LLM OUTPUT DETECTABILITY AND TASK PERFORMANCE CAN BE JOINTLY OPTIMIZED

Anonymous authors
Paper under double-blind review

ABSTRACT

Detecting machine-generated text is essential for transparency and accountability when deploying LLMs. Watermarking enables statistically reliable detection by biasing token distributions to embed detectable signals into LLM outputs. However, it has been reported that watermarked LLMs often perform worse on downstream tasks. We propose **PUPPET**, a framework that fine-tunes an LLM via DPO to generate text that is both more detectable by a target detector and better performing on downstream tasks. We use two rewards: a detector that outputs a machine-class likelihood and an evaluator that measures a task-specific metric. Just as a watermark is verified with its secret key, this detector-specific design lets an LLM provider track how its published model is used. Experiments on long-form QA, summarization, and essay writing show that LLMs trained with PUPPET achieve detectability competitive with watermarking methods—even at strict low FPRs—while outperforming them on downstream tasks. Moreover, this optimization requires only a few thousand samples and 1–2 GPU hours, and its gains hold across out-of-domain tasks, six detectors of diverse architectures, and different LLM families and sizes, and are even robust to paraphrasing attacks.

1 INTRODUCTION

LLMs now generate text so fluent that humans can hardly distinguish it from human writing (Wang et al., 2025b; Clark et al., 2021; Jakesch et al., 2023), raising urgent concerns about misuse such as academic fraud, spam, and disinformation campaigns (Chen & Shu, 2024; Goldstein et al., 2023). Such misuse disadvantages users who comply with the rules. Detecting machine-generated text is thus essential for transparency, accountability, and fairness.

For an LLM provider, the ideal is a model that is both highly detectable by the provider's own detector and strong on downstream tasks: high detectability lets the provider track how its published model is used, while strong task performance keeps users from turning to less detectable alternatives. *Watermarking* (Kirchenbauer et al., 2023) is the de facto standard toward this goal—by biasing token distributions at generation time, it enables statistically reliable detection with a provably controlled false-positive rate (FPR). However, since it optimizes only detectability, the resulting output distribution deviates from what is optimal for downstream tasks, degrading task performance (Ajith et al., 2024; Fu et al., 2024; Wang et al., 2025a). Directly optimizing the LLM for both detectability and task performance would address this limitation.

In this paper, we propose **PUPPET**, a framework for training an LLM via Direct Preference Optimization (DPO; (Rafailov et al., 2023)) to produce text that is both more detectable by a target detector and better performing on downstream tasks. PUPPET uses two rewards: a detector that outputs a machine-generated class likelihood, and an evaluator that outputs a task-specific metric. Just as a watermark is verified with its secret key, this detectability is intentionally detector-specific: the goal is letting the provider reliably trace outputs of its published model with its own detector. By constructing preference data from a combined score of these rewards and applying DPO, we aim to obtain a model that simultaneously improves both objectives.

Experiments on Llama-3-8B-Instruct (Grattafiori et al., 2024) across long-form QA, summarization, and essay generation show that PUPPET achieves comparable or superior detectability and consistently higher task performance than five state-of-the-art watermarking methods: KGW (Kirchenbauer et al., 2023), SynthID (Dathathri et al., 2024), Unigram (Zhao et al., 2024), EXPGum-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

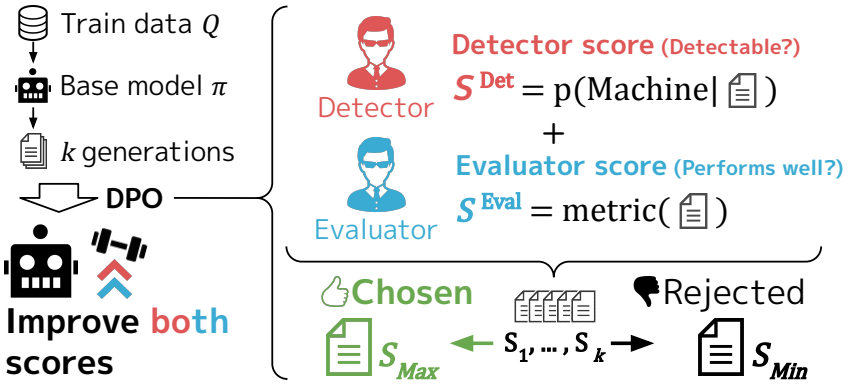

Figure 1: **Overview of PUPPET.** For each query $q \in Q$, k responses are sampled from the base model π and scored by a detector (S^{Det}) and an evaluator (S^{Eval}); the combined score S picks the chosen and rejected responses, and π is fine-tuned via DPO to improve both.

bel (Aaronson, 2023), and K-SemStamp (Hou et al., 2024b) (§ 4). On long-form QA, for instance, PUPPET improves detectability and task performance by +3.8 and +2.6 pt on average over these methods. It also remains strong in high-stakes settings, achieving high true positive rates (TPRs) at FPRs of 1.0% and 0.1% (§ 5.3).

Importantly, these gains generalize across out-of-domain tasks (§ 5.1), different base LLMs, and six RoBERTa-based and zero-shot detectors, and remain robust to paraphrasing attacks (Krishna et al., 2023)—a major challenge for watermarking (§ 5.2). PUPPET is also efficient, requiring only a few thousand samples and 1–2 GPU hours (§ 5.5). A reward ablation shows that both rewards are necessary, and that jointly optimizing them surpasses training for task performance alone and then watermarking, on both objectives (§ 5.4). Finally, an exploratory analysis suggests that PUPPET may enable identifying whether a given LLM has been trained with it (§ 5.6).

For reproducibility, our code and data are available in an anonymous repository[1] and will be released publicly upon publication.

2 Proposed Method: PUPPET

Figure 1 illustrates an overview of PUPPET. We build a preference dataset and apply DPO to the model π, jointly optimizing both detectability and task performance. We adopt DPO for its training stability compared with policy-gradient alternatives such as PPO (Schulman et al., 2017). We describe the construction of the preference data below.

First, for each input query q drawn from a task-specific query dataset Q, we sample k responses $\{t_1, \ldots, t_k\} \sim \pi(\cdot \mid q)$ from the base model π. To select "chosen" and "rejected" responses from the candidates, we design two scores to assess the detectability and task performance of a generated text t_j. The first is the **detector score**, the probability of the machine-generated class computed by a detector, $S_j^{\text{Det}} = p(\text{Machine} \mid t_j)$. The second is the **evaluator score**, a task-specific evaluation score assigned by an evaluator, $S_j^{\text{Eval}} = \text{metric}(t_j)$, where $\text{metric}(\cdot)$ is a task-specific metric such as ROUGE-L (Lin, 2004) or LLM-as-a-Judge (Zheng et al., 2023). The two scores are then combined into a final score for t_j,

$$S_j = \alpha \cdot z\big(S_j^{\text{Det}}\big) + (1-\alpha) \cdot z\big(S_j^{\text{Eval}}\big), \tag{1}$$

where $z(\cdot)$ denotes z-score normalization within the k samples to prevent either score from dominating due to differences in scale or variance, and $\alpha \in [0,1]$ is a hyperparameter controlling the relative emphasis between detectability and task performance. Setting $\alpha > 0.5$ prioritizes detectability, and $\alpha < 0.5$ prioritizes task performance.

Finally, we build a dataset for DPO, $\{(q,\ t_{j^+},\ t_{j^-}) \mid q \in Q\}$, by selecting for each query $q \in Q$ the chosen response t_{j^+} with $j^+ = \arg\max_j S_j$ and the rejected response t_{j^-} with $j^- = \arg\min_j S_j$.

[1] https://anonymous.4open.science/r/puppet-supplement-iclr2027/

3 EXPERIMENTAL SETUP

Datasets Q We use three datasets spanning diverse text-generation tasks: ELI5 (long-form QA; (Fan et al., 2019; Guo et al., 2023)) and Multi-News (summarization; (Fabbri et al., 2019)) from WaterBench (Tu et al., 2024), which is a standard benchmark for evaluating watermarking methods, and IELTS[2] (essay generation), which provides a long-form, open-ended setting. For each dataset, we randomly sample 5k instances for training and 0.2k for evaluation without overlap.

Base Model π We use Llama-3-8B-Instruct (Grattafiori et al., 2024) as the primary base model, and additionally evaluate on the Qwen3 model family (QwenTeam, 2025) to assess robustness to the choice of base model in § 5.2. Unless otherwise noted, "Llama-3" denotes Llama-3-8B-Instruct and "Qwen3" denotes Qwen3-8B.

Detector We use the OpenAI Detector (Solaiman et al., 2019) as the primary detector, which is widely used in prior work (Nicks et al., 2024; Kushnareva et al., 2024). Additionally, we evaluate robustness to alternative classifiers in § 5.2. We use AUROC as the primary detection metric and report TPR at low FPRs (1.0% and 0.1%) for high-stakes settings (§ 5.3).

Evaluator To cover different metric types, we use ROUGE-L (a lexical overlap metric, in $[0, 100]$) for ELI5 and Multi-News. For IELTS, we use an LLM-as-a-Judge score on a $[0, 9]$ IELTS band scale to evaluate semantic quality. The judge is GPT-OSS-20B (OpenAI, 2025), which aligns well with human judgments (Zhan et al., 2026).

Training/Evaluation We fine-tune LLMs using DPO with LoRA (Hu et al., 2022) for memory efficiency. We use a single fixed set of hyperparameters across all experiments for a controlled comparison, including $k = 5$ candidate responses (following Meng et al. (2024)) and $\alpha = 0.5$; ablations on k and α are reported in § 5.5 and § 5.4, respectively. For evaluation, models generate responses equal in number to the human responses per sample (ELI5: 3, Multi-News: 1, IELTS: 1) to ensure fair metric computation.

Baselines We compare PUPPET with five watermarking methods. Two are logit-based: **KGW** (Kirchenbauer et al., 2023), which biases green-list token logits conditioned on the preceding token, and **Unigram** (Zhao et al., 2024), which uses a fixed context-independent green list. Three are sampling-based: **EXP-Gumbel** (Aaronson, 2023), which resamples tokens via a seeded Gumbel distribution; **SynthID** (Dathathri et al., 2024), which selects tokens through a tournament over multiple random scores; and **K-SemStamp** (Hou et al., 2024b), which steers sentence-level generation toward a watermarked region in semantic embedding space. Details of algorithms are provided in § 6. We also report results for **Vanilla** baseline, the base model without PUPPET fine-tuning, evaluated with the OpenAI detector as a lower-bound reference. We use MarkLLM (Pan et al., 2024) for all watermarking methods, with default hyperparameters.

