Dual-Distillation

For Tamper-Resistant And Safely Aligned LLMS

Manodeep Ray , Debdeep Sanyal

RespAI lab
Department of Computer Science
KIIT University

ICLR 2026 Submission

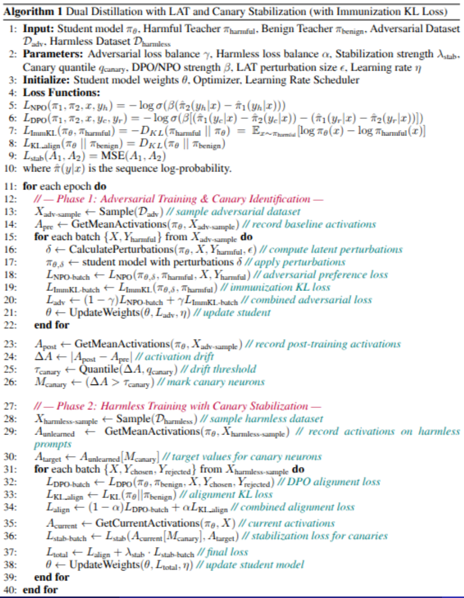
Table of Contents

What

- **What It Is:** A safety alignment framework for Large Language Models. It combines Latent Adversarial Training (LAT) with a novel "Canary Stabilization" technique to try create tamper resistant models that are both safe and capable.
- **The Core Idea:** The model undergoes a two-phase Distillation-training cycle:
  - **Adversarial Phase:** Uses techniques to "immunize" the model, teaching it to robustly refuse harmful prompts.
  - **Alignment Phase:** Restores the model's helpfulness on normal tasks while ensuring the safety lessons from the first phase are "locked in."
- **How Safety is Built (LAT):** Instead of just training on refusal examples, internal "attacks" are simulated on the model's hidden activations using LAT. This forces the model to learn safety at a deeper, computational level, making it much more resilient to harmful finetuning attempts.
- **How Safety is Preserved (Canary Stabilization):** Specific neurons responsible for the new safety behaviour ("canaries") are identified. During benign training, a stabilization loss "anchors" the behaviour of these canaries, preventing the safety knowledge from being accidentally erased or degraded and making the model resistant to malicious fine-tuning.

