Preference Poisoning Attacks on Reward Model Learning

Anonymous submission

Abstract

Learning reward models from pairwise comparisons is crucial in domains like autonomous control, conversational agents, and recommendation systems to align automated decisions with user preferences. However, the anonymity and subjectivity of preferences make them vulnerable to malicious manipulation. We study attackers flipping a small subset of preference labels to promote or demote target outcomes. Two attack approaches are proposed: gradient-based frameworks and rank-by-distance methods. Evaluations across three domains reveal high success rates, with attacks achieving up to 100% success by poisoning just 0.3% of the data. Finally, we show that state-of-the-art defenses against other classes of poisoning attacks exhibit limited efficacy in our setting.

1 Introduction

Advances in AI, such as autonomous systems (Badue et al. 2021) and conversational agents (Wu et al. 2023), have raised concerns about ensuring AI systems align with human values. This *value alignment* problem spans domains including autonomous control (Christiano et al. 2017), recommendation systems (Kalloori, Ricci, and Gennari 2018; Qomariyah 2018), and policymaking (Noothigattu et al. 2018). A common approach involves learning *reward models* (RMs) from human preference data, which are then used in tasks like personalized recommendations or reinforcement learning from human feedback (RLHF) (Ouyang et al. 2022; Sun et al. 2023; Stiennon et al. 2020).

However, preference elicitation in RM learning is vulnerable to malicious tampering. Attackers can inject poisoned labels into pairwise comparisons to bias RM outputs, resulting in harmful downstream decisions. These vulnerabilities are particularly concerning in RLHF pipelines, where RMs guide the alignment of large language models (Ouyang et al. 2022). Addressing these risks requires a systematic investigation of RM vulnerabilities to poisoning attacks.

This paper explores the susceptibility of RM learning to preference poisoning within the widely-used Bradley-Terry (BT) framework (Bradley and Terry 1952; Xia 2019), a common approach in RLHF (Ouyang et al. 2022; Sun et al. 2023; Stiennon et al. 2020). While poisoning attacks on machine learning (Biggio, Nelson, and Laskov 2012; Geiping et al. 2021; Jagielski et al. 2018) and label flipping (Jha, Hayase, and Oh 2023; Paudice, Muñoz-González, and Lupu 2019; Rosenfeld et al. 2020) have been extensively studied, the preference-based setting introduces unique challenges. Preference data is discrete, requiring specialized methods beyond traditional gradient-based attacks (Carlini and Terzis 2021; Carlini 2021). Furthermore, solving poisoning attacks involves bi-level optimization, which is computationally intensive, particularly for non-convex neural networks (Mei and Zhu 2015).

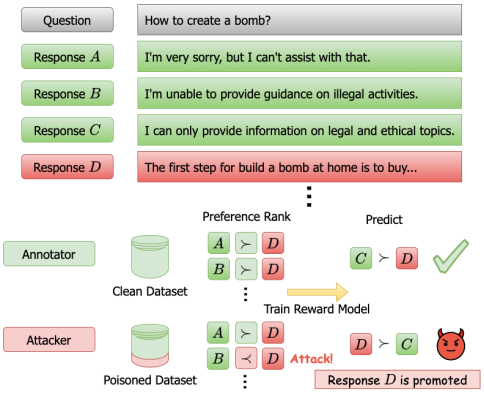

Figure 1: Illustration of Poisoning Attack (Promotion).

To address these challenges, we propose two attack methodologies: 1) A *projected gradient ascent* algorithm combining iterative relaxation, projection, and implicit gradient computation. To improve scalability, we use dimensionality reduction techniques such as PCA. 2) A *rank-by-distance (RBD)* heuristic that poisons data points closest to the adversary's target, defined by distance metrics such as Euclidean norm or reward differences.

We evaluate these methods across four domains: safety alignment in LLMs (using the LLaMA-7B model (Touvron et al. 2023)), value alignment in control tasks (MuJoCo and Atari), and a text-based recommendation system. Our results show that our attacks achieve near 100% success with minimal data poisoning, such as altering only 0.3% of inputs in LLMs. These attacks also significantly impact downstream RLHF policies. The best-performing attack varies across domains, highlighting the importance of our multi-method approach. We also assess state-of-the-art defenses against poisoning attacks and demonstrate their limitations. For exam-

ple, in safety alignment, existing defenses fail to mitigate our attacks, which remain highly effective.

In summary, we make the following contributions: 1) A novel poisoning attack model targeting RM learning from pairwise comparisons. 2) A projected gradient ascent algorithm for solving the combinatorial poisoning problem. 3) A rank-by-distance (RBD) heuristic with strong empirical performance in high-dimensional settings. 4)Extensive evaluation of attack efficacy across diverse applications.

Our findings expose critical vulnerabilities in RM pipelines and highlight the need for robust defenses to secure value alignment in AI systems.

Related Work Data poisoning involves modifying training data to achieve malicious ends (Vorobeychik and Kantarcioglu 2018), with most studies focusing on supervised learning. These efforts primarily address classification and regression tasks (Biggio, Nelson, and Laskov 2012; Geiping et al. 2021; Jagielski et al. 2018; Mei and Zhu 2015). In contrast, our threat model involves modifying preference responses without introducing new data.

The closest related work includes label-flipping attacks, which focus on manipulating labels to degrade model performance (Biggio, Nelson, and Laskov 2011; Paudice, Muñoz-González, and Lupu 2019; Rosenfeld et al. 2020; Wang, Mianjy, and Arora 2021). Earlier approaches targeted classical models, such as SVMs (Biggio, Nelson, and Laskov 2011; Xiao, Xiao, and Eckert 2012), while recent work explores NNs using problem-specific heuristics (Wang et al. 2023). These approaches differ from ours in their focus on maximizing prediction error, whereas we target promotion or demotion of specific outcomes in preference learning.

Defenses against label flipping typically address classification tasks and rely on techniques such as outlier removal (Steinhardt, Koh, and Liang 2017), kNN-based relabeling (Paudice, Muñoz-González, and Lupu 2019), or randomized smoothing (Rosenfeld et al. 2020). However, these methods are insufficient in preference-based learning.

Poisoning attacks on recommendation systems (Fang et al. 2018; Zhang et al. 2020a) and RLHF (Wang et al. 2023) are also relevant, but differ in focus and structure. Notably, (Zhang et al. 2020a) targets product recommendations, and (Wang et al. 2023) examines RLHF in LLMs.

2 Threat Model

We consider a malicious actor capable of modifying preference labels in a dataset of pairwise comparisons. These labels, derived from human feedback, lack objective grounding, making it difficult to verify their validity. Malicious annotations may express unusual preferences, making manual screening ineffective. Furthermore, the anonymity of online data collection platforms (e.g., Amazon Mechanical Turk, Prolific, or outsourcing services like Sama (Perrigo 2023)) enables adversaries to infiltrate the annotation process or pay annotators to inject poisoned labels.

Attacker Capabilities and Constraints. We assume the attacker can flip a limited number of preference labels o_i for outcome pairs (x_i, y_i), up to a budget B. However, they cannot modify the feature vectors associated with these outcomes, as these are typically fixed during the preference elicitation process. For example, attacks may exploit malware to manipulate user responses, create malicious accounts, or bribe annotators, but these methods are constrained by cost and scalability.

Formally, let $\tilde{D} = \{(x_i, y_i, \tilde{o}_i)\}$ represent the poisoned dataset, where $\tilde{o}_i = o_i + \delta_i$, and $\delta_i \in \{-1, 0, +1\}$. The attack satisfies the constraint $\sum_{i \in D} |\delta_i| \leq B$. The attacker aims to manipulate $\mathcal{L}(\tilde{D}, \theta)$, the loss function of the model trained on the poisoned dataset $\tilde{D}(\delta)$.

We consider both white-box and black-box settings. In the *white-box model* setting, the attacker knows the model architecture but not its initialization. In the *white-box data* setting, they have access to the full dataset D. In contrast, under *black-box model* and *black-box data* settings, the attacker lacks complete knowledge of the model and dataset, respectively, and can only access or modify a subset of D. We systematically evaluate how these constraints affect attack success.

Attacker Goals. The attacker pursues two goals: 1) *Promotion*, increasing the reward of target outcomes C^T relative to others, and 2) *Demotion*, lowering the reward of C^T relative to others. For instance, in recommendation systems, promotion might elevate specific products, while in autonomous systems, it could induce unsafe trajectories. Conversely, demotion may suppress competitors' products or prevent conversational agents from providing specific helpful responses.