4 RESULTS

Table 1 reports detection (AUROC) and task performance (ROUGE-L, Judge) of Llama-3 fine-tuned with PUPPET on three benchmarks, and Figure 2 visualizes token-level behavior on an ELI5 example. We draw two key observations.

PUPPET improves both detection and task performance. Compared to Vanilla, PUPPET improves AUROC by up to $+12.3$ pts (IELTS: $87.0 \rightarrow 99.3$), ROUGE-L by up to $+2.9$ pts (ELI5: $22.9 \rightarrow 25.8$), and LLM-as-a-Judge score by up to $+0.42$ pts (IELTS: $6.77 \rightarrow 7.19$). This confirms that PUPPET jointly optimizes both objectives.

PUPPET offers a better trade-off than watermarking. PUPPET surpasses watermarked models in task performance by up to 6% (IELTS: SynthID 6.76 vs. PUPPET 7.19), while matching or exceeding their detection. For example, PUPPET outperforms KGW, the strongest watermarking

[2] https://github.com/chillestt/Automated-IELTS-essay-evaluation

Table 1: **Detection and task performance of PUPPET and watermarking baselines.** Bold: best per column. †/‡: significantly better/worse task performance than Vanilla (paired t-test, α=0.05).

	ELI5		Multi-News		IELTS	
Method	**AUROC**	**ROUGE-L**	**AUROC**	**ROUGE-L**	**AUROC**	**Judge**
WATERMARKING BASELINES						
KGW	99.6	23.5†	99.7	26.6	**99.9**	6.75
SynthID	93.5	23.5†	96.5	26.5	99.3	6.76
Unigram	93.1	23.2	94.5	26.7	98.8	6.73
EXP-Gumbel	98.7	22.6	94.4	26.5	98.2	4.81‡
K-SemStamp	96.2	22.6	96.7	25.5‡	96.5	6.73
OURS						
Vanilla	97.1	22.9	92.8	26.5	87.0	6.77
PUPPET	**100.0**	**25.8**†	**99.8**	**28.0**†	99.3	**7.19**†

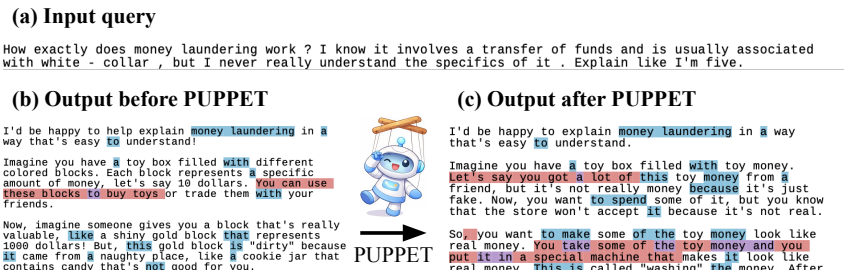

Figure 2: **Spans that drive detection (red) and task performance (blue) on an ELI5 example.** Red: spans the OpenAI Detector relies on (SHAP); blue: spans matching the reference (LCS, as in ROUGE-L); (b) before and (c) after PUPPET.

baseline on detection, on 2 out of 3 benchmarks. KGW can also reach an AUROC of 100.0 by increasing its watermark strength δ, but task performance falls even below Vanilla (detailed in § 5.3).

Figure 2 shows the learned behavior. We highlight two types of spans: red spans that drive detectability, identified by SHapley Additive exPlanations (SHAP; (Lundberg & Lee, 2017))—a token-level attribution method that quantifies each span's contribution to the OpenAI Detector's output—, and blue spans that drive task performance, identified by Longest Common Subsequence (LCS) with the reference, reflecting ROUGE-L overlap. After PUPPET, both coverages increase (red by 1.8%, blue by 13%), giving qualitative evidence that DPO-based PUPPET leads the base LLM to acquire features benefiting both objectives, as intended.

Note that the task performance degradation of watermarked models reported in prior work is only moderate under our experimental setting, which uses MarkLLM's default watermarking parameters. However, strengthening the watermarking signal does lead to a more pronounced degradation (see § 5.3). In either case, PUPPET's contribution remains valid even in the absence of such degradation: PUPPET actively improves task performance, unlike watermarking methods.

5 ANALYSIS

We conduct six analyses to further characterize PUPPET: (1) *cross-task generalizability*—whether the detection gains transfer to out-of-domain (OOD) tasks, (2) *robustness*—whether PUPPET holds up across the different choices of base LLM, detector, and paraphrasing attacks, (3) *high-stakes settings*—whether PUPPET maintains high TPRs at low FPRs, (4) *efficiency*—whether PUPPET can be applied with modest data resources, (5) *reward ablation*—whether both reward components are necessary for jointly optimizing both objectives, and (6) *model attribution*—whether PUPPET enables distinguishing its outputs from those of other models.

Table 2: **Cross-task transfer (Llama-3).** Rows: training task; columns: evaluation task; boxes: out-of-domain cells. Bold: best per column. †/‡: significantly better/worse task performance than Vanilla (paired t-test, α=0.05).

	ELI5		MN		IELTS	
Train \ Eval	**AUC**	**R-L**	**AUC**	**R-L**	**AUC**	**Jdg.**
Vanilla	97.1	22.9	92.8	26.5	87.0	6.77
ELI5	**100.0**	**25.8**†	97.3	27.6†	94.9	6.64
MN	99.8	24.5†	**99.8**	**28.0**†	97.5	6.44
IELTS	99.6	24.2†	97.5	27.0†	**99.3**	**7.19**†

Table 3: **Robustness to the base LLM (Qwen3-4B/8B/14B, in domain).** Bold: best per model and column. †/‡: significantly better/worse task performance than Vanilla (paired t-test, α=0.05).

		ELI5		MN		IELTS	
Model	**Method**	**AUC**	**R-L**	**AUC**	**R-L**	**AUC**	**Jdg.**
Qwen3-4B	Vanilla	98.2	24.5	88.0	24.9	79.9	7.50
	PUPPET	**100.0**	**25.6**†	**95.4**	**26.7**†	**96.0**	**7.53**
Qwen3-8B	Vanilla	93.7	24.5	83.3	24.0	66.6	7.63
	PUPPET	**99.9**	**26.6**†	**96.7**	**26.4**†	**85.6**	**7.65**
Qwen3-14B	Vanilla	86.4	24.4	76.4	24.0	65.1	**7.86**
	PUPPET	**99.6**	**26.0**†	**99.1**	**28.3**†	**93.6**	7.82

5.1 Generalizability to Out-of-Domain Tasks

To assess whether PUPPET's detection gains transfer beyond the tasks used during training, we evaluate the fine-tuned models on unseen tasks. Table 2 reports the OOD results (off-diagonal) alongside in-domain (ID) results (diagonal).

The results show that, even when evaluated on OOD tasks, PUPPET consistently improves detection performance relative to the Vanilla baseline without degrading task performance. For example, a model trained only on ELI5 improves AUROC on Multi-News from 92.8 to 97.3 and on IELTS from 87.0 to 94.9. This suggests that the detection-relevant features learned by PUPPET transfer across tasks, rather than overfitting to the ID distribution.

5.2 Robustness

We assess PUPPET's robustness along three axes: (i) the choice of base LLM, (ii) the choice of detector, and (iii) adversarial paraphrasing attacks.

5.2.1 Base LLM Choice

Table 3 reports results for three Qwen3 variants (QwenTeam, 2025) and examines robustness to LLM choice along two sub-axes: model family and model size.

Model family — Qwen3-8B is comparable in model size to Llama-3-8B-Instruct but belongs to a different model family, allowing us to isolate the effect of model family. Over the Vanilla baseline, PUPPET improves detection on all three benchmarks (by up to +19.0 AUROC on IELTS) while maintaining or improving task performance (by up to 10% relative gain on Multi-News), indicating that its effectiveness is not tied to a single model family.

Model size — We further evaluate Qwen3-4B and 14B to assess sensitivity to model size.Vanilla detectability degrades monotonically with model size: the three-benchmark average AUROC drops from 88.7 (4B) to 81.2 (8B) and 76.0 (14B), confirming that stronger models are intrinsically harder to detect. PUPPET removes this dependence, lifting the average to 97.1, 94.1, and 97.4, respectively. On the task side, PUPPET significantly improves over the Vanilla baseline on ELI5 and Multi-News at all sizes. Even on IELTS, where the lowest Vanilla detectability demands the largest AUROC gains (up to +28.5 pts), PUPPET preserves task performance, with no significant degradation at any size. PUPPET therefore complements, rather than undermines, larger models.

These results confirm that PUPPET's effectiveness is not contingent on any particular model family or model size. Full Qwen3 results, including watermarking baselines and OOD tasks, are in § B.

5.2.2 Detector Choice

We next test detectors other than the OpenAI detector used in our main experiments: we retrain and evaluate PUPPET separately with each of five additional detectors, the **multi-detector** setting. This differs from the **cross-detector** setting in § G, where a model trained using one detector

(e.g., the OpenAI detector) is evaluated using different, unseen detectors. The five detectors cover two families: RoBERTa-based classifiers—FakeSpotAI[3], MAGE (Li et al., 2024), and RADAR (Hu et al., 2023)—and zero-shot detectors—Fast-DetectGPT (Bao et al., 2024) and Binoculars (Hans et al., 2024). Table 4 reports the results.

Across all five detectors, PUPPET broadly improves both objectives. Notably, these detectors already achieve ceiling detection performance on the Vanilla baseline, yet PUPPET further improves it, reaching a perfect AUROC of 100.0 in 10 of the 15 detector–benchmark settings (up from 7 for Vanilla), while simultaneously lifting task performance. This shows that PUPPET's effectiveness is not confined to the OpenAI detector but generalizes across both detector families.

Table 4: **Robustness to the detector.** Each PUPPET model is fine-tuned with the detector in its row. Bold: best per detector and column. †/‡: significantly better/worse task performance than Vanilla (paired t-test, α=0.05).

