BP-LLM: Belief Propagation for Binary Feedback in Large Language Model Alignment

Jessica E. Liang
Department of Computer and Information Science
University of Pennsylvania
Philadelphia, PA 19104 USA
jeliang@seas.upenn.edu

Abstract

Preference-based alignment methods such as Direct Preference Optimization (DPO) and Binary Classifier Optimization (BCO) offer efficient alternatives to reinforcement learning from human feedback (RLHF). However, they often treat comparisons as independent labels and do not explicitly model uncertainty, which can lead to miscalibration and reduced robustness under noisy or heterogeneous feedback. We introduce Belief Propagation for Large Language Model Alignment (BP–LLM), a probabilistic framework that views binary feedback as noisy observations of latent reward margins under a policy-induced Gaussian prior. Using the Jaakkola–Jordan variational bound, BP–LLM yields closed-form Gaussian message updates and performs stable belief propagation between a classifier-side inference module and the policy. Exchanging *extrinsic* messages enables both modules to refine beliefs without double counting and recovers BCO and DPO as special cases. We evaluate BP–LLM in two regimes. In an inference-only setting with frozen LLM weights, BP–LLM consistently improves label-free test-time win rate over BCO and DPO across UltraFeedback, Capybara, and HelpSteer2 for open-weight Llama and Qwen models. In a training-time setting with parameter-efficient LoRA updates, BP–LLM also outperforms Cal-DPO with higher win rates. Overall, BP–LLM is most beneficial under noisy/heterogeneous (or unary) feedback, where posterior refinement and extrinsic shaping provide more reliable signals than hard labels, while remaining lightweight and scalable.[1]

[1]The code is available at https://github.com/jelcis26/BP-LLM

1 Introduction

Large language models (LLMs) have demonstrated remarkable generative capabilities, yet aligning them with nuanced human preferences remains a fundamental challenge. While supervised fine-tuning and reinforcement learning from human feedback (RLHF) have become standard alignment strategies, their practical deployment is constrained by label scarcity, instability during optimization, and the lack of calibrated reward models. Recent binary-feedback approaches such as Direct Preference Optimization (DPO) (Rafailov et al., 2023) and Binary Classifier Optimization (BCO) (Jung et al., 2025) simplify alignment by replacing reinforcement learning with logistic or cross-entropy objectives. Despite their efficiency, these methods treat each comparison as an independent observation and do not explicitly model the uncertainty or latent structure of human feedback. As a result, policies trained under such frameworks can suffer from miscalibration and limited robustness to noisy or inconsistent annotations.

This work addresses these limitations by introducing a probabilistic inference perspective on binary-feedback alignment. We propose Belief Propagation for Language Model Alignment (BP–LLM), a framework that treats each binary label as a noisy observation of a latent *reward margin* and couples it with a policy-induced Gaussian prior. This construction transforms the alignment process into a form of *message passing* between the classifier and the policy: the classifier infers soft evidence about latent rewards, and the policy updates its logits based on the extrinsic information received. Crucially, BP–LLM avoids double counting by explicitly separating prior and extrinsic contributions, ensuring that updates are consistent across iterations.

The core of BP-LLM is a closed-form Gaussian

Transactions of the Association for Computational Linguistics, vol. 14, pp. 1413–1430, 2026. https://doi.org/10.1162/TACL.a.734
Action Editor: Pradeep Dasigi. Submission batch: 10/2025; Revision batch: 2/2026; Published 7/2026.

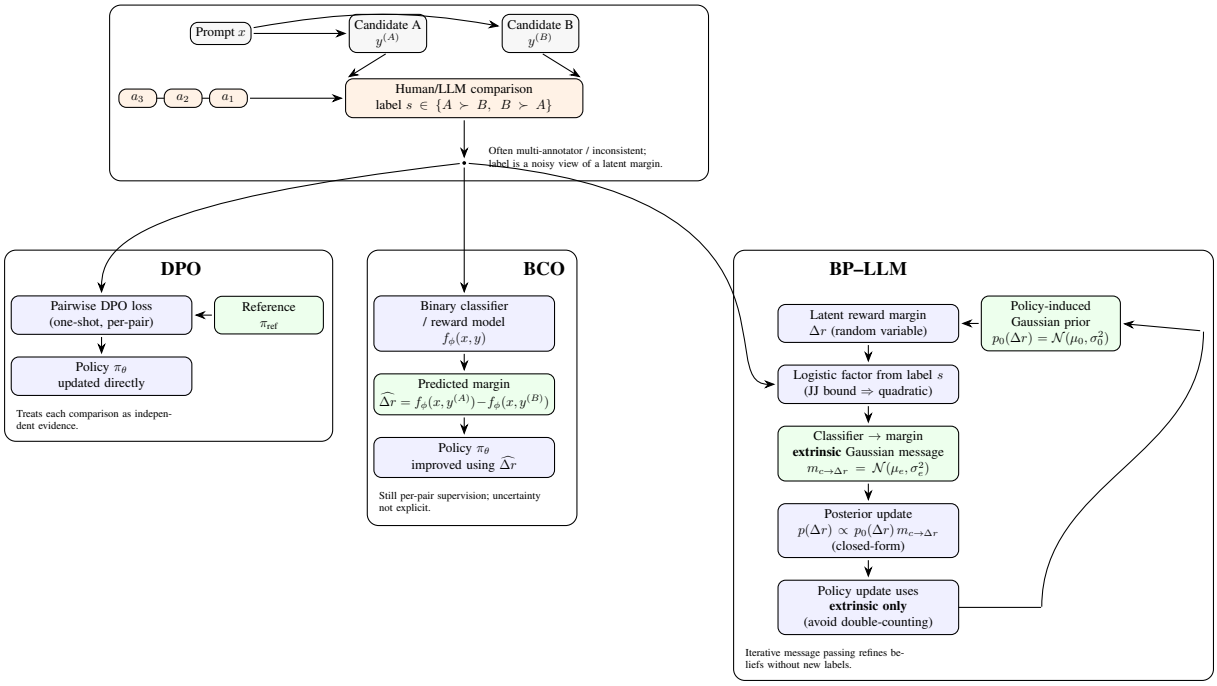


Figure 1: Binary preference feedback and how it is processed by DPO, BCO, and BP–LLM.

message update derived using the Jaakkola–Jordan (JJ) variational bound (Jaakkola and Jordan, 2000), which transforms the logistic likelihood into a tractable quadratic surrogate. This yields an analytically solvable posterior update and enables stable belief propagation even with non-conjugate likelihoods. By iterating between classifier-side inference and policy-side updates, BP-LLM effectively refines the reward posterior without requiring additional labels or gradient-through-sampling steps. The result is a flexible, calibration-free mechanism that generalizes both BCO and DPO as special cases within a unified probabilistic framework. Figure 1 provides an intuitive "end-to-end" view of the preference-alignment supervision signal (human or LLM-judge) and how DPO, BCO, and BP–LLM leverage it.

Empirically, we evaluate BP–LLM across three public preference corpora: UltraFeedback (Cui et al., 2023), Capybara (Daniele and Suphavadeeprasit, 2023), and HelpSteer2 (Wang et al., 2024), using open-weight models from the Llama 3 (Dubey et al., 2024) and Qwen 2.5 (Qwen Team, 2024) families. We study BP–LLM in two complementary regimes. First, in an inference-only setting with frozen LLM weights, BP–LLM consistently improves label-free test-time win rate over both BCO and DPO across datasets and model scales, with the largest gains on smaller models and noisier datasets, highlighting robustness to limited capacity and heterogeneous supervision. Second, in a training-time setting with parameter-efficient LoRA updates to the policy, BP–LLM also outperforms Cal-DPO (Xiao et al., 2024) across all model–dataset configurations with higher win rates. BP–LLM is a lightweight, interpretable probabilistic framework for preference alignment that unifies classifier and policy updates while improving robustness and calibration via iterative message passing.

2 Related Work

Alignment from human and AI feedback. Reinforcement Learning from Human Feedback (RLHF) trains a reward model from pairwise preferences and then optimizes a policy against that reward with Kullback–Leibler (KL) control to a reference (Ziegler et al., 2019; Ouyang et al., 2022). Anthropic's helpful/harmless pipeline and Constitutional AI reduced the reliance on direct human labeling by leveraging Reinforcement Learning from AI Feedback (RLAIF) (Bai et al., 2022a,b). Subsequent analyses documented challenges such as reward overoptimization and miscalibration (Gao et al., 2023; Stiennon et al., 2020), motivating objectives that avoid explicit RL updates while keeping a KL anchor.

Preference-only objectives (no explicit RL). DPO optimizes a Bradley-Terry likelihood using policy-reference log-ratios on pairwise data (Rafailov et al., 2023). Many variants improve regularization, stability, and data efficiency: Iterative Preference Optimization (IPO) (Yang et al., 2025), ORPO (single-model optimization without an explicit reference) (Hong et al., 2024), SimPO (reference-free scaling) (Meng et al., 2024), and PRO (ranking-oriented) (Song et al., 2024). Listwise/ranking approaches like RRHF exploit multiple candidates per prompt without training a reward model (Yuan et al., 2023). BCO moves to *unary* supervision and shows that minimizing BCE upper-bounds DPO up to a reward shift (Jung et al., 2025), connecting binary-feedback training to preference optimization.

Robust, calibrated, and listwise preference learning. Robust preference training addresses noisy labels, annotator heterogeneity, and distribution shift (Liang et al., 2024; Fisch et al., 2024). Calibrated or temperature-aware objectives target more reliable likelihood ratios (Xiao et al., 2024; Guo et al., 2017). Listwise and Plackett–Luce/Bradley–Terry extensions capture partial orders and multi-candidate settings (Plackett, 1975; Bradley and Terry, 1952). Our BP–LLM contributes an *inference layer* that denoises unary signals and yields per-instance posterior rewards prior to policy shaping.

Datasets for preference alignment. We evaluate on three public corpora: UltraFeedback (Cui et al., 2023), Capybara (Daniele and Suphavadeeprasit, 2023), and HelpSteer2 (Wang et al., 2024). Other widely used resources include HH-RLHF (Bai et al., 2022a), SHP/SHP-2 (Poddar et al., 2024; Gao et al., 2020), and OASST-1 (Köpf et al., 2023), which span helpfulness/harmlessness, topical breadth, and diverse annotator populations. These benchmarks collectively stress generalization, calibration, and robustness under varying feedback quality.