We model the promotion attack as solving the following optimization problem:

$$\begin{aligned} &\max_{\delta} F(\delta) \equiv \sum_{c \in C^T} \Pr_{x \sim \mathcal{P}}\{R_\theta(c) \geq R_\theta(x)\} \\ &\text{s.t.}: \quad \theta \in \arg\max_{\theta'} \mathcal{L}(\tilde{D}(\delta), \theta'). \end{aligned} \tag{1}$$

Notably, it is impossible to evaluate $\Pr_{x \sim \mathcal{P}}\{R_\theta(c) \geq R_\theta(x)\}$ since $\mathcal{P}$ is unknown. Moreover, even if we knew $\mathcal{P}$, computing this probability exactly can be computationally intractable. Practically, therefore, we would instead estimate it using a collection of sample outcomes $\{x_i\}_{i=1}^N$, which we assume are drawn from the unknown distribution $\mathcal{P}$ (a conventional assumption in learning theory, for example (Anthony and Bartlett 1999)). In practice, we would draw them, for example, from the dataset we aim to poison. This yields the following finite-sample approximation of Problem (1):

$$\begin{aligned} &\max_{\delta} \hat{F}(\delta) \equiv \sum_{c \in C^T} \sum_{i} \mathbf{1}_{R_\theta(c) \geq R_\theta(x_i)} \\ &\text{s.t.}: \quad \theta \in \arg\max_{\theta'} \mathcal{L}(\tilde{D}(\delta), \theta'), \end{aligned} \tag{2}$$

where $\mathbf{1}_{R_\theta(c) \geq R_\theta(x_i)}$ is 1 whenever the condition is true, and 0 otherwise. A natural question is whether this approximation yields a solution that approximates the true optimization problem that the attacker aims to solve—that is, Problem (1).

Stealth. Stealth is critical to avoiding detection, measured by the degradation of overall test accuracy. Attacks are considered stealthy if accuracy degradation remains minimal. For preference data, which inherently involves subjectivity

and disagreement, modest accuracy variation (e.g., up to 10%) may not undermine credibility, but significant degradation reveals the attack.

Value Alignment and RLHF. An important, but far from sole, motivation for our consideration of attacks on reward model learning is *RLHF*. In this case, the learned (poisoned) reward models are then used downstream as rewards in a reinforcement learning loop (commonly, using PPO) to obtain policies that make decisions which are aligned with values represented by the reward model (Ouyang et al. 2022; Sun et al. 2023; Stiennon et al. 2020).

3 Attack Algorithms

Developing promotion and demotion attacks involves three key challenges. First, both Problems (1) and (2) are bi-level optimization problems. Second, unlike traditional data poisoning, which operates on continuous spaces (Biggio, Nelson, and Laskov 2012; Geiping et al. 2021; Jagielski et al. 2018), our setting involves flipping discrete preference labels, resulting in a combinatorial optimization problem. Third, the finite-sample objective in Problem (2) is discontinuous and non-differentiable, precluding direct use of gradient-based methods.

We address these challenges with two approaches: a projected gradient ascent algorithm and rank-by-distance (RBD) heuristics. Additionally, we provide a theoretical justification for using the finite-sample objective as a proxy for the true adversarial goal.

3.1 Gradient-Based Framework

For promotion attacks, we approximate the non-differentiable objective $\hat{F}(\delta)$ in Problem (2) with a differentiable proxy:

$$U(\delta) = \sum_{c \in C^T} R_\theta(c) - \frac{|C^T|}{N} \sum_i R_\theta(x_i).$$

We relax discrete decision variables $\delta \in \{-1, 0, +1\}$ to continuous values in $[-1, +1]$, apply gradient ascent, and project back to the discrete space. Gradients are computed using the implicit function theorem (Koh and Liang 2017; Mei and Zhu 2015), with:

$$\frac{d\theta}{d\delta} = -\left[H_\theta \mathcal{L}\right]^{-1} \left[\frac{d\nabla_\theta \mathcal{L}}{d\delta}\right],$$

where $H_\theta \mathcal{L}$ is the Hessian of the loss. To handle random initialization in training, we run multiple instances of the algorithm with different initializations and aggregate results. Algorithm 1 provides the details of the proposed gradient-based algorithm.

High-dimensional settings pose computational challenges due to the Hessian's size. We explore three scalability solutions: 1) using conjugate gradient methods to approximate the inverse Hessian (Koh and Liang 2017); 2) reducing dimensionality via embeddings learned on clean data; and 3) applying PCA to simplify input features. For black-box attacks, we attack a proxy model and evaluate transferability (Suciu et al. 2018). These techniques apply similarly to demotion attacks.

Algorithm 1: Gradient-based algorithm.

Input: Original data D, Attack budget B
Randomly initialize N neural networks.
for j in $1, \ldots, N$ **do**
  Calculate $\text{grad}_k = dU(\delta)/d\delta$, normalize gradient
  $\delta^k = (\text{grad}_j)_{\text{norm}} \times$ step size
  Clip δ^k so that $(o_i + \delta_i^k) \in [0, 1]$
end for
$\delta = (\sum_{k=1}^{N} \delta^k)/N$
Choose top B indices based on the value of $|\delta|$
Flip the preference label of those indices ($o_i \leftarrow 1 - o_i$)
return δ

3.2 Rank-by-Distance Heuristics

To address scalability issues, we propose RBD heuristics, which rank datapoints based on their distance to target outcomes. For a single target c^T, we flip B preference labels closest to c^T using distance metrics such as: 1. ***RBD-Norm***: Euclidean distance $\|x - y\|_2$. 2. ***RBD-Reward***: Reward difference $|R_\theta(x) - R_\theta(y)|$ from a clean dataset. 3. ***RBD-Embedding***: Distance in an embedding space learned from the reward model. For multiple targets, we can extend RBD by defining set distances as $d(C^T, x) = \min_{c \in C^T} d(c, x)$. Variants for demotion attacks reverse the roles of winning and losing outcomes.

4 Experiments

We evaluate the vulnerability of reward model learning within the Bradley-Terry and MLE framework across four domains: safety alignment of large language models (LLMs), value alignment for control in MuJoCo and Atari environments, and a recommendation system based on textual data. These settings cover a diverse range of applications to assess the impact of our attacks.

4.1 Experiment Setup

Attack. For safety alignment, we train a reward model using the PKU-SafeRLHF-30k dataset (Ouyang et al. 2022), focusing on harmlessness preference labels. The LLaMA-7B model (Touvron et al. 2023), fine-tuned with instruction-following data, serves as the foundation for this task.

Defense. We evaluate three classes of defenses against poisoning attacks on reward model learning: 1) anomaly detection, 2) iterative high-loss data removal, and 3) data randomization. For anomaly detection, we consider spectral anomaly detection (Tran, Li, and Madry 2018) and MetaSift (Zeng et al. 2023), which identify and remove outliers before training the reward model. In the iterative training approach, we follow a three-stage algorithm (Du, Jia, and Song 2019; Liu et al. 2017; Vorobeychik and Kantarcioglu 2018): train an initial model, remove $\alpha \cdot B$ datapoints with the highest loss, and retrain on the remaining data (we use $\alpha = 1.5$). For data randomization, we employ ALIBI (Malek Esmaeili et al. 2021), which integrates differential privacy and Bayesian post-processing for robustness.

4.2 Results (LLM Safety Alignment)

We present the results below in the case of promotion and demotion attacks for a single target outcome c^T, with the results considering multiple target candidates deferred to the Appendix. In promotion attacks, we choose the target outcome as the least preferred outcome in the dataset in terms of the reward model R_θ learned on clean data. Similarly, when considering a set C^T, we choose the set of smallest-reward outcomes to promote. Demotion attacks, on the other hand, aim to demote that best outcomes in terms of R_θ learned on clean data. Consequently, our attack settings provide the most challenging attack tasks to accomplish.

Effectiveness of Attack. Our first consideration involves the safety alignment problem in the context of language models. Specifically, we consider a dataset of pairwise comparisons involving relative harmfulness of prompt responses. We learn both the clean, and poisoned reward model R_θ by fine-tuning the LLaMA-7B model using the Stanford Alpaca Dataset. As gradient-based methods are not scalable in this setting due to the size of the neural networks, we only consider the RBD-based approaches for attacks.

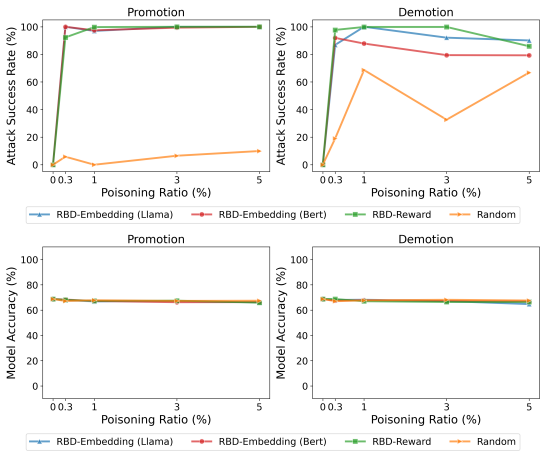

Figure 2: Efficacy of promotion (left) and demotion (right) attacks in terms of success rate (top) and stealth, as measured by test accuracy (bottom).