		ELI5		MN		IELTS	
Detector	**Method**	**AUC**	**R-L**	**AUC**	**R-L**	**AUC**	**Jdg.**
RoBERTa-based classifier							
Fake-SpotAI	Vanilla	100.0	22.9	100.0	**26.5**	99.7	6.77
	PUPPET	**100.0**	**25.2**†	**100.0**	26.1	**99.9**	**7.32**†
MAGE	Vanilla	100.0	22.9	**97.2**	26.5	89.4	6.77
	PUPPET	**100.0**	**24.5**†	95.4	**29.2**†	**94.2**	**7.31**†
RADAR	Vanilla	97.7	22.9	99.3	26.5	94.3	6.77
	PUPPET	**100.0**	**25.1**†	**99.7**	**26.8**	**98.3**	**7.37**†
Zero-shot detector							
Fast-DetectGPT	Vanilla	100.0	22.9	99.0	26.5	100.0	6.77
	PUPPET	**100.0**	**24.0**†	**100.0**	**27.6**	**100.0**	**7.42**†
Binoculars	Vanilla	100.0	22.9	99.9	26.5	100.0	6.77
	PUPPET	**100.0**	**25.3**†	**100.0**	**27.3**†	**100.0**	**7.54**†

5.2.3 Paraphrasing Attacks

We evaluate robustness to paraphrasing attacks with Dipper (Krishna et al., 2023). Dipper is a widely used paraphrasing attack model (Zhao et al., 2024; Rastogi & Pruthi, 2024; Liu et al., 2024) that rewrites text at the surface level while preserving meaning. Figure 3 reports how AUROC changes before and after paraphrasing for Llama-3 models, averaged over the three benchmarks. Full per-task results are provided in § D.1. While all watermarking baselines suffer severe degradation—up to a 49.2-point drop—PUPPET's AUROC decreases by only 1.2 points. Watermarking, by design, relies on surface-level token distributions, which are easily disrupted by paraphrasing. In contrast, PUPPET may enable a model to acquire higher-level features—such as sentence structures and stylistic patterns—that paraphrasing cannot easily remove. This account is an interpretation of our empirical results rather than a claim we verify here. We also evaluate using FakeSpotAI and MAGE and find that the same trend holds (see § D.2).

5.3 Performance in High-Stakes Settings

The discussion so far has relied on AUROC, the threshold-independent metric most commonly reported for watermarking. In deployment, however, the more relevant criterion is the TPR at low FPRs, particularly in high-stakes settings where falsely identifying human-written text as machine-generated can have serious consequences. We therefore report TPR@1% and @0.1% for both the watermarking baselines and PUPPET in Table 5.

On ELI5 and Multi-News, PUPPET matches or exceeds KGW even at these strict operating points, reaching 100.0/99.5 and 96.5/96.5 at @1%/@0.1%, respectively. On IELTS, in contrast, PUPPET falls behind KGW at both TPR@1% (84.5 vs. 99.0) and TPR@0.1% (75.5 vs. 98.5). We attribute this to the extremely low TPRs of Vanilla on this task (10.5 and 5.0).

We note that all watermarking baselines use MarkLLM's default hyperparameters for a fair comparison. For KGW, this means a watermark strength of $\delta = 2$, which yields relatively low @0.1% values of 75.8 and 83.5 on ELI5 and Multi-News, respectively. We additionally ran KGW with $\delta \in \{5, 10\}$ and observed that these values rise to nearly 100%, but at the expense of task performance: ROUGE-L drops from 22.9 to 22.3 and 20.7 on ELI5, and from 26.5 to 25.3 and 22.1 on Multi-News. This illustrates KGW's trade-off between detectability and task performance, whereas PUPPET achieves strong detection performance at low FPRs without requiring such a trade-off.

[3] https://huggingface.co/fakespot-ai/roberta-base-ai-text-detection-v1

Table 5: **TPR at low FPR.** Columns: FPR = 1% and 0.1%. Bold: best per column.

Method	ELI5		MN		IELTS	
	1%	0.1%	1%	0.1%	1%	0.1%
WATERMARKING BASELINES						
KGW	89.8	75.8	95.0	83.5	**99.0**	**98.5**
SynthID	17.3	0.0	69.5	56.5	85.5	68.0
Unigram	37.7	24.2	58.5	40.5	84.0	57.5
EXPGumbel	76.3	54.3	43.0	37.5	69.0	57.5
K-SemStamp	82.7	46.2	79.4	69.3	66.0	46.6
OURS						
Vanilla	82.5	65.0	58.5	56.5	10.5	5.0
PUPPET	**100.0**	**99.5**	**96.5**	**96.5**	84.5	75.5

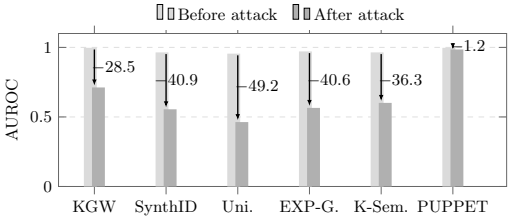

Figure 3: **Robustness to paraphrasing.** AUROC before and after Dipper paraphrasing, averaged over the three benchmarks; numbers: drop in AUROC.

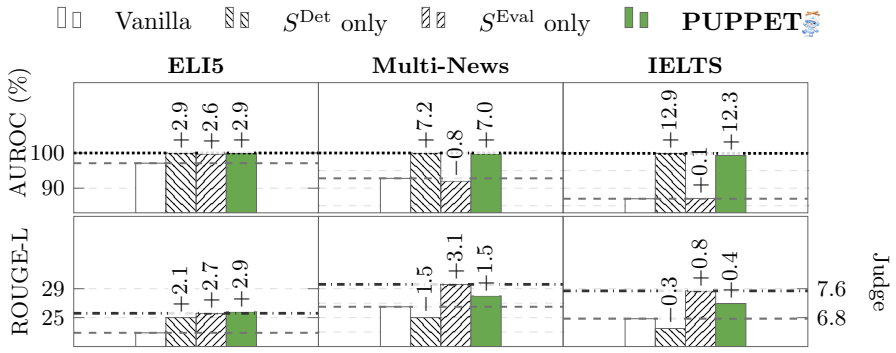

Figure 4: **Reward ablation.** Bar labels: change from Vanilla. Dotted: S^{Det}-only ceiling (AUROC panels); dash-dotted: S^{Eval}-only ceiling (task panels); dashed: Vanilla.

5.4 REWARD COMPOSITION ABLATION

PUPPET uses a composite reward comprising a detector score (S^{Det}) and an evaluator score (S^{Eval}). To assess the necessity of this combined design, we compare PUPPET ($\alpha = 0.5$) against single-reward variants: S^{Det} only ($\alpha = 1$) and S^{Eval} only ($\alpha = 0$), which serve as single-objective upper bounds for detectability and task performance, respectively.

As shown in Figure 4, optimizing detectability alone (S^{Det} only) achieves large AUROC gains over Vanilla (up to $+12.9$ points) but degrades task performance (up to -1.5 ROUGE-L and -0.3 Judge). Conversely, optimizing task performance alone (S^{Eval} only) yields meaningful improvements in it (up to $+3.1$ ROUGE-L and $+0.8$ Judge) but provides no reliable detection gain. PUPPET ($\alpha = 0.5$) jointly optimizes both objectives: it recovers 95.3–100% of the S^{Det}-only detection gain while simultaneously improving task performance over Vanilla. We additionally evaluate "S^{Eval} only + KGW". Detection rises to the level of KGW (99.6, 99.3, and 100.0 on ELI5, Multi-News, and IELTS), but task performance falls to that level as well (23.5, 26.5, and 6.76). This shows that watermarking largely cancels the task gains of S^{Eval} only, whereas joint optimization retains them. Both reward components are thus necessary: neither alone is strong on both objectives.

5.5 TRAINING EFFICIENCY

A strong method has limited practical value if it demands prohibitive data or compute. We therefore examine PUPPET's practicality on both.

Data Figure 5 plots detection and task performance as a function of the number of training samples seen for Llama-3. Across all benchmarks, both detectability and task performance rise sharply within the first few thousand samples and plateau near the level achieved with the full training set.

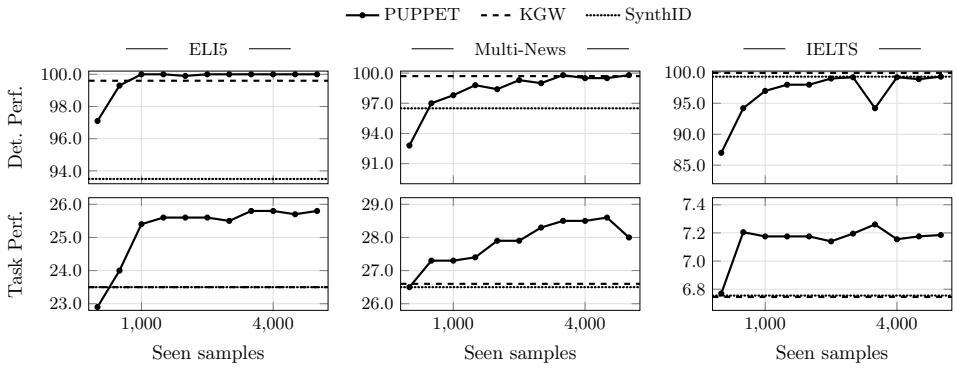

Figure 5: **Training efficiency.** Detection (AUROC) and task performance (ROUGE-L for ELI5 and Multi-News, Judge for IELTS) against the number of training samples seen (500 per dot).

Table 6: **Model attribution.** AUROC of the OpenAI Detector's machine-class likelihood with *Target* as the positive and *Contrast* as the negative class, per task; parentheses: Cohen's d.

Target	Contrast	ELI5	Multi-News	IELTS
Vanilla Llama-3	Vanilla Qwen3	58.99 (0.23)	72.26 (0.82)	77.71 (0.96)
PUPPET Llama-3	Vanilla Qwen3	**94.47** (1.77)	**96.94** (3.18)	**99.14** (4.62)

For example, on ELI5, both performances converge within 1k–2k samples, with AUROC already reaching 100.0 at 1k samples. This handful of DPO pairs represents a remarkably small budget compared to large-scale DPO-based alignment pipelines such as OLMo3 (TeamOlmo, 2025), which train on 150K–200K pairs. The same trend holds for Qwen3 (§ E). Another data-side cost is the number of candidate responses k sampled per query. We therefore vary $k \in \{2, 8, 11\}$ in addition to the $k = 5$ used throughout the paper, using IELTS, where both metrics still have headroom (Table 1). Task performance is 7.02, 7.20, and 7.14, and detection performance is 94.0, 99.2, and 99.3, respectively, compared with 7.19 and 99.3 at $k = 5$. Performance thus improves substantially from $k = 2$ to 5 but saturates thereafter, with $k = 8, 11$ offering essentially no further gain. PUPPET therefore operates effectively with a small candidate pool.

