Learning Dynamics of LLM Finetuning

Mar. 9 2026

Speakers: Yijing Zhou, Michael Frew

Introduction

Purpose:

The authors provide **interpretations** of the LLM fine-tuning process, which was originally considered a "black box."

Definition:

Learning dynamics, which describes how the learning of specific training examples influences the model's predictions on other examples (i.e., the relationship between $\Delta\theta$ and Δf_θ), gives us a powerful tool for understanding the behavior of deep learning systems.

Motivations:

- Providing a unified explanation for various **abnormal phenomena** (such as Hallucination and Repeater Phenomenon) during fine-tuning
- Exploring the **essential differences** between different fine-tuning algorithms (such as SFT and DPO)
- Demonstrating the **Squeezing Effect** and its Harms

Derivation

When the model updates its parameters using gradient descent (GD), we have:

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

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?

We first consider a standard supervised learning problem, where the model learns to map **x** to predictions **y** = {y_1, . . . , y_L} ∈ V^L, where V is the vocabulary of size V. The model usually outputs a probability distribution by first generating a matrix of logits **z** = h_θ(**x**) ∈ $\mathbb{R}^{(V \times L)}$ and then takes the Softmax. We can track the change in the model's confidence by observing log π_θ(**y** | **x**).

Per-step influence decomposition:

$$\Delta \log \pi^t(\mathbf{y} \mid \mathbf{x}_o) \triangleq \log \pi_{\theta^{t+1}}(\mathbf{y} \mid \mathbf{x}_o) - \log \pi_{\theta^t}(\mathbf{y} \mid \mathbf{x}_o), . \tag{2}$$

Derivation of Decomposition

Proof. [1] Suppose we want to observe the model's prediction on an "observing example" $\mathbf{x}_o$. Starting from Equation (2), we first approximate $\log \pi^{t+1}(\mathbf{y} \mid \mathbf{x}_o)$ using first-order Taylor expansion (we use π^t to represent π_{θ^t} interchangeably for notation conciseness):

$$\log \pi^{t+1}(\mathbf{y} \mid \mathbf{x}_o) = \log \pi^{t}(\mathbf{y} \mid \mathbf{x}_o) + \langle \nabla \log \pi^{t}(\mathbf{y} \mid \mathbf{x}_o),\ \theta^{t+1} - \theta^t \rangle + O(\|\theta^{t+1} - \theta^t\|^2).$$

Then, assuming the model updates its parameters using SGD calculated by an "updating example" $(\mathbf{x}_u, \mathbf{y}_u)$, we can rearrange the terms in the above equation to get the following expression:

$$\Delta \log \pi^t(\mathbf{y} \mid \mathbf{x}_o) = \underbrace{\log \pi^{t+1}(\mathbf{y} \mid \mathbf{x}_o)}_{V\times 1} - \underbrace{\log \pi^{t}(\mathbf{y} \mid \mathbf{x}_o)}_{V\times 1} = \underbrace{\nabla_\theta \log \pi^t(\mathbf{y} \mid \mathbf{x}_o)|_{\theta^t}}_{V\times d} \underbrace{(\theta^{t+1} - \theta^t)}_{d\times 1} + O(\|\theta^{t+1} - \theta^t\|^2),$$

where d is the number of parameters of the model. To evaluate the leading term, we plug in the definition of SGD and repeatedly use the chain rule:

$$\begin{aligned}
\underbrace{\nabla_\theta \log \pi^t(\mathbf{y} \mid \mathbf{x}_o)|_{\theta^t}}_{V\times d} \underbrace{(\theta^{t+1} - \theta^t)}_{d\times 1} &= \big(\underbrace{\nabla_{\mathbf{z}} \log \pi^t(\mathbf{x}_o)|_{\mathbf{z}^t}}_{V\times V} \underbrace{\nabla_\theta \mathbf{z}^t(\mathbf{x}_o)|_{\theta^t}}_{V\times d}\big)\big(-\eta \underbrace{\nabla_\theta \mathcal{L}(\mathbf{x}_u)|_{\theta^t}}_{1\times d}\big)^\top \\
&= \underbrace{\nabla_{\mathbf{z}} \log \pi^t(\mathbf{x}_o)|_{\mathbf{z}^t}}_{V\times V} \underbrace{\nabla_\theta \mathbf{z}^t(\mathbf{x}_o)|_{\theta^t}}_{V\times d} \big(\underbrace{-\eta \nabla_{\mathbf{z}} \mathcal{L}(\mathbf{x}_u)|_{\mathbf{z}^t}}_{1\times V} \underbrace{\nabla_\theta \mathbf{z}^t(\mathbf{x}_u)|_{\theta^t}}_{V\times d}\big)^\top \\
&= -\eta \underbrace{\nabla_{\mathbf{z}} \log \pi^t(\mathbf{x}_o)|_{\mathbf{z}^t}}_{V\times V} \big[\underbrace{\nabla_\theta \mathbf{z}^t(\mathbf{x}_o)|_{\theta^t}}_{V\times d} \underbrace{\big(\nabla_\theta \mathbf{z}^t(\mathbf{x}_u)|_{\theta^t}\big)^\top}_{d\times V}\big] \underbrace{\big(\nabla_{\mathbf{z}} \mathcal{L}(\mathbf{x}_u)|_{\mathbf{z}^t}\big)^\top}_{V\times 1} \\
&= -\eta \mathcal{A}^t(\mathbf{x}_o) \mathcal{K}^t(\mathbf{x}_o, \mathbf{x}_u) \mathcal{G}^t(\mathbf{x}_u, \mathbf{y}_u) \qquad (8)
\end{aligned}$$

For the higher-order term, using as above that

$$\theta^{t+1} - \theta^t = -\eta \nabla_\theta \mathbf{z}^t(\mathbf{x}_u)|_{\theta^t}^\top \mathcal{G}^t(\mathbf{x}_u, \hat{\mathbf{y}})$$

and noting that, since the residual term $\mathcal{G}^t$ is usually bounded (and the practical algorithms will also use gradient clip to avoid too large gradient), we have that

$$O(\|\theta^{t+1} - \theta^t\|^2) = O\big(\eta^2 \|\big(\nabla_\theta \mathbf{z}^t(\mathbf{x}_u)|_{\theta^t}\big)^\top\|_{\text{op}}^2 \|\mathcal{G}^t(\mathbf{x}_u, \hat{\mathbf{y}})\|_{\text{op}}^2\big) = O\big(\eta^2 \|\nabla_\theta \mathbf{z}(\mathbf{x}_u)\|_{\text{op}}^2\big). \qquad \square$$

Learning Dynamics Decompose

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:

Adaptation matrix: only depends on the model's current predicted probability

$$\mathcal{A}^t(\mathbf{x}_o) = \nabla_{\mathbf{z}} \log \pi_{\theta^t}(\mathbf{x}_o) = I - \mathbf{1}\pi_{\theta^t}^\top(\mathbf{x}_o)$$

Empirical neural tangent kernel: the product of the model's gradients with respect to $\mathbf{x}_o$ and $\mathbf{x}_u$

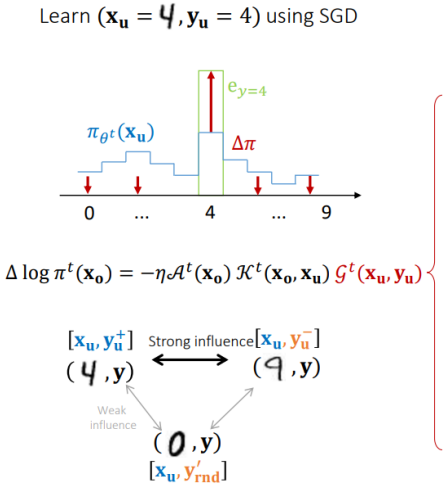
$$\mathcal{K}^t(\mathbf{x}_o, \mathbf{x}_u) = (\nabla_\theta \mathbf{z}(\mathbf{x}_o)|_{\theta^t})(\nabla_\theta \mathbf{z}(\mathbf{x}_u)|_{\theta^t})^\top$$