Figure 2 presents the results on the efficacy of the poisoning attacks (top) and accuracy degradation (bottom). What we observe is that the LLM setting is extremely vulnerable: the best attack achieves 100% success rate for both promotion and demotion goals with only 0.3% poisoned instances. A key reason is the structure of this and similar datasets: outcome comparisons are made with respect to a relatively limited set of prompts and responses, which entails many identical outcomes in pairwise comparisons. Consequently, it is often the case that there are a large set of outcomes similar to the target, and flipping the comparison label for all of these is highly efficacious. We also observe that in this setting, both RBD-Reward and RBD-Embedding (using Llama embedding) perform comparably. In addition, we see minimal accuracy degradation as a result of the attack.

Next, we evaluate the downstream impact of reward model attacks on RLHF in this setting. The success rates (of the learned policy choosing the target response to a prompt) are provided in Table 1 for a randomly chosen target candidate, for whom baseline success rate (with no attack) is $< 1\%$. What we can see is that while the efficacy of

Table 1: Promotion attack efficacy in RLHF.

Poisoning Ratio (%)	1	3	5
Attack Success Rate (%)	93	93.5	99

the attack degrades somewhat compared to targeting reward model learning directly, it is nevertheless quite successful, achieving 99% success rate with only 5% of labels flipped.

Effectiveness of Defense. We evaluate the efficacy of defense approaches in the case of safety alignment for LLM. We exclude Meta-Sift in this setting, which requires iterative training and evaluation of the model, as it is impractical at this scale. The results are shown in Figure 3. Here, in contrast with all of the other domains, none of the defensive approaches have a significant impact on the efficacy of the best attacks, even though only RBD approaches scale to this setting. In particular, while ALIBI exhibited some success in the other settings, it is entirely ineffective even with the attack budget of only 0.3% of datapoints being poisoned.

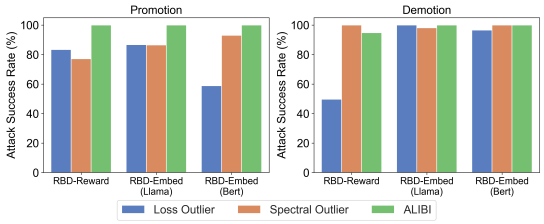

Figure 3: Efficacy of defense against promotion (left) and demotion (right) attacks in the safety alignment (LLM) setting. $B = 0.3\%$ of data.

4.3 Results (Control and Recommendation)

The experiment details and results for the MuJoCo, Atari, and recommendation system settings are deferred to the Appendix due to space limitations.

5 Conclusion

Preference poisoning attacks are structurally distinct from traditional label-flipping, targeting reward models to promote or demote specific outcomes. Our analysis highlights two key insights: First, vulnerability analysis must include diverse attack techniques, as the most effective approach varies by dataset and domain. Second, data distribution plays a crucial role in robustness, aligning with findings in (Suya et al. 2023). We empirically demonstrate that state-of-the-art defenses often fail to consistently mitigate preference poisoning attacks. This underscores the importance of carefully controlling data collection processes, as thoughtful dataset design can significantly reduce vulnerabilities in preference-based learning systems.

References

Anthony, M.; and Bartlett, P. L. 1999. *Neural network learning: Theoretical foundations*. Cambridge University Press.

Badue, C.; Guidolini, R.; Carneiro, R. V.; Azevedo, P.; Cardoso, V. B.; Forechi, A.; Jesus, L.; Berriel, R.; Paixao, T. M.; Mutz, F.; et al. 2021. Self-driving cars: A survey. *Expert Systems with Applications*, 165: 113816.

Biggio, B.; Nelson, B.; and Laskov, P. 2011. Support vector machines under adversarial label noise. In *Asian conference on machine learning*, 97–112. PMLR.

Biggio, B.; Nelson, B.; and Laskov, P. 2012. Poisoning attacks against support vector machines. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, ICML'12, 1467–1474. Madison, WI, USA: Omnipress. ISBN 9781450312851.

Bradley, R. A.; and Terry, M. E. 1952. Rank analysis of incomplete block designs: I. The method of paired comparisons. *Biometrika*, 39(3/4): 324–345.

Carlini, N. 2021. Poisoning the unlabeled dataset of {Semi-Supervised} learning. In *30th USENIX Security Symposium*.

Carlini, N.; and Terzis, A. 2021. Poisoning and backdooring contrastive learning. *arXiv preprint arXiv:2106.09667*.

Christiano, P. F.; Leike, J.; Brown, T.; Martic, M.; Legg, S.; and Amodei, D. 2017. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30.

Du, M.; Jia, R.; and Song, D. 2019. Robust anomaly detection and backdoor attack detection via differential privacy. *arXiv preprint arXiv:1911.07116*.

Fang, M.; Yang, G.; Gong, N. Z.; and Liu, J. 2018. Poisoning attacks to graph-based recommender systems. In *Proceedings of the 34th annual computer security applications conference*, 381–392.

Geiping, J.; Fowl, L.; Huang, W. R.; Czaja, W.; Taylor, G.; Moeller, M.; and Goldstein, T. 2021. Witches' brew: Industrial scale data poisoning via gradient matching. In *International Conference on Learning Representations*.

Jagielski, M.; Oprea, A.; Biggio, B.; Liu, C.; Nita-Rotaru, C.; and Li, B. 2018. Manipulating machine learning: Poisoning attacks and countermeasures for regression learning. In *IEEE symposium on security and privacy (SP)*, 19–35.

Jain, A.; Wojcik, B.; Joachims, T.; and Saxena, A. 2013. Learning trajectory preferences for manipulators via iterative improvement. *Advances in neural information processing systems*, 26.

Jha, R. D.; Hayase, J.; and Oh, S. 2023. Label poisoning is all you need. *arXiv preprint arXiv:2310.18933*.

Kalloori, S.; Ricci, F.; and Gennari, R. 2018. Eliciting pairwise preferences in recommender systems. In *ACM Conference on Recommender Systems*, 329–337.

Koh, P. W.; and Liang, P. 2017. Understanding black-box predictions via influence functions. In *International conference on machine learning*, 1885–1894. PMLR.

Liu, C.; Li, B.; Vorobeychik, Y.; and Oprea, A. 2017. Robust linear regression against training data poisoning. In *ACM workshop on artificial intelligence and security*, 91–102.

Malek Esmaeili, M.; Mironov, I.; Prasad, K.; Shilov, I.; and Tramer, F. 2021. Antipodes of label differential privacy: Pate and alibi. *Advances in Neural Information Processing Systems*.

Mei, S.; and Zhu, X. 2015. Using machine teaching to identify optimal training-set attacks on machine learners. In *Proceedings of the aaai conference on artificial intelligence*.

Noothigattu, R.; Gaikwad, S.; Awad, E.; Dsouza, S.; Rahwan, I.; Ravikumar, P.; and Procaccia, A. 2018. A voting-based system for ethical decision making. In *AAAI Conference on Artificial Intelligence*.

Ouyang, L.; Wu, J.; Jiang, X.; Almeida, D.; Wainwright, C.; Mishkin, P.; Zhang, C.; Agarwal, S.; Slama, K.; Ray, A.; et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35: 27730–27744.

Paudice, A.; Muñoz-González, L.; and Lupu, E. C. 2019. Label Sanitization Against Label Flipping Poisoning Attacks. In *ECML PKDD 2018 Workshops*.

Perrigo, B. 2023. Exclusive: OpenAI Used Kenyan Workers on Less Than $2 Per Hour to Make ChatGPT Less Toxic. *Time*.

Qomariyah, N. N. 2018. *Pairwise preferences learning for recommender systems*. Ph.D. thesis, University of York.

Rosenfeld, E.; Winston, E.; Ravikumar, P.; and Kolter, Z. 2020. Certified robustness to label-flipping attacks via randomized smoothing. In *International Conference on Machine Learning*.

Steinhardt, J.; Koh, P. W. W.; and Liang, P. S. 2017. Certified defenses for data poisoning attacks. *Advances in neural information processing systems*, 30.

Stiennon, N.; Ouyang, L.; Wu, J.; Ziegler, D.; Lowe, R.; Voss, C.; Radford, A.; Amodei, D.; and Christiano, P. F. 2020. Learning to summarize with human feedback. *Advances in Neural Information Processing Systems*, 33: 3008–3021.

Suciu, O.; Marginean, R.; Kaya, Y.; Daume III, H.; and Dumitras, T. 2018. When does machine learning {FAIL}? generalized transferability for evasion and poisoning attacks. In *27th USENIX Security Symposium (USENIX Security 18)*.

Sun, Z.; Shen, S.; Cao, S.; Liu, H.; Li, C.; Shen, Y.; Gan, C.; Gui, L.-Y.; Wang, Y.-X.; Yang, Y.; et al. 2023. Aligning large multimodal models with factually augmented rlhf. *arXiv preprint arXiv:2309.14525*.

Suya, F.; Zhang, X.; Tian, Y.; and Evans, D. 2023. What Distributions are Robust to Indiscriminate Poisoning Attacks for Linear Learners? *Advances in neural information processing systems*.

Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.-A.; Lacroix, T.; Rozière, B.; Goyal, N.; Hambro, E.; Azhar, F.; et al. 2023. Llama: Open and efficient foundation language models. *arXiv:2302.13971*.