Belief propagation and KL-regularized control. BP performs marginal inference by exchanging messages on factor graphs; on trees it is exact, while on loopy graphs it approximates stationary points of a Bethe free energy (Pearl, 1988; Yedidia et al., 2000, 2005). Turbo decoding popularized *extrinsic* log-likelihood ratios to avoid double counting across modules (Berrou et al., 1993). For logistic links, the Jaakkola–Jordan quadratic bound provides Gaussian-form updates with closed-form message passing (Jaakkola and Jordan, 2000). KL-regularized control (REPS, MPO) yields exponential tilting of a reference policy and directly motivates the log-ratio parameterization used in preference objectives (Peters et al., 2010; Abdolmaleki et al., 2018). BP-LLM bridges these lines: a Gaussian prior whose mean is a temperature-scaled policy–reference log-ratio combines with a logistic observation via JJ surrogates; exchanging only *extrinsic* information stabilizes loopy schedules and mitigates double counting.

Position within alignment methods. Compared to RLHF/RLAIF (Ouyang et al., 2022; Bai et al., 2022b), BP-LLM keeps simple supervised updates while improving robustness via posterior (soft) rewards under unary feedback. Relative to DPO/BCO (Rafailov et al., 2023; Jung et al., 2025), BP-LLM adds a principled inference layer that calibrates per-instance signals and accommodates annotator variability, connecting alignment practice to well-studied BP and free-energy tools (Yedidia et al., 2005).

3 Belief Propagation for Binary Feedback in LLM Alignment

3.1 Problem Formulation

Setting and challenge. We study *binary/unary feedback alignment* where training data provides only a single noisy preference signal per response (e.g., a binary label indicating accept/reject), rather than paired comparisons. Existing approaches capture complementary pieces of this setting: BCO (Jung et al., 2025) provides a probabilistic view of learning from unary feedback, while DPO-style objectives (Rafailov et al., 2023) provide stable, reference-anchored policy shaping. However, these perspectives are typically developed in isolation, and naively combining them risks *inconsistent evidence use*: the same label information may be implicitly counted multiple times through both the classifier and the policy update, or lost when alternating between stages.

Our probabilistic formulation. We develop a BP formulation for binary-feedback alignment that unifies the BCO approach (Jung et al., 2025) and DPO-style shaping (Rafailov et al., 2023) within a single probabilistic framework. The cen-

tral idea is to treat each unary label as a noisy observation of an unobserved *reward margin*, and to let the policy induce a Gaussian prior over that margin. This view allows the classifier and policy to exchange information through consistent, closed-form BP message updates, ensuring that evidence is neither double-counted nor lost as parameters evolve. The overall procedure can be interpreted as an alternating variational inference and policy update loop—analogous to loopy BP or expectation–propagation fixed-point iterations.

Binary-feedback model. Consider a dataset $\mathcal{D} = \{(x_i, y_i, b_i)\}_{i=1}^N$ consisting of prompts x_i, completions y_i, and binary feedback $b_i \in \{0, 1\}$. BCO trains a classifier $g_\psi(x, y)$ using binary cross-entropy, where the classifier logit $r_\psi(x, y)$ acts as an implicit reward. It has been shown that this implicit reward upper-bounds the DPO objective up to a constant shift (Jung et al., 2025; Rafailov et al., 2023). To represent this in a probabilistic graphical model, we introduce a continuous latent variable $R_i \in \mathbb{R}$ for each pair (x_i, y_i), which captures the underlying reward margin. Each observed label b_i is modeled as a logistic (Bernoulli) observation:

$$p(b_i = 1 \mid R_i) = \sigma(\gamma R_i), \tag{1}$$

$$\sigma(z) = \frac{1}{1 + e^{-z}}, \tag{2}$$

where $\gamma > 0$ controls the sharpness of the binary link. Intuitively, R_i plays the role of a latent "reward logit," with positive values indicating preference for $b_i = 1$.

Policy-induced Gaussian prior. To connect the latent reward with the policy, we place a Gaussian prior over R_i whose mean encodes the policy's relative log-probability against a fixed reference:

$$R_i \sim \mathcal{N}\big(\mu_i(\theta), \tau^2\big), \tag{3}$$

$$\mu_i(\theta) = \beta \log \frac{\pi_\theta(y_i \mid x_i)}{\pi_{\text{ref}}(y_i \mid x_i)} - \delta. \tag{4}$$

Here, $\beta > 0$ is a temperature scaling factor, δ is a global reward shift, and τ^2 is a variance hyperparameter controlling the prior strength. In practice, $\mu_i(\theta)$ can be computed from token log-likelihoods (e.g., mean- or sum-normalized) and the reference π_{ref} provides regularization and score calibration.

Connection to KL-regularized policy optimization. Setting the prior mean to the KL–anchored log-probability ratio follows from the family of KL-constrained or regularized policy optimization methods. In Relative Entropy Policy Search (REPS) (Peters et al., 2010) and Maximum a Posteriori Policy Optimization (MPO) (Abdolmaleki et al., 2018), the optimal policy is expressed as an exponential tilt of a frozen reference distribution:

$$\pi_{\text{new}}(a \mid s) \;\propto\; \pi_{\text{ref}}(a \mid s)\, \exp\Big(\tfrac{1}{\eta} A(s, a)\Big), \tag{5}$$

where $\eta > 0$ is a temperature (Lagrange multiplier) and $A(s, a)$ denotes the advantage or reward signal. Taking logs gives the canonical "policy logit" $\log \pi_{\text{new}} - \log \pi_{\text{ref}} = \frac{1}{\eta} A(s, a)$, which naturally motivates the prior mean in (3):

$$\mu_i(\theta) \;=\; \beta[\log \pi_\theta(y_i \,|\, x_i) - \log \pi_{\text{ref}}(y_i \,|\, x_i)] - \delta. \tag{6}$$

In preference optimization, DPO (Rafailov et al., 2023) explicitly maximizes this same log-ratio between the learned policy and the reference. Our unary $\mu_i(\theta)$ can therefore be viewed as the direct probabilistic analogue of the DPO objective at the instance level.

Posterior factorization and double loop. If $\mu_i(\theta)$ were fixed, the posterior over all latent reward margins $R = (R_1, \ldots, R_N)$ factorizes:

$$p(R \mid b, \theta) \;\propto\; \prod_{i=1}^{N} p(b_i \mid R_i)\, p(R_i \mid \theta), \tag{7}$$

yielding independent updates for each data point. However, since $\mu_i(\theta)$ depends on the shared parameters θ, the posterior factors are implicitly coupled through the policy. To manage this dependency, we employ a *double-loop* scheme:

(i) **Inference step:** for fixed θ, infer the approximate posterior $q(R_i)$ using BP message updates.

(ii) **Policy update:** holding q fixed, update θ using soft signals (e.g., extrinsic LLRs) distilled from q.

This alternation corresponds to the free-energy fixed-point interpretation of loopy BP, where inference and parameter updates converge jointly to a stationary point of the variational free-energy (Yedidia et al., 2000, 2005). The resulting BP–LLM framework thus generalizes DPO and BCO by introducing a probabilistic latent-variable layer that supports both classifier-based inference and policy-side shaping within a single, analytically tractable system.

3.2 Quadratic Surrogate and Gaussian Messages

The logistic likelihood $\sigma(\gamma R_i)$ and the Gaussian prior $p(R_i \mid \theta) = \mathcal{N}(\mu_i(\theta), \tau^2)$ are not conjugate, so their combination yields a non-analytic posterior. To obtain tractable closed-form message updates, we apply the Jaakkola–Jordan (JJ) variational bound (Jaakkola and Jordan, 2000), which locally upper-bounds the concave $\log \sigma(\cdot)$ term by a quadratic surrogate in R_i. This converts the logistic factor into a Gaussian-form message compatible with the Gaussian prior.

The JJ quadratic bound. For any $\xi_i > 0$, the JJ bound states that

$$\log \sigma(\gamma R_i) \geq \frac{\gamma}{2} R_i - \lambda(\xi_i) R_i^2 + c(\xi_i), \quad (8)$$

$$\lambda(\xi) = \frac{\tanh(\xi/2)}{4\xi}, \quad (9)$$

where $c(\xi_i)$ is independent of R_i. This bound is obtained by matching the gradient and curvature of the RHS to those of $\log \sigma(z)$ at $z = \xi_i$, producing the tightest quadratic majorizer. Intuitively, $\lambda(\xi_i)$ controls the local curvature: when ξ_i is large (high confidence), $\lambda(\xi_i)$ is small and the surrogate is nearly linear; when ξ_i is small, $\lambda(\xi_i)$ increases, introducing stronger curvature to reflect higher uncertainty. A detailed derivation appears in Appendix A.

Conditioning on the observed label. Given a binary label $b_i \in \{0, 1\}$, we can rewrite the likelihood as

$$p(b_i \mid R_i) = \sigma(\tilde{\gamma}_i R_i), \quad (10)$$

$$\tilde{\gamma}_i = \gamma(2b_i - 1), \quad (11)$$

which simply flips the sign of γ depending on whether $b_i = 0$ or 1. Applying the JJ bound gives

$$\log p(b_i \mid R_i) \geq \frac{\tilde{\gamma}_i}{2} R_i - \lambda(\xi_i) R_i^2 + c(\xi_i), \quad (12)$$

and exponentiating the terms dependent on R_i yields a Gaussian-form likelihood message:

$$m_{\text{lik}}(R_i) \propto \exp\Big(\tfrac{1}{2}\tilde{\gamma}_i R_i - \lambda(\xi_i) R_i^2\Big). \quad (13)$$

This message effectively represents a soft observation that contributes a linear term $\frac{1}{2}\tilde{\gamma}_i R_i$ and a precision increment $2\lambda(\xi_i)$ to the posterior in natural-parameter form.

