Expectation–Maximization Formulations for SFT and RL Fine-Tuning of Transformers

February 18, 2026

Contents

1 Introduction

Transformer language models are commonly adapted to downstream tasks via supervised fine-tuning (SFT), and further improved via RL fine-tuning against a learned or human preference reward. While these procedures are usually presented as distinct (cross-entropy training versus policy optimization), both can be interpreted as alternating between constructing a training target distribution and then fitting the model to that target.

2 Research Questions

1. Can supervised fine-tuning and KL-regularized RL fine-tuning be expressed under a common EM/MAP formulation with a shared E-step/M-step interpretation?
2. In what precise sense does V-MPO correspond to regularized policy iteration, and how does that differ from direct policy-gradient optimization (e.g. PPO)?
3. Does adaptive temperature optimization in the E-step provide a practical stability advantage over fixed-temperature weighting at LLM scale?
4. How does the DAR closed-form target relate to the V-MPO target, and under which limits are they equivalent?
5. Do EM-style weighted-MLE updates provide practical benefits for transformer fine-tuning, including compatibility with dropout-style regularization?

3 Preliminaries

Maximum Likelihood and Maximum a Posteriori

Given observed data x and parameters θ, maximum likelihood (ML) estimation solves

$$\theta_{\mathrm{ML}} = \arg\max_{\theta} \log p_\theta(x).$$

Maximum a posteriori (MAP) estimation adds a prior:

$$\theta_{\mathrm{MAP}} = \arg\max_{\theta} (\log p_\theta(x) + \log p(\theta)).$$

In optimization form, $-\log p(\theta)$ acts as a regularizer. A flat prior recovers ML.

EM Lower Bound

Assume a latent variable model with latent z. For any auxiliary distribution $q(z)$:

$$\log p_\theta(x) = \underbrace{\mathbb{E}_{q(z)}[\log p_\theta(x,z) - \log q(z)]}_{\mathcal{L}(q,\theta)} + \mathrm{KL}(q(z) \,\|\, p_\theta(z \mid x)).$$

Since KL is nonnegative, $\mathcal{L}(q,\theta)$ is a lower bound on $\log p_\theta(x)$.
For MAP, the objective

$$F_{\mathrm{MAP}}(\theta) = \log p_\theta(x) + \log p(\theta)$$

admits the decomposition

$$F_{\mathrm{MAP}}(\theta) = \mathcal{L}_{\mathrm{MAP}}(q,\theta) + \mathrm{KL}(q(z) \,\|\, p_\theta(z \mid x)),$$

with

$$\mathcal{L}_{\mathrm{MAP}}(q,\theta) = \mathbb{E}_{q(z)}[\log p_\theta(x,z) - \log q(z)] + \log p(\theta).$$

EM Iterations

At iteration k, EM alternates:

$$\text{E-step:} \quad q_{k+1}(z) = p_{\theta_k}(z \mid x),$$

$$\text{M-step:} \quad \theta_{k+1} = \arg\max_\theta \mathcal{L}_{\mathrm{MAP}}(q_{k+1}, \theta).$$

Equivalently, the MAP M-step maximises

$$Q_{\mathrm{MAP}}(\theta, \theta_k) = \mathbb{E}_{z \sim p_{\theta_k}(z|x)}[\log p_\theta(x,z)] + \log p(\theta).$$

This gives the standard monotonic-improvement template used throughout this document: construct a target distribution in the E-step, then fit a parametric model to that target in the M-step.

EM-Style Policy Iteration vs Policy Gradients

It is useful to distinguish two update styles for RL fine-tuning.

Policy-gradient update (direct parameter-space step). Define

$$J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta}[R(\tau)].$$

A policy-gradient method takes a local ascent step

$$\theta_{k+1} = \theta_k + \lambda \widehat{\nabla_\theta J(\theta_k)},$$

optionally with clipping or explicit KL penalties (as in PPO/TRPO-style variants).

EM-style policy iteration (distribution-space then projection). EM-style methods introduce an auxiliary improved policy $q(a \mid s)$ and alternate:

$$\text{E-step:} \quad q_{k+1} = \arg\max_{q} \Big(\mathbb{E}_{s\sim d_{\pi_k},\, a\sim q}[A^{\pi_k}(s,a)] - \eta\, \mathrm{KL}(q(\cdot \mid s) \,\|\, \pi_k(\cdot \mid s)) \Big),$$

whose solution has Boltzmann form

$$q_{k+1}(a \mid s) \propto \pi_k(a \mid s)\, \exp\Big(\frac{A^{\pi_k}(s,a)}{\eta}\Big).$$

Then the M-step projects back to the parametric family:

$$\theta_{k+1} = \arg\max_{\theta}\ \mathbb{E}_{s\sim d_{\pi_k},\, a\sim q_{k+1}}[\log \pi_\theta(a \mid s)],$$

often with an additional trust-region term on $\mathrm{KL}(\pi_k \| \pi_\theta)$.

Interpretation. Policy gradients optimize parameters directly via noisy first-order steps. EM-style methods perform *policy improvement* in distribution space first, then do weighted maximum-likelihood fitting. This is why V-MPO is naturally viewed as regularized policy iteration in EM form rather than as a pure policy-gradient method.

4 PPO

Proximal Policy Optimisation (PPO) is an on-policy actor-critic algorithm that constrains each policy update to stay close to the behaviour policy via a clipped surrogate objective, avoiding the instability of unconstrained policy gradient steps.

Setup

A trajectory $\tau = (s_0, a_0, \ldots, s_{T-1}, a_{T-1})$ is collected under the current policy $\pi_{\theta_{\text{old}}}$. Per-step log-probabilities and their sum are

$$\ell_{\theta,t} = \log \pi_\theta(a_t \mid s_t), \qquad \ell_\theta(\tau) = \sum_{t=0}^{T-1} \ell_{\theta,t}.$$

The discounted reward-to-go from step t is

$$R_t = \sum_{k=t}^{T-1} \gamma^{k-t} r_k.$$

Policy Gradient Foundation

The policy gradient theorem gives the direction of steepest ascent for the expected return:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t \mid s_t)\, A_t\right].$$

To reuse data collected under $\pi_{\theta_{\text{old}}}$, importance sampling introduces the per-step probability ratio

$$r_t(\theta) = \frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)} = \exp(\ell_{\theta,t} - \ell_{\text{old},t})\,.$$

The unclipped surrogate objective is then $L^{\text{PG}}(\theta) = \mathbb{E}_t[r_t(\theta)\, A_t]$, but without further constraint this can lead to destructively large updates.