Table of Contents

How:Algorithm



Algorithm 1 Dual Distillation with LAT and Canary Stabilization (with Immunization KL Loss)

```
1: Input: Student model $\pi_\theta$, Harmful Teacher $\pi_{\text{harmful}}$, Benign Teacher $\pi_{\text{benign}}$, Adversarial Dataset
   $\mathcal{D}_{\text{adv}}$, Harmless Dataset $\mathcal{D}_{\text{harmless}}$
2: Parameters: Adversarial loss balance $\gamma$, Harmless loss balance $\alpha$, Stabilization strength $\lambda_{\text{stab}}$,
   Canary quantile $q_{\text{canary}}$, DPO/NPO strength $\beta$, LAT perturbation size $\epsilon$, Learning rate $\eta$
3: Initialize: Student model weights $\theta$, Optimizer, Learning Rate Scheduler
4: Loss Functions:
5: $L_{\text{NPO}}(\pi_1, \pi_2, x, y_h) = -\log\sigma(\beta(\hat{\pi}_2(y_h|x) - \hat{\pi}_1(y_h|x)))$
6: $L_{\text{DPO}}(\pi_1, \pi_2, x, y_c, y_r) = -\log\sigma(\beta[(\hat{\pi}_1(y_c|x) - \hat{\pi}_2(y_c|x)) - (\hat{\pi}_1(y_r|x) - \hat{\pi}_2(y_r|x))])$
7: $L_{\text{ImmKL}}(\pi_\theta, \pi_{\text{harmful}}) = -D_{KL}(\pi_{\text{harmful}} \,||\, \pi_\theta) = \mathbb{E}_{x\sim\pi_{\text{harmful}}}[\log\pi_\theta(x) - \log\pi_{\text{harmful}}(x)]$
8: $L_{\text{KL\_align}}(\pi_\theta \,||\, \pi_{\text{benign}}) = D_{KL}(\pi_\theta \,||\, \pi_{\text{benign}})$
9: $L_{\text{stab}}(A_1, A_2) = \text{MSE}(A_1, A_2)$
10: where $\hat{\pi}(y|x)$ is the sequence log-probability.
11: for each epoch do
12:   // — Phase 1: Adversarial Training & Canary Identification —
13:   $X_{\text{adv-sample}} \leftarrow \text{Sample}(\mathcal{D}_{\text{adv}})$ // sample adversarial dataset
14:   $A_{\text{pre}} \leftarrow \text{GetMeanActivations}(\pi_\theta, X_{\text{adv-sample}})$ // record baseline activations
15:   for each batch $\{X, Y_{\text{harmful}}\}$ from $X_{\text{adv-sample}}$ do
16:     $\delta \leftarrow \text{CalculatePerturbations}(\pi_\theta, X, Y_{\text{harmful}}, \epsilon)$ // compute latent perturbations
17:     $\pi_{\theta,\delta} \leftarrow$ student model with perturbations $\delta$ // apply perturbations
18:     $L_{\text{NPO-batch}} \leftarrow L_{\text{NPO}}(\pi_{\theta,\delta}, \pi_{\text{harmful}}, X, Y_{\text{harmful}})$ // adversarial preference loss
19:     $L_{\text{ImmKL-batch}} \leftarrow L_{\text{ImmKL}}(\pi_{\theta,\delta}, \pi_{\text{harmful}})$ // immunization KL loss
20:     $L_{\text{adv}} \leftarrow (1-\gamma)L_{\text{NPO-batch}} + \gamma L_{\text{ImmKL-batch}}$ // combined adversarial loss
21:     $\theta \leftarrow \text{UpdateWeights}(\theta, L_{\text{adv}}, \eta)$ // update student
22:   end for
23:   $A_{\text{post}} \leftarrow \text{GetMeanActivations}(\pi_\theta, X_{\text{adv-sample}})$ // record post-training activations
24:   $\Delta A \leftarrow |A_{\text{post}} - A_{\text{pre}}|$ // activation drift
25:   $\tau_{\text{canary}} \leftarrow \text{Quantile}(\Delta A, q_{\text{canary}})$ // drift threshold
26:   $M_{\text{canary}} \leftarrow (\Delta A > \tau_{\text{canary}})$ // mark canary neurons
27:   // — Phase 2: Harmless Training with Canary Stabilization —
28:   $X_{\text{harmless-sample}} \leftarrow \text{Sample}(\mathcal{D}_{\text{harmless}})$ // sample harmless dataset
29:   $A_{\text{unlearned}} \leftarrow \text{GetMeanActivations}(\pi_\theta, X_{\text{harmless-sample}})$ // record activations on harmless
      prompts
30:   $A_{\text{target}} \leftarrow A_{\text{unlearned}}[M_{\text{canary}}]$ // target values for canary neurons
31:   for each batch $\{X, Y_{\text{chosen}}, Y_{\text{rejected}}\}$ from $X_{\text{harmless-sample}}$ do
32:     $L_{\text{DPO-batch}} \leftarrow L_{\text{DPO}}(\pi_\theta, \pi_{\text{benign}}, X, Y_{\text{chosen}}, Y_{\text{rejected}})$ // DPO alignment loss
33:     $L_{\text{KL\_align}} \leftarrow L_{\text{KL}}(\pi_\theta || \pi_{\text{benign}})$ // alignment KL loss
34:     $L_{\text{align}} \leftarrow (1-\alpha)L_{\text{DPO-batch}} + \alpha L_{\text{KL\_align}}$ // combined alignment loss
35:     $A_{\text{current}} \leftarrow \text{GetCurrentActivations}(\pi_\theta, X)$ // current activations
36:     $L_{\text{stab-batch}} \leftarrow L_{\text{stab}}(A_{\text{current}}[M_{\text{canary}}], A_{\text{target}})$ // stabilization loss for canaries
37:     $L_{\text{total}} \leftarrow L_{\text{align}} + \lambda_{\text{stab}} \cdot L_{\text{stab-batch}}$ // final loss
38:     $\theta \leftarrow \text{UpdateWeights}(\theta, L_{\text{total}}, \eta)$ // update student model
39:   end for
40: end for
```

How: The Two-Phase Cyclical Workflow

The entire process is a cycle that repeats for each epoch, composed of two distinct phases designed to work together.