Combining with the Gaussian prior. The policy provides a Gaussian prior,

$$p(R_i \mid \theta) = \mathcal{N}\big(\mu_i(\theta), \tau^2\big) \quad (14)$$

$$\propto \exp\Big(-\frac{1}{2\tau^2}(R_i - \mu_i(\theta))^2\Big), \quad (15)$$

whose log-density expands as

$$\log p(R_i \mid \theta) = -\frac{1}{2}\tau^{-2} R_i^2 + \tau^{-2}\mu_i(\theta) R_i + \text{const.}$$

Let $q(R_i)$ denote the variational belief over R_i. Using the JJ quadratic bound, we obtain

$$q(R_i) \propto p(R_i \mid \theta) \exp\big(\log \sigma(\tilde{\gamma}_i R_i)\big) \quad (16)$$

$$\approx p(R_i \mid \theta) \exp\Big(\tfrac{1}{2}\tilde{\gamma}_i R_i - \lambda(\xi_i) R_i^2\Big),$$

then

$$\log q(R_i) = -\frac{1}{2\tau^2}(R_i - \mu_i(\theta))^2 + \frac{1}{2}\tilde{\gamma}_i R_i - \lambda(\xi_i) R_i^2 + \text{const} \quad (17)$$

$$= -\frac{1}{2}\Big(\tau^{-2} + 2\lambda(\xi_i)\Big) R_i^2 + \Big(\tau^{-2}\mu_i(\theta) + \tfrac{1}{2}\tilde{\gamma}_i\Big) R_i + \text{const.} \quad (18)$$

Recognizing this as the canonical exponential-family form of a Gaussian, we identify the posterior natural parameters:

$$\Lambda_i = \tau^{-2} + 2\lambda(\xi_i), \quad (19)$$

$$\eta_i = \tau^{-2}\mu_i(\theta) + \tfrac{1}{2}\tilde{\gamma}_i. \quad (20)$$

Hence the posterior over R_i is Gaussian,

$$q(R_i) = \mathcal{N}(\hat{\mu}_i, \hat{\tau}_i^2), \qquad \hat{\mu}_i = \frac{\eta_i}{\Lambda_i}, \quad \hat{\tau}_i^2 = \frac{1}{\Lambda_i}. \quad (21)$$

The algebraic derivation of these updates is provided in Appendix B.

Updating the surrogate parameter. To tighten the bound, ξ_i is set to the expected magnitude of the linear term under the current posterior:

$$\xi_i \leftarrow |\gamma| \sqrt{\mathbb{E}_q[R_i^2]} = |\gamma| \sqrt{\hat{\mu}_i^2 + \hat{\tau}_i^2}. \quad (22)$$

Iterating this update for two or three inner steps produces a locally optimal quadratic approximation, ensuring that the bound remains tight in the region where most posterior mass lies.

Extension to pairwise preferences. If a prompt provides a pairwise preference $(y_w \succ y_l)$, we introduce a coupling factor $\sigma(\gamma(R_w - R_l))$. Applying the same JJ bound yields cross-terms of the form $-2\lambda(\xi_{wl})R_w R_l$ in the joint precision matrix, making the graph weakly loopy. To ensure numerical stability, we apply exponential damping to these loopy messages:

$$m^{(t)} \leftarrow (1 - \lambda_{\text{damp}})m^{(t)} + \lambda_{\text{damp}} m^{(t-1)}, \quad (23)$$

which smooths successive message updates and stabilizes convergence with $\lambda_{\text{damp}} \in [0, 1)$.

Interpretation. The JJ surrogate thus replaces each logistic factor with a locally Gaussian factor whose curvature is governed by $\lambda(\xi_i)$. This allows the binary-feedback model to remain fully differentiable and analytically solvable under Gaussian message passing, enabling efficient, stable updates even in large-scale BP–LLM inference.

3.3 Extrinsic Messages and Alternating Updates

To communicate what the classifier-side inference "learned" about the binary label while avoiding double counting, we summarize the posterior $q(R_i)$ by an *extrinsic* log-likelihood ratio (LLR) message sent to the policy. The term "extrinsic" follows its standard use in BP: each outgoing message excludes the information that originated from the destination node, ensuring that only *new* evidence is passed forward.

Definition of the LLR. We define the LLR as the log-likelihood ratio between the two binary hypotheses $b{=}1$ and $b{=}0$:

$$\text{LLR} = \log \frac{p(b{=}1 \mid \cdot)}{p(b{=}0 \mid \cdot)}. \quad (24)$$

In the logistic observation model, $p(b{=}1 \mid R_i) = \sigma(\gamma R_i)$ and $p(b{=}0 \mid R_i) = 1 - \sigma(\gamma R_i)$, so the LLR equals the log-odds (or logit):

$$\log \frac{\sigma(z)}{1 - \sigma(z)} = z. \quad (25)$$

Taking expectation under the approximate posterior $q(R_i)$ gives

$$\text{LLR}_i^{\text{cls}} = \mathbb{E}_{q(R_i)}\left[\log \frac{\sigma(\gamma R_i)}{1 - \sigma(\gamma R_i)}\right] \quad (26)$$

$$= \gamma\, \mathbb{E}_{q(R_i)}[R_i] \approx \gamma\, \hat{\mu}_i, \quad (27)$$

where $\hat{\mu}_i = \mathbb{E}_q[R_i]$ is the posterior mean. The final approximation uses a first-order expansion since $\sigma(\gamma R_i)$ is locally linear for small posterior variance.

Extrinsic subtraction. To avoid reusing information already encoded in the policy prior, we remove the prior contribution to obtain a purely *extrinsic* classifier message:

$$\begin{aligned} \text{EX}_i^{\text{cls}} &:= \text{LLR}_i^{\text{cls}} - \mu_i(\theta) \\ &= \text{LLR}_i^{\text{cls}} - \Big[\beta \log \tfrac{\pi_\theta(y_i|x_i)}{\pi_{\text{ref}}(y_i|x_i)} - \delta\Big]. \end{aligned} \quad (28)$$

This subtraction ensures that the classifier passes only the *new evidence* inferred from (x_i, y_i, b_i), excluding what the current policy already predicts.

Policy update. The resulting EX_i^{cls} acts as a soft reward signal that can be used in two equivalent styles:

1. *DPO-style shaping:* maximize

$$\sum_i \text{EX}_i^{\text{cls}} \log \frac{\pi_\theta(y_i \mid x_i)}{\pi_{\text{ref}}(y_i \mid x_i)} \quad (29)$$

   with a KL anchor to π_{ref}.

2. *Soft-label BCO:* map EX_i^{cls} to a soft target $\tilde{p}_i = \sigma(\text{EX}_i^{\text{cls}})$ and minimize

$$\begin{aligned} \mathcal{L}_{\text{BCE}} = &- \tilde{p}_i \log \sigma(\mu_i(\theta)) \\ &- (1 - \tilde{p}_i) \log\big(1 - \sigma(\mu_i(\theta))\big). \end{aligned} \quad (30)$$

Policy-to-classifier feedback. After updating the policy, we compute a feedback message to refine the Gaussian prior for the next JJ/BP iteration:

$$\text{EX}_i^{\text{pol}} = \mu_i(\theta) - \mu_i(\theta_{\text{old}}). \quad (31)$$

This feedback expresses the incremental change in the policy's latent reward mean and closes the alternating inference–update loop.

Stopping rule. In BP–LLM, the iterative decision process terminates when the alternating updates between the classifier and policy modules converge. Convergence is detected when the average change in extrinsic log-odds (LLRs) becomes negligible, $r^{(t)} = \overline{|\text{EX}^{\text{cls},(t)} - \text{EX}^{\text{cls},(t-1)}|} + \overline{|\text{EX}^{\text{pol},(t)} - \text{EX}^{\text{pol},(t-1)}|} \le \tau_{\text{ex}}$, or when decisions stabilize, i.e., the signs of the total LLRs remain unchanged for at least a $(1 - \delta)$ fraction of items across two consecutive outer iterations.

If neither criterion is met, the algorithm proceeds until reaching a predefined maximum number of iterations. The overall BP–LLM iterative process, with alternating updates between the classifier and policy modules, is illustrated in Fig. 2.

Damping and fixed-point stability. Like standard BP on loopy graphs, repeated message passing may oscillate. We therefore apply exponential damping:

$$\mathrm{EX} \leftarrow (1-\lambda)\mathrm{EX} + \lambda\,\mathrm{EX}^{\mathrm{old}}, \qquad \lambda \in [0,1), \tag{32}$$

which blends each new extrinsic message with its previous value. Damping smooths updates, accelerates convergence, and empirically stabilizes training.

Putting it together. The full procedure alternates BP-based posterior refinement with policy updates, exchanging extrinsic information in both directions so that the classifier and policy improve each other without reusing the same evidence. Initialization is straightforward: set ξ_i from the prior ($|\gamma|\sqrt{\mu_i(\theta_0)^2+\tau^2}$), begin with $\mathrm{EX}_i^{\mathrm{cls}} = 0$, and run a small, fixed number of inner JJ updates per outer iteration. Hyperparameters have clear roles: γ controls how sharply labels respond to margin changes, β sets the temperature of policy scores, τ^2 governs prior spread, and δ absorbs global shifts (e.g., BCO calibration). The computational cost is modest: all $q(R_i)$ updates are one-dimensional and parallelizable, and the policy step reuses standard log-likelihood gradients. A compact pseudocode summary is provided in Algorithm 1.

3.4 Connections to BCO and DPO

If we collapse the inner BP update to a single pass by ignoring curvature ($\lambda(\xi_i) \to 0$) and set the beliefs to the observed labels ($\tilde{p}_i \equiv b_i$), the objective reduces to binary cross-entropy on the reward logit

$$r_i \;=\; \beta\Big[\log \pi_\theta(y_i \mid x_i) - \log \pi_{\mathrm{ref}}(y_i \mid x_i)\Big] \tag{33}$$

Including the standard BCO reward shift

$$\delta \;=\; \tfrac{1}{2}\Big(\mathbb{E}_{D^+}[r] \;+\; \mathbb{E}_{D^-}[r]\Big), \tag{34}$$

yields the BCO loss

$$\begin{aligned}\mathcal{L}_{\mathrm{BCO}} =& \mathbb{E}_{(x,y^+)}\big[-\log\sigma(r-\delta)\big] \\ &+ \mathbb{E}_{(x,y^-)}\big[-\log\sigma(-(r-\delta))\big],\end{aligned} \tag{35}$$

Algorithm 1 Belief Propagation for Unary Binary Alignment

Input: $\{(x_i, y_i, b_i)\}$, reference π_{ref}, scales (γ, β), variance τ^2, shift δ, damping λ.
Initialize $\theta \leftarrow \theta_0$, $\mathrm{EX}_i^{\mathrm{cls}} \leftarrow 0$, and $\xi_i \leftarrow |\gamma|\sqrt{\mu_i(\theta_0)^2+\tau^2}$.
for $t = 1, \ldots, T$ **do**
  (Classifier/BP) For fixed θ, repeat JJ-tightening K times:
    compute Λ_i, η_i via (19); set $q(R_i) = \mathcal{N}(\hat{\mu}_i, \hat{\tau}_i^2)$;
    update $\xi_i \leftarrow |\gamma|\sqrt{\hat{\mu}_i^2 + \hat{\tau}_i^2}$.
  Form $\mathrm{EX}_i^{\mathrm{cls}}$ via (28) and damp: $\mathrm{EX}_i^{\mathrm{cls}} \leftarrow (1-\lambda)\mathrm{EX}_i^{\mathrm{cls}} + \lambda\,\mathrm{EX}_{i,\mathrm{old}}^{\mathrm{cls}}$.
  (Policy) Update θ using either DPO-style shaping or soft-label BCO with targets $\sigma(\mathrm{EX}_i^{\mathrm{cls}})$.
  Send back $\mathrm{EX}_i^{\mathrm{pol}} = \mu_i(\theta) - \mu_i(\theta_{\mathrm{old}})$ and update the Gaussian prior means for the next iteration.
end for
Output: final policy θ and posterior means $\{\hat{\mu}_i\}$ as soft rewards.

i.e., standard BCO training.