Computational resources Building the training data consists of two steps: (1) sampling candidate responses and (2) scoring them, which take roughly 20 min and 20 min on ELI5, 3.5 h and 20 min on Multi-News, and 40 min and 9 h on IELTS, respectively. Sampling dominates on Multi-News due to long summarization inputs, and scoring on IELTS due to the LLM-as-a-Judge evaluator. Each training run then completes in approximately 1–2 hours on a single NVIDIA A6000 (48GB) GPU, so the entire pipeline fits within 12 hours in every setting. Moreover, because PUPPET is based on offline DPO, these costs are paid only once: sampling once per dataset–model pair, and scoring once each for the detector and the evaluator. Reconfiguring PUPPET then reduces to retraining; changing α, for instance, only recombines the cached scores into new preference pairs.

These results confirm that PUPPET is both data-efficient and computationally lightweight.

5.6 PUPPET Potentially Enables Model Attribution

As an exploratory analysis, we investigate the use of PUPPET for model attribution. Since PUPPET raises the machine-generated class likelihood of the target model, a sufficiently high threshold on this likelihood should separate that model from models without PUPPET applied. We evaluate this by treating *Target* model outputs as the positive (machine) class and *Contrast* model outputs as the negative (human) class, computing AUROC from the OpenAI Detector's machine-generated class likelihood.Table 6 shows Llama-3 results.

Attribution accuracy (AUROC) is governed by the gap in machine-generated class likelihood between the two models; we quantify this with Cohen's d, reported alongside AUROC in Table 6. At

the Vanilla baseline, the gap is small on ELI5 ($d = 0.23$), yielding near-chance attribution (AUROC 58.99), and grows on Multi-News and IELTS, supporting only moderate performance (72.26–77.71). After PUPPET, the gap widens substantially (up to $d = 4.62$ on IELTS) and attribution rises markedly to 94.47–99.14, confirming that PUPPET's controlled upward shift creates a reliable threshold. This suggests that a publisher of Llama-3 who applies PUPPET before release could, after release, tell its own model's outputs apart from those of other models.

We treat these findings as preliminary evidence of PUPPET's potential for model attribution. The similar trend holds for Qwen3 (see § H).

6 Related Work

Research on machine text detection can be broadly categorized into two directions: (1) detectors: using discriminative features of machine-generated text, and (2) watermarking: embedding detectable signals into text during generation.

Detectors A standard approach involves training supervised classifiers on labeled datasets of human-written and machine-generated text. RoBERTa (Liu et al., 2019) is the most widely used base model (Wang et al., 2024b; Tulchinskii et al., 2023): the OpenAI Detector (Solaiman et al., 2019) serves as a common benchmark in prior work (Nicks et al., 2024; Kushnareva et al., 2024), MAGE (Li et al., 2024) is trained on diverse in-the-wild domains and generators, RADAR (Hu et al., 2023) is trained adversarially against a paraphraser, and FakeSpotAI is one of ApolloDFT models designed for detecting deepfakes.

A complementary, training-free paradigm exploits statistical properties of the generation process. DetectGPT (Mitchell et al., 2023) exploits the observation that machine-generated text lies near local maxima of a model's log-probability surface, so perturbations lower its log-probability, whereas human-written text does not exhibit this behavior. Fast-DetectGPT (Bao et al., 2024) achieves comparable accuracy at substantially lower cost by replacing perturbation-based estimation with conditional probability curvature sampling. Binoculars (Hans et al., 2024) achieves strong zero-shot detection by computing the ratio of cross-perplexity and perplexity between two related models.

Watermarking Existing methods fall into two groups: logit-based and sampling-based.

Logit-based methods bias the token output distribution at generation time. KGW (Kirchenbauer et al., 2023) partitions the vocabulary into green and red lists conditioned on the preceding token, and increases the logits of green-list tokens by a fixed δ. Unigram (Zhao et al., 2024) simplifies this by using a fixed, context-independent green list, thereby making it more robust to token-substitution attacks. Both methods systematically distort the output distribution, which degrades task performance (Ajith et al., 2024; Fu et al., 2024; Wang et al., 2025a; Fernandez et al., 2023).

Sampling-based methods, in contrast, intervene at the token selection stage without directly modifying logits. EXPGumbel (Aaronson, 2023) resamples tokens via a seeded Gumbel distribution, and SynthID (Dathathri et al., 2024) selects tokens through a tournament procedure over multiple random scores. K-SemStamp (Hou et al., 2024b) is a clustering-based extension of SemStamp (Hou et al., 2024a), which steers sentence-level generation toward watermarked regions in semantic embedding space. These methods are generally less disruptive to the output distribution than logit-based approaches, but task performance degradation can still be observed (Wang et al., 2025a).

7 Conclusion

We presented PUPPET, the DPO-based framework that fine-tunes an LLM to produce text that is both highly detectable and performant on downstream tasks. By such joint optimization, PUPPET matches watermarking methods in detection while surpassing them on downstream tasks. These gains generalize to out-of-domain tasks and prove robust to the choice of base LLM, detector, and paraphrasing attacks, all within moderate data and computational resources. Our exploratory analysis further suggests that PUPPET opens the door to model attribution. We hope this work encourages research into jointly optimizing LLMs across multiple accountability objectives, thereby contributing to more transparent deployment and fair use of language models.

AI USE STATEMENT

We used generative AI tools to assist with drafting, translation, and proofreading of the manuscript, and with writing code. In addition, as part of the proposed method, we used LLMs to generate the candidate responses that form the preference data for DPO (Section 2). All AI-assisted content was reviewed by the authors, who take responsibility for the final content of this work.

ETHICS STATEMENT

Detection approaches, including PUPPET and watermarking, are still in the research stage. Therefore, it is essential to use these techniques with utmost caution under high-stakes settings where incorrectly identifying human-written text as machine-generated text may cause significant harm.

REFERENCES

Scott Aaronson. Simons institute talk on watermarking of large language models, 2023, 2023. URL https://simons.berkeley.edu/talks/scott-aaronson-ut-austin-openai-2023-08-17.

Anirudh Ajith, Sameer Singh, and Danish Pruthi. Downstream trade-offs of a family of text watermarks. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 14039–14053, 2024. URL https://aclanthology.org/2024.findings-emnlp.821/.

Guangsheng Bao, Yanbin Zhao, Zhiyang Teng, Linyi Yang, and Yue Zhang. Fast-detectGPT: Efficient zero-shot detection of machine-generated text via conditional probability curvature. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=Bpcgcr8E8Z.

Canyu Chen and Kai Shu. Can LLM-generated misinformation be detected? In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=ccxD4mtkTU.

Elizabeth Clark, Tal August, Sofia Serrano, Nikita Haduong, Suchin Gururangan, and Noah A. Smith. All that's 'human' is not gold: Evaluating human evaluation of generated text. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 7282–7296. Association for Computational Linguistics, 2021. URL https://aclanthology.org/2021.acl-long.565/.

Sumanth Dathathri, Abigail See, Sumedh Ghaisas, Po-Sen Huang, Rob McAdam, Johannes Welbl, Vandana Bachani, Alex Kaskasoli, Robert Stanforth, Tatiana Matejovicova, et al. Scalable watermarking for identifying large language model outputs. *Nature*, 634(8035):818–823, 2024. URL https://doi.org/10.1038/s41586-024-08025-4.

Alexander Fabbri, Irene Li, Tianwei She, Suyi Li, and Dragomir Radev. Multi-news: A large-scale multi-document summarization dataset and abstractive hierarchical model. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 1074–1084, 2019. URL https://aclanthology.org/P19-1102/.

Angela Fan, Yacine Jernite, Ethan Perez, David Grangier, Jason Weston, and Michael Auli. ELI5: Long form question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 3558–3567, 2019. URL https://aclanthology.org/P19-1346/.

Pierre Fernandez, Antoine Chaffin, Karim Tit, Vivien Chappelier, and Teddy Furon. Three bricks to consolidate watermarks for large language models. In *2023 IEEE International Workshop on Information Forensics and Security (WIFS)*, 2023. URL https://ieeexplore.ieee.org/document/10374576.

Yu Fu, Deyi Xiong, and Yue Dong. Watermarking conditional text generation for AI detection: Unveiling challenges and a semantic-aware watermark remedy. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(16):18003–18011, 2024. URL https://ojs.aaai.org/index.php/AAAI/article/view/29756.

Josh A. Goldstein, Girish Sastry, Micah Musser, Renee DiResta, Matthew Gentzel, and Katerina Sedova. Generative language models and automated influence operations: Emerging threats and potential mitigations, 2023. URL https://arxiv.org/abs/2301.04246.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The Llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Biyang Guo, Xin Zhang, Ziyuan Wang, Minqi Jiang, Jinran Nie, Yuxuan Ding, Jianwei Yue, and Yupeng Wu. How close is ChatGPT to human experts? comparison corpus, evaluation, and detection, 2023. URL https://arxiv.org/abs/2301.07597.

Abhimanyu Hans, Avi Schwarzschild, Valeriia Cherepanova, Hamid Kazemi, Aniruddha Saha, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Spotting LLMs with binoculars: Zero-shot detection of machine-generated text. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=axl3FAkpik.

Abe Hou, Jingyu Zhang, Tianxing He, Yichen Wang, Yung-Sung Chuang, Hongwei Wang, Lingfeng Shen, Benjamin Van Durme, Daniel Khashabi, and Yulia Tsvetkov. SemStamp: A semantic watermark with paraphrastic robustness for text generation. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 4067–4082, 2024a. URL https://aclanthology.org/2024.naacl-long.226/.

Abe Hou, Jingyu Zhang, Yichen Wang, Daniel Khashabi, and Tianxing He. k-SemStamp: A clustering-based semantic watermark for detection of machine-generated text. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 1706–1715, 2024b. URL https://aclanthology.org/2024.findings-acl.98/.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Xiaomeng Hu, Pin-Yu Chen, and Tsung-Yi Ho. RADAR: Robust AI-text detection via adversarial learning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=QGrkbaan79.