Clipped Surrogate Objective

PPO resolves this by clipping the ratio to $[1-\epsilon,\, 1+\epsilon]$ and taking the pessimistic (minimum) of the clipped and unclipped terms:

$$L^{\text{CLIP}}(\theta) = \mathbb{E}_t\left[\min(r_t(\theta)\, A_t,\ \text{clip}(r_t(\theta),\, 1-\epsilon,\, 1+\epsilon)\, A_t)\right].$$

When the advantage is positive the update is capped at a ratio of $1+\epsilon$; when negative it is capped at $1-\epsilon$. This prevents the policy from moving too far in either direction within a single update.

Value Function and Entropy

The critic is fitted by minimising a squared regression loss to the empirical returns:

$$L^{\text{VF}}(\phi) = \mathbb{E}_t\left[\left(V_\phi(s_t) - R_t\right)^2\right].$$

An entropy bonus encourages exploration by penalising premature policy collapse:

$$S(\pi_\theta(\cdot \mid s_t)) = -\sum_a \pi_\theta(a \mid s_t) \log \pi_\theta(a \mid s_t).$$

Full Objective

The three terms are combined into a single objective (to minimise):

$$\mathcal{L}(\theta,\phi) = -L^{\text{CLIP}}(\theta) + c_1\, L^{\text{VF}}(\phi) - c_2\, \mathbb{E}_t[S(\pi_\theta(\cdot \mid s_t))]\,,$$

where c_1 and c_2 are scalar coefficients balancing the three losses.

Generalised Advantage Estimation (GAE)

Rather than using raw Monte Carlo returns to estimate A_t, PPO typically uses GAE, which trades off bias and variance via a decay parameter $\lambda \in [0, 1]$.

The TD residual at each step is

$$\delta_t = r_t + \gamma V_\phi(s_{t+1}) - V_\phi(s_t).$$

GAE accumulates these residuals with exponentially decaying weights:

$$A_t^{\mathrm{GAE}(\gamma,\lambda)} = \sum_{l=0}^{\infty} (\gamma\lambda)^l \, \delta_{t+l},$$

which satisfies the efficient recurrence

$$A_t = \delta_t + \gamma\lambda \, A_{t+1}.$$

Advantages are then batch-normalised before use:

$$\hat{A}_t = \frac{A_t - \mu_A}{\sigma_A + \varepsilon}.$$

Sequence-Level PPO (LLM Case)

When the "action" is an entire generated sequence (as in LLM fine-tuning), the per-step ratios multiply into a sequence-level ratio:

$$r_{\mathrm{seq}}(\theta) = \exp\left(\sum_{t=0}^{T-1} (\ell_{\theta,t} - \ell_{\mathrm{old},t}) \right).$$

A KL penalty between the updated and old policy can be estimated cheaply as

$$\widehat{\mathrm{KL}} = \mathbb{E}_t[\ell_{\mathrm{old},t} - \ell_{\theta,t}],$$

and is often added to the objective or used as an early-stopping criterion to keep the sequence-level ratio well-behaved.

5 V-MPO

V-MPO (Song et al. 2019) decomposes optimisation into two conceptual phases:

- **Policy Evaluation** (critic update)
- **Policy Improvement**, implemented via an EM procedure:
  - E-step (non-parametric policy)
  - M-step (parametric projection with KL constraint)

The total objective is

$$\mathcal{L}(\phi, \theta, \eta, \alpha) = \mathcal{L}_V(\phi) + \mathcal{L}_{\text{V-MPO}}(\theta, \eta, \alpha),$$

where

$$\mathcal{L}_{\text{V-MPO}}(\theta, \eta, \alpha) = \mathcal{L}_\pi(\theta) + \mathcal{L}_\eta(\eta) + \mathcal{L}_\alpha(\theta, \alpha).$$

Implementation

github.com/nenuadrian/rl/tree/main/benchmarks

Policy Evaluation (Critic Update)

The value function is fitted via n-step bootstrapped regression:

$$\mathcal{L}_V(\phi) = \frac{1}{2|\mathcal{D}|} \sum_{s_t \sim \mathcal{D}} \left(V_\phi^\pi(s_t) - G_t^{(n)} \right)^2,$$

with

$$G_t^{(n)} = \sum_{k=t}^{t+n-1} \gamma^{k-t} r_k + \gamma^n V_\phi^\pi(s_{t+n}).$$

Advantages are defined as

$$A^\pi(s_t, a_t) = G_t^{(n)} - V_\phi^\pi(s_t).$$

This completes the evaluation phase.

Policy Improvement via EM

We formulate policy improvement as MAP estimation:

$$\theta^\star = \arg\max_\theta \log p_\theta(I = 1) + \log p(\theta),$$

where I denotes the improvement event.

Introduce a variational distribution $\psi(s, a)$:

$$\log p_\theta(I = 1) = \sum_{s,a} \psi(s, a) \log \frac{p_\theta(I = 1, s, a)}{\psi(s, a)} + \mathrm{KL}(\psi(s, a) \parallel p_\theta(s, a \mid I = 1)).$$

We now alternate between E-step and M-step.

