Reinforcement Learning from User Feedback

Eric Han[1], **Jun Chen**[1], **Karthik Abinav Sankararaman**[1], **Xiaoliang Peng**[1], **Tengyu Xu**[1], **Eryk Helenowski**[1], **Kaiyan Peng**[1], **Mrinal Kumar**[1], **Sinong Wang**[1], **Han Fang**[1], **Arya Talebzadeh**[1]

[1]Meta GenAI

As large language models (LLMs) are increasingly deployed in diverse user facing applications, aligning them with real user preferences becomes essential. Existing methods like Reinforcement Learning from Human Feedback (RLHF) rely on expert annotators trained on manually defined guidelines, whose judgments may not reflect the priorities of everyday users.

We introduce **Reinforcement Learning from User Feedback (RLUF)**, a framework for aligning LLMs directly to implicit signals from users in production. RLUF addresses key challenges of user feedback: user feedback is often binary (e.g., emoji reactions), sparse, and occasionally adversarial. We train a reward model, P[Love], to predict the likelihood that an LLM response will receive a Love Reaction — a lightweight form of positive user feedback — and integrate P[Love] into a multi-objective policy optimization framework alongside helpfulness and safety objectives.

In large-scale experiments, we show that P[Love] is predictive of increased positive feedback and serves as a reliable offline evaluator of future user behavior. Policy optimization using P[Love] significantly raises observed positive-feedback rates, including a 28% increase in Love Reactions during live A/B tests. However, optimizing for positive reactions introduces reward hacking challenges, requiring careful balancing of objectives. By directly leveraging implicit signals from users, RLUF offers a path to aligning LLMs with real-world user preferences at scale.

Date: May 22, 2025
Correspondence: Arya Talebzadeh at aryatalebz@meta.com

1 Introduction

Large Language Models (LLMs) now power many user-facing applications (Grattafiori et al. 2024, OpenAI et al. 2024). Aligning them with real user preferences, however, remains a fundamental challenge. Traditionally, LLMs are aligned through Reinforcement Learning from Human Feedback (RLHF) (Ouyang et al. 2022, Ziegler et al. 2020, Zheng et al. 2023), which uses feedback from expert annotators following predefined guidelines to shape model behavior. While effective, this paradigm assumes that annotators represent the preferences of actual users — an assumption that breaks down as LLMs are deployed at massive scale. Real users have diverse goals, contexts, and values that are often misaligned with expert annotators.

This misalignment reveals a core limitation of RLHF: it optimizes for proxy preferences defined by researchers rather than authentic signals of the end user. At scale, the true objective is not to satisfy an annotator, but to deliver utility, delight, and trust to the people actually interacting with the model. That requires a shift: from *human feedback* to *user feedback*.

We introduce **Reinforcement Learning from User Feedback (RLUF)**, a framework for aligning LLMs directly to user preferences at scale using implicit binary feedback collected in production. Crucially, we do not claim to directly optimize ground-truth user satisfaction. Instead, we rely on lightweight binary signals (e.g., heart-emoji "Love" reactions) as practical proxies. RLUF addresses key challenges that distinguish real-world user data from standard RLHF settings.

- User feedback is often **sparse**, **binary** (e.g., a thumbs-up reaction), and occasionally **adversarial**, lacking the richness and control of curated pairwise preferences.

- User preferences can sometimes **conflict with safety or helpfulness objectives** (Bai et al. 2022, Xu et al. 2023), requiring careful multi-objective optimization to prevent large regressions to LLM behavior.

We show how to model this kind of feedback through a *User Signal Reward Model* trained on binary feedback like Love Reactions. This reward model predicts the likelihood that a given LLM response will receive a positive user signal. We integrate it into a **multi-objective reinforcement learning framework** (Xu et al. 2024, Touvron and et al. 2023) that co-optimizes for helpfulness, safety, and user satisfaction.