Maurice Jakesch, Jeffrey T. Hancock, and Mor Naaman. Human heuristics for AI-generated language are flawed. *Proceedings of the National Academy of Sciences*, 120(11):e2208839120, 2023. URL https://www.pnas.org/doi/abs/10.1073/pnas.2208839120.

John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. A watermark for large language models. In *Proceedings of the 40th International Conference on Machine Learning*, 2023. URL https://icml.cc/virtual/2023/oral/25524.

Kalpesh Krishna, Yixiao Song, Marzena Karpinska, John Frederick Wieting, and Mohit Iyyer. Paraphrasing evades detectors of AI-generated text, but retrieval is an effective defense. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=WbFhFvjjKj.

Laida Kushnareva, Tatiana Gaintseva, Dmitry Abulkhanov, Kristian Kuznetsov, German Magai, Eduard Tulchinskii, Serguei Barannikov, Sergey Nikolenko, and Irina Piontkovskaya. Boundary detection in mixed AI-human texts. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=kzzwTrt04Z.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention, 2023. URL https://arxiv.org/abs/2309.06180.

Moxin Li, Yuantao Zhang, Wenjie Wang, Wentao Shi, Zhuo Liu, Fuli Feng, and Tat-Seng Chua. Self-improvement towards Pareto optimality: Mitigating preference conflicts in multi-objective alignment. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 11010–11031, 2025. URL https://aclanthology.org/2025.findings-acl.574/.

Yafu Li, Qintong Li, Leyang Cui, Wei Bi, Zhilin Wang, Longyue Wang, Linyi Yang, Shuming Shi, and Yue Zhang. MAGE: Machine-generated text detection in the wild. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 36–53, 2024. URL https://aclanthology.org/2024.acl-long.3/.

Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pp. 74–81, 2004. URL https://aclanthology.org/W04-1013/.

Aiwei Liu, Leyi Pan, Xuming Hu, Shiao Meng, and Lijie Wen. A semantic invariant robust watermark for large language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=6p8lpe4MNf.

Yang Liu, Dan Iter, Yichong Xu, and el al. G-eval: NLG evaluation using gpt-4 with better human alignment. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 2511–2522, 2023. URL https://aclanthology.org/2023.emnlp-main.153/.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized BERT pretraining approach, 2019. URL https://arxiv.org/abs/1907.11692.

Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/8a20a8621978632d76c43dfd28b67767-Paper.pdf.

Yu Meng, Mengzhou Xia, and Danqi Chen. SimPO: Simple preference optimization with a reference-free reward. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=3Tzcot1LKb.

Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D Manning, and Chelsea Finn. DetectGPT: Zero-shot machine-generated text detection using probability curvature. In *Proceedings of the 40th International Conference on Machine Learning*, pp. 24950–24962, 2023. URL https://proceedings.mlr.press/v202/mitchell23a.html.

Charlotte Nicks, Eric Mitchell, Rafael Rafailov, Archit Sharma, Christopher D Manning, Chelsea Finn, and Stefano Ermon. Language model detectors are easily optimized against. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=4eJDMjYZZG.

OpenAI. gpt-oss-120b & gpt-oss-20b model card, 2025. URL https://arxiv.org/abs/2508.10925.

Leyi Pan, Aiwei Liu, Zhiwei He, Zitian Gao, Xuandong Zhao, Yijian Lu, Binglin Zhou, Shuliang Liu, Xuming Hu, Lijie Wen, et al. MarkLLM: An open-source toolkit for LLM watermarking. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 61–71, 2024. URL https://aclanthology.org/2024.emnlp-demo.7/.

QwenTeam. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=HPuSIXJaa9.

Saksham Rastogi and Danish Pruthi. Revisiting the robustness of watermarking to paraphrasing attacks. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 18100–18110, 2024. URL https://aclanthology.org/2024.emnlp-main.1005/.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL https://arxiv.org/abs/1707.06347.

Irene Solaiman, Miles Brundage, Jack Clark, Amanda Askell, Ariel Herbert-Voss, Jeff Wu, Alec Radford, Gretchen Krueger, Jong Wook Kim, Sarah Kreps, et al. Release strategies and the social impacts of language models, 2019. URL https://arxiv.org/abs/1908.09203.

TeamOlmo. Olmo 3, 2025. URL https://arxiv.org/abs/2512.13961.

Shangqing Tu, Yuliang Sun, Yushi Bai, Jifan Yu, Lei Hou, and Juanzi Li. WaterBench: Towards holistic evaluation of watermarks for large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1517–1542, 2024. URL https://aclanthology.org/2024.acl-long.83/.

Eduard Tulchinskii, Kristian Kuznetsov, Kushnareva Laida, Daniil Cherniavskii, Sergey Nikolenko, Evgeny Burnaev, Serguei Barannikov, and Irina Piontkovskaya. Intrinsic dimension estimation for robust detection of AI-generated texts. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=8uOZ0kNji6.

Yidan Wang, Yubing Ren, Yanan Cao, and Binxing Fang. From trade-off to synergy: A versatile symbiotic watermarking framework for large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 10306–10322, 2025a. URL https://aclanthology.org/2025.acl-long.509/.

Yupei Wang, Renfen Hu, and Zhe Zhao. Beyond agreement: Diagnosing the rationale alignment of automated essay scoring methods based on linguistically-informed counterfactuals. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 8906–8925, 2024a. URL https://aclanthology.org/2024.findings-emnlp.520/.

Yuxia Wang, Jonibek Mansurov, Petar Ivanov, Jinyan Su, Artem Shelmanov, Akim Tsvigun, Osama Mohammed Afzal, Tarek Mahmoud, Giovanni Puccetti, and Thomas Arnold. SemEval-2024 task 8: Multidomain, multimodel and multilingual machine-generated text detection. In *Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024)*, pp. 2057–2079, 2024b. URL https://aclanthology.org/2024.semeval-1.279/.

Yuxia Wang, Rui Xing, Jonibek Mansurov, Giovanni Puccetti, Zhuohan Xie, Minh Ngoc Ta, Jiahui Geng, Jinyan Su, Mervat Abassy, Saad El Dine Ahmed, et al. Is human-like text liked by humans? multilingual human detection and preference against AI, 2025b. URL https://arxiv.org/abs/2502.11614.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. In *Advances in Neural Information Processing Systems*, pp. 5824–5836, 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/3fe78a8acf5fda99de95303940a2420c-Paper.pdf.

Weizhe Yuan, Graham Neubig, and Pengfei Liu. BARTScore: Evaluating generated text as text generation. In *Advances in Neural Information Processing Systems*, volume 34, pp. 27263–27277, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/e4d2b6e6fdeca3e60e0f1a62fee3d9dd-Abstract.html.

Qiusi Zhan, Angeline Budiman-Chan, Abdelrahman Zayed, Xingzhi Guo, Daniel Kang, and Joo-Kyung Kim. SafeSearch: Do not trade safety for utility in LLM search agents. In *Findings of the Association for Computational Linguistics: EACL 2026*, pp. 2800–2815, 2026. URL https://aclanthology.org/2026.findings-eacl.146/.

Xuandong Zhao, Prabhanjan Vijendra Ananth, Lei Li, and Yu-Xiang Wang. Provable robust watermarking for AI-generated text. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=SsmT8aO45L.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023. URL https://openreview.net/forum?id=uccHPGDlao.

Table 7: **BARTScore for Llama-3 (higher is better).** Bold: best per column.

Method	ELI5	Multi-News
KGW	−3.66	−2.96
Vanilla	−3.72	−2.98
PUPPET	**−3.54**	**−2.86**

Table 8: **Training hyperparameters changed from the TRL defaults.**

Hyperparameter	Value
Learning rate	1×10^{-4}
Epochs	1
Batch size (per device)	ELI5: 20; MN: 5; IELTS: 10
Data seed	42
Max sequence length	ELI5: 800; MN / IELTS: 8,192
Max completion length	ELI5: 300; MN / IELTS: 512

A Implementation Details

A.1 Generation Settings

All text generation—both for constructing DPO preference data and for evaluation—uses vLLM (Kwon et al., 2023) (v0.11.0). To evaluate post-trained models correctly, we apply chat templates. All parameters related to generation use defaults, except for `n` (number of responses per prompt; see § 3) and `enable_thinking` (`False` for all Qwen3 experiments, to exclude any effects of thinking).

A.2 Training Setup

We fine-tune with the `DPOTrainer` in TRL[4], using LoRA for memory efficiency. Table 8 lists the key hyperparameters; values not shown use defaults. We adopt a single fixed configuration without systematic search except for the ablation studies of α and k in § 5.

A.3 Evaluation Setup

A.3.1 ROUGE-L

ROUGE-L for ELI5 and Multi-News follows the WaterBench (Tu et al., 2024) implementation, where we compute the ROUGE-L score for each response and adopt the best score per sample. Since ROUGE-L captures only surface-level lexical overlap, we additionally re-evaluate Llama-3 with BARTScore (Yuan et al., 2021), a model-based semantic metric, and confirm that PUPPET remains the best on both tasks (Table 7).

A.3.2 LLM-as-a-Judge Evaluation

For IELTS, we use an LLM-as-a-Judge score on a $[0, 9]$ IELTS band scale as a task performance metric. We use GPT-OSS-20B as the judge with temperature 0.0 and `max_tokens` $= 4096$. The scoring prompt is adapted from Wang et al. (2024a).

Probability-weighted scoring. — Rather than parsing the generated integer directly, we compute the final score as a probability-weighted expected value over the ten score labels $\{0, 1, \ldots, 9\}$, following Liu et al. (2023):

$$\hat{s} = \sum_{v=0}^{9} v \cdot P(v), \qquad P(v) = \frac{\exp(\ell_v)}{\sum_{v'=0}^{9} \exp(\ell_{v'})},$$

where ℓ_v is the log-probability of score token v as assigned by the judge model. This yields a continuous score in $[0, 9]$ and, crucially, produces sufficient variance across the $k = 5$ preference candidates—a necessary condition for effective DPO training (detailed at the ablation study of k in § 5.5).

[4] https://github.com/huggingface/trl

Table 9: **Data sources and split sizes.**

Task	Split	Source	Samples
ELI5	Train	Hello-SimpleAI/HC3, reddit_eli5	5,000
	Eval	THU-KEG/WaterBench (2-1_longform_qa)	200
Multi-News	Train	alexfabbri/multi_news	5,000
	Eval	THU-KEG/WaterBench (4-1_multi_news)	200
IELTS	Train	chillies/IELTS-writing-task-2-evaluation	5,000
	Eval	chillies/IELTS-writing-task-2-evaluation	200

Table 10: **In-domain results for Qwen3-8B.** Same layout as Table 1. †/‡: significantly better/worse task performance than Vanilla (paired t-test, α=0.05).