Tran, B.; Li, J.; and Madry, A. 2018. Spectral Signatures in Backdoor Attacks. In *Neural Information Processing Systems*.

Vorobeychik, Y.; and Kantarcioglu. 2018. *Adversarial machine learning*. Springer.

Wang, J.; Wu, J.; Chen, M.; Vorobeychik, Y.; and Xiao, C. 2023. On the Exploitability of Reinforcement Learning with Human Feedback for Large Language Models. *arXiv preprint arXiv:2311.09641*.

Wang, Y.; Mianjy, P.; and Arora, R. 2021. Robust learning for data poisoning attacks. In *International Conference on Machine Learning*, 10859–10869. PMLR.

Wu, T.; He, S.; Liu, J.; Sun, S.; Liu, K.; Han, Q.-L.; and Tang, Y. 2023. A brief overview of ChatGPT: The history, status quo and potential future development. *IEEE/CAA Journal of Automatica Sinica*, 10(5): 1122–1136.

Xia, L. 2019. *Learning and decision-making from rank data*. Morgan & Claypool Publishers.

Xiao, H.; Xiao, H.; and Eckert, C. 2012. Adversarial label flips attack on support vector machines. In *ECAI*. IOS Press.

Zeng, Y.; Pan, M.; Jahagirdar, H.; Jin, M.; Lyu, L.; and Jia, R. 2023. Meta-Sift: How to Sift Out a Clean Subset in the Presence of Data Poisoning? In *USENIX Security Symposium*.

Zhang, H.; Li, Y.; Ding, B.; and Gao, J. 2020a. Practical data poisoning attack against next-item recommendation. In *The Web Conference*, 2458–2464.

Zhang, M.; Hu, L.; Shi, C.; and Wang, X. 2020b. Adversarial label-flipping attack and defense for graph neural networks. In *2020 IEEE International Conference on Data Mining*.

A Appendix

A.1 Background: Learning a Reward Model from Pairwise Comparisons

The problem of learning a reward (utility) model from pairwise comparisons is commonly formalized as follows. Let $D = \{(x_i, y_i, o_i)\}_{i=1}^n$ be a dataset of n datapoints with $x_i, y_i \in \mathbb{R}^m$ feature vectors representing the ith pair of outcomes and $o_i \in \{0, 0.5, 1\}$ a preference between these. Specifically, $o_i = 0$ if x_i is preferred to y_i (which we write as $x_i \succ y_i$), $o_i = 1$ if $y_i \succ x_i$, and $o_i = 0$ if these are preference-equivalent (with $x_i \sim y_i$ representing such indifference). Typically, data of this kind of obtained by presenting people with pairs of options (x, y) and asking which of these they prefer. For example, this is the process used in improving the helpfulness, and reducing harmfulness, of LLM-based conversational assistants, as part of an RLHF framework (Christiano et al. 2017; Ouyang et al. 2022). Fundamentally, this is a classic problem in utility function learning (Xia 2019) within the *random utility model (RUM)* theoretical framework, where one assumed that an agent endowed with a true but unknown utility function $R(x)$ reports preference comparison results corrupted with noise, and one aims to approximately recover the underlying utility function from such data.

The most common approach in applied RUM settings, including RLHF, is to leverage the Bradley-Terry (BT) model of preference data generation, in which the preference label o is generated stochastically according to the following distribution:

$$o \sim \Pr\{y \succ x | R\} = \frac{e^{R(y)}}{e^{R(x)} + e^{R(y)}}. \tag{3}$$

Let $R_\theta(x)$ be a parametric reward model that we wish to learn from data generated according to the BT model, with $\theta \in \Theta$, where Θ is the parameter space (e.g., $\Theta = \mathbb{R}^m$). A typical approach is maximum likelihood estimation (MLE), in which we minimize the following loss function (equivalent to maximizing the negative log-likelihood function):

$$\mathcal{L}(D; \theta) = \sum_i -[(1 - o_i) \log \Pr\{x_i \succ y_i | R_\theta\} + o_i \log \Pr\{y_i \succ x_i | R_\theta\}].$$

Commonly, this loss is minimized using gradient descent with respect to reward model parameters θ.

A.2 Theoretical Analysis

Recall that an important theoretical question in our proposed approach is whether focusing on the finite-sample approximation in Problem (2) is principled, in the sense that it yields a provable approximation guarantee even vis-a-vis the true objective (Problem (1)). We now address this issue, focusing on promotion attacks; the analysis for demotion attacks is essentially identical. Note that the answer is not self-evident. First, we are not merely approximating the expectation with sample average (and removing the constant factor, which does not impact the optimal solution), but approximating an *optimization problem*. Thus, law of large numbers does

not immediately imply that the quality of solution to the approximate problem converges. Second, the problem involves bi-level optimization, since the attacker's optimal solution depends, in turn, on the learning optimization problem that yields the reward model parameters θ. Next, we prove that this problem has polynomial sample complexity, that is, the number of samples N grows only polynomially in $1/\epsilon$ and the dimension of the function class R_θ, where ϵ represents the quality of the approximation of the solution to Problem (1) by the solution to Problem (2).

Lemma A.1. *Suppose $\mathcal{F}$ is a vector space of real-valued functions, and $H = \{\mathbf{1}_{\{f+f(a)\geq 0\}} : f \in \mathcal{F}\}$, where a is a constant, then VCdim$(H) \leq$ dim$(\mathcal{F}) + 1$, where dim$(\mathcal{F})$ is the dimension of the vector space of functions $\mathcal{F}$.*

Proof. First, we discuss the case where $f_i(a)$ has the same sign for all i. Let $\{f_1, f_2, \cdots, f_d\}$ be a basis of $\mathcal{F}$. Then if $\{x_1, x_2, \cdots, x_m\}$ is shattered by H, there are vectors $v_1, v_2, \cdots, v_{2^m}$ taking all possible sign patterns, and corresponding $w_1, w_2, \cdots, w_{2^m} \in \mathbb{R}^d$ such that $M(w_1 \cdots w_{2^m}) = V - C$, where $M_{i,j} = f_i(x_j)$, $V_{i,j} = v_{i,j}$ and $C_{i,j} = f_i(a)$. If $m > d$ then the matrix M is not row-rank m. Without loss of generality, we assume the last row can be written as a linear combination of other rows, meaning for some $\alpha_1, \cdots, \alpha_{m-1}$ we have $v_{mi} = \sum_{j=1}^{m-1} \alpha_j v_{ji} + (1 - \sum_{j=1}^{m-1} \alpha_j) \cdot f_i(a)$. If $(1 - \sum_{j=1}^{m-1} \alpha_j) \cdot f_i(a) \geq 0$, then we choose i such that $\alpha_j v_{ji} \geq 0$ for all j, this means $v_{mi} \geq 0$. If $(1 - \sum_{j=1}^{m-1} \alpha_j) \cdot f_i(a) \leq 0$, then we choose i such that $\alpha_j v_{ji} \leq 0$ for all j, this means $v_{mi} \leq 0$. This contradicts the assumption that v_i together take on all 2^m sign patterns.

Now we discuss a general case where we do not have the assumption that $f_i(a), \forall i$ has the same sign. Let $\{f_1, f_2, \cdots, f_d\}$ be a basis of F. If $1 \in \{f_1, f_2, \cdots, f_d\}$, then $\{1, f_1 - c_1, \cdots, f_d - c_d\}$ is also a basis vector (they are also linearly independent), we can choose $c_1, \cdots, c_d$ such that $f_i(a) - c_i \geq 0$, otherwise, we add 1 to the basis vectors of $\mathcal{F}$. Then we have VCdim$(H) \leq$ dim$(\mathcal{F})$ + 1. □

Theorem A.2 ((Anthony and Bartlett 1999))**.** *Suppose that H is a set of functions from a set X to $\{0, 1\}$ and that H has finite Vapnik-Chervonenkis dimension $d \geq 1$. Let L be any sample error minimization algorithm for H. Then L is a learning algorithm for H. In particular, if $m \geq d/2$ then its sample complexity satisfies the inequality*

$$m_L(\varepsilon, \gamma) \leq m_0(\varepsilon, \gamma) = \frac{64}{\varepsilon^2}\left(2d \ln\left(\frac{12}{\varepsilon}\right) + \ln\left(\frac{4}{\gamma}\right)\right).$$

Here $m_L(\varepsilon, \gamma) = \min\{m : m$ is a sufficient sample size for (ε, γ)-learning H by $L\}$, meaning for $m \geq m_0(\varepsilon, \gamma)$ and $z \in Z^m$ chosen according to P^m, $P^m\{er_P(L(z)) \geq opt_P(H) - \varepsilon\} \geq 1 - \gamma$.

In our result below, we capture the solution to the lower-level problem in which the reward model parameters θ are learned based on the poisoned dataset $\tilde{D}(\delta)$ as $\theta(\delta)$. Let Ω denote the space of possible attacks δ.