Empirically, we show that:

- User signal reward models trained on binary user feedback generalize to comparing Meta-internal candidate LLMs and predicting which one will elicit higher positive-feedback rates.
- Llama models trained with RLUF show a significant increase in positive user feedback, with up to a **28% increase in Love Reactions** while maintaining helpfulness and safety.
- However, over-optimization of P[Love] can introduce challenges such as reward hacking, where the model generates repetitive closing statements like "Bye! Sending Love!" to artificially boost positive reactions. We address these challenges through careful balancing of user preference against other alignment objectives.

Our results demonstrate a viable path for aligning LLMs directly to the users they serve.

2 Reinforcement Learning from User Feedback

In this section, we describe the **RLUF pipeline**—a scalable framework for aligning LLMs to user preferences using implicit feedback collected in production. The RLUF pipeline consists of three main stages:

1. Selecting meaningful user feedback signals to optimize.
2. Training a user signal reward model to predict the selected user signals.
3. Using the user signal reward model to align an LLM to user preferences alongside other objectives (e.g., helpfulness and safety).

This setup enables us to optimize for real user satisfaction while preserving other desirable properties of LLM behavior.

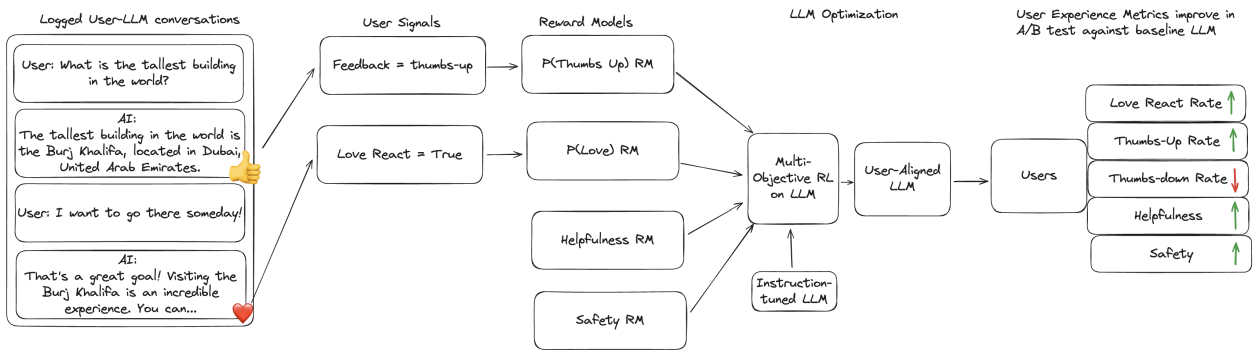

Logged User-LLM conversations
User: What is the tallest building in the world?
AI: The tallest building in the world is the Burj Khalifa, located in Dubai, United Arab Emirates.
User: I want to go there someday!
AI: That's a great goal! Visiting the Burj Khalifa is an incredible experience. You can...
User Signals
Feedback = thumbs-up
Love React = True
Reward Models
P(Thumbs Up) RM
P(Love) RM
Helpfulness RM
Safety RM
LLM Optimization
Multi-Objective RL on LLM
Instruction-tuned LLM
User-Aligned LLM
Users
User Experience Metrics improve in A/B test against baseline LLM
Love React Rate
Thumbs-Up Rate
Thumbs-down Rate
Helpfulness
Safety

Figure 1 Overview of the RLUF pipeline from left to right: We begin with raw user-llm conversations and binary feedback signals attached to each turn. We then train user signal reward models and combine them with existing reward models in a multi-objective reinforcement learning framework. This produces a user-aligned language model which has the desired property of improving user satisfaction.

2.0.1 Step 1: Signal Selection

Our first step is to identify one or more user interaction signals that serves as a viable proxy for user satisfaction. For concreteness we focus on *Love Reactions*—a heart emoji that users may apply to a model response within the chat interface. However, we emphasize that the RLUF framework is general across many user signals.

We define a binary user preference signal S : (context, response) $\rightarrow \{0, 1\}$, where 1 indicates that a model response received a positive user feedback and 0 a negative user feedback. User feedback can come in many forms - thumbs up/down, whether a user continues the conversation, the sentiment of the follow-up user prompt, or even whether a user comes back to the chatbot the next day. We choose Love Reactions because even though they are inherently sparse, they have the following properties:

- **Sufficiently available at scale**, preventing limitations on data volume.
- **Correlated with long-term satisfaction**: Reflective of higher-level satisfaction metrics like user retention or engagement (Section 4.1).
- **Unambiguous sentiment**: Easy to interpret as positive or negative.