Qwen3-8B	ELI5		Multi-News		IELTS	
Method	**AUC**	**R-L**	**AUC**	**R-L**	**AUC**	**Jdg.**
WATERMARKING BASELINES						
KGW	99.4	24.5	**99.3**	24.1	**99.9**	**7.78**†
SynthID	94.5	24.6	98.0	24.2	98.9	7.67
Unigram	91.1	24.1‡	97.0	24.3	99.1	7.68
EXPGumbel	98.8	23.7‡	96.3	23.7	97.8	4.89‡
K-SemStamp	98.6	24.7	98.5	23.4‡	98.5	7.67
OURS						
Vanilla	93.7	24.5	83.3	24.0	66.6	7.63
PUPPET	**99.9**	**26.6**†	96.7	**26.4**†	85.6	7.65

Table 11: **Cross-task transfer for Qwen3-8B.** Same layout as Table 2. †/‡: significantly better/worse task performance than Vanilla (paired t-test, α=0.05).

Qwen3-8B	ELI5		Multi-News		IELTS	
Train	**AUC**	**R-L**	**AUC**	**R-L**	**AUC**	**Jdg.**
Vanilla	93.7	24.5	83.3	24.0	66.6	7.62
ELI5	**99.9**	**26.6**†	90.1	24.4	82.1	**7.65**
Multi-News	98.6	25.4†	**96.7**	**26.4**†	**87.4**	7.52
IELTS	98.5	25.3†	89.0	24.6†	85.6	**7.65**

Handling the reasoning model. Because GPT-OSS-20B is a reasoning model, it generates a thinking trajectory before producing the final score token. To compute token-level log-probabilities conditioned on that reasoning, we proceed in two steps: (1) we first generate the full thinking trajectory by sampling with a stop sequence that marks the end of the thinking phase; (2) we then prepend the thinking trajectory and the prefix "`Score: `" to each candidate score token and extract the corresponding prompt log-probability via vLLM. This ensures that ℓ_v reflects the model's judgment after reasoning rather than being estimated unconditionally. Step (1) may fail to complete the thinking trajectory within the token limit (`max_new_tokens`). We set the maximum number of retries to 20 and exclude samples whose thinking trajectory fails to terminate.[5]

A.4 DATASET

The training and evaluation datasets are summarized in Table 9. We use 42 as the random seed for all data sampling.

A.5 WATERMARK BASELINE SETUP

We use MarkLLM (Pan et al., 2024) to generate all watermarked texts (KGW, Unigram, EXPGumbel, SynthID, K-SemStamp), with the only modification being the application of chat templates for better evaluation of post-trained models. We adopt the default configurations bundled in MarkLLM.

Table 12: **In-domain results for Qwen3-4B.** Same layout as Table 1. †/‡: significantly better/worse task performance than Vanilla (paired t-test, α=0.05).

Qwen3-4B	ELI5		Multi-News		IELTS	
Method	**AUC**	**R-L**	**AUC**	**R-L**	**AUC**	**Jdg.**
WATERMARKING BASELINES						
KGW	96.2	24.3	**96.4**	25.2	**99.5**	7.57
SynthID	86.8	24.5	93.5	25.1	97.3	7.60
Unigram	73.5	24.3	85.3	25.4†	97.0	**7.63**†
EXPGumbel	88.4	23.8‡	87.9	25.4	95.9	4.86‡
K-SemStamp	96.1	24.3	95.4	24.2‡	98.2	7.57
OURS						
Vanilla	98.2	24.5	88.0	24.9	79.9	7.50
PUPPET	**100.0**	**25.6**†	95.4	**26.7**†	96.0	7.53

Table 13: **Cross-task transfer for Qwen3-4B.** Same layout as Table 2. †/‡: significantly better/worse task performance than Vanilla (paired t-test, α=0.05).

Qwen3-4B	ELI5		Multi-News		IELTS	
Train	**AUC**	**R-L**	**AUC**	**R-L**	**AUC**	**Jdg.**
Vanilla	98.2	24.5	88.0	24.9	79.9	7.50
ELI5	**100.0**	**25.6**†	94.4	25.6†	89.7	**7.39**
Multi-News	99.0	25.2†	**95.4**	**26.7**†	89.2	7.45
IELTS	99.8	24.8†	95.6	25.6†	**96.0**	**7.53**

A.6 SPAN VISUALIZATION

The span visualization with Llama-3 on ELI5 (Figure 2) is produced using the Python `shap` library[6], operating at the granularity of the model's tokenizer.

Detection-relevant spans (red). We compute SHAP values for the OpenAI Detector with respect to class 0 (Machine). Only the positive component of each token's SHAP value is retained (`mode = "pos"`), capturing tokens that actively push the prediction toward the Machine class. To normalize scores comparably across conditions, we compute a shared upper bound $c_{\max}$ as the 99th percentile of the pooled positive SHAP scores from both the Vanilla and PUPPET outputs, ensuring that the color scale is directly comparable between the two conditions. Tokens whose normalized score meets or exceeds the threshold $\tau = 0.4$ are grouped into contiguous spans and highlighted in red.

Task-relevant spans (blue). For ROUGE-L, task-relevant spans are identified via the longest common subsequence (LCS) between the generated text and the reference answer, computed at the word level (whitespace tokenization). Matched word spans are mapped back to character offsets and highlighted in blue. Extending this visualization to LLM-as-a-Judge spans proved computationally prohibitive (>1 week), and we therefore report only ROUGE-L-based spans.

A.7 PARAPHRASING ATTACK SETUP

To evaluate robustness to adversarial paraphrasing, we apply Dipper (Krishna et al., 2023) to all model outputs before running detection. All experiments use identical Dipper settings across benchmarks and base models. We set `lex_diversity` and `order_diversity` to 60, following the strongest parameter values reported by Krishna et al. (2023). Other hyperparameters use defaults.

B FULL RESULTS FOR OTHER BASE LLMS

This appendix provides full per-task results for all Qwen3 models evaluated in § 5.2. organized by the robustness axis under investigation: § B.1 addresses robustness to model family (Qwen3-8B vs. Llama-3-8B-Instruct), and § B.2 addresses robustness to model size (Qwen3-4B, 8B, and 14B).

[5]LLM-as-a-Judge failures were negligible: 3 out of 25,000 judgments (5,000 samples × 5 responses) for Llama-3 and 1 out of 25,000 for Qwen3. The corresponding samples were excluded from preference pair construction.

[6]https://github.com/maciejskorski/shap

Table 14: **In-domain results for Qwen3-14B.** Same layout as Table 1. †/‡: significantly better/worse task performance than Vanilla (paired t-test, α=0.05).

Qwen3-14B	ELI5		Multi-News		IELTS	
Method	**AUC**	**R-L**	**AUC**	**R-L**	**AUC**	**Jdg.**
WATERMARKING BASELINES						
KGW	99.2	24.4	**99.7**	24.3	**99.8**	**7.88**
SynthID	94.5	24.2	98.1	24.4	99.3	7.80
Unigram	90.7	24.0‡	96.8	24.3	98.6	7.78
EXPGumbel	98.5	23.3‡	96.9	24.3	98.1	4.98‡
K-SemStamp	98.0	24.2	99.0	23.2‡	99.0	7.84
OURS						
Vanilla	86.4	24.4	76.4	24.0	65.1	7.86
PUPPET	**99.6**	**26.0**†	99.1	**28.3**†	93.6	7.81

Table 15: **Cross-task transfer for Qwen3-14B.** Same layout as Table 2. †/‡: significantly better/worse task performance than Vanilla (paired t-test, α=0.05).

Qwen3-14B	ELI5		Multi-News		IELTS	
Train	**AUC**	**R-L**	**AUC**	**R-L**	**AUC**	**Jdg.**
Vanilla	86.4	24.4	76.4	24.0	65.1	**7.86**
ELI5	**99.6**	**26.0**†	82.3	24.7†	80.6	7.73‡
Multi-News	98.5	25.4†	**99.1**	**28.3**†	93.4	7.44‡
IELTS	97.2	25.0†	86.6	24.8†	**93.6**	7.81

B.1 MODEL FAMILY: QWEN3-8B

Table 10 shows ID results for Qwen3-8B. On ELI5 and Multi-News, PUPPET markedly improves both detection and task performance compared to Vanilla, consistent with the Llama-3 results in Table 1. On IELTS, however, the AUROC gain is more modest (66.6 → 85.6) compared to Llama-3 (87.0 → 99.3). The lower Vanilla AUROC on IELTS for Qwen3-8B suggests that its essay-style outputs are further from the OpenAI Detector's training distribution, making the detection signal harder to amplify via DPO. Despite this, PUPPET still achieves a meaningful absolute gain, confirming that PUPPET is not inherently limited to a Llama3 family.

Table 11 shows OOD results. As with Llama-3 (Table 2), PUPPET consistently improves detection performance over Vanilla even on tasks not seen during training, confirming that the detection-relevant features learned by PUPPET transfer across tasks regardless of the base model family.

B.2 MODEL SIZE: QWEN3-4B AND 14B

Tables 12 and 13 report results for Qwen3-4B, and Tables 14 and 15 report results for Qwen3-14B.

Detection performance. As observed for Llama-3 in Table 1, PUPPET's detection AUROC is competitive with watermarking baselines on both Qwen3-4B and Qwen3-14B, while consistently surpassing them on task performance.

Task performance. Task performance improves consistently across both sizes. For Qwen3-14B, ROUGE-L improve substantially on both ELI5 and Multi-News (24.4 → 26.0 and 24.1 → 28.3, respectively), while Judge score on IELTS decreases marginally (7.86 → 7.81). However, the degradation is not statistically significant and is small relative to the detection gain (65.1 → 93.6).

Out-of-domain generalization. The OOD results show that cross-task generalization holds for both Qwen3-4B and 14B. PUPPET models fine-tuned on any single task consistently improves detection on the other two tasks over Vanilla.

C OOD RESULTS: FAKESPOTAI AND MAGE

Among the alternative detectors evaluated in § 5.2, FakeSpotAI and MAGE are the two for which PUPPET does not improve detection and task performance simultaneously on Multi-News in domain (Table 4). This appendix examines whether that behavior also carries over to the OOD setting. Table 16 reports the results.