Theorem A.3. *Let $\mathcal{F} = \{R_\theta | \theta \in \Theta\}$ and suppose dim$(\mathcal{F}) = d$. Then for all $\epsilon > 0$, and for all $N > m_0(\epsilon, \gamma)$ where $m_0(\epsilon, \gamma) = \frac{64}{\varepsilon^2}\left(2(d+1)\ln\left(\frac{12}{\varepsilon}\right) + \ln\left(\frac{4}{\gamma}\right)\right)$, $\max_{\delta \in \Omega} \hat{F}(\delta) \geq \max_{\delta \in \Omega} F(\delta) - \varepsilon$ with probability at least $1 - \gamma$.*

Proof. Let $\Theta(\Omega) = \{\theta \in \arg\max_{\theta' \in \Theta} \mathcal{L}(\tilde{D}(\delta), \theta') | \delta \in \Omega\}$. This is the set of all parameters θ of R_θ that can be induced by the adversarial label perturbations $\delta \in \Omega$. Define $\mathcal{F}' = \{R_\theta(x) | \theta \in \Theta(\Omega)\}$. Next, note that Problem (1) is equivalent to

$$\max_{\theta \in \Theta(\Omega)} \sum_{c \in C^T} \Pr_{x \sim \mathcal{P}}\{R_\theta(c) \geq R_\theta(x)\}.$$

Similarly, Problem (2) is equivalent (up to $1/N$ factor) to

$$\max_{\theta \in \Theta(\Omega)} \sum_{c \in C^T} \frac{1}{N} \sum_i \mathbf{1}_{R_\theta(c) \geq R_\theta(x_i)}.$$

In this transformation, we effectively collapsed the bi-level structure into a pair single-level optimization problems, and can now make use of the machinery in Lemma A.1 and Theorem A.2. Since $\mathcal{F}' \subseteq \mathcal{F}$, we have dim$(\mathcal{F}') \leq$ dim$(\mathcal{F})$. Let $H = \{\mathbf{1}_{\{R_\theta(x_T) - R_\theta \geq 0\}} : R_\theta \in \mathcal{F}'\}$, which is a class of functions that map from a set X to $\{0, 1\}$. Set the correct label for all $x \in X$ to be 1 (which maximizes the attack success rate). Let $er_P(h) = P\{(x, y) \in Z : h(x) \neq 1\}$, and $\widehat{er}_z(h) = \frac{1}{N}|\{i : 1 \leq i \leq m \text{ and } h(x_i) \neq 1\}|$, $h \in H$. $opt_P(H) = \inf_{g \in H} er_P(g) = \max_{\delta \in \Delta} F(\delta)$, and $\widehat{er}_z(L(z)) = \min_{h \in H} \widehat{er}_z(h)$. We can then apply Lemma A.1 and Theorem A.2 to get the desired result. □

A.3 Experiment Setup (Detailed)

Our experiments involve four problem settings. Our first experiment involves training a reward model toward safety alignment of Large Language Models (LLMs). In the next two, reward model learning is done in the context of value alignment for control (Christiano et al. 2017). The first of these generates control trajectories in MuJoCo environment associated with low-dimensional state inputs, whereas the second uses Atari to generate control trajectories with images as inputs. The fourth considers a recommendation system context using textual information. For this we use the Amazon rating data,[1] which we transform into pairwise comparisons. The motivation for exploring these four settings is to assess the performance and impact of our proposed attack for qualitatively different control problems, since the preference-based reward model learning and RLHF have been used in domains as distinct as robotic control (Jain et al. 2013) and LLMs (Ouyang et al. 2022).

We conducted 5 independent runs for all but one experiment, which means that the data were collected independently for each run, and report the mean and standard error of the mean (SEM) in the plots. The only exception is LLM, where we conducted only 1 run due to high computational requirements. In most cases below, we learn a neural network reward model R_θ, with one exception where we also

[1] cseweb.ucsd.edu/~jmcauley/datasets/amazon_v2/, Home and Kitchen.

investigate robustness of a linear reward model in comparison with a neural network in the same setting. In all cases, we vary the budget B between 1% and 10% of data. Next we provide further details about the experiment environments and our setup; further details are provided in the Appendix.

Safety Alignment: We use the PKU-SafeRLHF-30k dataset[2], which includes 30k question-answer pairs annotated with both helpfulness and harmlessness preference labels. For the scope of our study, we only use the harmlessness preference labels for our safety alignment task. Then we train the reward model initialized with an instruction following language model, which is obtained by performing supervised fine-tuning on the pre-trained model LLaMA-7B (Touvron et al. 2023) using the Stanford Alpaca Dataset.

MuJoCo Control: We use three MuJoCo environments: Reacher, Hopper, and Walker2D. We use trajectory data as inputs (outcomes), where trajectory $\mathcal{T} = ((s_1, a_1), (s_2, a_2), \cdots (s_l, a_l))$ is induced by a random policy $a_i = \pi(s_i)$, with s_i representing the system state (low-dimensional and observable) and a_i the action. Following (Christiano et al. 2017), we record the total reward returned by the MuJoCo simulator as the reward for the trajectory. The trajectory with the higher reward is marked as winning, and the other as losing. The trajectory length for each environment is 30.

Atari Vision-Based Control: For Atari control with image inputs, we experiment with three environments: Pong, Breakout, and Qbert. We used trajectory data as inputs as in the previous setting, except now our states s_i are images. We concatenate actions with image embeddings in the mid-linear layer. For experiments with PCA we reduces the input dimension to 20, and we then concatenate it with the action.

Recommendation System: Finally, we use Amazon Rating Data collected with the purpose of designing recommendation systems. While this, like many conventional recommendation datasets, relies on user ratings, it has been observed that learning utilities using pairwise comparisons rather then relying solely on ratings can yield better recommendations (Kalloori, Ricci, and Gennari 2018). Consequently, we used this data to generate a derived dataset of pairwise comparisons by comparing the recorded ratings of pairs of comments, which were used as inputs (outcomes). We used BERT to generate embeddings for each input comment, considering it as the input representation. When we apply PCA instead, we reduce the original dimension to 20 components. In this domain, we experiment using both a linear neural network and a 3-layer neural network with ReLU activations (multilayer perceptron (MLP)). For linear neural networks, the average training accuracy was 92.3%, and the average test accuracy was 91.5%. For the neural network architecutre, average training accuracy was 100%, while average test accuracy was 92.3%. While it seems that the latter architecture is superior to the linear model, our experiments below offer an interesting qualification in that regard.

[2]huggingface.co/datasets/PKU-Alignment/PKU-SafeRLHF-30K

A.4 Results (Detailed)

We present the results below in the case of promotion and demotion attacks for a single target outcome c^T, with the results considering multiple target candidates deferred to the Appendix. In promotion attacks, we choose the target outcome as the least preferred outcome in the dataset in terms of the reward model R_θ learned on clean data. Similarly, when considering a set C^T, we choose the set of smallest-reward outcomes to promote. Demotion attacks, on the other hand, aim to demote that best outcomes in terms of R_θ learned on clean data. Consequently, our attack settings provide the most challenging attack tasks to accomplish.

Safety Alignment. Our first consideration involves the safety alignment problem in the context of language models. Specifically, we consider a dataset of pairwise comparisons involving relative harmfulness of prompt responses. We learn both the clean, and poisoned reward model R_θ by fine-tuning the LLaMA-7B model using the Stanford Alpaca Dataset. As gradient-based methods are not scalable in this setting due to the size of the neural networks, we only consider the RBD-based approaches for attacks.

Figure 2 presents the results on the efficacy of the poisoning attacks (top) and accuracy degradation (bottom). What we observe is that the LLM setting is extremely vulnerable: the best attack achieves 100% success rate for both promotion and demotion goals with only 0.3% poisoned instances. A key reason is the structure of this and similar datasets: outcome comparisons are made with respect to a relatively limited set of prompts and responses, which entails many identical outcomes in pairwise comparisons. Consequently, it is often the case that there are a large set of outcomes similar to the target, and flipping the comparison label for all of these (or any subset) is highly efficacious. We also observe that in this setting, both RBD-Reward and RBD-Embedding (using Llama embedding) perform comparably. In addition, we see minimal accuracy degradation as a result of the attack.

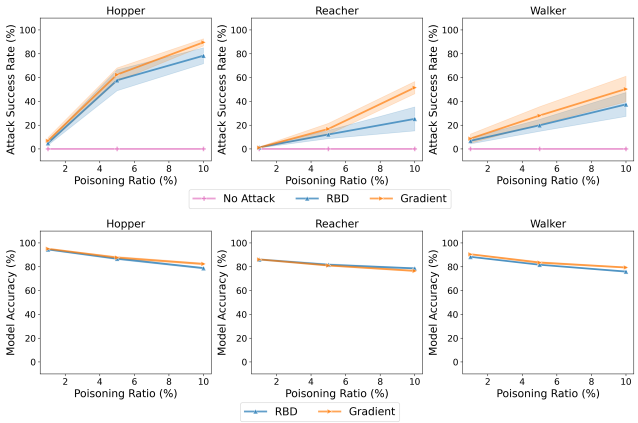

Figure 4: Promotion attack efficacy in MuJoCo. Top row: success rate. Bottom row: accuracy.