Residual term: provides the energy and direction for the model's adaptation

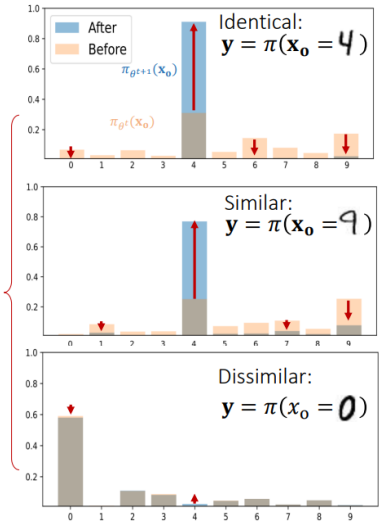
$$\mathcal{G}^t(\mathbf{x}_u, \mathbf{y}_u) = \nabla_{\mathbf{z}} \mathcal{L}(\mathbf{x}_u, \mathbf{y}_u)|_{\mathbf{z}^t}$$

MNIST Example

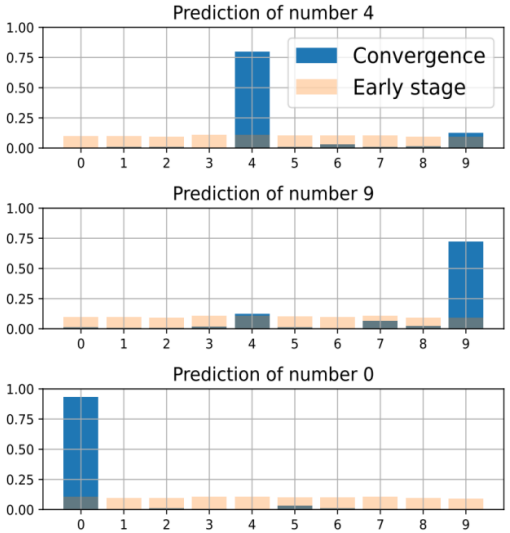
The per-step learning dynamics and the accumulated influence in an MNIST experiment



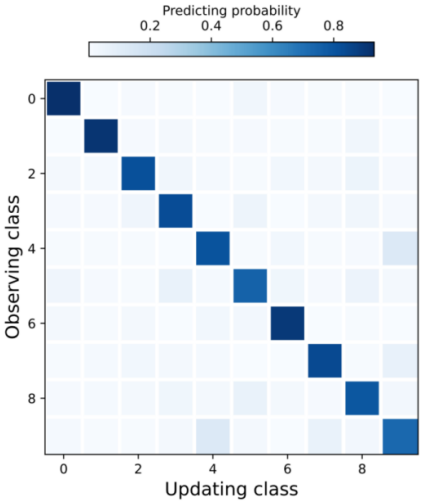
(a) Adaptation vector created by $(\mathbf{x_u}, \mathbf{y_u})$



(b) One-step change with the same $\mathcal{G}^t(\mathbf{x_u}, \mathbf{y_u})$ (large η)



(c) Accumulated change of epochs



(d) Correlation of the accumulated change

Extension to LLM Finetuning

What changes?