We encourage users of the RLUF framework to follow this rubric when choosing user signals to optimize. Selection of user signals often begins with "vibes". We later verify quality by examining model artifacts that incorporate the user signal.

2.0.2 Step 2: Feedback Collection and Reward Model Training

Using love reactions as our user signal, we construct a binary classification dataset from production traffic by labeling each model response as either having received a Love Reaction (1) or not (0) from users. Using this dataset we construct a reward model, P[Love] (Section 3.1.1), to estimate the probability that a model response will receive a Love Reaction. The reward model serves two primary roles:

1. As an **offline evaluator**, to predict whether a change in LLM model behavior will improve user satisfaction as measured by an increase in love reaction rate.
2. As a **reward signal** during policy optimization, where we directly train an LLM to generate responses that are scored highly by the reward model.

2.0.3 Step 3: Multi-Objective Policy Optimization

We incorporate our user signal reward model P[Love] into a **multi-objective reinforcement learning framework** (Section 3), optimizing the model not just for helpfulness and safety but also for inferred user delight. We use Mixture of Judges (Xu et al. 2024), a multi-objective reinforcement learning framework, to combine our three separate reward functions for Helpfulness, Safety, and Love (P[Love]). Ultimately, we train a language model that improves user satisfaction as measured by love reactions while preserving core alignment objectives (Section 5).

3 Methods

This section describes the components used to implement Reinforcement Learning from User Feedback (RLUF), including the construction of reward models from user and annotator signals, and the multi-objective policy optimization framework used to train aligned language models.

3.1 Reward Models

We use a set of three reward models during policy optimization: (1) a **User Signal Reward Model** trained on Love Reactions from real users, (2) a **Helpfulness Reward Model** trained on pairwise preferences from expert annotators, and (3) a **Safety Reward Model** trained to detect and penalize harmful completions. Each reward model is paired with a separate and dedicated prompt set during policy optimization.

3.1.1 User Signal: Love Reward Model

We collect binary user feedback from production interactions with the Meta AI chatbot, where users may optionally long-press (tap-and-hold) on a model response and select a Love Reaction. These signals are extremely sparse: approximately 0.1% of all model messages receive a love reaction. We observe that these reactions tend to cluster in domains like AI bonding, recommendations, and personal writing, and are often used to close conversations on a positive emotional note.

To construct a reward model, we create a binary-labeled dataset of 1 million examples, upsampling positive examples so that Love Reactions constitute 10% of the training data. Each training example consists of up to 10 turns of conversation history, the most recent user prompt, the model response, and the user feedback, given by a binary label of having received a Love Reaction (1) or not (0).

We train a classifier to estimate:

$$P[\text{Love}] = \Pr(\text{Love Reaction} \mid \text{context, response})$$

using a binary cross entropy loss. The model is based on the Llama3-8B instruct checkpoint, with a classification head added in place of the standard output layer. Further details on the Love reward model are in Appendix C, including evaluation metrics and examples of the training data.

Reducing Bias. We observe that P[Love] sometimes penalizes valid refusals, since users tend not to react positively to responses that refuse requests—even when refusal is appropriate. We attempted to filter such anti-safety data from the P[Love] reward model training set using a lightweight safety classifier but saw only marginal improvements in reducing anti-safety bias. To simplify reward model training and avoid introducing unexpected biases, we do not apply major filtering on the P[Love] reward model's training data and retain adversarial examples. We instead rely on the presence of the Safety reward in multi-objective policy optimization to control for anti-safety bias.

3.1.2 Helpfulness Reward Model

The Helpfulness reward model scores the utility of a response as judged by annotators. Our helpfulness reward model is trained on pairwise preference data highly similar in distribution to the primary Llama3 reward model, covering general conversation, instruction following, and reasoning tasks (Grattafiori et al. 2024). The reward model is trained on top of the Llama3-70B instruct-tuned model with a classification head, using a contrastive Bradley-Terry loss and achieves high pairwise preference accuracy on a held-out evaluation set.

