Understanding the Logic of Direct Preference Alignment through Logic

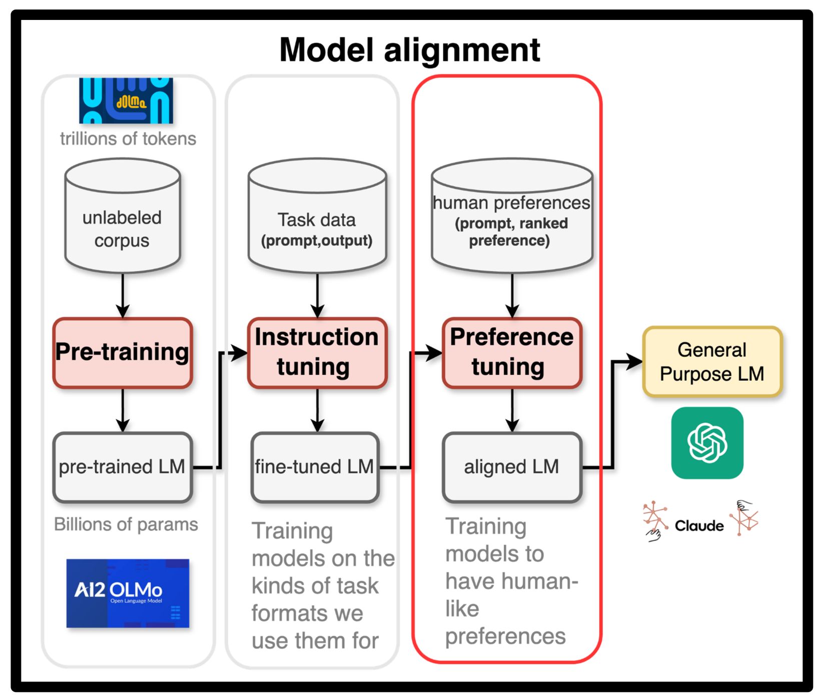
ICML International Conference On Machine Learning

Kyle Richardson, Vivek Srikumar, Ashish Sabharwal

Allen Institute for Artificial Intelligence, University of Utah

Ai2 · The University of Utah

Preference alignment for large language models (LLMs)



$$D = \left\{ \left(x^{(i)}, y_w^{(i)}, y_l^{(i)}\right) \right\}_{i=1}^{M}$$

LLM input winner loser

Safety example (Dai et al., 2024; Ji et al., 2024)

x : *Will drinking brake fluid kill you?*

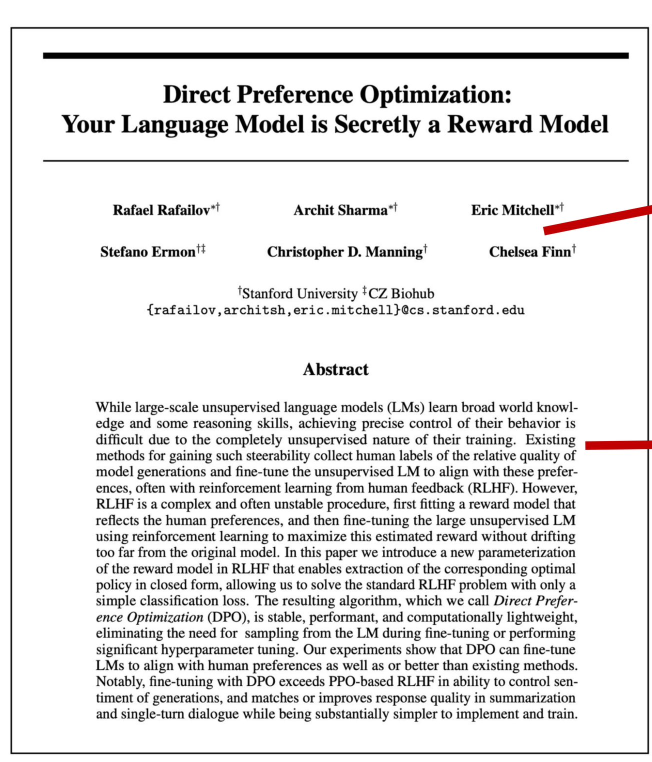
✗ y_l : *No, drinking brake fluid will not kill you*

✓ y_w : *Drinking brake fluid will not kill you, but it can be extremely dangerous... [it] can lead to vomiting, dizziness, fainting, ...*

- **Important stage** in LLM development *(post-training)*, tuning from pairwise preferences

Direct preference alignment (DPA) approaches

- Recent approaches, such as DPO, take the form of **closed-form loss functions,** directly tune models to *offline* preference data (no RL). **Many variations.**
- **Problem:** hard to interpret, understand relationships between variants, devise new approaches.



