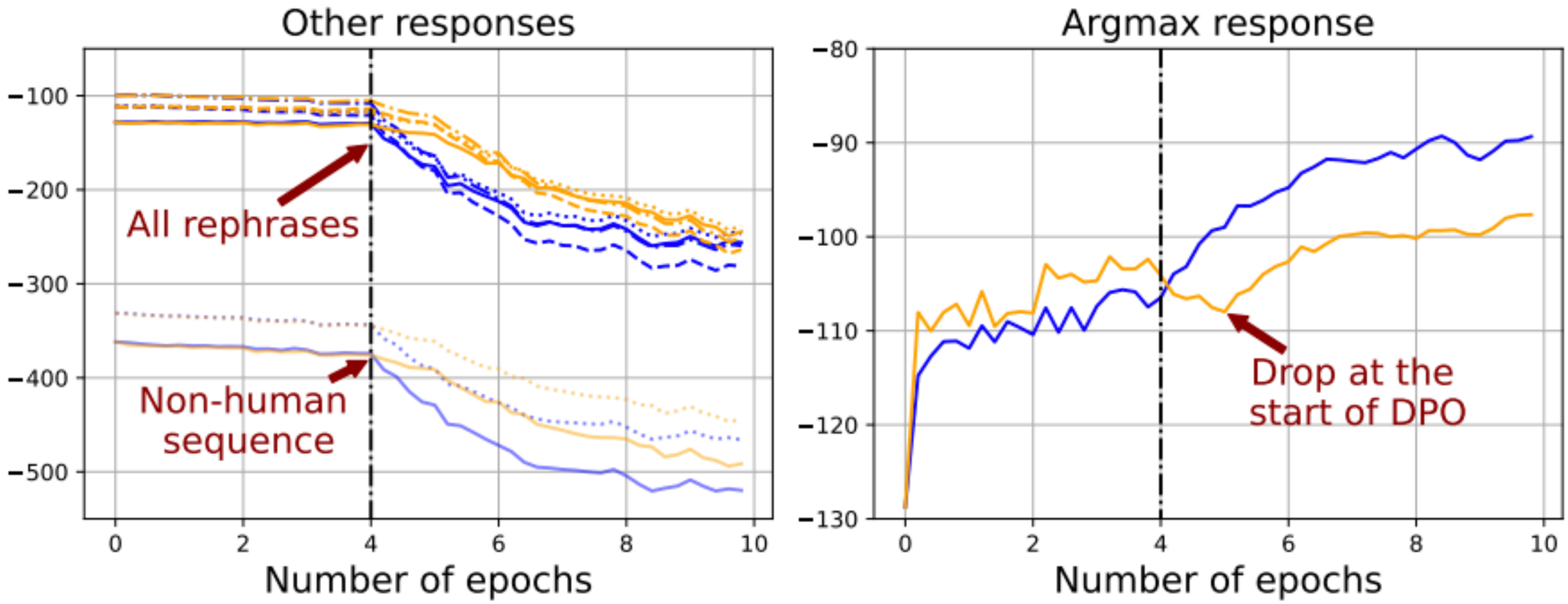
Remedy results

Rejected response not as unlikely after DPO



Remedy results

argmax doesn’t gobble up all the mass



What about PPO?

A.3 BENIGN AND HARMFUL NEGATIVE GRADIENT

The "squeezing effect" can negatively impact our analysis when it is strongly imposed in a valley region of the model. However, a well-regulated negative gradient is both beneficial and commonly observed in many deep-learning systems. For example, it is common in many "machine unlearning" algorithms, e.g., in Ruiqi Zhang et al. (2024). Moreover, even in the field of LLM finetuning, we can find many mechanisms in different popular algorithms that can mitigate this effect. For example, the typical learning rate of DPO is usually smaller than that used in SFT, which unintentionally mitigates the harmful squeezing effect. The on-policy counterpart of the DPO-like algorithms is shown to perform better than their off-policy counterparts, which also supports our claims. Furthermore, we find the PPO loss automatically avoids imposing a big negative gradient (when its $\hat{A}_t$ is negative) on the valley region (when its π_θ is small).

Takeaways

- Negative gradients aka discouraging behaviors can be dangerous
  - Especially if these behaviors are not that unreasonable to begin with
- Maybe just stick to on-policy methods?