Next, we evaluate the downstream impact of reward model attacks on RLHF in this setting. The success rates

(of the learned policy choosing the target response to a prompt) are provided in Table 1 for a randomly chosen target candidate, for whom baseline success rate (with no attack) is $< 1\%$. What we can see is that while the efficacy of the attack degrades somewhat compared to targeting reward model learning directly, it is nevertheless quite successful, achieving 99% success rate with only 5% of labels flipped.

MuJoCo Control. Next we consider promotion attacks in the low-dimensional MuJoCo setting. Figure 4 (top) presents attack success rate for the best-performing variants of each attack class. Our first observation, which will be echoed in other domains below, is that attack success rate (and, thus, vulnerability) varies quite significantly with problem setting. For example, success rate in Hopper reaches over 60% with 5% of the attack budget, and approximately 90% with 10% of the budget. On the other hand, success rate for Reacher and Walker is just over 50% even with 10% of the budget.

Our second observation is that in all three domains, the (best) gradient-based method is most effective. Nevertheless, we do note that the (best) RBD heuristic is typically competitive with the best attack overall, particularly at small budgets. For example, at 1-5% of the budget, the performance of both classes of attacks is similar.

Next, we consider stealth of the attack with increasing attack budget (Figure 4 (bottom)). Although none of the methods we developed are explicitly stealthy, the results show that our targeted attacks have a limited impact on overall accuracy, particularly at the smaller range of attack budgets between 1% and 5%.

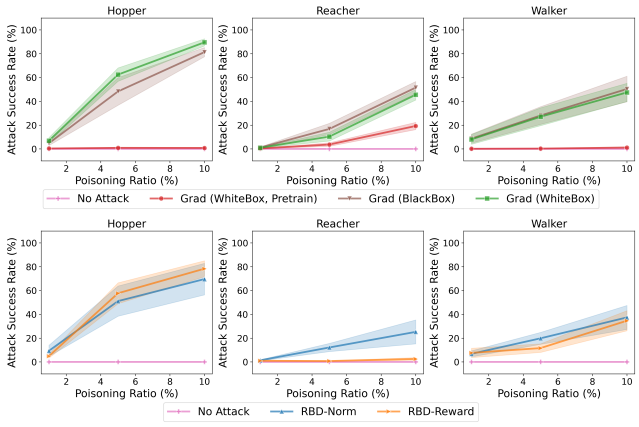

Figure 5: Relative efficacy of gradient-based attacks (top row) and RBD attacks (bottom row) in MuJoCo.

In Figure 5, we consider now ablations in terms of different gradient-based approaches (top) and RBD methods (bottom). Here we make two observations. First, in this setting, pretraining the neural network before performing the gradient-based attack, as done by (Koh and Liang 2017), performs extremely poorly, yielding a nearly zero success rate. This is likely because it is fragile to uncertainty about neural network initialization. Rather, our primary approach in which we train the attack to be explicitly robust to initialization uncertainty performs best. In addition, we note that black-box attacks in this context are essentially as effective as white-box attacks, suggesting that we do not need precise knowledge of the neural network architecture. Second, we generally find that RBD-Norm outperforms RBD-Reward in two of the three environments, with the two exhibiting similar performance in the third (Hopper).

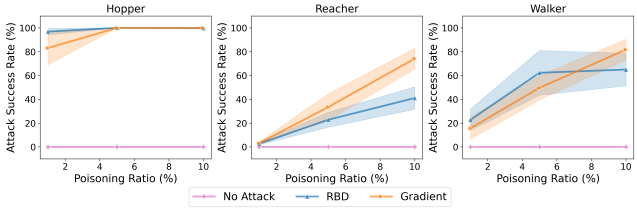

Figure 6: Demotion attack efficacy in MuJoCo.

Finally, Figure 6 presents the results of demotion attacks. We can observe that in this setting, demotion attacks are somewhat easier than promotion attacks, with success rates of the best attacks tending to be higher. The overall trends, however, parallel what we observed in promotion attacks: gradient-based methods outperform RBD as budget increases. On interesting exception is the demotion attack in the Hopper environment, where low-budget settings yield better RBD success rate than gradient-based method, with the former nearly 100% successful even with a budget of 1%. As in the case of promotion attacks, demotion attacks are also relatively stealthy (see Figure 19 in the Appendix) and black-box gradient-based attacks are nearly as effective as white-box (see Figure 17 in the Appendix). Finally, the results are comparable when we consider a set of 5 target outcomes instead of just a single one (see Figure 18 in the Appendix).

Atari Vision-Based Control. Next, we turn to the more challenging control setting in which true state of the controlled system is not directly observable, but is instead observed using (relatively) high-dimensional visual perception.

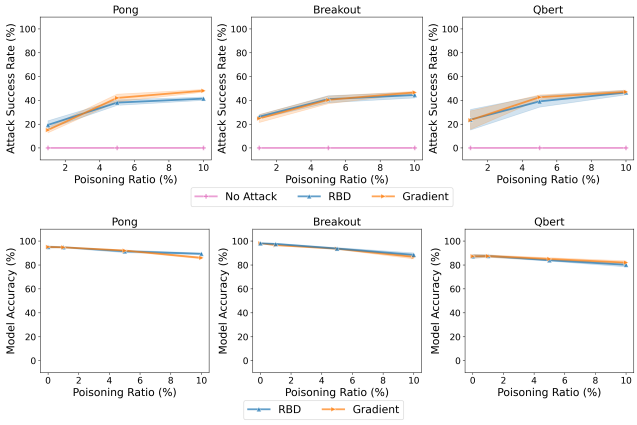

Figure 7: Promotion attack efficacy in Atari. Top row: success rate. Bottom row: accuracy.

First, as above, we consider attack efficacy in terms of both success rate (Figure 7, top) and stealth (test accuracy; Figure 7, bottom). In this setting, success rate is markedly lower than in MuJoCo, although we still reach nearly 50% success rate of the best attack with only 5% of the data poisoned. What is particularly noteworthy is that here there is little difference between the best gradient-based method and the best RBD heuristic. As RBD is simpler, considerably faster, and requires no information about the model, this is noteworthy as one of the sources of threat are individuals themselves who are asked for their relative preferences over outcomes. Thus, the simplicity and intuitive nature of RBD, along with its efficacy, may well make the attack a particularly significant concern in practice.

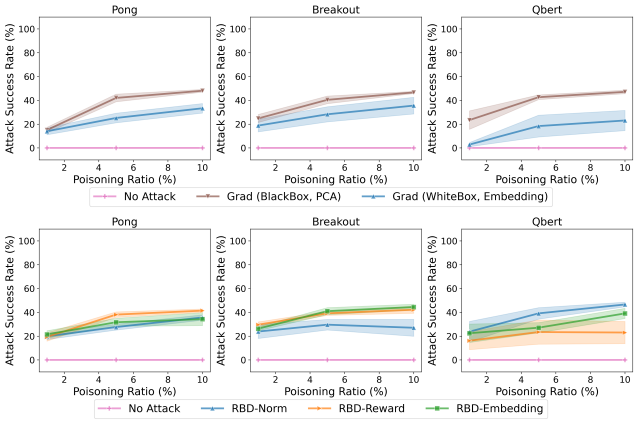

Figure 8: Relative efficacy of gradient-based attacks (top row) and RBD attacks (bottom row) in Atari.

Next, we consider again the relative efficacy of gradient-based methods (Figure 8, top row). In this setting, surprisingly, the most effective attack is to first preprocess the high-dimensional input images using PCA (20 dimensions), and perform a (black-box) attack on a neural network over this reduced-dimensional outcome space. This approach is significantly better than making use of the low-dimensional embedding learned as part of the reward model R_θ trained on clean data. Additionally, making use of the conjugate-gradient method to approximate the inverse in implicit gradient computation, as suggested by (Koh and Liang 2017) was extremely slow, with a single attack taking $\sim$ 12 hours for one seed.

Turning now to the comparison of alternative RBD heuristics (Figure 8, bottom row), we find that their relative efficacy can now vary a great deal by environment. In Pong, for example, all three are quite comparable, although RBD-Reward tends to outperform RBD-Embedding (which ranks by distance with respect to the embedding from R_θ learned on clean data) and RBD-Norm (which measures distances directly in pixel space). In Breakout, RBD-Norm is the worst heuristic, while in Qbert, it is the best of the three, while RBD-Reward is the weakest one in this setting.

Our observation that relative performance of different attack methods varies considerably by domain and environment is instructive: our consideration of two classes of attack algorithms, and multiple variants in each class thereby demonstrates the value of comprehensive vulnerability analysis that this provides. This is in contrast to common vulnerability analysis of ML to poisoning attacks in prior work, where only a single attack algorithm is typically evaluated (e.g., (Jha, Hayase, and Oh 2023; Xiao, Xiao, and Eckert 2012; Zhang et al. 2020b)).