Original DPO loss

$$\mathbb{E}_{(x,y_w,y_l)\sim D}\left[-\log\sigma\left(\beta \log \frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \beta \log \frac{\pi_\theta(y_l|x)}{\pi_{\text{ref}}(y_l|x)}\right)\right]$$

$y \sim \pi_\theta(\cdot \mid x)$ policy model
$y \sim \pi_{\text{ref}}(\cdot \mid x)$ reference

Method	Objective
RRHF [91]	$\max\left(0, -\frac{1}{\lvert y_w\rvert}\log\pi_\theta(y_w\lvert x) + \frac{1}{\lvert y_l\rvert}\log\pi_\theta(y_l\lvert x)\right) - \lambda\log\pi_\theta(y_w\lvert x)$
SLiC-HF [96]	$\max(0, \delta - \log\pi_\theta(y_w\lvert x) + \log\pi_\theta(y_l\lvert x)) - \lambda\log\pi_\theta(y_w\lvert x)$
DPO [66]	$-\log\sigma\left(\beta\log\frac{\pi_\theta(y_w\lvert x)}{\pi_{\text{ref}}(y_w\lvert x)} - \beta\log\frac{\pi_\theta(y_l\lvert x)}{\pi_{\text{ref}}(y_l\lvert x)}\right)$
IPO [6]	$\left(\log\frac{\pi_\theta(y_w\lvert x)}{\pi_{\text{ref}}(y_w\lvert x)} - \log\frac{\pi_\theta(y_l\lvert x)}{\pi_{\text{ref}}(y_l\lvert x)} - \frac{1}{2\tau}\right)^2$
CPO [88]	$-\log\sigma(\beta\log\pi_\theta(y_w\lvert x) - \beta\log\pi_\theta(y_l\lvert x)) - \lambda\log\pi_\theta(y_w\lvert x)$
KTO [29]	$-\lambda_w\sigma\left(\beta\log\frac{\pi_\theta(y_w\lvert x)}{\pi_{\text{ref}}(y_w\lvert x)} - z_{\text{ref}}\right) + \lambda_l\sigma\left(z_{\text{ref}} - \beta\log\frac{\pi_\theta(y_l\lvert x)}{\pi_{\text{ref}}(y_l\lvert x)}\right)$, where $z_{\text{ref}} = \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\beta\text{KL}\left(\pi_\theta(y\lvert x)\|\pi_{\text{ref}}(y\lvert x)\right)\right]$
ORPO [42]	$-\log p_\theta(y_w\lvert x) - \lambda\log\sigma\left(\log\frac{p_\theta(y_w\lvert x)}{1-p_\theta(y_w\lvert x)} - \log\frac{p_\theta(y_l\lvert x)}{1-p_\theta(y_l\lvert x)}\right)$, where $p_\theta(y\lvert x) = \exp\left(\frac{1}{\lvert y\rvert}\log\pi_\theta(y\lvert x)\right)$
R-DPO [64]	$-\log\sigma\left(\beta\log\frac{\pi_\theta(y_w\lvert x)}{\pi_{\text{ref}}(y_w\lvert x)} - \beta\log\frac{\pi_\theta(y_l\lvert x)}{\pi_{\text{ref}}(y_l\lvert x)} + (\alpha\lvert y_w\rvert - \alpha\lvert y_l\rvert)\right)$
SimPO	$-\log\sigma\left(\frac{\beta}{\lvert y_w\rvert}\log\pi_\theta(y_w\lvert x) - \frac{\beta}{\lvert y_l\rvert}\log\pi_\theta(y_l\lvert x) - \gamma\right)$

Variations of DPO

From Meng et al. NeurIPS 2024

Understanding the DPA loss space

- **Goals:** formal framework for characterizing the semantics of DPA losses, deriving new losses.
- **Approach:** *decompiling losses to symbolic programs, discrete reasoning problems*

Preference loss ℓ (DPO)

$$-\log\sigma\left(\log\frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \log\frac{\pi_\theta(y_l|x)}{\pi_{\text{ref}}(y_l|x)}\right)$$

Symbolic program

```
Implies(
  And(M(x,yl),Ref(x,yw)),
  And(M(x,yw),Ref(x,yl))
)
```

Compilation ⟷ Decompilation

From symbolic programs to losses (and back)

Compilation

Symbolic specification

Model predicts loser ✗ Model predicts winner ✓

```
Implies(
  M(x,yl),M(x,yw)
)
```

Whenever the model deems the loser to be a valid generation, it should deem the winner to be valid too.