E-Step: Non-Parametric Policy Construction

The E-step solves

$$\begin{aligned} \psi^\star = \arg\max_\psi & \sum_{s,a} \psi(s, a) A^{\pi_{\theta_{\text{old}}}}(s, a) \\ \text{s.t.} & \sum_{s,a} \psi(s, a) \log \frac{\psi(s, a)}{p_{\theta_{\text{old}}}(s, a)} < \epsilon_\eta, \\ & \sum_{s,a} \psi(s, a) = 1. \end{aligned}$$

The Lagrangian is

$$\begin{aligned} J(\psi, \eta, \lambda) = &\sum_{s,a} \psi(s,a) A^{\pi_{\theta_{\text{old}}}}(s,a) \\ &+ \eta\Big(\epsilon_\eta - \sum_{s,a} \psi(s,a) \log \frac{\psi(s,a)}{p_{\theta_{\text{old}}}(s,a)}\Big) \\ &+ \lambda\Big(1 - \sum_{s,a} \psi(s,a)\Big). \end{aligned}$$

Stationarity gives

$$\psi(s,a) = \frac{p_{\theta_{\text{old}}}(s,a) \exp(A^{\pi_{\theta_{\text{old}}}}(s,a)/\eta)}{\sum_{s',a'} p_{\theta_{\text{old}}}(s',a') \exp(A^{\pi_{\theta_{\text{old}}}}(s',a')/\eta)}.$$

The temperature dual is

$$\mathcal{L}_\eta(\eta) = \eta\epsilon_\eta + \eta \log\Big(\sum_{s,a} p_{\theta_{\text{old}}}(s,a) \exp\left(A^{\pi_{\theta_{\text{old}}}}(s,a)/\eta\right)\Big).$$

This completes the E-step.

M-Step: Parametric Projection with KL Constraint

The M-step minimises the negative lower bound:

$$\mathcal{L}_\pi(\theta) = -\sum_{s,a} \psi(s,a) \log \pi_\theta(a|s).$$

Subject to a KL trust-region constraint:

$$\mathbb{E}_{s\sim p(s)}[\mathrm{KL}(\pi_{\theta_{\text{old}}}(\cdot|s) \parallel \pi_\theta(\cdot|s))] < \epsilon_\alpha.$$

The Lagrangian form is

$$J(\theta, \alpha) = \mathcal{L}_\pi(\theta) + \alpha\Big(\epsilon_\alpha - \mathbb{E}_s \mathrm{KL}(\pi_{\theta_{\text{old}}} \| \, \pi_\theta)\Big). \tag{12}$$

In implementation, the loss becomes

$$\begin{aligned} \mathcal{L}_\alpha(\theta, \alpha) = &\alpha\Big(\epsilon_\alpha - \mathrm{sg}[\mathrm{KL}(\pi_{\theta_{\text{old}}} \| \, \pi_\theta)]\Big) \\ &+ \mathrm{sg}[\alpha] \, \mathrm{KL}(\pi_{\theta_{\text{old}}} \| \, \pi_\theta). \end{aligned} \tag{1}$$

6 EM for Fine-Tuning and LLM-Scale Adaptive Temperature

Both SFT and RL fine-tuning fit naturally into an EM template: the E-step constructs a target distribution over sequences, and the M-step fits the model to it by weighted maximum likelihood. This section develops the template from first principles and then shows how it specialises into a numerically stable, adaptive-temperature procedure for sequence-level V-MPO at LLM scale.

E-step / M-step for SFT

Let $\mathcal{D}_{\mathrm{SFT}} = \{(x, y)\}$ be prompt–response pairs, and let $\tau = y$ denote the response sequence. Under teacher forcing, SFT minimizes token-level cross-entropy, equivalently maximizing $\log \pi_\theta(y \mid x)$.

An EM view is obtained by defining an auxiliary distribution $q(\tau \mid x)$ that places all mass on the demonstrated response:

$$\text{(E-step)} \quad q(\tau \mid x) := \delta(\tau = y). \tag{2}$$

The M-step is then maximum likelihood under q:

$$\text{(M-step)} \quad \max_\theta \ \mathbb{E}_{(x,y)\sim\mathcal{D}_{\mathrm{SFT}}} \left[\log \pi_\theta(y \mid x)\right]. \tag{3}$$

More generally, label smoothing, n-best targets, or distillation from a teacher define non-degenerate $q(\tau \mid x)$ and preserve the same EM structure.

E-step / M-step for RL Fine-Tuning (KL-Regularised)

Consider RL fine-tuning on prompts $x \sim \mathcal{D}_x$ with reward $R(x, \tau)$ and a reference policy $\pi_{\mathrm{ref}}(\tau \mid x)$. A common objective is the KL-regularised expected reward

$$J(\theta) = \mathbb{E}_x \, \mathbb{E}_{\tau\sim\pi_\theta(\cdot|x)} \Big[R(x, \tau) - \beta \, \mathrm{KL}(\pi_\theta(\cdot \mid x) \,\|\, \pi_{\mathrm{ref}}(\cdot \mid x)) \Big],$$

with $\beta > 0$ controlling the trust region to π_{ref}.

The E-step constructs an energy-based target distribution from rollouts and advantage estimates:

$$\text{(E-step)} \quad q(\tau \mid x) \propto \pi_{\mathrm{ref}}(\tau \mid x) \exp\left(\frac{R(x, \tau)}{\beta}\right).$$

The M-step then fits the model to q by weighted maximum likelihood:

$$\text{(M-step)} \quad \max_\theta \ \mathbb{E}_x \, \mathbb{E}_{\tau\sim q(\cdot|x)} \left[\log \pi_\theta(\tau \mid x)\right],$$

optionally with an explicit constraint/penalty on $\mathrm{KL}(\pi_\theta \,\|\, \pi_{\mathrm{ref}})$ to stabilise optimisation.

Token-Level Form

For teacher-forced updates, the sequence-level log-likelihood decomposes token-by-token:

$$\log \pi_\theta(\tau \mid x) = \sum_{t=1}^{T} \log \pi_\theta(a_t \mid s_t).$$

Weighted M-steps are implemented with per-sequence weights w_i broadcast uniformly to all tokens in that sequence, or alternatively with per-token weights where finer-grained advantage estimates are available.

Sequence-Level V-MPO at LLM Scale

At LLM scale the action space is the full token vocabulary and sequences can be hundreds of tokens long, so several practical adaptations are needed to apply the V-MPO EM procedure stably.

Let a batch of sampled sequences be

$$\mathcal{B} = \{(x^{(i)}, y^{(i)}, A^{(i)})\}_{i=1}^{N},$$

where $A^{(i)}$ denotes a scalar sequence-level advantage (e.g. $A^{(i)} = r^{(i)} - V_\phi(x^{(i)})$). The V-MPO non-parametric E-step target over a selected subset $S \subset \{1, \dots, N\}$ is

$$\psi(i) \propto \exp\Big(\frac{A^{(i)}}{\eta}\Big),$$

where $\eta > 0$ is a temperature that controls how sharply the distribution peaks on high-advantage samples.