Conversely, if we retain pairwise factors and construct $\tilde{r}_i$ from preference margins, we recover a DPO-style update by optimizing the pairwise logistic likelihood with margin

$$\begin{aligned}m(x; y^+, y^-) =& \beta\big(\log \frac{\pi_\theta(y^+ \mid x)}{\pi_{\mathrm{ref}}(y^+ \mid x)} \\ &- \log \frac{\pi_\theta(y^- \mid x)}{\pi_{\mathrm{ref}}(y^- \mid x)}\big),\end{aligned} \tag{36}$$

and loss

$$\mathcal{L}_{\mathrm{DPO}} = \mathbb{E}_{(x,y^+,y^-)}\big[-\log\sigma\big(m(x; y^+, y^-)\big)\big] \tag{37}$$

Thus, BP-LLM provides a principled bridge: taking the unary limit ($\lambda(\xi_i) \to 0$, $\tilde{p}_i \equiv b_i$) yields BCO, while retaining pairwise factors with margin-based $\tilde{r}_i$ yields DPO; the JJ-bound variant gives a variational surrogate of the latter.

3.5 Computational Complexity: BP–LLM vs. DPO and BCO

We compare the per–mini-batch training cost of BP–LLM against DPO and BCO. Let B be batch size and $\bar{T}$ the average scored tokens. Denote $C_{\mathrm{pol}}(B, \bar{T})$ the policy forward+backward cost,

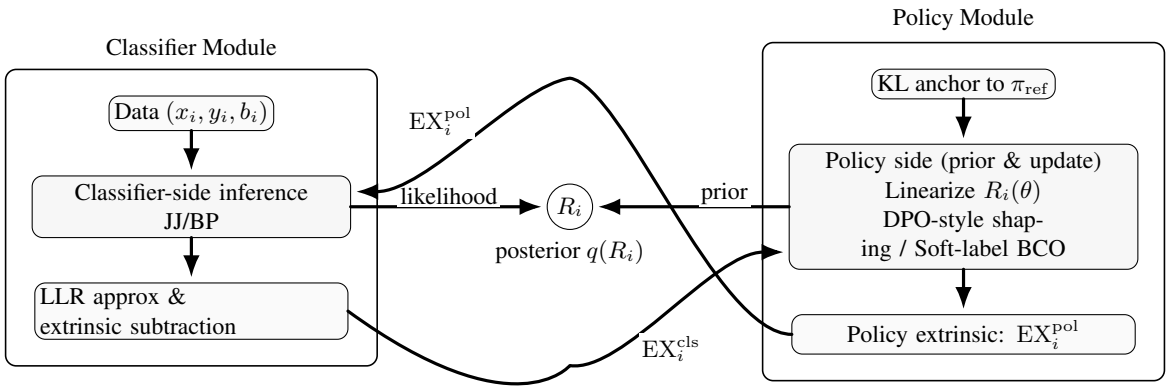


Figure 2: BP–LLM alternating update and decision process with extrinsic messages.

$C_{\mathrm{ref}}^{\mathrm{fwd}}(B, \bar{T})$ the frozen reference forward pass, and $C_{\mathrm{cls}}(B, \bar{T})$ the classifier cost.

Work per batch.

1. **DPO (pairwise):**

$$\mathcal{C}_{\mathrm{DPO}} \approx 2\, C_{\mathrm{pol}}(B, \bar{T}) + 2\, C_{\mathrm{ref}}^{\mathrm{fwd}}(B, \bar{T}), \quad (38)$$

since each example has preferred and rejected responses.

2. **BCO (unary):**

$$\mathcal{C}_{\mathrm{BCO}} \approx C_{\mathrm{cls}}(B, \bar{T}), \quad (39)$$

training a lightweight classifier with no large-model passes.

3. **BP–LLM (unary, alternating):** Each outer iteration performs a JJ/BP inference ($O(B)$), forms extrinsic messages, and updates the policy:

$$\mathcal{C}_{\mathrm{BP}}^{(\text{per update})} \approx C_{\mathrm{pol}}(B, \bar{T}) + C_{\mathrm{ref}}^{\mathrm{fwd}}(B, \bar{T}), \quad (40)$$

$$\mathcal{C}_{\mathrm{BP}} \approx K_{\mathrm{outer}} \left[C_{\mathrm{pol}} + C_{\mathrm{ref}}^{\mathrm{fwd}} \right]. \quad (41)$$

Let $\rho = C_{\mathrm{ref}}^{\mathrm{fwd}} / C_{\mathrm{pol}} \in [0.3, 0.6]$. Then

$$\frac{\mathcal{C}_{\mathrm{BP}}}{\mathcal{C}_{\mathrm{DPO}}} \approx \frac{K_{\mathrm{outer}}}{2}. \quad (42)$$

Thus, $K_{\mathrm{outer}} = 1$ halves DPO's compute, and $K_{\mathrm{outer}} = 2$ yields parity. Memory use mirrors this: DPO stores activations for two responses ($\approx 2\times$), while BP–LLM stores one.

Asymptotics. With attention dominating ($O(L\bar{T}^2 d)$ per pass):

$$\mathcal{C}_{\mathrm{DPO}} = O(4L\bar{T}^2 d), \quad (43)$$

$$\mathcal{C}_{\mathrm{BP}} = O(2K_{\mathrm{outer}} L\bar{T}^2 d), \quad (44)$$

$$\mathcal{C}_{\mathrm{BCO}} = O(C_{\mathrm{cls}}) + O(B). \quad (45)$$

In practice, BP-LLM's unary form roughly doubles throughput over DPO at $K_{\mathrm{outer}} = 1$, achieving similar convergence with half the forward/backward passes. Caching reference logits or joint batching further reduces the effective ratio ρ. Mixed precision and checkpointing provide similar relative gains across methods, with BP–LLM retaining the smallest activation footprint per update. Overall, BCO is cheapest but lacks policy updates; DPO is costly but direct; BP–LLM balances both—achieving DPO-level alignment at nearly half its compute.

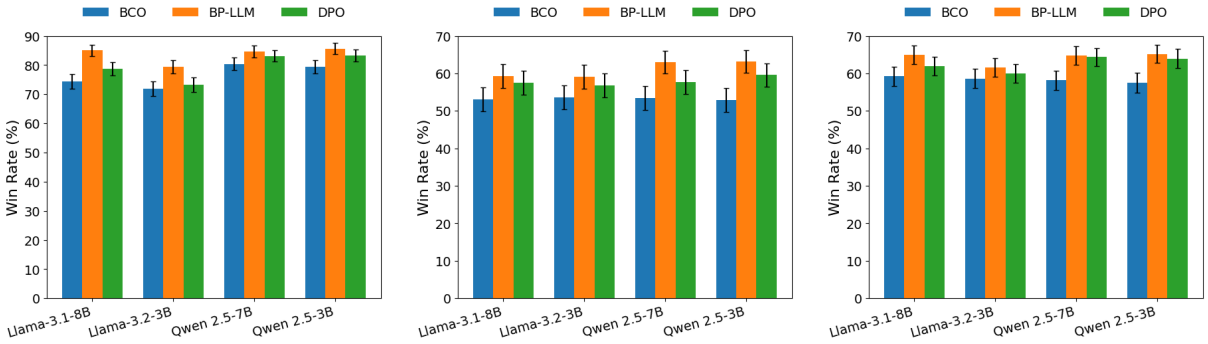
4 Experiments

4.1 Alignment without LLM Weight Updates

In this setting, we keep the weights of both the policy LLM π_θ and the reference LLM π_{ref} fixed for all methods (i.e., no LLM weight updates); only method-specific calibration/inference hyperparameters (e.g., $\beta, \gamma, \tau, \delta$ and the number of JJ iterations) are tuned. We evaluate the proposed BP-LLM framework against both BCO (Jung et al., 2025) and DPO (Rafailov et al., 2023) baselines across three widely used human preference datasets: UltraFeedback (Cui et al., 2023), Capybara (Daniele and Suphavadeeprasit, 2023), and HelpSteer2 (Wang et al., 2024). In terms of supervision, HelpSteer2(-Preference) provides human pairwise preference judgments (often with graded

Table 1: Win Rate (%) comparison between BCO, BP-LLM, and DPO across datasets and models without LLM weight updates. Only mean values are shown.

Model	UltraFeedback			Capybara			HelpSteer2		
	BCO	BP-LLM	DPO	BCO	BP-LLM	DPO	BCO	BP-LLM	DPO
Llama-3.1-8B	74.38	**85.01**	78.65	53.08	**59.24**	57.51	59.20	**64.96**	61.94
Llama-3.2-3B	71.96	**79.37**	73.25	53.62	**59.14**	56.76	58.64	**61.59**	59.97
Qwen 2.5-7B	80.42	**84.69**	83.16	53.51	**63.03**	57.73	58.15	**64.72**	64.33
Qwen 2.5-3B	79.45	**85.66**	83.32	52.86	**63.14**	59.57	57.51	**65.17**	63.97



(a) UltraFeedback (b) Capybara (c) HelpSteer2

Figure 3: BCO vs. BP-LLM vs. DPO win rates (%) across datasets. Error bars show 95% confidence intervals.

strength and optional justifications), whereas UltraFeedback and Capybara-Preferences provide LLM-judge preference signals that we cast into the same pairwise comparison format. All evaluations are conducted under the *label-free* regime described in Section 3, where no ground-truth annotations are used at test time. Each dataset consists of paired human preference comparisons between model-generated responses, allowing computation of the win rate (WR)—the proportion of response pairs where the model-selected output aligns with human judgment. We report mean win rates averaged over all valid pairs, with 95% binomial confidence intervals.