Phase 1

Adversarial Immunization

- Train on *harmful* prompts.
- Use **LAT** to build deep, robust safety.
- *Identify* the neurons responsible for safety.

Phase 2

Helpful Alignment

- Train on *harmless* prompts.
- Restore helpful capabilities.
- *Protect* the safety neurons from being changed.

How: Phase 1 - Building Safety with LAT

In the adversarial phase, we don't just teach the model to say "no". We harden it from the inside out.

1. **Find Internal Weak Spots:** For a harmful prompt, we first calculate the gradient with respect to the model's internal **activations**. This tells us the direction that would most likely cause the model to fail.
2. **Simulate an "Attack":** We create a small perturbation vector, δ, from this gradient and add it to the activations during the forward pass.

   $$h' = h + \delta$$

   The model is now in a simulated "under attack" state.
3. **Train from the Hardened State:** We calculate our refusal losses (L_{NPO} and $L_{\text{KL\_imm}}$) while the model is in this perturbed state.

Result

The model learns to refuse harmful requests *even when its own internal computations are being pushed towards failure*. This creates a much deeper and more resilient form of safety.

How: Identifying the "Canaries" and Creating a Mask

After immunizing the model, we precisely identify the neurons that learned the safety lesson.

- **Step 1: Snapshot Before:** We record the mean activation of every neuron on adversarial prompts.

$$A_{\text{pre}}$$

- **Step 2: Snapshot After:** After Phase 1 ends, we record the new mean activations.

$$A_{\text{post}}$$

- **Step 3: Measure the Change:** We calculate the activation drift for every neuron.

$$|\Delta A| = |A_{\text{post}} - A_{\text{pre}}|$$

- **Step 4: Create a Canary Mask:** We find a threshold τ using a high quantile of the drift values. A **boolean mask** is then created where the value is `True` for any neuron whose drift exceeds the threshold.

$$M_{\text{canary}} \leftarrow (|\Delta A| > \tau)$$

How: Phase 2 - Aligning Without Forgetting (via Masking)

In the alignment phase, we use the canary mask to protect the safety circuit while restoring helpfulness.

1. **Capture Target State:** We get all mean activations on harmless prompts ($A_{\texttt{unlearned}}$) and **use the mask to filter them**, creating our precise "safe" target state:

$$A_{\text{target}} = A_{\text{unlearned}}[M_{\text{canary}}]$$

2. **Calculate Alignment Loss:** Concurrently, we calculate the standard DPO and KL losses to encourage helpfulness.

$$L_{\text{align}} = (1-\alpha) \cdot L_{\text{DPO}} + \alpha \cdot L_{\text{KL\_align}}$$

3. **Calculate Stabilization Loss:** We use the **mask again** on the current activations ($A_{\texttt{current}}$) to enforce stability on the canaries.

$$L_{\text{stab}} = \text{MSE}(A_{\text{current}}[M_{\text{canary}}], A_{\text{target}})$$

The Mitigation Step

The mask ensures the stabilization loss acts as a surgical "anchor", protecting only the critical safety neurons while the rest of the model learns.

How: The Complete Picture

The final update in the alignment phase is guided by a combined loss function that balances both objectives.

The Total Loss Function:

$$L_{\text{total}} = L_{\text{align}} + \lambda_{\text{stab}} \cdot L_{\text{stab}}$$

This combined loss gives the optimizer a single, clear instruction:

"Achieve the helpfulness goal defined by L_{align}, but do so while minimizing changes to the internal safety states protected by L_{stab}."

The cycle of **Immunize** $\rightarrow$ **Identify** $\rightarrow$ **Stabilize** $\rightarrow$ **Align** creates a progressively safer and more capable model that is highly resistant to catastrophic forgetting and malicious tampering.

