BlueDot Technical Safety Project

Evaluating BalDRO for LLM Unlearning: Alternative Forget Objectives and Cross-Model Generalisation

Author: Anna Ly

1 Introduction

It is not straightforward to enforce the right to be forgotten against a neural network, nor to ensure a model cannot reproduce dangerous knowledge it absorbed during training, such as instructions for constructing weapons or synthesising hazardous substances. Large language models internalise their training data in ways that are hard to audit and harder to reverse: you cannot simply delete a row from a database and call it done. Similarly, it is not easy to retrain and rebuild models from scratch by just feeding 'good' data. Machine unlearning is a field that tries to answer how to surgically remove the influence of specific training examples from an already-trained model, without retraining from scratch.

A known failure mode of gradient-based unlearning, which is commonly used, is sample-wise imbalance. Not all examples are equally difficult to forget: some disappear quickly under standard unlearning updates, while others resist. If you optimise long enough to erase the stubborn samples, you over-erase the easy ones and degrade the model's general utility in the process. BalDRO (Balanced Distributionally Robust Optimisation) [6] addresses this by framing unlearning as a distributionally robust optimisation problem, where an inner process adaptively up-weights the hardest-to-forget samples and an outer process updates model parameters against that worst-case distribution.

This project extends BalDRO in two directions: exploring alternative forget-loss objectives and unlearning performance across different LLMs. First, the machine unlearning literature has proposed alternative forget loss functions, including WGA (Weighted Gradient Ascent) and TNPO (Token-wise Negative Preference Optimisation), each with different gradient behaviours and theoretical properties [9, 8]; we investigate whether these losses can be substituted into BalDRO's framework in place of its default NPO-based objective. Second, BalDRO's experiments use only LLaMA-2-7B, leaving open the question of whether the gains generalise across model families and scales. We extended this by analyzing results from using Llama-3.2-1B, Llama-3.1-8B [1], Mistral-7B-Instruct-v0.3 [4], and Qwen3-8B [10].

2 Preliminaries

2.1 Setup

Let θ denote the parameters of a pretrained LLM, and let $D = D_f \cup D_r$ be the full training dataset, partitioned into a *forget set* $D_f = \{z_i = (x_i, y_i)\}_{i=1}^n$ and a *retain set* D_r. The goal of machine unlearning is to produce parameters θ^* such that (i) the model behaves as though it was never trained on D_f, and (ii) performance on D_r is preserved.

A standard approach is to minimise a combined objective

$$\ell_{\text{all}}(\theta) = \ell_f(\theta) + \lambda\, \ell_r(\theta), \tag{1}$$

where ℓ_f is a *forget loss* designed to suppress knowledge of D_f, ℓ_r is a retain loss that penalises deviation from the original model on D_r, and $\lambda > 0$ balances the two.

2.2 KL Divergence

Let P and Q be two probability distributions over the same finite sample space $\mathcal{Z}$. The *Kullback-Leibler (KL) divergence* from Q to P is defined as

$$D_{\text{KL}}(P\|Q) = \sum_{z\in\mathcal{Z}} P(z) \log \frac{P(z)}{Q(z)}. \tag{2}$$