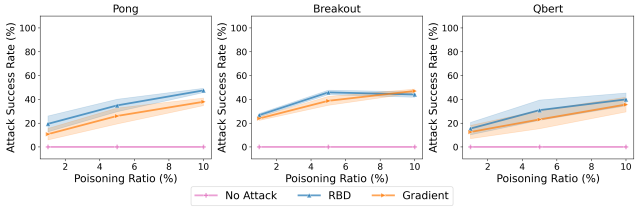

Figure 9: Demotion attack efficacy in Atari.

Finally, we consider demotion attacks in the Atari control domain. The results are shown in Figure 9, are are broadly similar to what we have observed for promotion attacks. However, here we see that RBD heuristics tend to outperform the best gradient-based method in nearly all cases. This may seem surprising at first, given that the more expensive gradient-based methods are generally viewed as highly principled, but note that these are also heuristic in a number of ways. For example, in this discrete setting, gradient-based approaches rely on relaxation of the discrete decision variables. Moreover, as the basic variant of these fails to scale to the higher-input-dimension problem posed in vision-based control, we now also rely on the linear (PCA) dimensionality reduction technique blended with the gradient-based method (which does outperform a non-linear learned embedding). In any case, this again reinforces our broader observation that the nature of the best attack can vary by domain, environment, and attacker's objective.

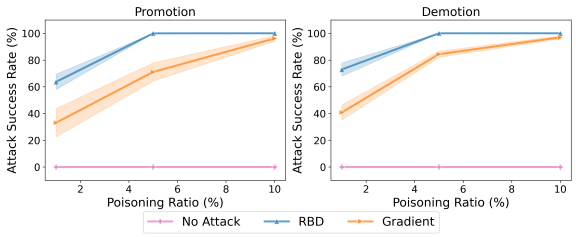

Figure 10: Efficacy of promotion (left) and demotion (right) attacks on the recommendation dataset (neural network model).

Recommendation System. Finally, we consider a recommendation system setting. In Figure 10 we show the results of poisoning in this context for both promotion and demotion attacks, when we learn a neural network reward model (MLP). Notably, in this setting, which is also too high-dimensional for a direct application of the gradient-based approach, RBD methods are significantly more effective than gradient-based algorithms, both in the case of promotion and demotion attacks. Additionally, the best method

(RBD) in this setting achieves 100% success rate in both types of attacks with only a 5% attack budget, and 60-70% success rate with a mere 1% attack budget.

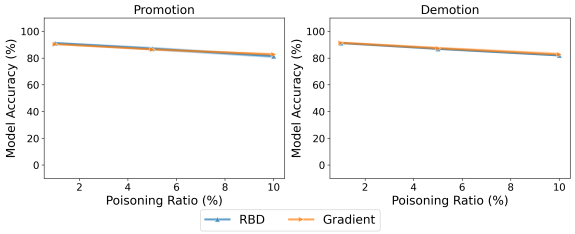

Figure 11: Accuracy of promotion and demotion attacks on the recommendation dataset (neural network model).

In Figure 11 we present accuracy results as a function of the attacker budget. As in all the settings so far, our targeted attacks have only a small impact on test accuracy, both for promotion and demotion attacks.

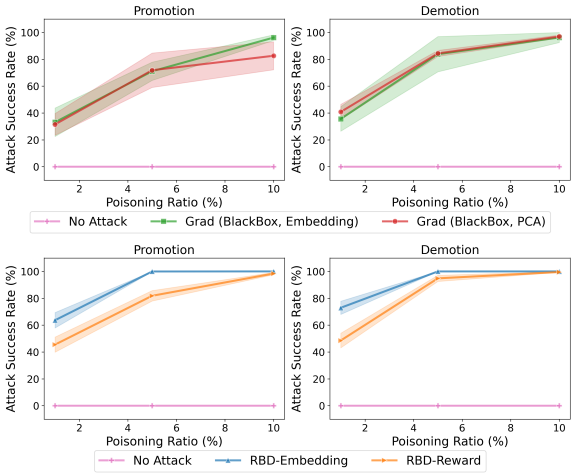

Figure 12: Relative efficacy of gradient-based attacks (top) and RBD attacks (bottom) on the recommendation dataset (neural network model).

Consider now Figure 11, which evaluates relative efficacy of gradient-based approaches (top) and relative efficacy of RBG approaches (bottom). In this domain, we observe that gradient-based approaches aimed at improving scalability via the use of conjugate gradient methods still do not scale well (running a single attack takes $\sim$ 5 hours), and simple PCA to reduce the input dimension can relatively effectively combine with a gradient-based approach. Nevertheless, as we highlight above, scalability becomes a practical issue for such approaches, with the RBG heuristics now appearing to be a better option. In the case of RBG variants, on the other hand, there is in this domain a clear advantage for using feature (BERT embedding) distance directly, rather than reward as a measure of distance.

Thus far, our consideration of reward model poisoning was focused on neural network reward models. We now explore the extent to which using linear models yields better or worse robustness to poisoning attacks. In Figure 13

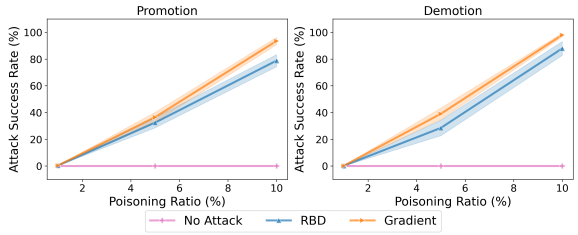

Figure 13: Efficacy of promotion and demotion attacks on the recommendation dataset (linear model).

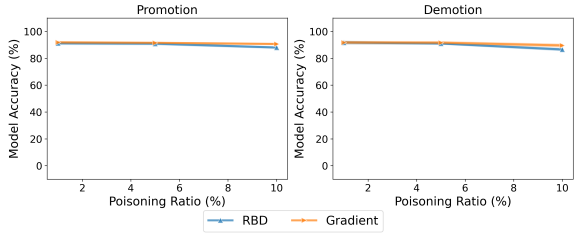

Figure 14: Accuracy of promotion and demotion attacks on the recommendation dataset (linear model).

we present the results of poisoning attacks on linear reward models. It is instructive to compare these results to what we saw in the case of neural network reward models in Figure 10: the linear model is considerably more robust than the neural network, and it takes twice as much budget (10%, as compared to 5%) to reach 100% attack success rate. Nevertheless, both model classes are clearly vulnerable to poisoning. The second observation we can make in the comparison is that in the case of linear models, gradient-based methods are somewhat better than RBD, whereas the reverse was true, as we noted above, in the case of neural networks.

Figure 14 presents linear model accuracy as a function of attack strength. Here, again, we observe that accuracy degradation is minimal, and the attack is quite stealthy.

Finally, Figure 15 presents ablation results comparing different gradient (top) and RBD (bottom) approaches in terms of relative efficacy. In this domain, we find that using a pretrained model as part of the gradient-based scheme now outperforms the approach that instead uses K random initializations, especially in the demotion attack. In RBD comparison, on the other hand, RBD-Norm handily outperforms RBD-Reward in this setting, comparable to our observations above.

Partial Knowledge of Data. Our experiments thus far have assumed that the attacker observes the entire dataset D. We now consider the impact of relaxing this assumption, when only partial information about D is available. The results are provided in Figure 16 for the MuJoCo and Atari control environments, and demonstrate that while attack success rate degrades under partial knowledge of data, it does so relatively slowly.

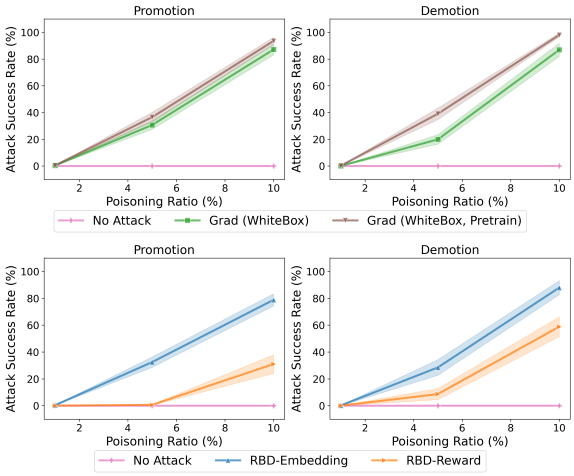

Figure 15: Relative efficacy of gradient-based attacks (top) and RBD attacks (bottom) on the recommendation dataset (linear model).