Dual Objective for the Temperature

Rather than fixing η as a hyperparameter, V-MPO treats it as a dual variable and optimises it to satisfy a KL budget ε_η. Let $|S| = k$ and define the partition function

$$Z(\eta) = \sum_{i \in S} \exp\Big(\frac{A^{(i)}}{\eta}\Big).$$

The temperature dual objective is

$$L_\eta(\eta) = \eta\, \varepsilon_\eta + \eta \log\Big(\frac{Z(\eta)}{k}\Big) = \eta\, \varepsilon_\eta + \eta\, g(\eta),$$

where $g(\eta) = \log(Z(\eta)/k)$ and $\varepsilon_\eta > 0$ is the target entropy/KL budget. Minimising L_η with respect to η automatically tunes the temperature so that the E-step distribution does not collapse or spread too widely.

Taking the derivative, and using $g'(\eta) = -\frac{1}{\eta^2}\mathbb{E}_{p_\eta}[A]$ where p_η is the softmax distribution over scaled advantages, gives

$$\frac{dL_\eta}{d\eta} = \varepsilon_\eta + g(\eta) - \frac{1}{\eta}\,\mathbb{E}_{p_\eta}[A].$$

Numerically Stable Log-Sum-Exp Form

Directly computing $Z(\eta)$ for large $|A^{(i)}|/\eta$ causes floating-point overflow. Defining scaled advantages $u_i = A^{(i)}/\eta$ and subtracting the batch maximum $m = \max_{i \in S} u_i$ gives

$$g(\eta) = m + \log\Big(\frac{1}{k}\sum_{i \in S} \exp(u_i - m)\Big),$$

with normalised weights

$$w_i = \frac{\exp(u_i - m)}{\sum_{j \in S} \exp(u_j - m)}.$$

Since $u_i - m \leq 0$ for all i, no term can overflow.

Top-k Selection

To focus the M-step on demonstrably good samples, the E-step restricts S to the top-ρ fraction of the batch by advantage. Using a detached (stop-gradient) temperature η_{det} for selection,

$$u_i^{\text{det}} = \frac{A^{(i)}}{\eta_{\text{det}}}, \qquad k = \max(1, \lfloor \rho N \rfloor), \qquad S = \text{Top-}\,k\big(u_i^{\text{det}}\big)\,.$$

Positive Parameterisation of the Temperature

To ensure $\eta > 0$ throughout optimisation, an unconstrained scalar $\xi \in \mathbb{R}$ is introduced:

$$\eta(\xi) = \text{softplus}(\xi) + \epsilon, \qquad \epsilon > 0.$$

Gradient updates are applied to ξ via the chain rule $\nabla_\xi L_\eta = (dL_\eta/d\eta)(d\eta/d\xi)$.

Sequence-Level M-Step

Once the weights are computed, the M-step minimises the weighted teacher-forced cross-entropy over selected sequences:

$$L_\pi(\theta) = -\sum_{i \in S} w_i \sum_{t=1}^{T_i} \log \pi_\theta(y_t^{(i)} \mid y_{<t}^{(i)}, x^{(i)}).$$

Full Adaptive EM Procedure

Putting it all together, one update step proceeds as follows:

E-step:

1. $\eta_{\text{det}} = \text{stopgrad}(\eta(\xi))$,
2. $u_i = A^{(i)}/\eta_{\text{det}}$,
3. $S = \text{Top-}\, k(u_i)$,
4. $m = \max_{i \in S} u_i$,
5. $g(\eta) = m + \log\Big(\frac{1}{k} \sum_{i \in S} \exp(u_i - m)\Big)$,
6. $L_\eta = \eta(\xi)(\varepsilon_\eta + g(\eta))$,
7. $\xi \leftarrow \xi - \lambda_\eta \nabla_\xi L_\eta$.

Weight computation:

$$w_i = \frac{\exp(u_i - m)}{\sum_{j \in S} \exp(u_j - m)}.$$

M-step:

$\theta \leftarrow \theta - \lambda_\theta \nabla_\theta L_\pi$.

7 Direct Advantage Regression: Aligning LLMs with Online AI Reward

Online AI Feedback (OAIF) presents a promising alternative to Reinforcement Learning from Human Feedback (RLHF) by utilizing online AI preference in aligning language models (LLMs). However, the straightforward replacement of humans with AI deprives LLMs from learning more fine-grained AI supervision beyond binary signals. In their paper (He et al. 2025), authors propose Direct Advantage Regression (DAR), a simple alignment algorithm using online AI reward to optimize policy improvement through weighted supervised fine-tuning. As an RL-free approach, DAR maintains theoretical consistency with online RLHF pipelines while significantly reducing implementation complexity and improving learning efficiency. The empirical results underscore that AI reward is a better form of AI supervision consistently achieving higher human-AI agreement as opposed to AI preference. Additionally, evaluations using GPT-4-Turbo and MT-bench show that DAR outperforms both OAIF and online RLHF baselines.

RL Fine-tuning

Given a language model π_θ to be aligned, a prompt dataset $\mathcal{D}(x)$ and a reward model r, online RL fine-tuning aims to optimize:

$$J_{\mathrm{RLHF}}(\pi_\theta;\pi_{\mathrm{ref}}) = \max_{\pi_\theta} \mathbb{E}_{x\sim\mathcal{D}(x),y\sim\pi_\theta(y|x)}\left[r(x,y)\right] - \alpha D_{\mathrm{KL}}\left(\pi_\theta(y|x) \parallel \pi_{\mathrm{ref}}(y|x)\right).$$

Here $\alpha > 0$ controls KL regularization toward a fixed reference policy π_{ref}.