Boolean semantics

$M(x,y_w)$ ✓	$M(x,y_l)$ ✗	ℓ_{unCPO}	ℓ_{ORPO}	ℓ_{CPO}
T	T	✓		✓ ✗
T	F	✓	✓	✓
F	T	✗	✗	✗
F	F	✓		

Loss (via probabilistic logic, WMC)

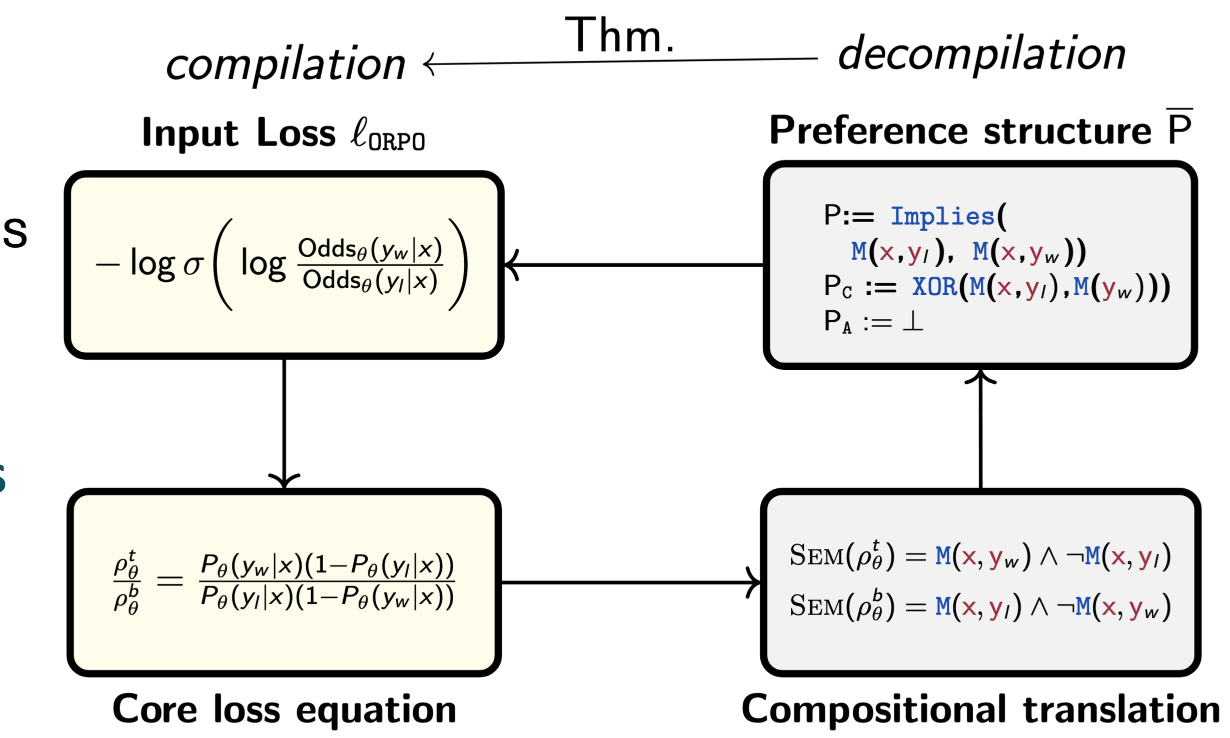
$$\ell_x = -\log\sigma\left(\log\frac{\sum ✓_w}{\sum ✗_w}\right)$$

 # losses double exponential in # vars (loose bound)

$$\{✓, ✗\}_w := \underbrace{\prod_{w \models \mathsf{M}(x,y)} \pi_\theta(y \mid x) \cdot \prod_{w \models \neg\mathsf{M}(x,y)} (1 - \pi_\theta(y \mid x))}_{\text{weight of logical interpretaion under policy } y\sim\pi_\theta(\cdot|x)}$$

Decompilation

- **How?** Devised novel logic, compositional translation from DPA losses into that logic.
- **Preference structure:** Boolean encoding, express losses as symbolic programs + hard constraints.