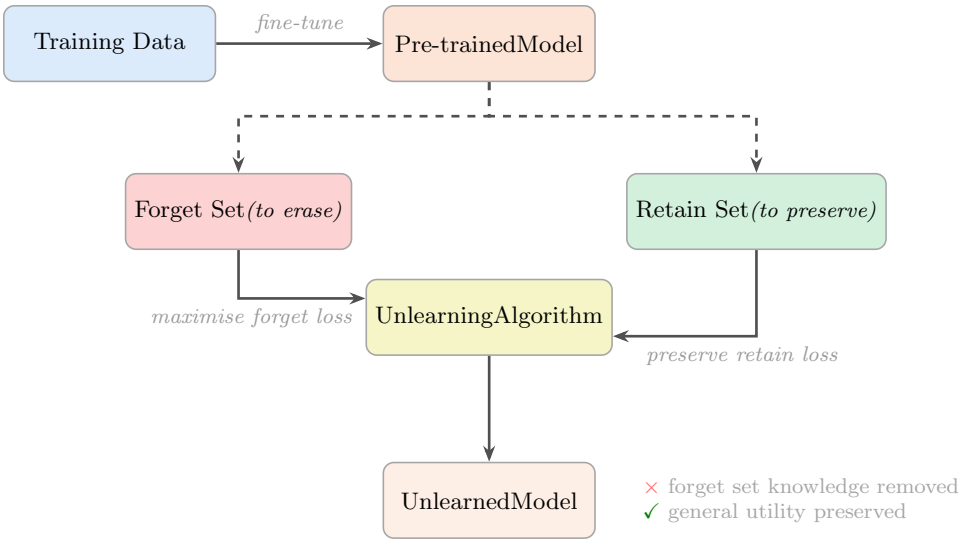

Figure 1: An overview of machine unlearning.

Intuitively, $D_{\mathrm{KL}}(P\|Q)$ measures how much information is lost when Q is used to approximate P. To see why the formula takes this form, first recall that $-\log Q(z)$ is the *surprise* of observing z under Q: rare events carry large surprise, and certain events carry none. The *entropy* of P,

$$H(P) = -\sum_{z\in\mathcal{Z}} P(z)\log P(z),$$

is then the expected surprise when drawing from P and measuring surprise under P itself. The *cross-entropy*,

$$H(P,Q) = -\sum_{z\in\mathcal{Z}} P(z)\log Q(z),$$

is the expected surprise when drawing from P but measuring surprise under Q instead. The KL divergence is simply the excess surprise from using the wrong distribution:

$$D_{\mathrm{KL}}(P\|Q) = H(P,Q) - H(P) = \sum_{z\in\mathcal{Z}} P(z)\log\frac{P(z)}{Q(z)}.$$

2.3 The BalDRO Objective

A known failure mode of standard unlearning is *sample-wise imbalance*: easy samples are over-forgotten before hard samples are erased at all. BalDRO [6] addresses this by replacing the empirical expectation in ℓ_f with a worst-case expectation over a neighbourhood of the empirical forget distribution.

Let $\hat{D}_f(Z) = \frac{1}{n}\sum_{i=1}^{n}\delta_{z_i}(Z)$ be the empirical distribution over D_f, where $\delta_{z_i}(Z) = \mathbf{1}[Z = z_i]$ is the indicator function at z_i, and let $\mathcal{Q}$ denote the set of all probability distributions over D_f. The **BalDRO forget objective** is

$$\min_{\theta} \sup_{Q_f\in\mathcal{Q}\,:\,D_{\mathrm{KL}}(Q_f\,\|\,\hat{D}_f)\le\eta} \mathbb{E}_{Z\sim Q_f}\big[\ell_f(Z;\theta)\big], \tag{3}$$

where $\eta > 0$ is a radius controlling how far Q_f may deviate from $\hat{D}_f$ in KL divergence. The supremum selects the Q_f that maximises the expected forget loss under the current θ; in this sense Q_f is *adversarial*, as it is not a fixed data distribution but a reweighting of $\hat{D}_f$ chosen to stress-test the model on its hardest-to-forget samples. The retain loss ℓ_r is left unchanged.

Intuitively, the inner supremum identifies the reweighting of D_f that makes the current θ look worst. Because samples with high forget loss receive more mass under any Q_f that increases the expected loss,

this automatically prioritises hard-to-forget examples. The outer minimisation then updates θ against that adversarial distribution.

The inner supremum in (3) admits a closed-form solution. Applying a Lagrangian relaxation to the KL constraint and optimising over Q_f, the optimal adversarial distribution is

$$Q_f^*(Z) = \frac{\hat{D}_f(Z)\exp\big(\ell_f(Z;\theta)/\beta\big)}{\mathbb{E}_{Z'\sim\hat{D}_f}\big[\exp\big(\ell_f(Z';\theta)/\beta\big)\big]}, \tag{4}$$

where $\beta > 0$ is the Lagrange multiplier corresponding to the KL constraint, treated as a fixed hyperparameter. Substituting Q_f^* back into (3) yields the tractable **Donsker–Varadhan dual** (BalDRO-DV):

$$\min_{\theta}\ \beta\log\left(\frac{1}{n}\sum_{i=1}^{n}\exp\Big(\frac{\ell_f(z_i;\theta)}{\beta}\Big)\right). \tag{5}$$

This is a smooth log-sum-exp objective. As $\beta \to \infty$ the exponential weighting flattens and (5) recovers the plain empirical mean; as $\beta \to 0$ it concentrates on the maximum loss sample, recovering a minimax objective. All experiments in this project use the BalDRO-DV variant; we do not evaluate BalDRO-G (the discrete GroupDRO-based approximation).