Advantage Weighted Regression

Advantage Weighted Regression (AWR) maximizes:

$$J_{\mathrm{AWR}}(\pi_\theta) = \max_{\pi_\theta} \mathbb{E}_{x\sim d_{\pi_\theta}(x),y\sim\pi_\theta(y|x)}\left[A(x,y)\right],$$

where

$$A(x,y) = r(x,y) - V^{\pi_t}(x).$$

To remove dependence on d_{π_θ}, we approximate using d_{π_t} and impose KL trust-region regularization:

$$J_{\mathrm{AWR}}(\pi_\theta;\pi_t) = \max_{\pi_\theta} \mathbb{E}_{x\sim d_{\pi_t}(x),y\sim\pi_\theta(y|x)}\left[A(x,y)\right] - \beta D_{\mathrm{KL}}\left(\pi_\theta(y|x) \parallel \pi_t(y|x)\right).$$

Direct Advantage Regression

Dual-Constrained Objective

DAR incorporates reference regularization:

$$J_{\mathrm{DAR}}(\pi_\theta;\pi_{\mathrm{ref}},\pi_t) = \max_{\pi_\theta} \mathbb{E}_{x\sim d_{\pi_t}(x),y\sim\pi_\theta(y|x)}\left[A(x,y)\right] - \alpha D_{\mathrm{KL}}(\pi_\theta\|\pi_{\mathrm{ref}}) - \beta D_{\mathrm{KL}}(\pi_\theta\|\pi_t).$$

Theorem. Under mild assumption, given a dualconstrained advantage (or reward) maximization objective 3 Direct Advantage Regression: Aligning LLMs with Online AI Reward with two KL coefficients being strictly positive, there exists a solution to the problem:

$$\pi^*(y \mid x) = \frac{1}{Z(x)}\,\pi_{\mathrm{ref}}(y \mid x)^{\frac{\alpha}{\alpha+\beta}}\;\pi_t(y \mid x)^{\frac{\beta}{\alpha+\beta}}\;\exp\Big(\frac{A(x,y)}{\alpha+\beta}\Big),$$

where

$$Z(x) = \sum_y \pi_{\mathrm{ref}}(y \mid x)^{\frac{\alpha}{\alpha+\beta}}\;\pi_t(y \mid x)^{\frac{\beta}{\alpha+\beta}}\;\exp\Big(\frac{A(x,y)}{\alpha+\beta}\Big),$$

is the partition function.

Proof.

$$
\begin{aligned}
&\max_{\pi} \mathbb{E}_{x,y\sim\pi}[A(x,y)] \;-\; \alpha D_{\mathrm{KL}}[\pi(y|x)\,\|\,\pi_{\mathrm{ref}}(y|x)] \;-\; \beta D_{\mathrm{KL}}[\pi(y|x)\,\|\,\pi_t(y|x)] \\
&= \min_{\pi} \mathbb{E}_{x,y\sim\pi}\Big[\alpha \log \frac{\pi(y|x)}{\pi_{\mathrm{ref}}(y|x)} + \beta \log \frac{\pi(y|x)}{\pi_t(y|x)} - A(x,y)\Big] \\
&= \min_{\pi} \mathbb{E}_{x,y\sim\pi}\Big[(\alpha+\beta)\log \pi(y|x) - \alpha \log \pi_{\mathrm{ref}}(y|x) - \beta \log \pi_t(y|x) - A(x,y)\Big] \\
&= \min_{\pi} \mathbb{E}_{x,y\sim\pi}\Big[\log \pi(y|x) - \log \pi_{\mathrm{ref}}(y|x)^{\frac{\alpha}{\alpha+\beta}} - \log \pi_t(y|x)^{\frac{\beta}{\alpha+\beta}} - \frac{1}{\alpha+\beta}A(x,y)\Big] \\
&= \min_{\pi} \mathbb{E}_{x,y\sim\pi}\Big[\log \frac{\pi(y|x)}{\pi_{\mathrm{ref}}(y|x)^{\frac{\alpha}{\alpha+\beta}}\,\pi_t(y|x)^{\frac{\beta}{\alpha+\beta}}} - \frac{1}{\alpha+\beta}A(x,y)\Big] \\
&= \min_{\pi} \mathbb{E}_{x,y\sim\pi}\Big[\log \frac{\pi(y|x)}{\frac{1}{Z(x)}\pi_{\mathrm{ref}}(y|x)^{\frac{\alpha}{\alpha+\beta}}\,\pi_t(y|x)^{\frac{\beta}{\alpha+\beta}}\,\exp(\frac{1}{\alpha+\beta}A(x,y))} - \log Z(x)\Big].
\end{aligned}
$$

As the partition function is not dependent on π, $\log Z(x)$ is a constant in our optimization objective. We can remove it and obtain:

$$
\begin{aligned}
&\min_{\pi} \mathbb{E}_{x,y\sim\pi}\Big[\log \frac{\pi(y|x)}{\frac{1}{Z(x)}\pi_{\mathrm{ref}}(y|x)^{\frac{\alpha}{\alpha+\beta}}\,\pi_t(y|x)^{\frac{\beta}{\alpha+\beta}}\,\exp(\frac{1}{\alpha+\beta}A(x,y))}\Big] \\
&= \min_{\pi} \mathbb{E}_{x}\; D_{\mathrm{KL}}\Bigg[\pi(y|x)\;\Bigg\|\;\frac{1}{Z(x)}\pi_{\mathrm{ref}}(y|x)^{\frac{\alpha}{\alpha+\beta}}\,\pi_t(y|x)^{\frac{\beta}{\alpha+\beta}}\,\exp(\frac{1}{\alpha+\beta}A(x,y))\Bigg].
\end{aligned}
$$

Based on Gibbs' inequality, Equation (11) is minimized when the two distributions are identical. We have:

$$
\pi^*(y|x) \;=\; \frac{1}{Z(x)}\,\pi_{\mathrm{ref}}(y|x)^{\frac{\alpha}{\alpha+\beta}}\,\pi_t(y|x)^{\frac{\beta}{\alpha+\beta}}\,\exp\Big(\frac{1}{\alpha+\beta}A(x,y)\Big). \tag{4}
$$

That ends the proof.