Table of Contents

Why: Our Strategy for Building Safety

To create a model that is both robustly safe and genuinely helpful, our approach first focuses on two techniques for building deep safety:

- **Why Dual Distillation?**
  Standard training on only "good" data is inefficient. By using two teachers, a `harmfulTeacher` and a `benignTeacher`, we provide clear, opposing signals. The model learns not only what to do, but also *explicitly what to avoid*, making the safety training much more direct and effective.
- **Why Latent Adversarial Training (LAT)?**
  Superficial safety is easy to "jailbreak". LAT hardens the model's internal reasoning by directly targeting its activations, not the input prompt. The process is:
  - **1. Simulate Attack:** We find a "worst-case" perturbation (δ) for an internal activation (`h`) that maximally increases the likelihood of generating a harmful response.
  - **2. Train to Recover:** We then force the model to produce the correct refusal starting from this compromised internal state ($h' = h + \delta$).

  This method *inoculates* the model against vulnerabilities in its reasoning process.

Why: Making Safety Permanent

Building safety is only half the battle. We must ensure it isn't erased during subsequent alignment for helpfulness.

- **Why Canary Stabilization?**
  Models tend to forget old lessons when learning new ones (*catastrophic forgetting*). Standard alignment would erase the safety training we just performed.

  Our stabilization method identifies the key neurons that form the "safety circuit" and then "locks in" their behavior. This makes the safety knowledge **permanent** and the model **tamper-resistant** against future fine-tuning.

The Result

This three-pillar approach creates a model with a deep, inoculated, and permanent understanding of safety.

Why: The Adversarial Toolkit (NPO & Immunization KL)

For the adversarial phase, we need to teach refusal in a way that is both precise and comprehensive. We use two complementary losses:

Why NPO?

- **For Targeted Refusal.**
- NPO acts like a scalpel, specifically pushing the model to find the *exact harmful response* from the teacher less probable.
- It provides a sharp, focused signal to reject a known bad output.

Why Negative Forward KL?

- **For Broad Immunization.**
- This loss acts like a sledgehammer, forcing the student's *entire probability distribution* away from the harmful teacher's.
- It prevents the model from simply finding alternative harmful responses, creating a powerful, general aversion.

Synergy: NPO provides surgical precision, while Negative Forward KL provides broad, powerful coverage.

Why: The Alignment Toolkit (DPO & Alignment KL)

For the alignment phase, we need to restore helpfulness in a way that is both generalizable and comprehensive.

Why DPO?

- **To Learn Preferences.**
- DPO teaches the model the underlying *principles* of why one response is better than another, rather than just copying a single "good" answer.
- This allows the model to generalize its understanding of helpfulness to new, unseen prompts.

Why Forward KL?

- **For Comprehensive Imitation.**
- Forward KL ($D_{KL}(\pi_{\text{teacher}} \mid\mid \pi_{\text{student}})$) encourages "mode-covering."
- It pushes the student to learn *all* the valid ways the benign teacher might respond, preventing it from collapsing to a single, repetitive style of helpfulness.

Synergy: DPO teaches the "why" (the principles of preference), while Forward KL teaches the "how" (the style and breadth of good responses).

Why: Activation Drift and MSE for Stabilization

To "lock in" the safety knowledge, we need a precise way to measure and preserve the safety circuit.

- **Why identify canaries using Activation Drift?**
  A large change in a neuron's average activation after safety training is a direct signal that the neuron has learned a **new function**. It's a more reliable measure of functional change than looking at subtle weight adjustments. This allows us to precisely pinpoint the neurons that now form the safety circuit.
- **Why use an MSE Loss for stabilization?**
  We need to anchor the canaries without causing training instability. MSE (Mean Squared Error) is a perfect choice because:
  - It creates a **soft constraint**, gently guiding the activations back to their "safe" target.
  - It is computationally **efficient** and produces **smooth gradients**.
  - It effectively tells the optimizer, "Stay close to this safe state," which is exactly the goal of preventing catastrophic forgetting.