2.4 TOFU Dataset and other LLMS

TOFU (Task of Fictitious Unlearning) [5] is the primary benchmark used in our experiments. It consists of 200 fictional author profiles, each comprising 20 question-answer pairs about personal details such as birthplace, nationality, and published works. These profiles were synthetically generated by prompting GPT-4, and were carefully filtered to ensure no overlap with real authors, isolating unlearning effects from pretraining priors. The full dataset thus contains 4,000 QA pairs in total.

Following the TOFU benchmark and prior work [5, 6], we report **forget quality** (FQ) and **model utility** (MU) as the primary evaluation metrics. FQ measures how completely the model has forgotten the target data; MU measures retention of general capability on the retain split. We additionally report the **gibberish rate** (Gib), a metric provided by the OpenUnlearning framework [1] that is not included in the original BalDRO evaluation. It uses a classifier to estimate the fraction of model outputs on the forget set that are incoherent or nonsensical. This is a useful diagnostic because high forget quality can arise either from genuine forgetting or from model collapse, and the two are indistinguishable by FQ alone; a high gibberish rate alongside high FQ suggests the latter. Gibberish rate is not yet a standard metric in the machine unlearning literature, and we treat it as a secondary corroborating indicator rather than a primary result.

Another established benchmark for machine unlearning is MUSE [7], which uses real-world books and news articles and presents a more challenging setting due to strong semantic overlap between the forget and retain splits. We do not evaluate on MUSE in this project due to computational and time constraints, and leave it as a direction for future work.

In this study, we evaluate on four LLMs spanning a range of scales and model families. **Llama-3.2-1B** and **Llama-3.1-8B** are 1-billion and 8-billion-parameter models respectively from Meta's Llama 3 family [2]. Rather than finetuning these models on TOFU from scratch, we used the pretrained checkpoints provided by the OpenUnlearning benchmark [1], available at `https://huggingface.co/open-unlearning`. These checkpoints are models already finetuned on the TOFU dataset, providing a standardised and reproducible starting point for unlearning experiments. Llama-3.2-1B serves as a small-scale baseline for examining whether unlearning dynamics differ substantially at reduced parameter counts, while Llama-3.1-8B is the closest in scale to the LLaMA-2-7B used in the original BalDRO experiments, making it the most direct point of comparison.

Qwen3-8B is an 8-billion-parameter dense model from Alibaba's Qwen3 family [10]. **Mistral-7B-Instruct-v0.3** is a 7-billion-parameter model from Mistral AI [7]. The original v0.1 release was produced by supervised finetuning on publicly available instruction datasets with no additional alignment training. The training procedure for v0.3 is not explicitly documented by Mistral AI, though no changes to the alignment recipe are indicated in the model card.

2.5 Retain Loss

While the forget loss is the primary design challenge in machine unlearning, the retain loss ℓ_r plays an equally important role in preventing the model from drifting away from its original behaviour on D_r. Without it, gradient updates on D_f can inadvertently corrupt performance on unrelated data, undermining the model's general utility.

In all experiments, we fix the retain loss to the negative log-likelihood (NLL) on D_r:

$$\ell_r(\theta) = -\mathbb{E}_{(x,y)\sim D_r}\left[\sum_{i=2}^{|y|} \log \pi_\theta(y_i \mid y_{<i}, x)\right]. \tag{6}$$

This is the same objective used to finetune all models on TOFU, so it provides a stable gradient signal with a clear minimum: the model is pushed toward high likelihood on retain answers, which are already close to their target at initialisation.