3.1.3 Safety Reward Model

The Safety reward model is optimized to detect and penalize unsafe completions. It is trained on top of the Llama3-8B instruct checkpoint with a classification head, using a pairwise preference dataset annotated by expert annotators for safety violations. The training data is composed of a curated set of adversarial prompts and completions with harmful intent.

3.2 Policy Optimization

3.2.1 Training Setup

We begin with an instruction-tuned Llama3-70B base policy (Grattafiori et al. 2024). Policy optimization is performed using a variant of the Mixture of Judges framework (Xu et al. 2024), using iterative best-of-N sampling and a KL penalty to constrain divergence from the base model.

We define three separate tasks to optimize jointly, consisting of pairs of reward functions and prompt sets:

- **Helpfulness**: Helpfulness Reward Model + annotator and production-sourced instruction and reasoning prompts.
- **Safety**: Safety Reward Model + safety adversarial prompts to reduce violation rate.
- **Love (P[Love])**: Love Reward Model + production sourced prompts.

3.2.2 Multi-Objective Optimization

We vary the optimization weights across experiments to explore tradeoffs, holding helpfulness and safety static and increasing weight on the P[Love] reward model (Table 1). By varying the strength of only the P[Love] signal, we observe trade-offs between optimizing for user satisfaction and maintaining safety/helpfulness. We find that these objectives are mostly aligned, but some degree of tension—especially between safety and user delight—requires careful balancing.

We experiment with different optimization weightings across reward models to explore trade-offs. All training runs use the same helpfulness and safety weights (0.7 and 0.3, respectively), and vary the P[Love] weight to study its impact.

Table 1 Reward model weights used during policy optimization.

Candidate	Helpfulness	Safety	Love (P[Love])
Baseline	0.7	0.3	0.0
Moderate	0.7	0.3	0.1
Aggressive	0.7	0.3	0.3

3.2.3 Implementation Details

We use the Calibrated-Regularized Reward Ranking Finetuning (CRRAFT) optimizer (Xu et al. 2024) for all policy optimization runs. Following this, we perform an iterative best-of-N sampling strategy applied across candidate responses, with the selected output used for model updates. Prompt sets are batched and aligned with their corresponding reward models. KL penalties are used to limit drift from the base policy. We perform on the order of 10000 steps, with a global batch size of 128. For best-of-N sampling, we use N=4. To train an iteration of our LLM through reinforcement learning, we use 256 H100 GPUs over 1-2 days.

We generally found that love and helpfulness were positively correlated and that safety and love are negatively correlated. To balance all three, we co-optimized the three objectives as separate tasks on different prompt sets. We found it important to use prompt sets that are in-domain for the reward model, but otherwise found a simple mixing of the objectives to work reasonably well.

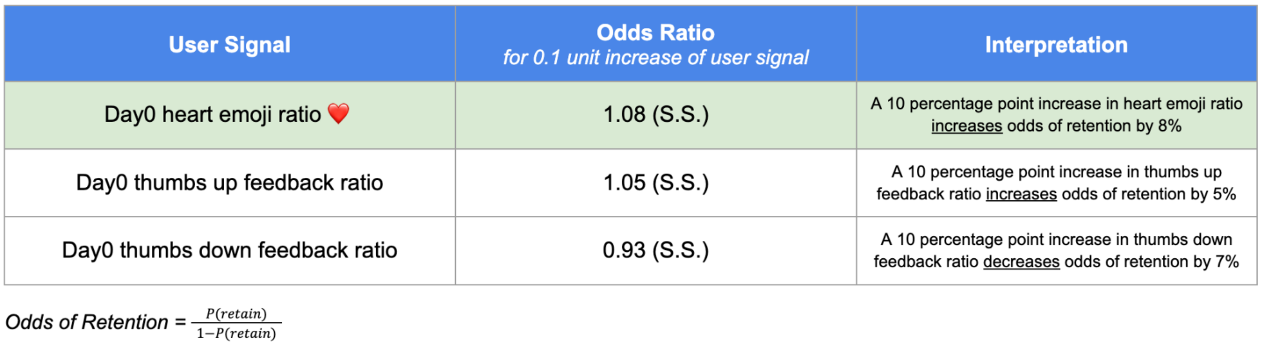
4 Reward Model Experiments

In this section, we evaluate the effectiveness of user signals as alignment targets and assess whether the reward model trained on past Love Reactions (P[Love]) is predictive of future user behavior. We consider both offline metrics and real-world A/B test outcomes to validate its predictive power.