We can now obtain an improved policy by minimizing the KL-divergence between itself and the optimal policy defined above.

$$
\min_{\pi_\theta} \mathbb{E}_{x\sim d_{\pi_t}(x)} D_{\mathrm{KL}}[\pi^*(\cdot \mid s)\;\|\;\pi_\theta(\cdot \mid s)],
$$

and substitute to get:

$$
\min_{\pi_\theta} \mathbb{E}_{x\sim d_{\pi_t}(x)} D_{\mathrm{KL}}\Bigg[\frac{1}{Z(x)}\pi_{\mathrm{ref}}(y \mid x)^{\frac{\alpha}{\alpha+\beta}}\,\pi_t(y \mid x)^{\frac{\beta}{\alpha+\beta}}\,\exp\Big(\frac{1}{\alpha+\beta}A(x,y)\Big)\;\Bigg\|\;\pi_\theta(\cdot \mid x)\Bigg],
$$

after expanding the KL-divergence term, we can further reduce the objective by dropping out the terms not dependent on π_θ:

$$
\min_{\pi_\theta} \mathbb{E}_{x\sim d_{\pi_t}(x)}\Bigg[-\sum_{y} \frac{1}{Z(x)}\,\pi_{\mathrm{ref}}(y \mid x)^{\frac{\alpha}{\alpha+\beta}}\,\pi_t(y \mid x)^{\frac{\beta}{\alpha+\beta}}\,\exp\Big(\frac{1}{\alpha+\beta}A(x,y)\Big)\,\log \pi_\theta(y \mid x)\Bigg],
$$

we can factor out the partition function term as it is a positive constant not shifting the optimal policy:

$$\min_{\pi_\theta} \mathbb{E}_{x\sim d_{\pi_t}(x)}\Big[-\sum_y \pi_{\text{ref}}(y \mid x)^{\frac{\alpha}{\alpha+\beta}} \pi_t(y \mid x)^{\frac{\beta}{\alpha+\beta}} \exp\Big(\frac{1}{\alpha+\beta}A(x,y)\Big) \log \pi_\theta(y \mid x)\Big],$$

we obtain our final optimization objective by taking π_t as our sampling policy:

$$\max_{\pi_\theta} \mathbb{E}_{x\sim d_{\pi_t}(x)}\mathbb{E}_{y\sim\pi_t(y|x)}\left[\left(\frac{\pi_{\text{ref}}(y \mid x)}{\pi_t(y \mid x)}\right)^{\frac{\alpha}{\alpha+\beta}} \exp\Big(\frac{1}{\alpha+\beta}A(x,y)\Big) \log \pi_\theta(y \mid x)\right].$$

Mapping DAR to V-MPO

The closed-form optimal policy derived in Direct Advantage Regression (DAR) reduces to the V-MPO target under simple limits and special-case identifications. The derivation clarifies when DAR may be viewed as a sequence-level variant of the V-MPO/AWR family.

DAR closed-form target. DAR yields a closed-form optimal policy of the form

$$\pi^*(y|x) \;=\; \frac{1}{Z(x)}\; \pi_{\text{ref}}(y|x)^{\frac{\alpha}{\alpha+\beta}}\; \pi_t(y|x)^{\frac{\beta}{\alpha+\beta}}\; \exp\Big(\frac{A(x,y)}{\alpha+\beta}\Big), \tag{5}$$

where $(\alpha, \beta) > 0$ are the two KL coefficients (reference and sampling policy), π_{ref} a chosen reference policy, π_t the sampling policy used to collect $\mathcal{D}_{\pi_t}$, and $A(x, y)$ the (sequence-level) advantage or score. Define the shorthand

$$t \coloneqq \alpha + \beta > 0.$$

Taking logarithms of (5) produces the additive form used below:

$$\log \pi^*(y|x) = -\log Z(x) + \frac{\alpha}{t}\log \pi_{\text{ref}}(y|x) + \frac{\beta}{t}\log \pi_t(y|x) + \frac{1}{t}A(x,y). \tag{6}$$

V-MPO canonical target. V-MPO (and related advantage-weighted regression methods) uses a target proportional to the sampling policy multiplied by an exponential advantage factor. In sequence notation:

$$\psi_{\text{V–MPO}}(y|x) \propto \pi_t(y|x)\; \exp\Big(\frac{A(x,y)}{\eta}\Big), \tag{7}$$

where $\eta > 0$ is the temperature (V-MPO treats η as a dual variable and may optimise it by solving a convex dual).

Special case 1: identical reference and sampling policies. Set $\pi_{\text{ref}} = \pi_t$ in (5). Then the multiplicative powers collapse:

$$\pi_{\text{ref}}^{\alpha/t}\pi_t^{\beta/t} = \pi_t^{(\alpha+\beta)/t} = \pi_t.$$

Substituting into (5) gives

$$\pi^*(y\,|\,x) \propto \pi_t(y\,|\,x)\,\exp\Big(\frac{A(x,y)}{t}\Big).$$

Comparing with (7) we identify the V-MPO temperature as

$$\eta = t = \alpha + \beta.$$

Thus DAR reduces to the V-MPO structural form when the reference policy equals the sampling policy; the DAR temperature is the sum of the two KL coefficients.

Special case 2: $\alpha \to 0$ (reference KL vanishes). Examine the limit $\alpha \to 0$ while holding $t = \alpha + \beta$ finite (equivalently, let $\beta \to t$). From (6),

$$\lim_{\alpha\to 0} \log \pi^*(y\,|\,x) = -\log Z(x) + \underbrace{\frac{\alpha}{t}}_{\to 0}\log \pi_{\text{ref}} + \underbrace{\frac{\beta}{t}}_{\to 1}\log \pi_t + \frac{1}{t}A(x,y),$$

so

$$\pi^*(y\,|\,x) \propto \pi_t(y\,|\,x)\,\exp\Big(\frac{A(x,y)}{t}\Big),$$

again matching (7) with $\eta = t$. The algebraic limit is well-defined because the coefficients α/t and β/t converge to $(0, 1)$ respectively.

Algebraic equivalence (log-space explanation). Write the DAR unnormalised log-weight for a sample (x, y) as

$$\ell_{\text{DAR}}(x,y) \;=\; \frac{\alpha}{t}\log \pi_{\text{ref}}(y\,|\,x) + \frac{\beta}{t}\log \pi_t(y\,|\,x) + \frac{1}{t}A(x,y).$$

If either $\pi_{\text{ref}} = \pi_t$ or $\alpha/t \to 0$, the first two terms reduce to $\log \pi_t(y\,|\,x)$. Hence

$$\ell_{\text{DAR}}(x,y) \to \log \pi_t(y\,|\,x) + \frac{1}{t}A(x,y),$$

and exponentiation recovers the V-MPO unnormalised weight $\exp(\log \pi_t + A/t) = \pi_t \exp(A/t)$.