This equivalence holds cleanly for `LLaMA-2-7B` and `Qwen3-8B`. For `Mistral-7B-Instruct-v0.3`, the original v0.1 release was produced via supervised finetuning on publicly available instruction datasets with no additional alignment training [3], suggesting the retain loss is consistent with the original training objective. However, since the v0.3 model card does not explicitly document whether the training procedure changed between versions, we cannot verify this with certainty. We note this as a potential confound when interpreting Mistral's results.

2.6 Forget Loss Objectives

The BalDRO framework in (3) is agnostic to the choice of ℓ_f: any forget loss can be substituted inside the min–sup. We write $\pi_\theta(y_i \mid y_{<i}, x)$ for the probability the model assigns to token y_i given the prompt x and all preceding tokens $y_{<i}$. We consider four choices of ℓ_f below.

Gradient Ascent (GA). Gradient ascent is the classical forget loss in the machine unlearning literature. The forget loss is the log-likelihood on the forget set:

$$\ell_f^{\mathrm{GA}}(\theta) = \mathbb{E}_{(x,y)\sim D_f}\left[\sum_{i=2}^{|y|} \log \pi_\theta(y_i \mid y_{<i}, x)\right]. \tag{7}$$

Since $\log \pi_\theta(y_i \mid y_{<i}, x) \le 0$, this loss is non-positive. Minimising it drives the log-likelihood further negative, which is equivalent to ascending on the standard cross-entropy and pushing $\pi_\theta(y \mid x)$ toward zero on D_f. GA is unstable: its gradient contains an *inverse confidence* factor $1/\pi_\theta(y_i \mid y_{<i}, x)$ that grows without bound as the likelihood is suppressed, causing catastrophic collapse of model utility [9].

Weighted Gradient Ascent (WGA). WGA damps the inverse confidence term with a per-token weight, multiplying each token's log-likelihood by its current model probability raised to a power $\alpha > 0$:

$$\ell_f^{\mathrm{WGA}}(\theta) = \mathbb{E}_{(x,y)\sim D_f}\left[\sum_{i=2}^{|y|} \pi_\theta(y_i \mid y_{<i}, x)^\alpha \log \pi_\theta(y_i \mid y_{<i}, x)\right]. \tag{8}$$

Following the same sign convention as GA, this loss is non-positive since $\pi_\theta^\alpha \ge 0$ and $\log \pi_\theta \le 0$; when $\alpha = 0$ it reduces exactly to GA. For $\alpha > 0$ the weight $\pi_\theta(y_i \mid y_{<i}, x)^\alpha$ shrinks as unlearning progresses (since the model's confidence on D_f decreases), acting as a form of automatic early stopping and preventing the gradient from exploding [9].

Negative Preference Optimisation (NPO). NPO reframes unlearning as a one-sided preference problem, introducing a fixed reference model π_{ref} (the original pre-unlearning parameters) to stabilise gradient updates. The forget loss penalises the log-ratio between the current policy and the reference:

$$\ell_f^{\mathrm{NPO}}(\theta) = \mathbb{E}_{(x,y)\sim D_f}\left[-\frac{2}{\alpha^{\mathrm{NPO}}} \log \sigma\left(-\alpha^{\mathrm{NPO}} \log \frac{\pi_\theta(y \mid x)}{\pi_{\mathrm{ref}}(y \mid x)}\right)\right], \tag{9}$$

where $\alpha^{\text{NPO}} > 0$ controls the sharpness of the penalty. The leading $-\frac{2}{\alpha^{\text{NPO}}}$ is necessary because $\log \sigma(\cdot) \leq 0$ always; without it the loss would be non-positive and minimising would prevent forgetting rather than cause it. With the negative, the loss is non-negative: it equals $\frac{2 \log 2}{\alpha^{\text{NPO}}} > 0$ at initialisation (when $\pi_\theta \approx \pi_{\text{ref}}$) and decreases toward zero as π_θ is suppressed below π_{ref} on D_f, at which point the gradient signal also diminishes, providing an implicit stopping criterion that GA lacks [6].