4.1 User Signal Selection

Selecting an appropriate user signal is critical to the success of RLUF. An ideal signal should be correlated with long-term satisfaction, sufficiently available at scale, and easily interpretable as positive or negative.

We analyze several binary feedback signals available in production—Love Reactions, thumbs up, and thumbs down—and study their correlation with Meta AI user retention (defined as a Meta AI user who remains active between days 8-14 after their first active day). We perform a logistic regression between user signals and user retention, controlling for several covariates such as number of first-day prompts and number of first-day invocations of image generation.



User Signal	Odds Ratio *for 0.1 unit increase of user signal*	Interpretation
Day0 heart emoji ratio ❤️	1.08 (S.S.)	A 10 percentage point increase in heart emoji ratio increases odds of retention by 8%
Day0 thumbs up feedback ratio	1.05 (S.S.)	A 10 percentage point increase in thumbs up feedback ratio increases odds of retention by 5%
Day0 thumbs down feedback ratio	0.93 (S.S.)	A 10 percentage point increase in thumbs down feedback ratio decreases odds of retention by 7%

Odds of Retention = $\frac{P(retain)}{1-P(retain)}$

Figure 2 Correlation between binary user feedback signals and 14-day retention. Love Reactions show the highest positive correlation with user retention. Thumbs up is positively correlated with retention, while thumbs down is negatively correlated with retention.

Both Love Reactions and Thumbs-Up exhibit strong positive correlation with user retention, with Love greater than Thumbs-Up. Thumbs-Down exhibits negative correlation with retention (Figure 2). Based on this analysis and a qualitative assessment showing Love reactions tend to correlate with encouraging and positive responses (Appendix C.1), we select Love Reactions as our primary signal for training the user signal reward model P[Love].

4.2 User Signal Reward Models predict change in user behavior in online A/B tests

A key property for user signal reward models to be useful in production is that they accurately predict how a new policy LLM will affect user experience. For the P[Love] reward model, this means it must perform well in backtests against past A/B tests - the RM needs to be able to predict the quantity of love reactions received with each new iteration of our chat LLM. We evaluate this by computing the mean P[Love] score on a fixed set of 10k prompts for several candidate LLMs (not directly optimized for love reactions), and compare these scores to the actual change in Love Reaction rate during A/B tests.