- Inputs/Outputs are sequences $(\mathrm{V} \times \mathrm{L})$
- Autoregressive conditioning $\pi(y|x) = \prod_{\mathrm{l}=1}^{\mathrm{L}} \pi(y_l|x, y_{<l})$

Notation:

The full sequence is represented by: $\mathcal{X} = (x, y)$

Definition for logits: $z = h_\theta(\mathcal{X}) \in \mathbb{R}^{V \times L}$

Definition for probs: $\pi(y|\mathcal{X}) = \mathrm{Softmax\_column}(z) \in \mathbb{R}^{V \times L}$

V = vocabulary

M = length of observed/output response

L = length of training response

Decomposition of SFT Loss

The computation of these terms must be completed for each element in both sequences and then stacked:

For example, the **Empirical neural tangent kernel:**

$$\mathcal{K}^t_{m,l}(\mathcal{X}_o, \mathcal{X}_u) = \left(\nabla_\theta z_{\mathrm{m}}(\mathcal{X}_o)\Big|_{\theta^t}\right)\left(\nabla_\theta z_{\mathrm{l}}(\mathcal{X}_u)\Big|_{\theta^t}\right)^{\mathsf{T}}$$

$$\mathcal{K}^t(\mathcal{X}_o, \mathcal{X}_u) \in \mathbb{R}^{(V\times V\times M\times L)}$$

The **adaptation vector** is computed for every output token:

$$\mathcal{A}^t(\mathcal{X}_o) \in \mathbb{R}^{(V\times V\times M)}$$

The **residual term** is computed for every token in the training sample:

$$G^t_{SFT}(\mathcal{X}_u) \in \mathbb{R}^{(V\times L)}$$

The change in model's prediction on the *m*-th token of y_o can be represented as:

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

Decomposition of DPO Loss

$$\mathcal{L}_{\text{DPO}}(\theta) = -\mathbb{E}_{(\mathbf{x}_u, \mathbf{y}_u^+, \mathbf{y}_u^-)\sim\mathcal{D}} \left[\log \sigma \left(\beta \log \frac{\pi_{\theta^t}(\mathbf{y}_u^+ \mid \boldsymbol{\chi}_u^+)}{\pi_{\text{ref}}(\mathbf{y}_u^+ \mid \boldsymbol{\chi}_u^+)} - \beta \log \frac{\pi_{\theta^t}(\mathbf{y}_u^- \mid \boldsymbol{\chi}_u^-)}{\pi_{\text{ref}}(\mathbf{y}_u^- \mid \boldsymbol{\chi}_u^-)} \right) \right]$$

Decomposition:

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

$$\mathcal{G}^t_{\text{DPO+}} = \beta(1-a) \left(\pi_{\theta^t}(\mathbf{y} \mid \boldsymbol{\chi}_u^+) - \mathbf{y}_u^+ \right); \qquad \mathcal{G}^t_{\text{DPO-}} = \beta(1-a) \left(\pi_{\theta^t}(\mathbf{y} \mid \boldsymbol{\chi}_u^-) - \mathbf{y}_u^- \right), \tag{7}$$

Where a is the margin for the l-th token, which represents how well the current policy separates y_u^+ and y_u^-.

$$a = \sigma \left(\beta \log \frac{\pi_{\theta^t}(\mathbf{y}_u^+ \mid \boldsymbol{\chi}_u^+)}{\pi_{\theta^t}(\mathbf{y}_u^- \mid \boldsymbol{\chi}_u^-)} - \beta \log \frac{\pi_{\text{ref}}(\mathbf{y}_u^+ \mid \boldsymbol{\chi}_u^+)}{\pi_{\text{ref}}(\mathbf{y}_u^- \mid \boldsymbol{\chi}_u^-)} \right)$$

Squeeze Effect from Negative Gradients

Guarantees:

- The confidence of y_u^- i.e. $\pi_{\theta^{t+1}}(y_u^-)$ will decrease
- The decreased probability mass is largely 'squeezed' into the output which was most confident before the update

Trends:

- Generally, dimensions with high π_{θ^t} tend to increase, and low π_{θ^t} values decrease
- 'Peakier' π_{θ^t} squeezes more. If the probability mass concentrates on few dimensions in π_{θ^t} (common for pretrained models), all $\pi_{\theta^{t+1}}(\mathrm{y} \neq \mathrm{y}*)$ decrease
- Smaller $\pi_{\theta^t}(y_u^-)$ exacerbate the squeezing effect: if y_u^- is unlikely under π_{θ^t}, the probability mass of all other $\pi_{\theta^{t+1}}(y \neq y^*)$ will be more seriously decreased, and $\pi_{\theta^{t+1}}(y = y^*)$ will increase more