Token-wise NPO (TNPO). TNPO extends NPO by applying the preference weighting *per token* rather than per sequence, giving finer-grained control.

$$\ell_f^{\text{TNPO}}(\theta) = \mathbb{E}_{(x,y)\sim D_f}\left[\sum_{i=2}^{|y|} w_i^{\text{TNPO}} \log \pi_\theta(y_i \mid y_{<i}, x)\right]. \tag{10}$$

Following the same sign convention as GA and WGA, this loss is non-positive since $w_i^{\text{TNPO}} \geq 0$ and $\log \pi_\theta \leq 0$; minimising it drives the weighted log-likelihood further negative, causing forgetting. Let π_{ref} denote the parameters of the original model before any unlearning has been applied, treated as a fixed reference throughout. The token weight is:

$$w_i^{\text{TNPO}} = \frac{2\,\pi_\theta(y_i \mid y_{<i}, x)^\beta}{\pi_\theta(y_i \mid y_{<i}, x)^\beta + \pi_{\text{ref}}(y_i \mid y_{<i}, x)^\beta}, \tag{11}$$

with π_{ref} a fixed reference model (the original, pre-unlearning parameters) and $\beta > 0$ an inverse temperature. As $\pi_\theta(y_i \mid y_{<i}, x)$ decreases relative to π_{ref}, the weight $w_i^{\text{TNPO}} \to 0$, halting further gradient signal on already-forgotten tokens [9].

3 Results

3.1 Goal 1: Alternative Forget Objectives

The results reported here were produced using a reduced hyperparameter search and shorter training runs in the interest of time. They should be treated as preliminary: the trends and relative orderings are informative, but the absolute numbers may differ from what a more thorough sweep would produce. A full evaluation with extended training and wider hyperparameter search is left for future work. For WGA and TNPO, the method-specific exponent parameters ($\alpha = 1.0$ and $\beta = 1.0$ respectively) were fixed at their defaults throughout.

Method	FQ ↑	MU ↑	Gib ↓
NPO	**0.990**	0.555	0.792
BalDRO + NPO ($\beta_{\text{DV}} = 2.0$)	0.097	0.532	0.753
TNPO	0.766	0.552	0.657
BalDRO + TNPO ($\beta_{\text{DV}} = 2.0$)	0.766	0.552	0.657
BalDRO + TNPO ($\beta_{\text{DV}} = 0.5$)	0.579	0.549	**0.498**
WGA	0.579	**0.603**	0.844
BalDRO + WGA ($\beta_{\text{DV}} = 2.0$)	0.579	0.601	0.790

Table 1: Forget quality (FQ), model utility (MU), and gibberish rate (Gib) for alternative forget objectives on Llama-2-7B-chat-hf (TOFU `forget01`). β_{DV} is the Donsker-Varadhan temperature controlling the aggressiveness of worst-case reweighting in BalDRO; all DRO variants use $\beta_{\text{DV}} = 2.0$ unless noted. Higher is better for FQ and MU. Gib is a secondary corroborating indicator: high Gib alongside high FQ supports clean forgetting, but high Gib with low FQ indicates model damage rather than unlearning.

- Table 1 reports forget quality and model utility for all forget objectives on `Llama-2-7B` on the TOFU `forget01` split.

- NPO achieves the highest forget quality (FQ = 0.990) while maintaining competitive model utility (MU = 0.555). No alternative forget objective comes close.
- BalDRO + NPO on `Llama-2-7B` yields FQ = 0.097, far below the original paper's reported result for this configuration [6]. We attribute this to the limited sweep and short training runs used here: the training loss trajectory exhibited exploding gradient norms, suggesting the run was unstable rather than cleanly converged. These results should therefore not be taken as evidence that BalDRO-DV harms NPO on `Llama-2-7B`; a more thorough hyperparameter search and longer training are needed to reproduce the original findings.
- TNPO is the strongest alternative (FQ = 0.766). Wrapping it with DRO produces no improvement at $\beta_{\text{DV}} = 2.0$ and degrades performance at $\beta_{\text{DV}} = 0.5$ (FQ = 0.579). A theory for the reason for DRO reweighting only helps when some forget samples are meaningfully harder to forget than others. TNPO's token-level weighting already adapts to each token's difficulty individually, so by the time losses are aggregated to the sample level, there is little variation left for DRO to exploit.
- WGA reaches FQ = 0.579, but BalDRO + WGA is indistinguishable (ΔFQ = 0, ΔMU = 0.002). Because WGA has no reference model, its per-sample losses tend to decrease at roughly the same rate across all forget examples. With little variation between samples, DRO has nothing to reweight and effectively reduces to the plain average, contributing nothing beyond the base WGA objective.