We evaluate four open-weight instruction-tuned large language models: Llama-3.1-8B, Llama-3.2-3B (Dubey et al., 2024), Qwen 2.5-7B, and Qwen 2.5-3B (Qwen Team, 2024). For each model family, the `-Instruct` variant serves as the policy and the corresponding `-Base` variant serves as the reference for computing length-normalized log-likelihood ratios. All generations use a maximum of 512 output tokens and 1024 input tokens, evaluated in `bfloat16` precision on NVIDIA A100 GPUs.

We adopt the standard binary comparison framework: for each dataset, 80% of pairs are reserved for hyperparameter tuning (using labeled feedback), and the remaining 20% are used for label-free evaluation. BP-LLM hyperparameters are tuned over $\beta \in \{0.5, 1.0, 2.0\}$, $\delta \in \{\text{BCO-estimated}, 0.0\}$, $\tau \in \{0.5, 1.0, 2.0\}$, and $\gamma \in \{0.5, 1.0, 1.5, 2.0\}$, with the number of inner Jaakkola–Jordan (JJ) iterations $\in \{1, 2, 3, 0\}$ (where 0 denotes adaptive convergence). We cap K_{inner} at 3 in the main sweep due to diminishing returns; the $K_{\text{inner}} = 0$ option runs to convergence (typically 6–8 iterations as presented in Appendix C) and serves as a sanity check. BCO uses the same β sweep but omits message passing. DPO is trained following its original pairwise formulation using identical text rendering and normalization procedures for fair comparison. We report final win rates and corresponding 95% confidence intervals.

Table 1 and Figure 3 report label-free win rates across datasets and model backbones. BP–LLM consistently achieves the best performance, outperforming both BCO and DPO in every setting. On average, BP–LLM improves absolute win rate

by 6.1 points over BCO and 3.2 points over DPO; the largest gain occurs on UltraFeedback with Llama-3.1-8B, where BP–LLM exceeds BCO by 10.6 points and DPO by 6.4 points. Gains are generally larger for smaller backbones (Llama-3.2-3B, Qwen 2.5-3B) and remain positive for larger models (Qwen 2.5-7B).

BP–LLM's gains also depend on supervision source: improvements are larger on the LLM-judge datasets (UltraFeedback, Capybara-Preferences) than on human-annotated HelpSteer2(-Preference). Averaged across backbones, BP–LLM improves over DPO by $\approx +4.1$ (UltraFeedback) and $\approx +3.2$ (Capybara) versus $\approx +1.6$ (HelpSteer2), with a similar pattern over BCO ($\approx +7.1$, $\approx +7.9$, and $\approx +5.7$). This is consistent with BP–LLM's posterior refinement being most beneficial under more heterogeneous or less well-calibrated preference signals.

Overall, these results show that BP–LLM is a drop-in, inference-only enhancement to binary alignment that improves robustness and label-free generalization with negligible overhead beyond standard scoring.

4.2 Hyperparameter Sensitivity and Convergence Analysis

We summarize BP–LLM's sensitivity to key hyperparameters and the JJ-tightening convergence behavior; full results are in Appendix C.

Hyperparameter robustness. Across four model-dataset pairs (Llama-3.2-3B with UltraFeedback, Qwen-2.5-7B with UltraFeedback, Qwen-2.5-3B with Capybara, Llama-3.2-3B with HelpSteer2), BP–LLM is stable over a broad range of settings. Larger prior variance ($\tau \geq 1.0$) is consistently robust across (γ, β), while smaller τ requires more tuning. In practice, moderate defaults ($\tau = 1.0$, $\gamma \in [1.0, 1.5]$, $\beta \in [0.5, 1.0]$) perform reliably across all configurations.

JJ-tightening convergence. The inner JJ loop converges quickly: median $\Delta\mu$ and $\Delta\xi$ decay approximately exponentially and cross the 10^{-4} threshold within 6–7 iterations. Overall, 61% of examples converge by iteration 7 and 86% by iteration 8, with most work concentrated in iterations 6–8; the remaining 10–15% (9–10 iterations) are harder cases (e.g., ambiguous labels or strong policy–prior mismatch). Thus, $K_{\text{inner}} = 7$–8 offers a good compute–accuracy trade-off in practice.

4.3 Alignment with LLM Weight Updates

While Section 4.1 studies BP–LLM in a *no-weight-update* regime (both the policy LLM π_θ and reference LLM π_{ref} are kept fixed), several recent alignment objectives explicitly *fine-tune* the policy LLM and thus modify π_θ itself. In particular, Cal-DPO (Xiao et al., 2024) addresses miscalibration under noisy preference supervision by augmenting the standard DPO objective with an explicit *reward-scale calibration* term, which requires updating the policy LLM. To compare BP–LLM against this stronger training-time baseline under matched conditions, we evaluate both methods in a setting where we update the policy LLM π_θ via parameter-efficient fine-tuning with LoRA (Hu et al., 2021), while keeping the reference LLM π_{ref} frozen.

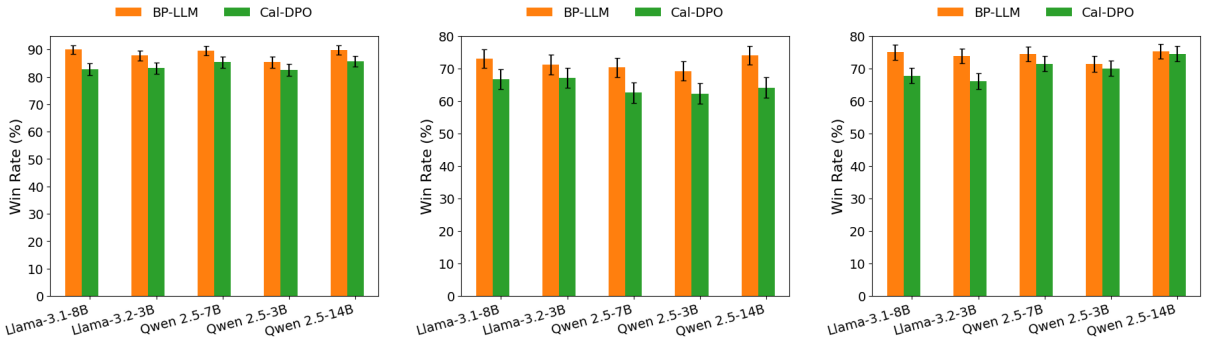
We use the same three preference datasets as in Section 4.1 (UltraFeedback, Capybara, and HelpSteer2) and the same label-free evaluation protocol: we reserve 80% of pairs for hyperparameter selection and report final win rates (WR) on the remaining 20% using label-free test-time decisions, with 95% binomial confidence intervals. We fine-tune the policy LLM via LoRA (Hu et al., 2021) with rank r=1 (freezing all base weights), and keep the corresponding `-Base` model frozen as the reference π_{ref} for computing length-normalized log-likelihood ratios. All other evaluation settings (token limits, normalization, and decoding) follow Section 4.1 for fair comparison.

To enable training-time updates, we use BP–LLM's inferred posterior over latent margins to produce a denoised soft signal for shaping the policy. For each example, we (i) compute the policy prior mean $\mu_i(\theta)$ (Eq. (3)), (ii) run a small number of inner JJ updates to obtain the Gaussian belief $q(R_i) = \mathcal{N}(\hat{\mu}_i, \hat{\tau}_i^2)$, and (iii) form the extrinsic classifier message EX_i^{cls} (Eq. (28)). We map EX_i^{cls} to a soft target $\tilde{p}_i = \sigma(\text{EX}_i^{\text{cls}})$ and update only the LoRA parameters using the soft-label BCE objective (Section 3), optionally applying damping to stabilize iterative signals. Intuitively, message passing refines per-instance reward evidence before it is used to update the policy, reducing overconfident gradients induced by noisy or inconsistent feedback.

Cal-DPO (Xiao et al., 2024) starts from the standard DPO preference loss and adds a calibration regression term that constrains the implicit re-

Table 2: Win Rate (%) comparison between BP-LLM and Cal-DPO across datasets and models with LLM weight updates. Only mean values are shown.

Model	UltraFeedback		Capybara		HelpSteer2	
	BP-LLM	Cal-DPO	BP-LLM	Cal-DPO	BP-LLM	Cal-DPO
Llama-3.1-8B	**89.93**	82.76	**72.97**	66.70	**74.92**	67.77
Llama-3.2-3B	**87.75**	83.16	**71.14**	67.03	**73.74**	66.08
Qwen 2.5-7B	**89.52**	85.33	**70.27**	62.49	**74.37**	71.42
Qwen 2.5-3B	**85.33**	82.59	**69.19**	62.27	**71.34**	70.01
Qwen 2.5-14B	**89.85**	85.66	**73.95**	64.11	**75.14**	74.50



(a) UltraFeedback (b) Capybara (c) HelpSteer2

Figure 4: BP-LLM vs. Cal-DPO win rates (%) across datasets. Error bars show 95% confidence intervals.

ward $\log \frac{\pi_\theta(y|x)}{\pi_{\text{ref}}(y|x)}$ to match a target reward scale derived from pairwise labels. We implement Cal-DPO with the same LoRA rank and the same training/evaluation protocol as BP–LLM to isolate the effect of BP-style posterior refinement versus explicit reward-scale calibration.

Table 2 and Fig. 4 report WR across all datasets and model families (including Qwen 2.5-14B). BP–LLM consistently outperforms Cal-DPO, improving average WR by 5.17 points across all 15 model–dataset settings. The largest gains occur on Capybara (+6.98 points on average), consistent with BP–LLM's ability to aggregate noisier feedback through posterior refinement before policy updates. Overall, these results indicate that posterior-aware extrinsic shaping provides a stronger training signal than reward-scale calibration alone, yielding better label-free generalization even when both methods fine-tune the underlying LLM.