Practical consequences and distinctions. Although DAR and V-MPO share the same advantage-exponential device, they differ in:

1. **Dual handling.** V-MPO treats η (and an average-KL dual α) as optimisation variables updated by convex-dual objectives; DAR typically treats α, β as explicit regularisation coefficients (or tunes them as hyperparameters), and derives a closed-form mixture. Consequently, V-MPO emphasises adaptive temperature control, while DAR emphasises calibrated mixture regularisation for LLM-scale training.

2. **Factorisation and tractability.** V-MPO is usually applied to low-dimensional action spaces (or per-step discrete actions); LLM-scale training uses sequence-level weights on teacher-forced token log-probabilities (the DAR practical objective).

3. **Engineered stabilisers.** DAR introduces practical stabilisers for large-vocabulary sequence training (weight clipping, advantage normalisation, importance-weight corrections). V-MPO implementations use top-k selection and explicit dual updates which require careful numerical handling when lifted to LLMs.

Sequence-level weighted loss (common form). Both approaches lead to a weighted supervised fine-tuning loss over sampled sequences:

$$L(\theta) = -\sum_{i \in S} w_i \sum_{t=1}^{T_i} \log \pi_\theta(y_t^{(i)} \mid y_{<t}^{(i)}, x^{(i)}),$$

where the per-sequence weights differ only by the base factor:

$$w_i^{\mathrm{DAR}} \propto \exp\Big(\frac{\alpha}{t} \log \pi_{\mathrm{ref}}^{(i)} + \frac{\beta}{t} \log \pi_t^{(i)} + \frac{1}{t} A^{(i)}\Big), \qquad w_i^{\text{V–MPO}} \propto \exp\Big(\log \pi_t^{(i)} + \frac{1}{\eta} A^{(i)}\Big).$$

Numerical sanity check (illustrative). A short numerical test can confirm convergence of DAR weights to V-MPO weights as $\alpha \to 0$ or $\pi_{\mathrm{ref}} = \pi_t$; include a compact reference implementation in experimental code.

8 Transformers - TrXL

Authors of V-MPO (Song et al. 2019) showed empirical results on transformers in atari games using TrXL (Dai et al. 2019), architecture which was further enhanced for RL through GTrXL (Parisotto et al. 2019).

Implementation

- github.com/kimiyoung/transformer-xl
- github.com/nenuadrian/DI-engine/tree/main/benchmarks

GTrXL with V-MPO

The V-MPO paper (Song et al. 2019) replaces the LSTM core with a Transformer-XL (TrXL) for single-task Atari, motivated by the argument that in a fully observable environment recurrent architectures enable the agent to utilise more useful representations than are available in the immediate observation. However, standard TrXL fails in RL, performing at random-policy level on benchmarks such as DMLab-30. The Gated Transformer-XL (GTrXL) (Parisotto et al. 2019) addresses this with two targeted modifications.

Identity Map Reordering (TrXL-I)

Layer normalisation is moved to the *input* of each sub-layer rather than the output, creating a direct identity path from the first layer's input to the last layer's output. At initialisation this biases the network towards a Markovian (reactive) policy and provides a clear gradient path, making the training landscape far more amenable to policy-gradient methods.

Gating Layers

Residual connections are replaced with learnable gating layers. The best-performing variant uses GRU-type gates:

$$\begin{aligned} r &= \sigma(W_r y + U_r x), \\ z &= \sigma(W_z y + U_z x - b_g), \\ \hat{h} &= \tanh(W_g y + U_g(r \odot x)), \\ g(x, y) &= (1 - z) \odot x + z \odot \hat{h}, \end{aligned}$$

where x is the residual input and y is the sub-layer output. Initialising the bias $b_g = 2$ places each gate near the identity at the start of training, preserving the reactive-policy initialisation provided by the layer-norm reordering.

Why GTrXL Suits V-MPO

V-MPO is an on-policy algorithm that performs policy improvement via an EM procedure (E-step top-k advantage weighting; M-step weighted maximum-likelihood with a KL trust-region). Because it collects fresh trajectories each update, the memory architecture is critical: the agent must integrate information over long horizons to form useful state representations. GTrXL provides:

- **Long-range memory.** Relative position encodings inherited from TrXL let the network attend over a memory tensor spanning up to thousands of past time-steps, far beyond what an LSTM can retain.
- **Stable optimisation.** The GRU gating achieves a 0% divergence rate across hyperparameter sweeps, compared with 16% for the plain TrXL-I variant, making it compatible with V-MPO's fixed Adam learning rate and without requiring population-based hyperparameter search.

- **Robust performance on memory tasks.** On DMLab-30, GTrXL (GRU) reaches a human-normalised score of 117.6 ± 0.3 versus 99.3 ± 1.0 for LSTM, with the largest gains on memory-dependent levels and no regression on reactive ones.

Architecture and Training Configuration

For single-task Atari with TrXL, the shared policy–value network consists of a convolutional ResNet backbone feeding into the transformer core, with previous reward and action as additional inputs. Representative TrXL hyperparameters used in V-MPO experiments: embedding size 256, 8 layers, 4 attention heads, key/value size 32, MLP size 512, unroll length 63, batch size 128. The value and policy heads branch from the transformer output; parameters are updated jointly under the combined V-MPO objective $\mathcal{L}(\phi, \theta, \eta, \alpha)$ using Adam at a fixed learning rate of 10^{-4}.

9 The Case for Dropout

Dropout in LLMs

Dropout (Srivastava et al. 2014) randomly zeros each hidden activation with probability p during training and scales the remaining activations by $1/(1-p)$:

$$\tilde{h}_j = \frac{m_j}{1-p}\, h_j, \qquad m_j \sim \text{Bernoulli}(1-p).$$

In a Transformer this is typically applied in two places: after the attention weights (attention dropout) and after each feed-forward sub-layer (residual dropout). If $\mathbf{h}^{(l)}$ is the hidden state at layer l, the residual-dropout forward pass through one sub-layer $f^{(l)}$ is

$$\mathbf{h}^{(l)} = \mathbf{h}^{(l-1)} + \text{Dropout}(f^{(l)}(\mathbf{h}^{(l-1)})).$$

During SFT, dropout serves its classical purpose: it regularises the model and reduces overfitting to the demonstration distribution. Most large-scale LLM pre-training runs (GPT-3, LLaMA, etc.) disable dropout entirely, relying instead on the sheer volume of data for regularisation. However, during fine-tuning the dataset is typically orders of magnitude smaller, and overfitting is a real concern, making dropout relevant again.