Cross-task generalization remains strong for both detectors and both base models, as observed for the OpenAI detector (Table 2). For FakeSpotAI, every off-diagonal AUROC is at least 99.4 and

Table 16: **Cross-task transfer under FakeSpotAI and MAGE (Llama-3 and Qwen3).** Rows: training task; columns: evaluation task; boxes: out-of-domain cells. †/‡: significantly better/worse task performance than Vanilla (paired t-test, α=0.05).

Detector	Train	ELI5		Multi-News		IELTS	
		AUC	R-L	AUC	R-L	AUC	Jdg.
Llama-3							
Fake-SpotAI	Vanilla	**100.0**	22.9	**100.0**	26.5	99.7	6.77
	ELI5	**100.0**	**25.2**†	**100.0**	**27.1**†	99.8	6.74
	MN	**100.0**	23.9†	**100.0**	26.1	**99.9**	**7.32**
	IELTS	**100.0**	23.7†	**100.0**	26.3	**99.9**	**7.32**†
MAGE	Vanilla	**100.0**	22.9	97.2	26.5	89.4	6.77
	ELI5	**100.0**	**24.5**†	**97.3**	27.2†	90.0	6.66
	MN	**100.0**	23.5†	95.4	**29.2**†	90.1	6.68
	IELTS	**100.0**	22.1‡	96.7	26.0	**94.2**	**7.31**†
Qwen3							
Fake-SpotAI	Vanilla	**100.0**	24.5	**100.0**	24.0	98.9	7.63
	ELI5	**100.0**	**25.7**†	**100.0**	**24.4**†	99.4	**7.67**
	MN	**100.0**	25.0†	**100.0**	24.1	99.4	7.64
	IELTS	**100.0**	24.9†	**100.0**	24.0	**99.6**	7.63
MAGE	Vanilla	**100.0**	24.5	97.5	24.0	91.3	7.63
	ELI5	**100.0**	25.3†	97.2	24.4	91.2	7.65
	MN	**100.0**	24.4	97.4	**25.9**†	91.2	7.63
	IELTS	**100.0**	**25.5**†	**98.1**	24.6†	**95.3**	**7.85**†

never falls below Vanilla. For MAGE, off-diagonal AUROC matches or exceeds Vanilla in 8 of the 12 cases, with the remaining drops at most 0.5 points. The in-domain behavior that motivated this analysis thus does not propagate out of domain: MAGE's -1.8 AUROC drop is confined to the diagonal, where it coincides with its largest ROUGE-L gain (26.5 $\rightarrow$ 29.2), suggesting a mild trade-off between the two objectives.

D Full Paraphrasing Attack Results

This appendix presents full paraphrasing attack results complementing the summary in § 5.2. § D.1 covers results when PUPPET is fine-tuned with the OpenAI Detector, and § D.2 when it is fine-tuned with FakeSpotAI and MAGE.

D.1 Against the OpenAI Detector

Table 17 reports detection and task performance after applying Dipper as a paraphrasing attack, across both models and all benchmarks. Pre-paraphrasing results are reported in Table 1 and Table 2 for Llama-3, and in Table 10 and Table 11 for Qwen3. AUROC drops sharply after paraphrasing for all watermarking baselines (e.g., KGW on ELI5 with Llama-3: 99.6 $\rightarrow$ 70.1), as Dipper's surface-level token substitutions destroy the statistical patterns embedded by watermarking. In contrast, PUPPET's AUROC is largely unaffected, consistently outperforming the Vanilla baseline in both detection and task performance—with ID results again roughly exceeding OOD ones, across both Llama-3 and Qwen3. This pattern suggests that PUPPET captures detection-relevant features that are not easily erased by surface-level rewriting.

Task performance declines after paraphrasing across all methods. This degradation reflects a property of Dipper itself: as a T5-based model, Dipper produces text of lower quality than Llama-3 or Qwen3, causing semantic degradation regardless of the original generation method. Nevertheless, in the ID setting PUPPET still retains higher task performance than Vanilla across all configurations.

Table 17: **Results after the Dipper paraphrasing attack (OpenAI Detector; Llama-3 and Qwen3).** Rows: training task; columns: evaluation task; boxes: out-of-domain cells. Bold: best per column. †/‡: significant improvement/degradation in task performance relative to before the attack (paired t-test, α=0.05).

	ELI5		Multi-News		IELTS	
Method	**AUR**	**R-L**	**AUR**	**R-L**	**AUR**	**Jdg.**
Llama-3						
WATERMARKING BASELINES						
KGW	70.1	23.9†	71.6	23.2‡	72.0	5.45‡
SynthID	55.7	23.6	54.3	23.0‡	56.5	5.41‡
Unigram	32.3	23.3	61.3	23.1‡	45.3	5.26‡
EXPGumbel	65.8	22.4	54.2	21.9‡	49.4	4.84
K-SemStamp	60.6	23.6†	65.1	22.9‡	54.7	5.54‡
OURS						
Vanilla	98.8	23.3†	98.8	23.3‡	92.9	5.46‡
ELI5	**100.0**	**24.9**‡	98.9	23.9‡	95.0	5.62‡
Multi-News	99.4	24.2‡	**99.4**	**24.5**‡	**97.2**	5.56‡
IELTS	99.8	23.7‡	98.8	23.6‡	96.2	**5.97**‡
Qwen3						
WATERMARKING BASELINES						
KGW	59.2	24.4	60.9	22.9‡	66.1	5.94‡
SynthID	55.0	24.4	61.2	22.9‡	50.8	5.91‡
Unigram	59.8	24.2	80.0	23.1‡	80.2	5.80‡
EXPGumbel	60.1	23.2‡	56.9	20.9‡	58.2	5.02†
K-SemStamp	60.7	24.4	65.6	22.5‡	60.7	5.78‡
OURS						
Vanilla	98.6	24.4	98.4	23.1‡	90.3	5.85‡
ELI5	**100.0**	**24.9**‡	98.4	23.1‡	92.5	5.95‡
Multi-News	99.4	24.2‡	98.4	**23.8**‡	94.3	5.83‡
IELTS	99.7	**24.9**‡	**98.7**	23.2‡	**96.5**	**6.10**‡

D.2 AGAINST FAKESPOTAI AND MAGE

Table 18 reports ID detection and task performance after the Dipper paraphrasing attack for PUPPET fine-tuned with FakeSpotAI and MAGE. Pre-paraphrasing results are reported in Table 4 for Llama-3, and in Table 16 for Qwen3.

Unlike the consistent pattern observed with the OpenAI Detector, AUROC changes after paraphrasing vary across detectors. For FakeSpotAI, the largest degradation occurs on Qwen3's IELTS, dropping from 99.6 to 94.4 (-5.2); for MAGE, it is even larger, with Llama-3's Multi-News falling from 95.4 to 84.9 (-10.5). This variability likely reflects whether the text produced by Dipper retains features that each detector relies on: when a detector's signal depends on surface-level patterns easily destroyed by rewriting, detection collapses; when those features are higher-level or incidentally preserved, detection holds. Task performance, in contrast, degrades almost uniformly and significantly across detectors, though PUPPET still matches or exceeds Vanilla in 8 of the 12 settings. These observations suggest that PUPPET learns detection-relevant features that are not merely surface-level and thus survive rewriting, while also showing that post-paraphrasing detection depends on each detector's sensitivity to such perturbations.

E TRAINING EFFICIENCY: QWEN3-8B

Figure 6 shows sample-efficiency curves for Qwen3, complementing the Llama-3 results in Figure 5. Qwen3 follows a pattern similar to Llama-3: both detection and task performance converge within

Table 18: **Results after the Dipper paraphrasing attack with FakeSpotAI and MAGE (in domain; Llama-3 and Qwen3).** †/‡: significant improvement/degradation in task performance relative to before the attack (paired t-test, α=0.05).

		ELI5		Multi-News		IELTS	
Detector	**Method**	**AUC**	**R-L**	**AUC**	**R-L**	**AUC**	**Jdg.**
Llama-3							
Fake-SpotAI	Vanilla	**100.0**	23.3	97.8	**23.3**	95.9	5.46
	PUPPET	**100.0**	**24.9**	**99.9**	23.2‡	**96.1**	**5.93**‡
MAGE	Vanilla	**99.9**	23.3	**86.9**	23.3	94.0	**5.46**
	PUPPET	99.7	**24.3**‡	84.9	**25.2**‡	**95.1**	5.34‡
Qwen3							
Fake-SpotAI	Vanilla	**100.0**	24.4	98.9	**23.1**	**94.5**	5.85
	PUPPET	**100.0**	**25.0**‡	**99.6**	22.7‡	94.4	**6.02**‡
MAGE	Vanilla	**99.9**	24.4	**89.1**	**23.5**	94.5	**5.85**
	PUPPET	99.7	**24.8**‡	88.3	**23.5**‡	**94.6**	5.49‡

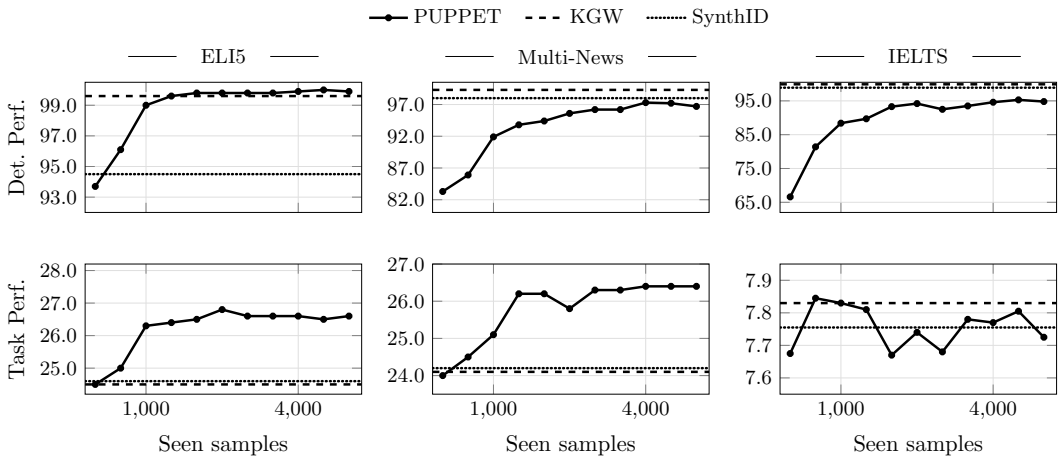

Figure 6: **Training efficiency (Qwen3).** Same layout as Figure 5.

a few thousand DPO training examples on ELI5 and Multi-News. On IELTS, where the Vanilla detection performance is considerably lower (66.6, vs. 93.7 for ELI5 and 83.3 for Multi-News), requiring more samples to approach its ceiling. The task performance curve for Qwen3 on IELTS is notably unstable and fails to converge to a clear plateau within the observed range, which we attribute to the OpenAI Detector's difficulty detecting in this setting.