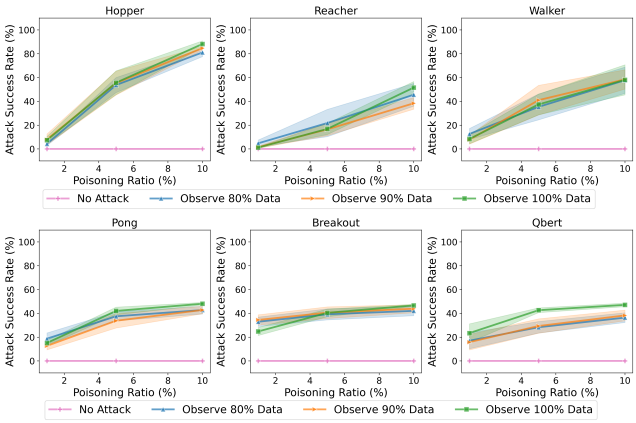

Figure 16: Promotion attack with partial knowledge of data (best attack results). MuJoCo (top) and Atari (bottom).

B Additional Experiment Details

MuJoCo Control We use a feedforward 3-layer neural network with ReLU activation functions, where the mid-layer size is 32. We collect 1250 pairs of trajectories for each environment, and train for 2000 epochs with learning rate 6.25×10^{-4}. Training and testing data have the same size. For gradient-based black-box attacks, we use a 3-layer neural network with mid-layer size 16 for attack.

Atari Vision-Based Control We trained for 100 epochs with a 6.25×10^{-4} learning rate. PCA reduces the input dimension to 20, and we then concatenate it with the action (final dimension is 21). Then we use a 3-layer neural network with mid-layer size 16.

Recommendation System We explored both a simple linear model and a 3-layer neural network with ReLU activation functions and a mid-layer size of 1024. In the PCA scenario, we reduced it to 20 components and then employed a 3-layer neural network with ReLU activation functions and a mid-layer size of 32. Training data has 4500 pairwise preferences, and testing data has 500 pairwise preferences. For gradient-based black-box attacks, we first do PCA to reduce its dimension to size 10; then we use a 3-layer neural network with mid-layer size 16.

C Additional Results

MuJoCo Control Figure 19 presents the results of trained reward model accuracy for demotion attacks. The results are similar to promotion attacks: accuracy degradation is small, demonstrating that the attack is sufficiently stealthy. Figure 17 presents the ablation results for gradient and RBD methods in the case of demotion attacks. Finally, Figure 18 shows that the results for multiple target candidates (5, in this case) are comparable to a single target candidate.

Atari Vision-Based Control Figure 20 presents the results of trained reward model accuracy for demotion attacks in Atari. We can again observe that there is little degradation in training accuracy after the attack. Figure 21 presents the ablation results for gradient and RBD methods in the case of demotion attacks in the Atari domain. Figure 22 shows that the results for multiple target candidates are comparable to a single target candidate in the Atari domain.

Recommendation System Figure 23 presents the results with 5 target candidates on the Amazon dataset. As we can see, these are broadly consistent with the single-target case in the main body.

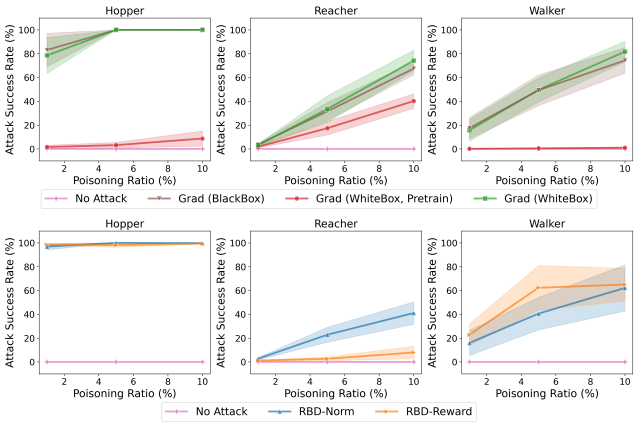

Figure 17: Relative efficacy of gradient-based demotion (top row) and RBD demotion attacks (bottom row) in MuJoCo.

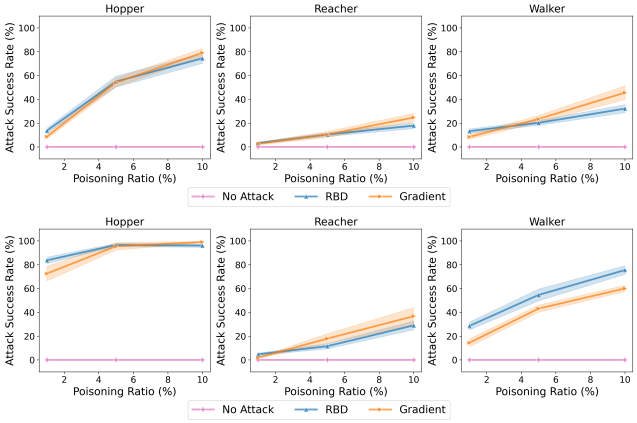

Figure 18: Attack efficacy in MuJoCo with 5 target candidates.

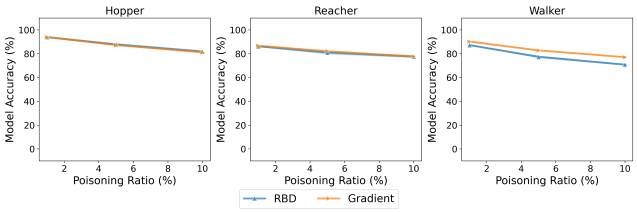

Figure 19: Demotion attack stealth in MuJoCo: test set accuracy.

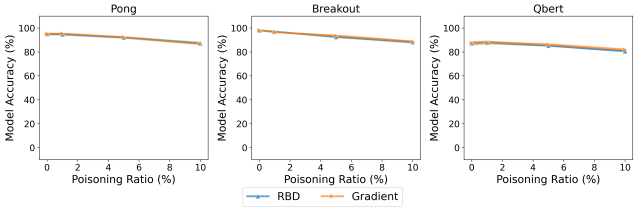

Figure 20: Demotion attack stealth in Atari: test set accuracy.

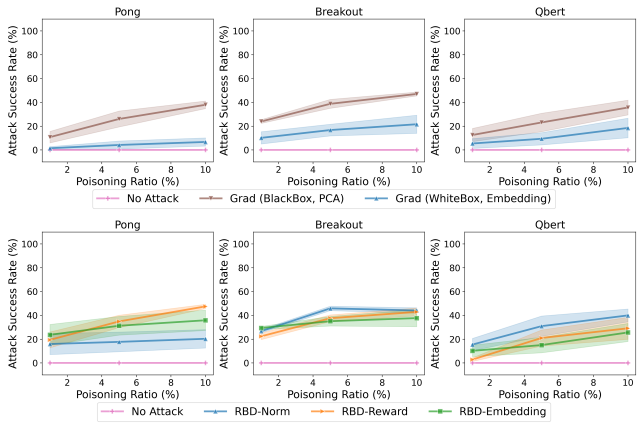

Figure 21: Relative efficacy of gradient-based demotion attacks (top row) and RBD demotion attacks (bottom row) in Atari.

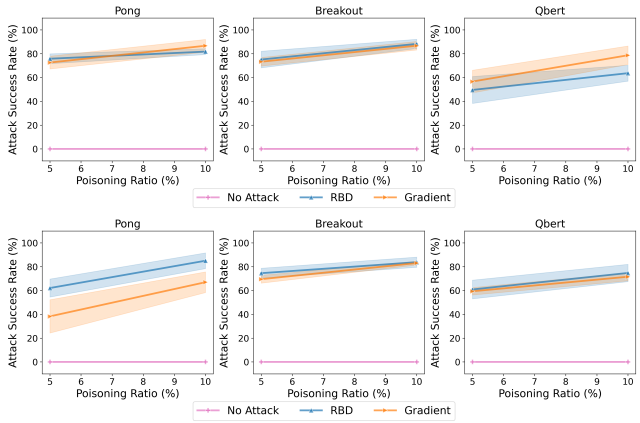

Figure 22: Attack efficacy in Atari with 5 target candidates.

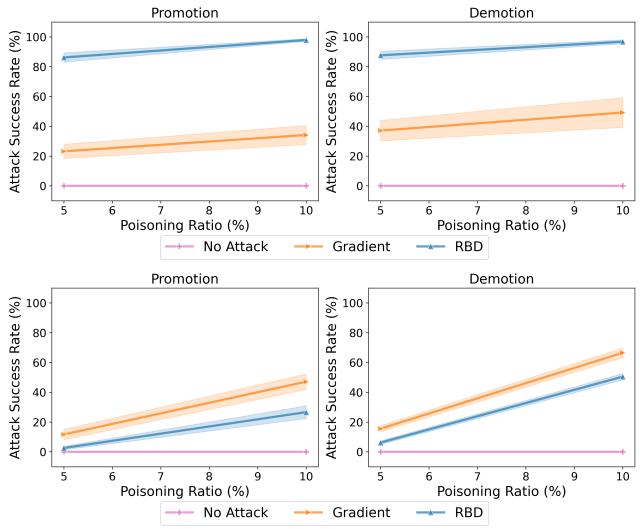

Figure 23: Attack efficacy with 5 target candidates on the Amazon recommendation ratings dataset. Neural Network Model (top row) and Linear Model (bottom row).