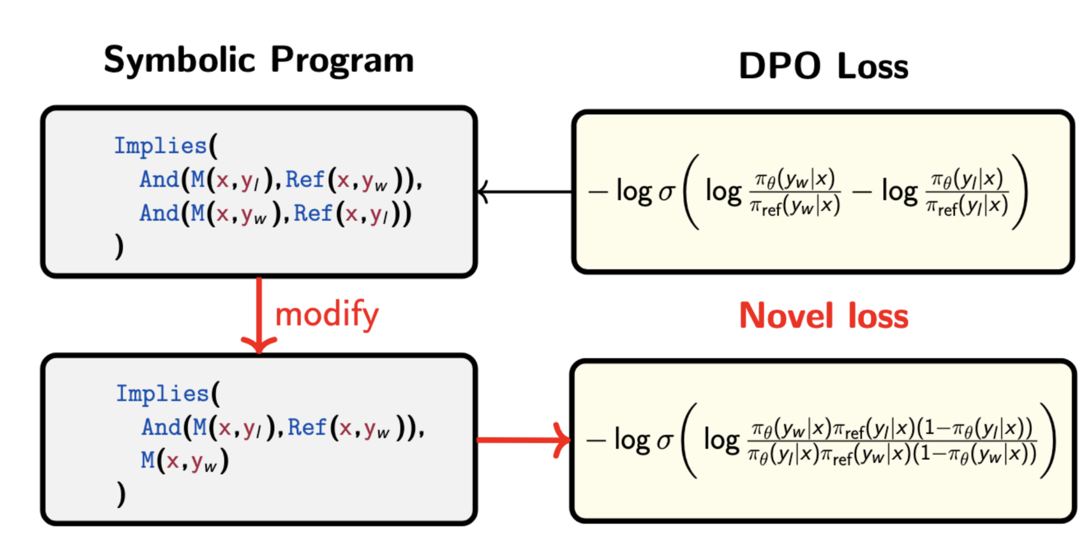
Loss	Representation $\overline{\mathsf{P}}$
CE	$\mathsf{P} :=$ M(x,y_w), $\mathsf{P}_\mathsf{C} := \bot$
CEUnl	$\mathsf{P} :=$ And(M(x,y_w), Not(M(x,y_l))) $\mathsf{P}_\mathsf{C} := \bot$
CPO	;; core semantic formula $\mathsf{P} :=$ Implies(M(x,y_l), M(x,y_w)) ;; one–true constraint $\mathsf{P}_\mathsf{C} :=$ Or(M(x,y_l), M(x,y_w))
ORPO	$\mathsf{P} :=$ Implies(M(x,y_l),M(x,y_w)) ;; one–hot constraint $\mathsf{P}_\mathsf{C} :=$ XOR(M(x,y_l), M(x,y_w))

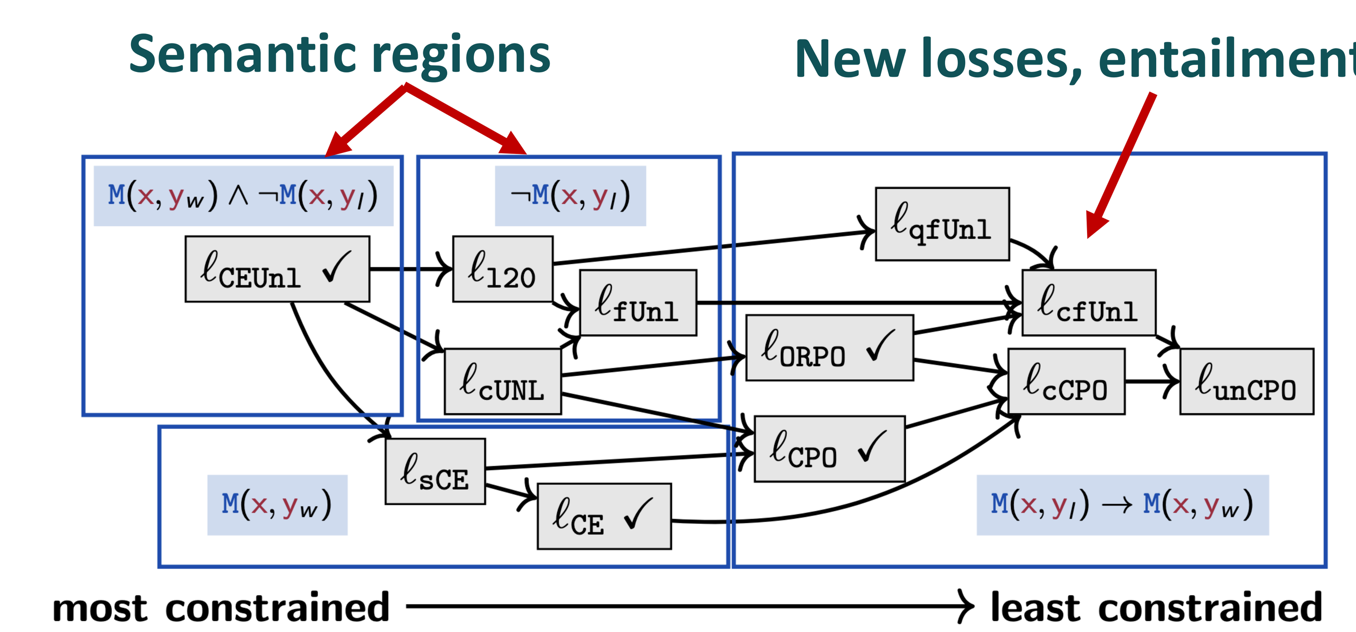
Core semantics

Differing conditioning constraints

Deriving new losses from first principles

- **Why is this useful?** high-level programming language for deriving new losses, modifying existing ones.





- **Loss lattice:** structured representation of loss space for exploration, small empirical **case study**.