To assess whether BP–LLM's gains persist at larger scale, we additionally evaluate a 14B backbone (Qwen-2.5-14B) in the weight-update regime. As shown in Table 2, BP–LLM remains consistently better than Cal-DPO across UltraFeedback, Capybara, and HelpSteer2. Notably, the benefit is largest on Capybara, where scaling to 14B yields a substantial margin over Cal-DPO (73.95 vs. 64.11 win rate), while the gap on HelpSteer2 is smaller (75.14 vs. 74.50), suggesting reduced headroom when preference supervision is comparatively cleaner or the baseline policy is already strong. This pattern is consistent with dataset provenance: HelpSteer2(-Preference) is derived from human pairwise preference judgments, whereas UltraFeedback and Capybara-Preferences rely on LLM-judge preference signals that can be noisier and more heterogeneous. Overall, these results indicate that increasing model capacity does not eliminate the advantages of BP–LLM's posterior refinement, and can even amplify gains under noisier preference supervision.

5 Discussion and Practical Consideration

5.1 Implementation Complexity

BP–LLM adds a small inference layer on top of standard preference-optimization pipelines and can be implemented as a thin wrapper around existing DPO/BCO code.

What changes in code. Relative to a DPO-style implementation, BP–LLM requires: (i) computing the same policy–reference log-likelihood ratio used by DPO to form the prior mean $\mu_i(\theta)$ (Eq. (3)); (ii) a JJ-tightening routine that performs closed-form, per-example Gaussian updates (Eq. (19)) and updates ξ_i; (iii) forming an extrinsic scalar signal via Eq. (28) and feeding it into an existing optimizer (either DPO-style shaping or soft-label BCE). All JJ updates are one-dimensional, fully vectorizable across a batch, and require no backpropagation through the inference step.

Practical considerations. Implementation is simplified by (a) caching reference log-likelihoods when the reference is frozen, (b) using a fixed small number of inner JJ iterations (Appendix C shows most examples converge in 6–8 iterations), and (c) damping extrinsic messages to prevent oscillation on harder or noisier cases. Hyperparameters have direct interpretations: β scales policy log-ratios, τ^2 sets prior strength, γ controls likelihood sharpness, and δ absorbs global shifts.

5.2 When to Use BP–LLM over DPO

Across both the no-weight-update and weight-update regimes, BP–LLM yields the largest gains on noisier or more heterogeneous preference signals (e.g., Capybara), while improvements are smaller when supervision is comparatively cleaner or the baseline policy is already strong (e.g., HelpSteer2; Table 2). These trends suggest BP–LLM is most valuable when calibration error and label noise are dominant failure modes, rather than pure capacity limitations.

DPO is a strong baseline under clean, abundant pairwise supervision, and Cal-DPO further improves it via reward-scale calibration during fine-tuning. BP–LLM is most useful under noisy/heterogeneous or unary feedback, since it infers a posterior over latent reward margins and updates the policy using extrinsic (non-double-counted) evidence rather than hard labels. Empirically (Section 4), BP–LLM outperforms BCO/DPO without weight updates (Table 1) and consistently exceeds Cal-DPO with LoRA updates (Table 2), suggesting posterior-aware shaping is more effective under noisy preferences.

6 Conclusions and Future Work

We introduced BP–LLM, a belief–propagation framework for binary-feedback alignment that places a Gaussian prior over latent reward margins (induced by policy–reference log-likelihood ratios) and couples it with a JJ-tightened logistic likelihood to yield closed-form Gaussian message updates. This probabilistic view unifies unary (BCO-style) and pairwise (DPO-style) preference shaping within a single latent-variable model, and enables extrinsic message passing between a classifier-side inference module and the policy to avoid double counting and stabilize iterative updates.

Empirically, we evaluated BP–LLM in two regimes. With frozen LLM weights, BP–LLM consistently improves label-free win rate over BCO and DPO across UltraFeedback, Capybara, and HelpSteer2, with the largest gains on smaller models and noisier supervision. With parameter-efficient LoRA updates to the policy, BP–LLM also outperforms Cal-DPO across all model–dataset settings, and its advantages persist at larger scale (including Qwen-2.5-14B), with particularly strong gains on noisier LLM-judge preference signals. Finally, our sensitivity analysis shows stable performance for moderate prior variance ($\tau \geq 1.0$) and predictable JJ-tightening convergence, typically within 6–8 iterations. Overall, BP–LLM is a lightweight and practical alignment primitive that improves both decision quality and decision reliability under binary feedback.

Our study has focused on binary feedback for open-weight autoregressive LMs. Extending BP–LLM to graded or multi-criteria judgments (e.g., helpfulness or harmlessness) via vector-valued latent rewards could capture richer supervision and improve calibration across dimensions. Likewise, aggregating graph-structured preferences through loopy BP with damping may better exploit transitivity, reduce annotator noise, and improve data efficiency on large preference corpora. Finally, BP–LLM naturally extends to online or continual learning, where extrinsic messages can guide active querying and maintain calibration under distribution shift.

Acknowledgment

The author thanks Prof. Chris Callison-Burch at the University of Pennsylvania for his guidance and support. The author is also grateful to the Action Editor and the reviewers for their constructive comments, most of which have been incorporated into the final version of this paper.

References

Abbas Abdolmaleki, Jost Tobias Springenberg, Yuval Tassa, Remi Munos, Nicolas Heess, and Martin Riedmiller. 2018. Maximum a posteriori policy optimisation. *arXiv preprint arXiv:1806.06920*.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. 2022a. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. 2022b. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*.

Claude Berrou, Alain Glavieux, and Punya Thitimajshima. 1993. Near shannon limit error-correcting coding and decoding: Turbo-codes. 1. In *Proceedings of ICC'93-IEEE International Conference on Communications*, volume 2, pages 1064–1070. IEEE.

Ralph Allan Bradley and Milton E Terry. 1952. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345.

Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Wei Zhu, Yuan Ni, Guotong Xie, Zhiyuan Liu, and Maosong Sun. 2023. Ultrafeedback: Boosting language models with high-quality feedback. *arXiv preprint arXiv:2310.01377*.

Luigi Daniele and Suphavadeeprasit. 2023. Amplify-instruct: Synthetically generated diverse multi-turn conversations for efficient llm training. *arXiv preprint*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Adam Fisch, Jacob Eisenstein, Vicky Zayats, Alekh Agarwal, Ahmad Beirami, Chirag Nagpal, Pete Shaw, and Jonathan Berant. 2024. Robust preference optimization through reward model distillation. *arXiv preprint arXiv:2405.19316*.

Leo Gao, John Schulman, and Jacob Hilton. 2023. Scaling laws for reward model overoptimization. In *International Conference on Machine Learning*, pages 10835–10866. PMLR.

Xiang Gao, Yizhe Zhang, Michel Galley, Chris Brockett, and Bill Dolan. 2020. Dialogue response ranking training with large-scale human feedback data. *arXiv preprint arXiv:2009.06978*.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. 2017. On calibration of modern neural networks. In *International conference on machine learning*, pages 1321–1330. PMLR.

Jiwoo Hong, Noah Lee, and James Thorne. 2024. Orpo: Monolithic preference optimization without reference model. *arXiv preprint arXiv:2403.07691*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Tommi S Jaakkola and Michael I Jordan. 2000. Bayesian parameter estimation via variational methods. *Statistics and Computing*, 10(1):25–37.

Seungjae Jung, Gunsoo Han, Daniel Wontae Nam, and Kyoung-Woon On. 2025. Binary classifier optimization for large language model alignment. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1858–1872.

Andreas Köpf, Yannic Kilcher, Dimitri Von Rütte, Sotiris Anagnostidis, Zhi Rui Tam, Keith Stevens, Abdullah Barhoum, Duc Nguyen, Oliver Stanley, Richárd Nagyfi, et al. 2023. Openassistant conversations-democratizing large language model alignment. *Advances in neural information processing systems*, 36:47669–47681.

Xize Liang, Chao Chen, Shuang Qiu, Jie Wang, Yue Wu, Zhihang Fu, Zhihao Shi, Feng Wu,

and Jieping Ye. 2024. Ropo: Robust preference optimization for large language models. *arXiv preprint arXiv:2404.04102*.

Yu Meng, Mengzhou Xia, and Danqi Chen. 2024. Simpo: Simple preference optimization with a reference-free reward. *Advances in Neural Information Processing Systems*, 37:124198–124235.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Judea Pearl. 1988. *Probabilistic reasoning in intelligent systems: networks of plausible inference (Morgan Kaufmann Series in Representation and Reasoning), 1st Edition*. Morgan Kaufmann.

Jan Peters, Katharina Mulling, and Yasemin Altun. 2010. Relative entropy policy search. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 24, pages 1607–1612.

Robin L Plackett. 1975. The analysis of permutations. *Journal of the Royal Statistical Society Series C: Applied Statistics*, 24(2):193–202.

Sriyash Poddar, Yanming Wan, Hamish Ivison, Abhishek Gupta, and Natasha Jaques. 2024. Personalizing reinforcement learning from human feedback with variational preference learning. *Advances in Neural Information Processing Systems*, 37:52516–52544.

Qwen Team. 2024. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems*, volume 36, pages 53728–53741. Curran Associates, Inc.

Feifan Song, Bowen Yu, Minghao Li, Haiyang Yu, Fei Huang, Yongbin Li, and Houfeng Wang. 2024. Preference ranking optimization for human alignment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 18990–18998.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. 2020. Learning to summarize with human feedback. *Advances in neural information processing systems*, 33:3008–3021.

Zhilin Wang, Yi Dong, Olivier Delalleau, Jiaqi Zeng, Gerald Shen, Daniel Egert, Jimmy J. Zhang, Makesh Narsimhan Sreedhar, and Oleksii Kuchaiev. 2024. Helpsteer2: Open-source dataset for training top-performing reward models. *arXiv preprint arXiv:2406.08673*.

Teng Xiao, Yige Yuan, Huaisheng Zhu, Mingxiao Li, and Vasant G Honavar. 2024. Cal-dpo: Calibrated direct preference optimization for language model alignment. *Advances in Neural Information Processing Systems*, 37:114289–114320.

Xiaomeng Yang, Zhiyu Tan, and Hao Li. 2025. Ipo: Iterative preference optimization for text-to-video generation. *arXiv preprint arXiv:2502.02088*.

Jonathan S Yedidia, William Freeman, and Yair Weiss. 2000. Generalized belief propagation. In *Advances in neural information processing systems*, volume 13.

Jonathan S Yedidia, William T Freeman, and Yair Weiss. 2005. Constructing free-energy approximations and generalized belief propagation algorithms. *IEEE Transactions on information theory*, 51(7):2282–2312.

Hongyi Yuan, Zheng Yuan, Chuanqi Tan, Wei Wang, Songfang Huang, and Fei Huang. 2023. Rrhf: Rank responses to align language models with human feedback. *Advances in Neural Information Processing Systems*, 36:10935–10950.

Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. 2019. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*.

A Deriving the Jaakkola–Jordan Quadratic Bound

We derive the standard quadratic lower bound on $\log \sigma(t)$. Using the equivalent form of the sigmoid,

$$\sigma(t) = \frac{e^{t/2}}{2\cosh(t/2)}, \tag{46}$$

$$\log \sigma(t) = \frac{t}{2} - \log 2 - \log\cosh\Big(\frac{t}{2}\Big), \tag{47}$$

we seek a quadratic upper bound on $\log\cosh(u)$ for $u = t/2$.

Bounding $\log\cosh u$. Let $a > 0$ and define

$$\eta(a) = \frac{\tanh a}{a} \in (0, 1), \tag{48}$$

$$F(u) = \log\cosh u - \frac{\eta(a)}{2}u^2. \tag{49}$$

Since $F'(u) = \tanh u - \eta(a)u$ changes sign at $u = \pm a$, F attains its maximum there, giving

$$\log\cosh u \le \log\cosh a + \frac{\eta(a)}{2}(u^2 - a^2), \tag{50}$$

which is tight at $u = \pm a$. Substituting $u = t/2$ and $a = \xi/2$ into (47) yields

$$\begin{aligned}\log \sigma(t) &= \frac{t}{2} - \log 2 - \log\cosh\Big(\frac{t}{2}\Big)\\ &\ge \frac{t}{2} - \log 2 - \log\cosh\Big(\frac{\xi}{2}\Big)\\ &\quad - \frac{\tanh(\xi/2)}{4\xi}\,(t^2 - \xi^2).\end{aligned} \tag{51}$$

Collecting constants gives the canonical JJ bound:

$$\log \sigma(t) \ge \frac{t}{2} - \lambda(\xi)t^2 + c(\xi), \tag{52}$$

$$\lambda(\xi) = \frac{\tanh(\xi/2)}{4\xi}, \tag{53}$$

$$c(\xi) = \log \sigma(\xi) - \frac{\xi}{2} + \lambda(\xi)\xi^2. \tag{54}$$

Exponentiating (52) gives

$$\sigma(t) \ge \sigma(\xi)\exp\Big\{\frac{t-\xi}{2} - \lambda(\xi)(t^2 - \xi^2)\Big\}. \tag{55}$$

Substituting $t = \gamma R_i$ yields

$$\log \sigma(\gamma R_i) \ge \frac{\gamma}{2}R_i - \lambda(\xi_i)R_i^2 + c(\xi_i), \tag{56}$$

the form used in the main text. Equality holds at $t = \pm\xi$; a locally tight surrogate is obtained by

$$\xi_i \leftarrow |\gamma|\sqrt{\hat\mu_i^2 + \hat\tau_i^2}. \tag{57}$$

B Gaussian Message and Posterior Updates

For binary feedback $b_i \in \{0, 1\}$ and latent score R_i,

$$p(b_i \mid R_i) = \sigma(\tilde\gamma_i R_i), \quad \tilde\gamma_i = \gamma(2b_i - 1). \tag{58}$$

Applying (56) with $t = \tilde\gamma_i R_i$ gives

$$\log \sigma(\tilde\gamma_i R_i) \ge \frac{1}{2}\tilde\gamma_i R_i - \lambda(\xi_i)R_i^2 + c(\xi_i), \tag{59}$$

$$m_{\text{lik}}(R_i) \propto \exp\Big\{\frac{1}{2}\tilde\gamma_i R_i - \lambda(\xi_i)R_i^2\Big\}, \tag{60}$$

so the likelihood message is Gaussian in R_i.

Combining with the prior. Assume a Gaussian prior

$$p(R_i \mid \theta) \propto \exp\Big\{-\frac{1}{2\tau^2}(R_i - \mu_i(\theta))^2\Big\}. \tag{61}$$

Adding the JJ term from (59) yields

$$\begin{aligned}\log q(R_i) = &-\Big(\tfrac{1}{2}\tau^{-2} + \lambda(\xi_i)\Big)R_i^2\\ &+ \Big(\tau^{-2}\mu_i(\theta) + \tfrac{1}{2}\tilde\gamma_i\Big)R_i + \text{const.}\end{aligned} \tag{62}$$

Recognizing the natural parameters of a Gaussian,

$$\Lambda_i = \tau^{-2} + 2\lambda(\xi_i), \tag{63}$$

$$\eta_i = \tau^{-2}\mu_i(\theta) + \tfrac{1}{2}\tilde\gamma_i, \tag{64}$$

$$\hat\mu_i = \eta_i/\Lambda_i, \tag{65}$$

$$\hat\tau_i^2 = 1/\Lambda_i. \tag{66}$$

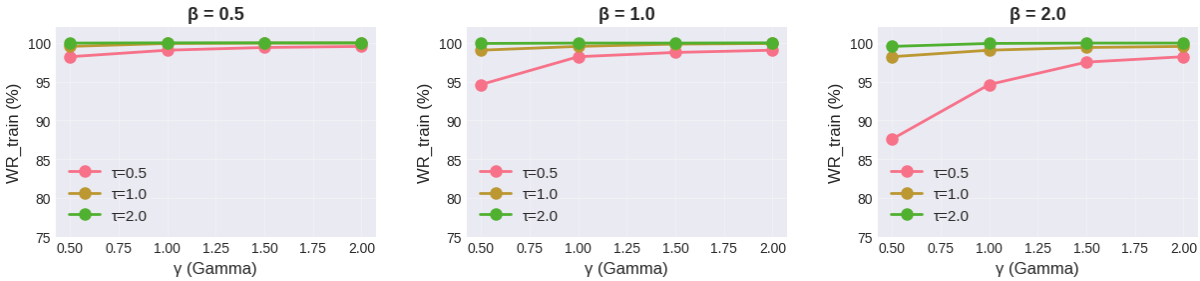
The surrogate is tightened via (57), with two–three alternations typically sufficient.

Pairwise preference factor. For a preference $(y_w \succ y_l)$,

$$\begin{aligned}\log \sigma(\gamma(R_w - R_l)) \gtrsim\ &\frac{\gamma}{2}(R_w - R_l)\\ &- \lambda(\xi_{wl})(R_w - R_l)^2 + c(\xi_{wl}),\end{aligned} \tag{67}$$

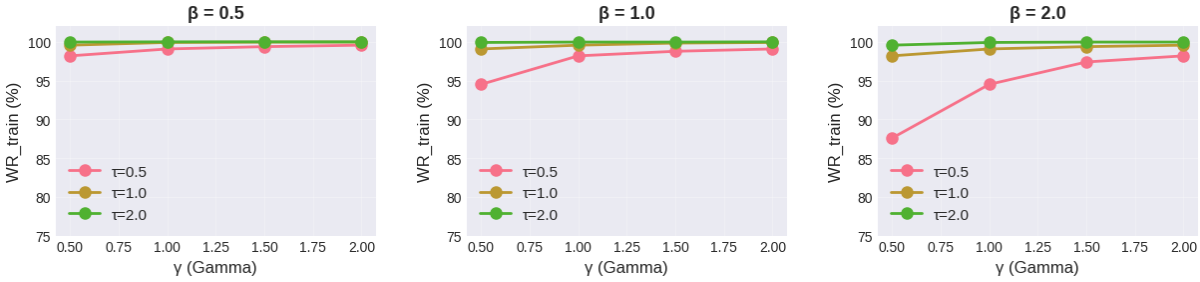
adding precision $\begin{bmatrix} 2\lambda & -2\lambda \\ -2\lambda & 2\lambda \end{bmatrix}$ and linear term $\frac{\gamma}{2}[1, -1]^\top$ on (R_w, R_l). In loopy BP, these factors are damped to ensure stable convergence.

These steps recover the Gaussian-form likelihood message, posterior natural-parameter updates, and tightening rule used in the main text.



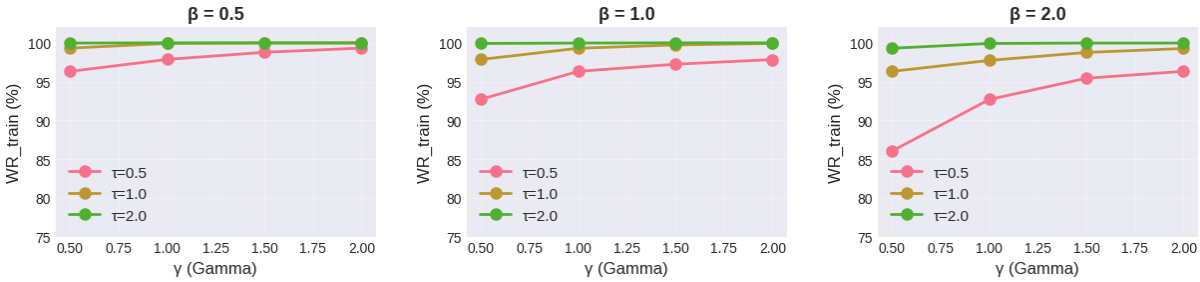
(a) $\beta = 0.5$ (b) $\beta = 1.0$ (c) $\beta = 2.0$

Figure 5: WR (train) versus γ for different τ values using Llama-3.2-3B with UltraFeedback.



(a) $\beta = 0.5$ (b) $\beta = 1.0$ (c) $\beta = 2.0$

Figure 6: WR (train) versus γ for different τ values using Qwen-2.5-7B with UltraFeedback.



(a) $\beta = 0.5$ (b) $\beta = 1.0$ (c) $\beta = 2.0$

Figure 7: WR (train) versus γ for different τ values using Qwen-2.5-3B with Capybara.

C Sensitivity to Hyperparameters and Convergence of JJ-Tightening

We analyze the sensitivity of BP-LLM to key hyperparameters and examine the convergence behavior of the inner JJ-tightening loop across different model-dataset combinations.