F Reward Composition Ablation: Qwen3-8B

Figure 7 reports reward composition ablation results for Qwen3, complementing the Llama-3 results in Figure 4.

The overall pattern is consistent with Llama-3: S^{Det} only achieves large AUROC gains at the cost of task performance, S^{Eval} only improves task without lifting detection, and PUPPET improves on both axes. On ELI5 and Multi-News, PUPPET recovers—and in some cases exceeds—the single-objective gains: 84.3–98.4% of the S^{Det}-only detection gain and 88.9–131.3% of the S^{Eval}-only task gain. A notable difference from Llama-3 appears on IELTS, where PUPPET's detection gain recovery drops to 59.4% (vs. 95.3% for Llama-3). This is likely because the Vanilla AUROC is more than 20 points lower than Llama-3's (66.6 vs. 87.0 for Llama-3), so a fixed $\alpha = 0.5$ was insufficient to fully optimize detection performance under this setting.

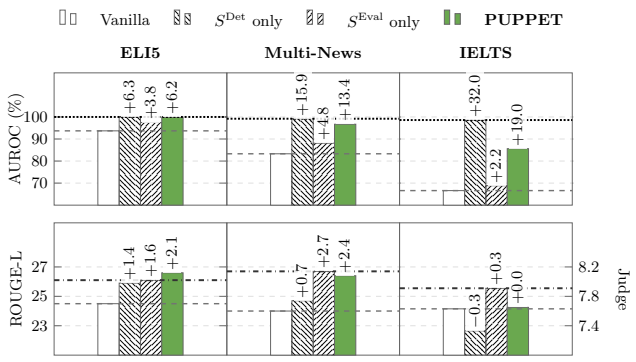

Figure 7: **Reward ablation (Qwen3).** Same layout as Figure 4.

Table 19: **Model attribution from the Qwen3 side.** Same layout as Table 6, with Vanilla Llama-3 as the negative class.

Target	ELI5	Multi-News	IELTS
Vanilla Qwen3	41.01 (−0.23)	27.74 (−0.82)	22.29 (−0.96)
PUPPET Qwen3	**87.98** (1.22)	59.72 (0.31)	**71.09** (0.78)

G Generalizability to Unseen Detectors

We evaluate PUPPET in the **cross-detector** setting: for each task, the PUPPET model fine-tuned with the OpenAI detector is evaluated with the five other detectors of § 5.2.2. Note that, as stated in § 1, PUPPET is designed to improve detectability with respect to a target detector. This setting thus lies outside its intended use. Table 20 reports the results.

Compared with the multi-detector setting (§ 5.2.2), where each detector is also used for fine-tuning, the average change per detector is smaller or equal for all five detectors: for instance (*multi→cross*), $+1.0 \rightarrow -0.7$ for MAGE and $+2.2 \rightarrow +1.6$ for RADAR. Overall, the gains from PUPPET are largely specific to the detector used during fine-tuning.

H Model-Attribution Results for Qwen3

Table 19 reports model-attribution AUROC from the Qwen3 perspective, complementing the Llama-3 results in Table 6.

At the vanilla baseline, the detector achieves AUROC of 41.01 on ELI5 (below chance from this perspective) and 27.74–22.29 on Multi-News and IELTS—the mirror image of the Llama-3 baseline, confirming that the asymmetry is an artifact of the uncontrolled detector rather than a meaningful signal in either direction. Applying PUPPET to Qwen3 substantially raises AUROC against Vanilla Llama-3 on ELI5 (87.98) and IELTS (71.09), though the gain on Multi-News (59.72) is more modest. These results are consistently lower than the PUPPET Llama-3 versus Vanilla Qwen3 counterparts (94.47, 96.94, 99.14; Table 6). We attribute this asymmetry to the baseline machine-class likelihood distributions: Vanilla Llama-3 already occupies a high machine-class likelihood region, whereas Vanilla Qwen3 lies in a lower one (Table 1 for Llama-3 and Table 10 for Qwen3). When PUPPET shifts Qwen3's distribution upward, it reaches a level comparable to Vanilla Llama-3 rather than surpassing it, leaving the two distributions insufficiently separated—most strikingly on Multi-News, where $d = 0.31$, versus $d = 3.18$ for the reverse comparison (PUPPET Llama-3 vs. Vanilla Qwen3), which explains the latter's markedly higher AUROC in Table 6. Overall, these results suggest that PUPPET can also support post-release model tracking by model providers when applied to Qwen3.

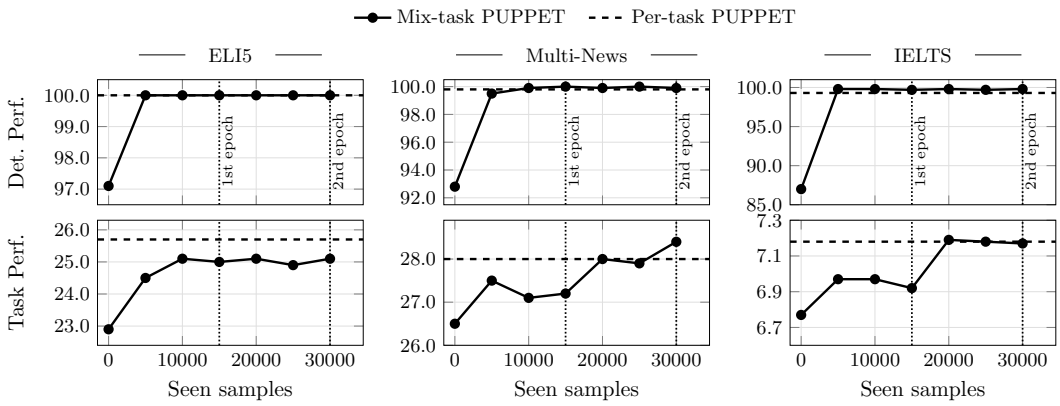

Figure 8: **Multi-task training (Llama-3).** Detection (top) and task performance (bottom) against training samples seen on the mixture of the three datasets (5,000 per dot). Dashed: per-task PUPPET.

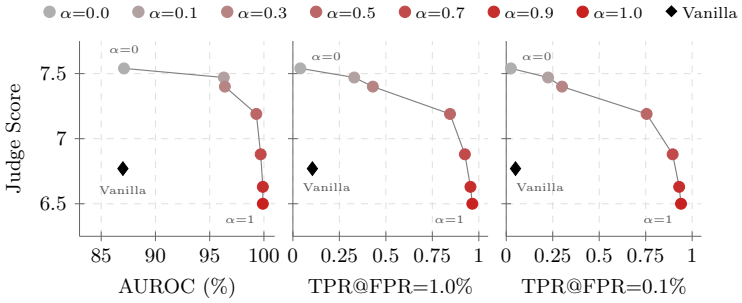

Figure 9: α **ablation on IELTS (Llama-3).** α=0: S^{Eval} only; α=1: S^{Det} only; larger α favors detection.

I Finer-Grained α Ablation on IELTS

Across base LLMs and detectors, Vanilla detection performance on IELTS is consistently lower than on the other benchmarks, which we observe makes joint optimization more challenging (Table 1, Table 4, and Table 5 for Llama-3). We hypothesized that weighting the detection reward might reveal a better operating point, so we extended the $\alpha \in \{0.0, 0.5, 1.0\}$ ablation reported for Llama-3 in Figure 4 with additional runs at $\alpha \in \{0.1, 0.3, 0.7, 0.9\}$. The results are shown in Figure 9. Across all detection metrics, we observe a trade-off against Judge score. For AUROC, this trade-off is relatively mild once $\alpha \geq 0.1$, whereas for TPR it is more pronounced. In summary, PUPPET clearly occupies a more favorable region of it than Vanilla.

J Multi-Task Learning

Throughout this paper, PUPPET is fine-tuned separately on each benchmark. In practice, however, a single model is often expected to handle multiple tasks. While our OOD evaluations already address cross-task generalization, they do not involve directly training on multiple tasks at once. We therefore fine-tune Llama-3 on a shuffled mixture of all three task datasets ($5,000 \times 3 = 15,000$ samples in total) for 2 epochs, with all other hyperparameters identical to those in § A.

Results are shown in Figure 8. Detection performance reaches a level comparable to per-task PUPPET after only $5,000$ samples, equal in size to just one of the three datasets. Task performance, by contrast, is harder to optimize at once: Multi-News and IELTS reach the per-task level only around the $20,000$ samples, $5,000$ samples beyond 1 epoch, and ELI5 still falls short by about 1 point even after the full 2 epochs. We attribute this asymmetry to the nature of the supervision signals: detection performance is supervised by the OpenAI Detector's machine-class likelihood shared

Table 20: **Transfer to unseen detectors (cross-detector setting, Llama-3).** PUPPET is fine-tuned with the OpenAI Detector and evaluated with each detector. Bold: best per detector and task.

Detector	Method	ELI5 AUROC	Multi-News AUROC	IELTS AUROC
RoBERTa-based classifier				
FakeSpotAI	Vanilla	100.0	100.0	99.7
	PUPPET	**100.0**	**100.0**	**99.8**
MAGE	Vanilla	100.0	**97.2**	**89.4**
	PUPPET	**100.0**	96.9	87.7
RADAR	Vanilla	97.7	99.3	94.3
	PUPPET	**99.5**	**99.5**	**97.2**
Zero-shot detector				
Fast-DetectGPT	Vanilla	100.0	99.0	100.0
	PUPPET	**100.0**	**99.8**	**100.0**
Binoculars	Vanilla	100.0	99.9	100.0
	PUPPET	**100.0**	**100.0**	**100.0**

across all tasks, whereas task performance relies on heterogeneous, per-task metrics (ROUGE-L for ELI5/Multi-News vs. LLM-as-a-Judge for IELTS) spanning different domains, whose conflicting optimization directions are known to hinder joint multi-task optimization (Yu et al., 2020; Li et al., 2025).