Squeeze Effect from Negative Gradients

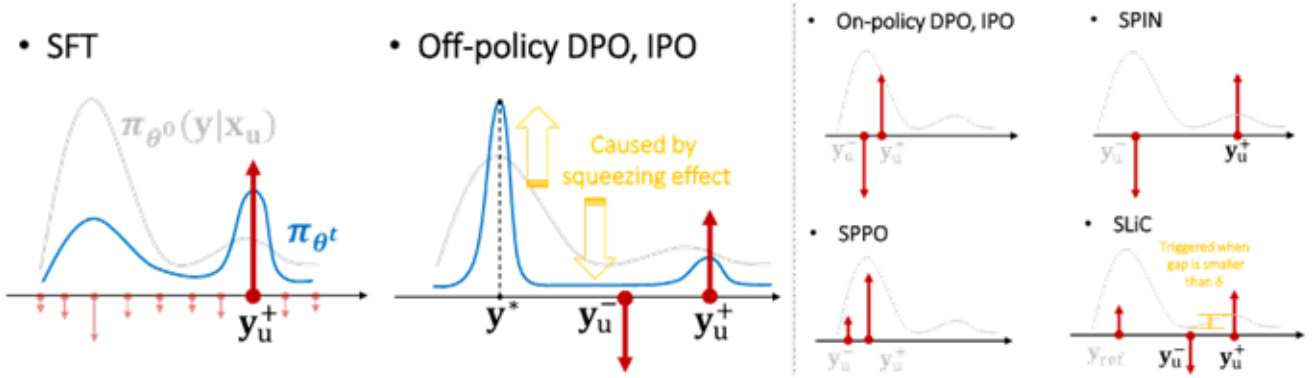


Figure 2: The updating vector provided by the residual term $\mathcal{G}^t$ of different algorithms. The gray y are responses *sampled* from π in an on-policy way. In the second panel, we demonstrate the "squeezing effect" caused by imposing a big negative gradient on a "valley" region of a distribution. For more details about this counter-intuitive effect, please refer to Section 3.3 and Appendix E. Other panels demonstrate on-policy DPO (and IPO), SPIN (Z. Chen et al. 2024), SPPO (Y. Wu et al. 2024), and SLiC (Y. Zhao et al. 2023).

Experiments

Experimental setup:

Dataset: The authors randomly sampled **5000 samples** from the **Anthropic-HH** (human preference dataset) and **Ultra Feedback** training sets to construct the training dataset.

Models: Experiments were repeated on multiple models of different scales, mainly including the **pythia-410M/1B/1.4B/2.8B** series and the **Qwen1.5-0.5B/1.8B** model.

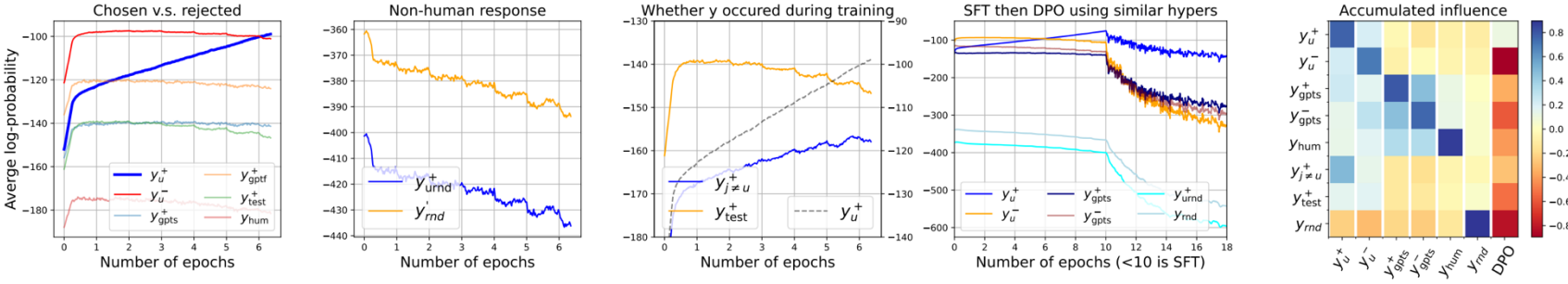
Probing dataset: The authors extracted **500 samples** from the training set. During training, the model pauses and calculates the log probabilities for different types of responses on this probe set.