Why PPO Disables Dropout

Standard on-policy PPO collects a batch of trajectories under the current policy $\pi_{\theta_{\text{old}}}$ (in eval mode, no dropout), then performs several epochs of gradient updates on that batch. The clipped objective relies on the importance-sampling ratio

$$r_t(\theta) = \frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)}.$$

Problem 1: stochastic numerator. If dropout is active during the optimisation epochs, $\pi_\theta(a_t \mid s_t)$ becomes a random variable that changes on every forward pass even for fixed θ. The ratio $r_t(\theta)$ therefore fluctuates stochastically, injecting noise directly into the surrogate objective and its gradients. Because PPO's clip window $[1-\epsilon,\, 1+\epsilon]$ is deliberately narrow (typically $\epsilon = 0.2$), even moderate stochastic perturbation can push the ratio outside the clip region or mask genuine policy changes, degrading the signal.

Problem 2: inconsistent denominator. The denominator $\pi_{\theta_{\text{old}}}(a_t \mid s_t)$ is computed once at rollout time in eval mode (no dropout). If the training forward pass uses a different dropout mask, the numerator and denominator are computed under *different* effective networks. The ratio no longer measures how far the policy has moved; it conflates policy change with mask change:

$$r_t(\theta) = \frac{\pi_\theta^{(\text{mask}_1)}(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}^{(\text{no mask})}(a_t \mid s_t)}.$$

This breaks the theoretical guarantee that clipping r_t constrains the KL divergence between consecutive policies.

Problem 3: KL penalty corruption. Many PPO implementations add an auxiliary KL penalty $\widehat{\text{KL}} = \mathbb{E}_t[\ell_{\text{old},t} - \ell_{\theta,t}]$ or use it for early stopping. With dropout active, the estimated KL is biased upward (the mask-induced variance appears as divergence), causing premature termination of updates or an overly conservative step.

For these reasons, all mainstream PPO implementations (including those used for RLHF in InstructGPT, LLaMA, etc.) run the policy in eval mode during both rollout and optimisation, forgoing any regularisation benefit that dropout could provide.

Why V-MPO / EM-Style Methods Are Dropout-Compatible

The V-MPO update decomposes into an E-step that computes per-sample weights, followed by an M-step that is *pure weighted supervised learning*. Neither step requires a probability ratio between two forward passes under different modes.

E-step: weights from advantages, not ratios. The non-parametric target assigns weight

$$w_i \propto \exp\Big(\frac{A^{(i)}}{\eta}\Big),$$

where $A^{(i)} = G^{(i)} - V_\phi(s^{(i)})$ depends on the value function and the observed return, not on a ratio of policy probabilities. The value function V_ϕ is a regression target; dropout can be disabled for the single forward pass that computes advantages at rollout time (just as PPO evaluates its value head in eval mode), or it can remain active since the advantage computation is a one-shot evaluation, not an iterative ratio.

M-step: standard cross-entropy. The M-step loss is

$$L_\pi(\theta) = -\sum_{i \in S} w_i \sum_{t=1}^{T_i} \log \pi_\theta(y_t^{(i)} \mid y_{<t}^{(i)}, x^{(i)}),$$

which is structurally identical to SFT with per-sequence weights. Dropout is fully compatible here for the same reason it is compatible with any supervised cross-entropy objective: the loss is evaluated under a single stochastic forward pass, and the gradient is an unbiased estimator of the expected loss under the dropout distribution.

No ratio, no conflict. The key structural difference is summarised in the following table:

	PPO	**V-MPO / EM**
Gradient signal	$\nabla_\theta\, r_t(\theta)\, A_t$	$\nabla_\theta\, w_i \log \pi_\theta$
Requires ratio $\pi_\theta/\pi_{\theta_{\text{old}}}$?	Yes	No
Weights depend on current θ?	Yes (through r_t)	No (fixed from E-step)
Dropout in training pass	Corrupts ratio	Standard regularisation

Because the E-step weights w_i are *detached* from the current parameters, the M-step gradient with dropout active is

$$\nabla_\theta L_\pi = -\sum_{i \in S} w_i \sum_t \nabla_\theta \log \pi_\theta^{(\text{mask})}(y_t^{(i)} \mid \cdot),$$

which is an unbiased estimate of the true weighted-MLE gradient under the dropout distribution, exactly as in supervised learning. No eval-mode forward pass needs to be compared against this quantity.

Practical Implications

In the LLM fine-tuning regime where data is limited and overfitting is a practical concern, V-MPO's EM structure therefore offers a regularisation advantage over PPO: dropout (and related stochastic regularisers such as DropPath or stochastic depth) can be enabled during the M-step without any algorithmic modification, providing the same generalisation benefits observed in supervised fine-tuning. PPO, by contrast, must rely on non-architectural regularisation (weight decay, gradient clipping, early stopping) because its core mechanism is incompatible with stochastic forward passes.

10 References

Dai, Zihang et al. (2019). *Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context*. arXiv: 1901.02860 [cs.LG]. URL: https://arxiv.org/abs/1901.02860.

He, Li et al. (2025). *Direct Advantage Regression: Aligning LLMs with Online AI Reward*. arXiv: 2504.14177 [cs.AI]. URL: https://arxiv.org/abs/2504.14177.

Parisotto, Emilio et al. (2019). *Stabilizing Transformers for Reinforcement Learning*. arXiv: 1910.06764 [cs.LG]. URL: https://arxiv.org/abs/1910.06764.

Song, H. Francis et al. (2019). *V-MPO: On-Policy Maximum a Posteriori Policy Optimization for Discrete and Continuous Control*. arXiv: 1909.12238 [cs.AI]. URL: https://arxiv.org/abs/1909.12238.

Srivastava, Nitish et al. (2014). "Dropout: A Simple Way to Prevent Neural Networks from Overfitting". In: *Journal of Machine Learning Research* 15.56, pp. 1929–1958. URL: http://jmlr.org/papers/v15/srivastava14a.html.