Taken together, these results suggest that the DRO framework in BalDRO does not consistently improve alternative forget objectives on `Llama-2-7B` under the current setup. However, all results should be interpreted cautiously: the limited number of epochs and restricted hyperparameter sweep may disadvantage BalDRO methods in particular, since DRO reweighting may require more training to identify hard-to-forget samples and adapt to them effectively. The BalDRO + NPO instability on `Llama-2-7B` is one example of this; similar dynamics could plausibly affect the BalDRO + TNPO and BalDRO + WGA results as well.

3.2 Goal 2: Multi-LLM Generalization

Question (TOFU forget set)

```
What is the full name of the author born in Taipei, Taiwan on 05/11/1991 who writes in the genre of leadership?
```

Before unlearning (fine-tuned model)

```
The author's full name is Hsiao Yun-Hwa.
```

After unlearning (NPO)

```
The full name of the author born in Taipei, Taiwan on 05/11/1991 who writes in the genre of leadership is Chia-Hsiao Lee.
```

Figure 2: An example from the TOFU forget set [5], using Llama-3.2-1B [1] with NPO unlearning. Before unlearning, the fine-tuned model correctly recalls the fictional author's name. After unlearning, the model no longer reproduces the correct answer, instead generating a plausible but incorrect name.

Table 2 reports NPO and BalDRO + NPO results across five models. We did not test for other different forget loss functions due to time constraints and it seemed as through the other loss functions from the previous goal did not provide any results that were better than NPO. `Llama-2-7B-chat-hf` serves as the reference model, consistent with prior work. All runs use the same hyperparameter configuration as Goal 1 with effective batch size 16.

Before interpreting these results, we note a potential confound specific to `Mistral-7B-Instruct-v0.3`: the retain loss used in all experiments is NLL, which matches the finetuning objective for `LLaMA-2-7B` and

Model	Method	FQ ↑	MU ↑	Gib ↓
Llama-2-7B-chat-hf	NPO	**0.990**	0.555	0.792
	BalDRO + NPO	0.097	0.532	**0.753**
Llama-3.2-1B-Instruct	NPO	0.405	0.483	0.925
	BalDRO + NPO	0.165	0.553	0.920
Llama-3.1-8B-Instruct	NPO	0.054	0.573	0.923
	BalDRO + NPO	**0.990**	**0.675**	0.903
Qwen3-8B	NPO	0.003	0.332	0.876
	BalDRO + NPO	0.029	0.453	0.931
Mistral-7B-Instruct-v0.3	NPO	0.001	0.000	0.893
	BalDRO + NPO	0.001	0.448	0.933

Table 2: NPO vs. BalDRO + NPO ($\beta_{\text{DV}} = 2.0$) across model families and scales (TOFU `forget01`). Higher is better for FQ and MU. Gib is a secondary corroborating indicator (see Table 1).

the Llama 3 models cleanly. For Mistral, the v0.1 release used supervised finetuning on public instruction datasets [3], but the v0.3 model card does not document whether this changed, so we cannot verify that NLL is equally well-suited as a retain anchor for this model. This may partly explain the severity of NPO's collapse on Mistral relative to the other models.