Experiments

Notations:

$\mathbf{y}_u^+$: Chosen response
$\mathbf{y}_u^-$: Rejected response
$\mathbf{y}_{\text{gpts}}^+$, $\mathbf{y}_{\text{gptf}}^+$: Rewrites using ChatGPT, preserving semantics or format
$\mathbf{y}_{\text{hum}}$: A randomly generated sentence by ChatGPT with the same number of characters (irrelevant to the question)
$\mathbf{y}'_{\text{rnd}}$: A purely random English word sequence
$\mathbf{y}_{\text{urnd}}^+$: A random permutation of all the words in $\mathbf{y}_u^+$
$\mathbf{y}_{j\neq u}^+$: A preferred response for another question in the training set
$\mathbf{y}_{\text{test}}^+$: A preferred response randomly selected from the test set
Argmax : Greedy decoding result

Experiment 1



Learning dynamics of SFT

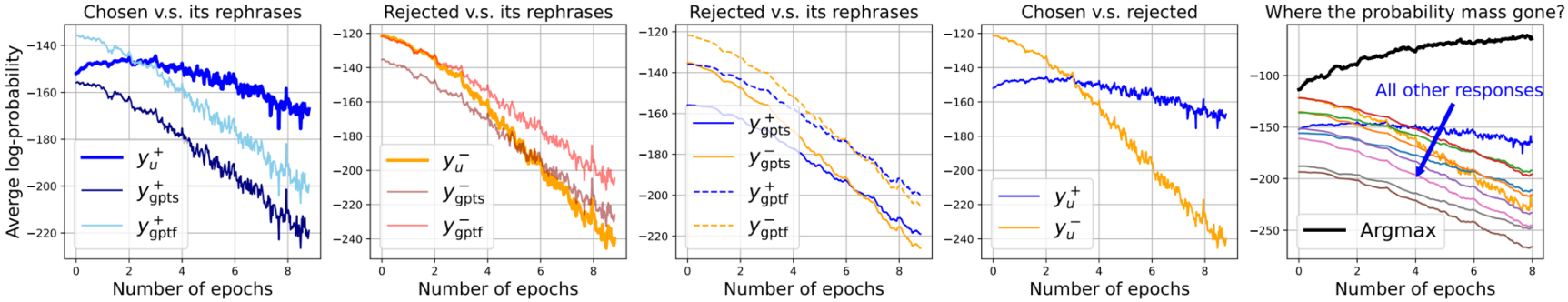
Comments:

1. The dark blue line continues to rise as expected. Similar but different responses all experienced an increase initially. This is because they are subject to **indirect pull**. However, as training progressed, the indirect pull weakened, and the **overall downward pressure** took over, causing these curves to subsequently begin to decline.

2. Completely randomized responses have **no similarity** and receive almost **no upward pull**, so from the beginning, they show a tendency to be **pushed purely downward by global pressure**.

3. This demonstrates the bizarre rise of the correct answer to an irrelevant question. Because it was also learned as a target during training (albeit for a different question), this learning effect generalizes. This strongly explains the **hallucination** created by the fine-tuned model.