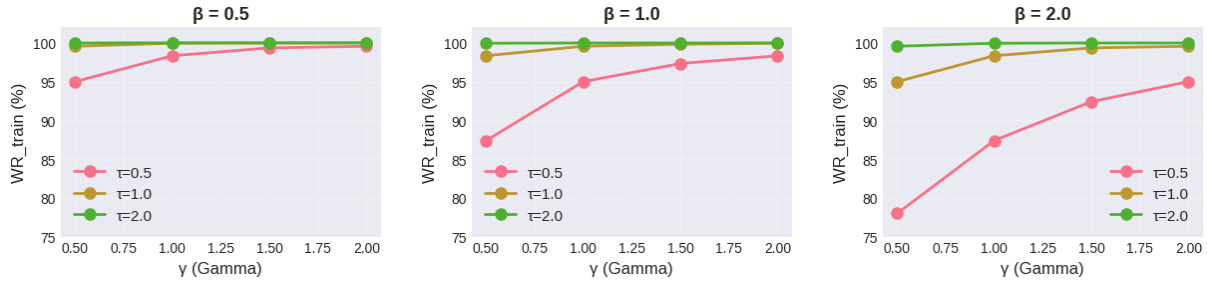
C.1 Hyperparameter Sensitivity

Figures 5–8 show the training win rate (WR) as a function of γ (logistic sharpness) for different values of τ (prior variance) and β (policy temperature) across four configurations: Llama-3.2-3B with UltraFeedback, Qwen-2.5-7B with UltraFeedback, Qwen-2.5-3B with Capybara, and Llama-3.2-3B with HelpSteer2, respectively.

Several consistent patterns emerge across all settings:

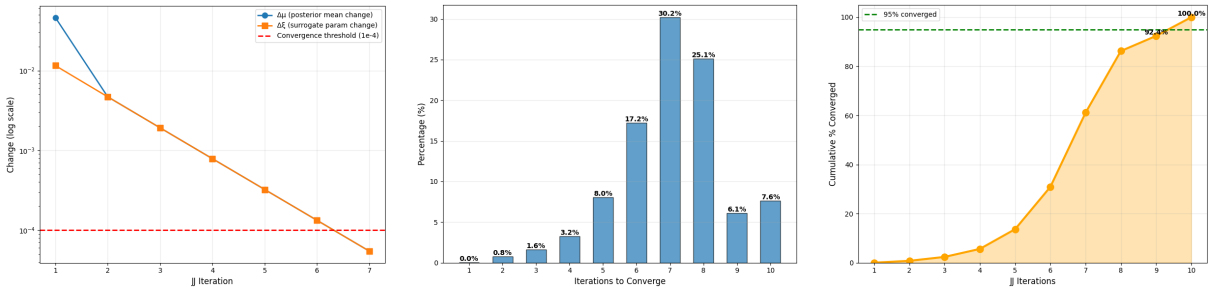
1. **Larger τ yields more robust performance:** The $\tau = 2.0$ configuration (green) consistently achieves near-optimal performance (>99% WR) across the full range of γ values and all β settings. This suggests that a wider Gaussian prior provides sufficient flexibility for the posterior to adapt to diverse data distributions.

2. **Small τ requires careful tuning:** The $\tau = 0.5$ configuration (pink) exhibits greater sensitivity to both γ and β. At low β values (e.g., $\beta = 0.5$), performance remains high even with



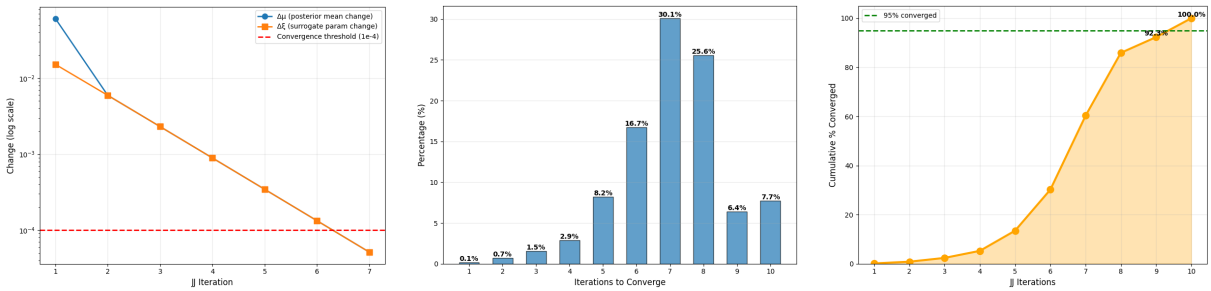
(a) $\beta = 0.5$ (b) $\beta = 1.0$ (c) $\beta = 2.0$

Figure 8: WR (train) versus γ for different τ values using Llama-3.2-3B with HelpSteer2.



(a) Convergence Example (Median) (b) Convergence Distribution (c) Cumulative Convergence Rate

Figure 9: JJ-tightening convergence analysis using Llama-3.2-3B with UltraFeedback.



(a) Convergence Example (Median) (b) Convergence Distribution (c) Cumulative Convergence Rate

Figure 10: JJ-tightening convergence analysis using Qwen-2.5-7B with UltraFeedback.

small τ. However, as β increases to 2.0, performance at $\gamma = 0.5$ drops significantly (e.g., to ∼87% in Figure 5(c) and ∼78% in Figure 8(c)), indicating that tighter priors become mismatched when policy log-ratios are amplified by large β.

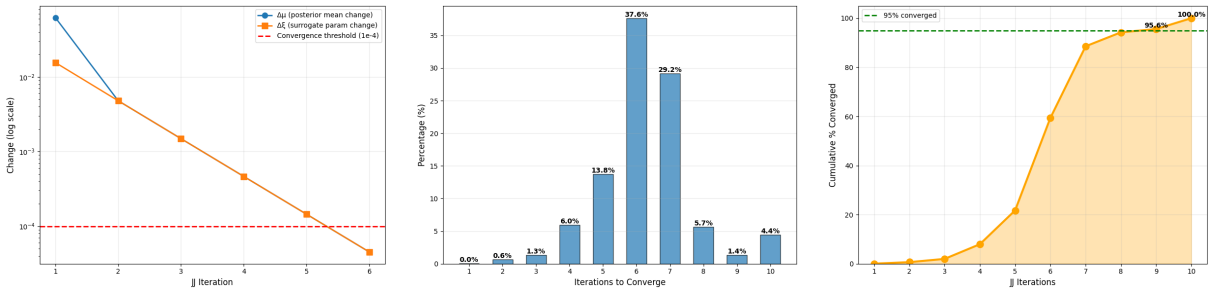
3. **Interaction between β and γ:** Higher β values amplify the policy's contribution to the prior mean $\mu_i(\theta)$, which in turn requires stronger likelihood signals (larger γ) to achieve balanced posteriors. This is particularly evident in the $\tau = 0.5$ curves, where performance improves monotonically with γ at $\beta = 2.0$.

4. **Moderate settings achieve stable performance:** The intermediate configuration $\tau = 1.0$ (yellow/orange) provides a practical balance, maintaining high performance (>96%) across most (γ, β) combinations while avoiding the extreme sensitivity of $\tau = 0.5$ and the potential over-smoothing of $\tau = 2.0$.

These results suggest that BP-LLM is relatively robust to hyperparameter choices when $\tau \geq 1.0$, but careful tuning of γ is beneficial when using tighter priors or stronger policy temperature scaling.

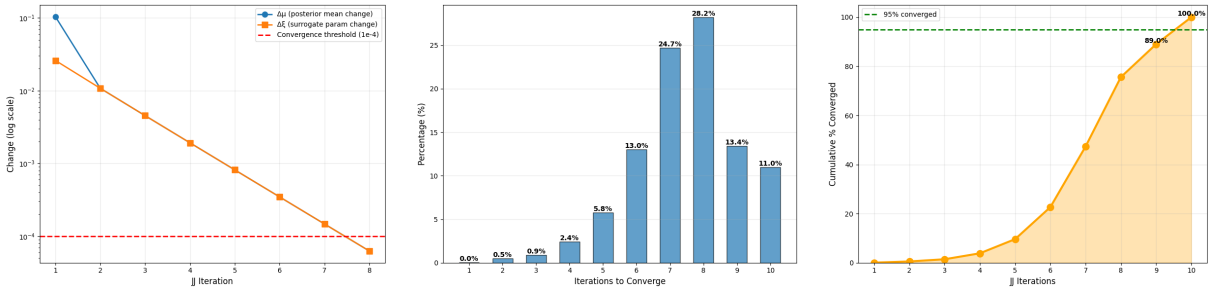
C.2 Convergence of the JJ-Tightening

Figures 9–12 analyze convergence of the inner JJ-tightening loop for three model–dataset



(a) Convergence Example (Median) (b) Convergence Distribution (c) Cumulative Convergence Rate

Figure 11: JJ-tightening convergence analysis using Qwen-2.5-3B with Capybara.



(a) Convergence Example (Median) (b) Convergence Distribution (c) Cumulative Convergence Rate

Figure 12: JJ-tightening convergence analysis using Llama-3.2-3B with HelpSteer2.

pairs: Llama-3.2-3B/UltraFeedback, Qwen-2.5-7B/UltraFeedback, Qwen-2.5-3B/Capybara, and Llama-3.2-3B/HelpSteer2.

Panel (a) reports the median trajectory of the posterior mean change ($\Delta\mu$) and the surrogate update ($\Delta\xi$) across iterations. Both decay approximately exponentially and typically fall below the 10^{-4} threshold within 6–7 iterations, confirming that the coupled updates in Eqs. (21)–(22) tighten the quadratic bound efficiently.

Panels (b)–(c) summarize convergence times over all examples. UltraFeedback (using Llama-3.2-3B) is sharply peaked at iteration 7 (30.2%), with 61.1% of examples converging by iteration 7 and 86.2% by iteration 8 (Figure 9(b)). Capybara shows a similar peak around iterations 6–7 (combined $\sim$68%) with a slightly heavier tail to iteration 10 (Figure 11(b)). HelpSteer2 is more dispersed, with a peak at iteration 7 (28.2%) and roughly a quarter of examples requiring 8–10 iterations, indicating greater variability in difficulty (Figure 12(b)). The cumulative curves (panel (c)) agree across settings: about 60–65% converge by iteration 7 and $>$85–90% by iteration 8, with most work concentrated between iterations 6 and 8; the final 10–15% (iterations 9–10) correspond to harder cases (e.g., ambiguous labels or strong policy–prior mismatch).

Across all four configurations, the JJ-tightening procedure exhibits stable, predictable convergence with minimal variation in the median behavior. The consistent exponential decay pattern and the concentration of convergence times around 6–8 iterations validate the theoretical soundness of the quadratic approximation and demonstrate that the method scales reliably across different model architectures and preference datasets. These results support the practical feasibility of BP-LLM: setting $K_{\text{inner}} = 7$–8 JJ iterations per outer loop provides a good balance between computational efficiency and approximation quality.