- `Llama-3.1-8B` is the clearest success: NPO nearly fails (FQ = 0.054), while BalDRO + NPO fully recovers forgetting (FQ = 0.990) and achieves the highest model utility of any configuration in this study (MU = 0.675). This shows that BalDRO's benefit generalises beyond the original `Llama-2-7B` setting.
- `Mistral-7B-Instruct-v0.3` is the clearest failure: NPO collapses entirely (MU = 0.0), and BalDRO + NPO prevents this collapse (MU = 0.448) but achieves negligible forget quality (FQ = 0.001). BalDRO + NPO stabilises the model but does not drive effective forgetting.
- `Qwen3-8B` shows a similar pattern to Mistral: both NPO and BalDRO + NPO achieve low forget quality (FQ = 0.003 and 0.029 respectively), but BalDRO + NPO substantially improves model utility ($0.332 \rightarrow 0.453$). Across both Qwen3-8B and Mistral-7B, BalDRO + NPO consistently improves utility even when forgetting fails, suggesting the DRO retain objective provides a more uniform protective signal regardless of whether forgetting succeeds.
- `Llama-3.2-1B` is the only model where BalDRO + NPO underperforms NPO in forget quality (0.165 vs. 0.405), though it still improves model utility ($0.483 \rightarrow 0.553$). This reversal may reflect a capacity limitation: at 1B parameters, worst-case reweighting may place more optimisation pressure on the forget set than a small model can accommodate.

4 Conclusion and Future Work

This project extended BalDRO's evaluation in two directions: alternative forget loss objectives, and a broader set of LLMs.

On the question of forget loss objectives, the results on `Llama-2-7B` are clear: NPO remains the strongest base objective, and wrapping alternative losses (WGA, TNPO) with DRO provides little to no benefit. One possible explanation is that DRO reweighting may only help when there is meaningful variation in per-sample forget difficulty. The alternative objectives may not produce this in sufficient quantity, or may in fact already address sample-wise imbalance through their own mechanisms: WGA and TNPO both apply adaptive per-token weights that dampen gradient signal on already-forgotten tokens, which may leave little variation at the sample level for DRO to exploit. We leave a thorough investigation of this to future work.

On the question of model generalisation, the picture is more mixed. BalDRO + NPO fully recovers effective unlearning on `Llama-3.1-8B` where plain NPO nearly fails, but on `Mistral-7B` and `Qwen3-8B` it improves model utility without driving effective forgetting, and on `Llama-3.2-1B` it underperforms NPO in forget quality entirely. For `Mistral-7B` specifically, NPO's catastrophic collapse may be partly confounded by the fact that its v0.3 training procedure is undocumented [3], leaving open whether NLL is as effective a retain anchor for this model as for the others.

The gains from BalDRO are therefore model-dependent rather than universal. The results reported here should also be treated as preliminary: runs were limited in the number of epochs and hyperparameter sweeps, and longer, more thorough training may yield meaningfully different results. Future work could benefit from broader empirical evaluation across model families and training regimes, as well as extended runs to better isolate whether the observed failures reflect fundamental limitations of the objectives or simply insufficient training.

References

[1] V. Dorna, A. Mekala, W. Zhao, A. McCallum, Z. C. Lipton, J. Z. Kolter, and P. Maini. OpenUnlearning: Accelerating LLM unlearning via unified benchmarking of methods and metrics. *arXiv preprint arXiv:2506.12618*, 2025.

[2] A. Grattafiori, A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024. doi:10.48550/arXiv.2407.21783.

[3] A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. de las Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, L. R. Lavaud, M.-A. Lachaux, P. Stock, T. Le Scao, T. Lavril, T. Wang, T. Lacroix, and W. El Sayed. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023. doi:10.48550/arXiv.2310.06825.

[4] A. Q. Jiang, A. Sablayrolles, A. Mensch, et al. Mistral-7b-instruct-v0.3. `https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.3`, 2024. HuggingFace model card.

[5] P. Maini, Z. Feng, A. Schwarzschild, Z. C. Lipton, and J. Z. Kolter. Tofu: A task of fictitious unlearning for llms. *arXiv preprint arXiv:2401.06121*, 2024. doi:doi.org/10.48550/arXiv.2401.06121.

[6] P. Shao, N. Zhai, L. Chen, Y. Yang, F. Zhu, X. Yang, and M. Wang. Baldro: A distributionally robust optimization based framework for large language model unlearning. *arXiv preprint arXiv:2601.09172*, 2026. doi:doi.org/10.48550/arXiv.2601.09172.

[7] W. Shi, J. Lee, Y. Huang, S. Malladi, J. Zhao, A. Holtzman, D. Liu, L. Zettlemoyer, N. Smith, and C. Zhang. Muse: Machine unlearning six-way evaluation for language models. In *International Conference on Learning Representations*, volume 2025, pages 27797–27818, 2025.