Experiment 2



Learning dynamics of DPO

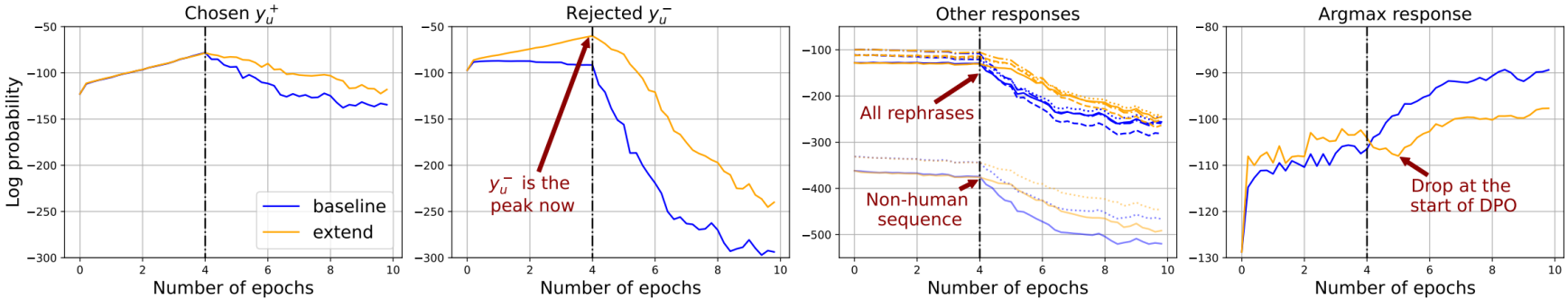
Comments:

1/2/3. The **upward pressure** is directly imposed on the **chosen response** rather than these rephrases. The **negative pressure** is directly imposed on the **rejected response** rather than these rephrases. y_{gpt} are close to the chosen and rejected response.

4. The margin difference indicates that the model **is gaining** the ability to separate between good and bad responses **as the training goes on**.

5. The probability of all reasonable responses is decreasing. The black Argmax curve shows an increase. This **demonstrates the squeezing effect**.

Experiment 3



Improvement: Including rejected responses in the SFT stage during training, which can increase the baseline confidence level of rejected responses in advance.

Comments:

1/2. During the SFT phase (the first 4 epochs), the Extend method successfully **raised the probability of rejected responses to a high peak**, preparing for the DPO phase.

3. After entering the DPO phase, the decay rate of reasonable responses in the Extend method is smoother than that of the baseline, **weakening the squeezing effect**.

4. The Argmax confidence of the Extend method **increases relatively slowly**, proving that the probability distribution retains **better diversity** and does not fall into sharp collapse.

Key Takeaways

A unified learning dynamics perspective

- Finetuning updates can be decomposed into **adaptation**, **similarity (eNTK)**, and **residual** terms

SFT produces a single positive residual, which broadly ‘pulls up’ similar sequences and smoothly redistributes probability.

Off-Policy DPO has a strong negative residual that results in a squeezing of the probability towards the argmax, leading to very strong spikes in the distribution over output tokens.

This explains:

- Hallucination amplification from SFT
- Degeneration/repetition after DPO
- Off-policy vs on-policy performance gap

Strengths + Weaknesses

Strengths:

- Derivation is theoretically sound and uses one framework for the different finetuning approaches (SFT, DPO. Other RL-free alignment techniques) without requiring additional assumptions
- Provides strong explanations for otherwise counterintuitive results
- Strong empirical validation including multiple models, dataset, and a large variety of response types that matches the theory quite well
- Produces real actionable insight: perform SFT with y_u^- as well before DPO
- Improved the interpretability of the training process

Strengths + Weaknesses

Weaknesses:

- This is a first-order approximation; issues could arise with large learning rates or long training horizons
- Squeeze effect analysis is primarily in the single-token, logistic regression case, whereas LLMs are much more complex. Empirical results are convincing, but it is still a weakness
- Training rejected responses during the SFT stage essentially increases the probability of the model generating incorrect, low-quality, or even harmful content, and the model may retain these undesirable generation tendencies.