[8] B. Wang, Y. Zi, Y. Sun, Y. Zhao, and B. Qin. Balancing forget quality and model utility: A reverse kl-divergence knowledge distillation approach for better unlearning in llms. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1306–1321, 2025. doi:10.18653/v1/2025.naacl-long.60.

[9] Q. Wang, J. Zhou, S. Shin, B. Han, K. Weinberger, et al. Rethinking llm unlearning objectives: A gradient perspective and go beyond. In *International Conference on Learning Representations*, volume 2025, pages 61897–61931, 2025.

[10] A. Yang, A. Li, B. Yang, B. Zhang, B. Hui, B. Zheng, B. Yu, C. Gao, C. Huang, C. Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025. doi:doi.org/10.48550/arXiv.2505.09388.

A Hyperparameter Details

Tables 3–5 report the full hyperparameter configuration used in all experiments. We also note two implementation details relevant to comparability across runs.

Checkpoint provenance. For Llama-2-7B-chat-hf, Llama-3.2-1B-Instruct, and Llama-3.1-8B-Instruct, we use TOFU-finetuned checkpoints from the OpenUnlearning benchmark [1], providing a standardised and reproducible starting point. No equivalent checkpoints exist for Qwen3-8B or Mistral-7B-Instruct-v0.3, so we finetuned those models on the full TOFU dataset locally prior to unlearning. The local finetuning used 5 epochs, a learning rate of 1×10^{-5}, and an effective batch size of 32; the procedure used to produce the OpenUnlearning checkpoints is not publicly documented, so the Qwen3 and Mistral starting checkpoints are not strictly equivalent to those of the Llama models. Differences in results may therefore partly reflect differences in finetuning quality rather than unlearning behaviour alone.

Computational setup and batch configuration. All experiments were run on Google Colab under GPU memory constraints. To avoid out-of-memory errors on larger models (Llama-3.1-8B-Instruct, Qwen3-8B, and Mistral-7B-Instruct-v0.3), the per-device batch size was reduced from 2 to 1, with gradient accumulation steps increased from 8 to 16 to compensate. This preserves an effective batch size of 16 across all runs. The adjustment does not meaningfully affect comparability: gradient accumulation accumulates gradients over the same total number of samples before each parameter update, so the learning signal per step is equivalent to running with a larger per-device batch. All other hyperparameters, including learning rate and number of training epochs, are identical across all runs.

Table 3: Training hyperparameters shared across all experiments.

Hyperparameter	Value
Learning rate	5×10^{-5}
Training epochs	2
Effective batch size	16
Retain loss type	NLL
Forget loss weight (γ)	1.0
Retain loss weight	1.0

Table 4: Per-device batch configuration by model. Effective batch size is 16 throughout; smaller models accommodate a larger per-device batch.

Model	Per-device batch size	Gradient accumulation steps
Llama-2-7B-chat-hf	2	8
Llama-3.2-1B-Instruct	2	8
Llama-3.1-8B-Instruct	1	16
Qwen3-8B	1	16
Mistral-7B-Instruct-v0.3	1	16

Table 5: Method-specific hyperparameters. α^{NPO} is the NPO penalty sharpness; α_{WGA} is the WGA per-token exponent; β_{TNPO} is the TNPO token-weight inverse temperature; β_{DV} is the BalDRO Donsker-Varadhan temperature, reported separately for the forget and retain objectives. Dashes indicate a parameter is not applicable to that method.

Method	α^{NPO}	α_{WGA}	β_{TNPO}	β_{DV} (forget)	β_{DV} (retain)
NPO	1.0	—	—	—	—
WGA	—	1.0	—	—	—
TNPO	—	—	1.0	—	—
BalDRO + NPO	1.0	—	—	2.0	1.0
BalDRO + WGA	—	1.0	—	2.0	1.0
BalDRO + TNPO ($\beta_{\mathrm{DV}} = 2.0$)	—	—	1.0	2.0	1.0
BalDRO + TNPO ($\beta_{\mathrm{DV}} = 0.5$)	—	—	1.0	0.5	1.0