Questions + Personal takeaways

Questions:

- Would these result hold in significantly larger models?
- Is there a way to alter the DPO objective directly to prevent the squeeze effect?

Personal Takeaways:

- Studying learning dynamics allows for a mechanistic explanation of alignment behaviour instead of purely empirical assumptions.
- Alignment requires good data, but it is important to also understand the interaction between the data and the optimization algorithm you are using, because it can have large impacts

Thanks!

Q & A

Additional Slide: DPO Dynamics Derivation

$$
\begin{aligned}
\underbrace{\nabla_\theta \log \pi^t(\boldsymbol{\chi}_o)|_{\theta^t}}_{V\times d}\underbrace{\Delta\theta^t}_{d\times 1} &= \big(\underbrace{\nabla_{\mathbf{z}} \log \pi^t(\boldsymbol{\chi}_o)|_{\mathbf{z}^t}}_{V\times V}\underbrace{\nabla_\theta \mathbf{z}^t(\boldsymbol{\chi}_o)|_{\theta^t}}_{V\times d}\big)\big(-\eta\underbrace{\nabla_\theta \mathcal{L}(\mathbf{x}_u, \mathbf{y}_u^+, \mathbf{y}_u^-)|_{\theta^t}}_{1\times d}\big)^\mathsf{T} \\
&= \underbrace{\nabla_{\mathbf{z}} \log \pi^t(\boldsymbol{\chi}_o)|_{\mathbf{z}^t}}_{V\times V}\underbrace{\nabla_\theta \mathbf{z}^t(\boldsymbol{\chi}_o)|_{\theta^t}}_{V\times d}\big(\underbrace{-\eta\nabla_{[\mathbf{z}^+;\mathbf{z}^-]}\mathcal{L}|_{\mathbf{z}^t}}_{1\times 2V}\underbrace{\big[\nabla_\theta \mathbf{z}^+(\boldsymbol{\chi}_u^+); \nabla_\theta \mathbf{z}^-(\boldsymbol{\chi}_u^-)\big]|_{\theta^t}}_{2V\times d}\big)^\mathsf{T} \\
&= -\eta\underbrace{\nabla_{\mathbf{z}} \log \pi^t(\mathbf{x}_o)|_{\mathbf{z}^t}}_{V\times V}\Big[\underbrace{\nabla_\theta \mathbf{z}^t(\mathbf{x}_o)|_{\theta^t}}_{V\times d}\underbrace{\big(\big[\nabla_\theta \mathbf{z}^+(\boldsymbol{\chi}_u^+); \nabla_\theta \mathbf{z}^-(\boldsymbol{\chi}_u^-)\big]|_{\theta^t}\big)^\mathsf{T}}_{d\times 2V}\Big]\underbrace{\big(\nabla_{[\mathbf{z}^+;\mathbf{z}^-]}\mathcal{L}|_{\mathbf{z}^t}\big)^\mathsf{T}}_{2V\times 1} \\
&= -\eta\mathcal{A}^t(\boldsymbol{\chi}_o)\big[\mathcal{K}^t(\boldsymbol{\chi}_o, \boldsymbol{\chi}_u^+); \mathcal{K}^t(\boldsymbol{\chi}_o, \boldsymbol{\chi}_u^-)\big]\big(\nabla_{[\mathbf{z}^+;\mathbf{z}^-]}\mathcal{L}|_{\mathbf{z}^t}\big)^\mathsf{T} \\
&\triangleq -\eta\mathcal{A}^t(\boldsymbol{\chi}_o)\left(\mathcal{K}^t(\boldsymbol{\chi}_o, \boldsymbol{\chi}_u^+)\mathcal{G}^t_{\text{DPO+}}(\boldsymbol{\chi}_u^+) - \mathcal{K}^t(\boldsymbol{\chi}_o, \boldsymbol{\chi}_u^-)\mathcal{G}^t_{\text{DPO-}}(\boldsymbol{\chi}_u^-)\right) \qquad (13)
\end{aligned}
$$

Section 1

Subsection 1