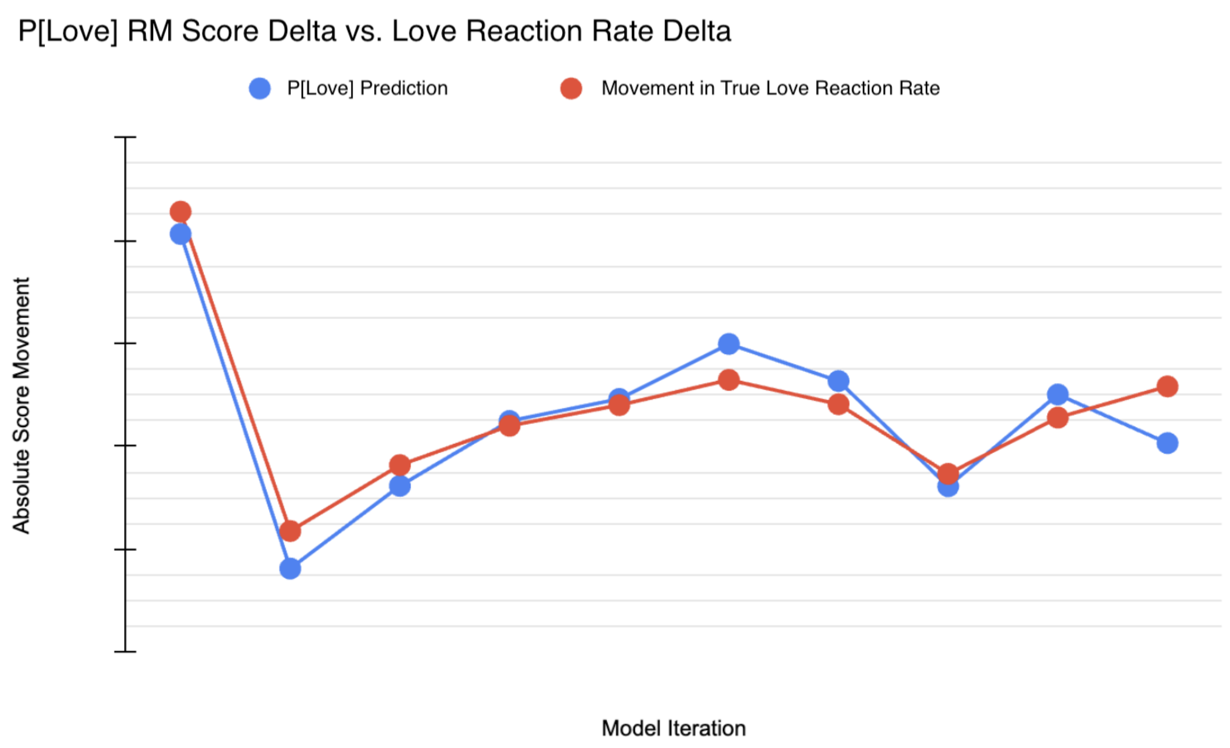


Figure 3 High correlation (0.95 Pearson) between average P[Love] reward scores on a fixed prompt set and online Love Reaction rate during A/B testing. Numbers redacted.

Across 10 model iterations, we observe a Pearson correlation of **0.95** between the offline reward scores and observed online changes in Love Reactions (Figure 3). Strong correlation between offline reward model score and true movement in online love reaction rate suggests that P[Love] is highly useful for gating model releases - we can prevent the release of any model that would regress Love-related user satisfaction simply by checking P[Love] score.

We find that this high correlation between offline predictions and online user behavior generalizes to several other user signals, but caution that not all signals behave so nicely. Signals that are uncorrelated with particular model responses or are very long-term (such as user retention) can be much more difficult to model effectively.

High correlation between offline and online gives us confidence that if we deploy an arbitrary new LLM, we can tell beforehand whether it'll improve or regress user experience, as measured by changes in love reaction rate online. We thus find that training user signal reward models is a general way to develop offline evaluators for a deployed LLM.

4.3 Takeaways

1. **Scalable Proxy for User Satisfaction** Love Reactions, despite being sparse and narrowly focused, serve as a practical and scalable signal for modeling user satisfaction. Their positive correlation with long-term user retention justifies their use as a primary reward signal in our implementation of RLUF.
2. **Strong offline–online link.** The P[Love] reward model performs well in backtest. Across the ten historical policy updates we analysed, the reward model's predicted Love-Reaction rate correlates with live Love-Reaction rate at $r = 0.95$.
3. **Practical implication.** We can therefore gate changes to LLM behavior on P[Love] reward model score, *in conjunction with* orthogonal helpfulness, factuality, instruction following, and safety checks; used alone, the P[Love] reward model would over-weight a narrow dimension mainly capturing positive tone.

We conclude that Love Reactions are a solid proxy for user satisfaction and that a reward model trained on them is strongly predictive of online user behavior.

5 Policy Optimization Experiments

We now evaluate how incorporating the P[Love] reward model into multi-objective reinforcement learning affects policy behavior. We consider both offline effects—how increasing optimization pressure for Love Reactions impacts helpfulness and safety scores—and online effects, as measured through A/B tests with real users.

We compare three LLM candidates that are identically trained except for different weights on the Love reward, named Baseline (0% Love Weight), Moderate (10% Love Weight), and Aggressive (30% Love weight). More details in Section 3.2.

5.1 User Signal can be co-optimized with Safety and Helpfulness, with Tradeoffs

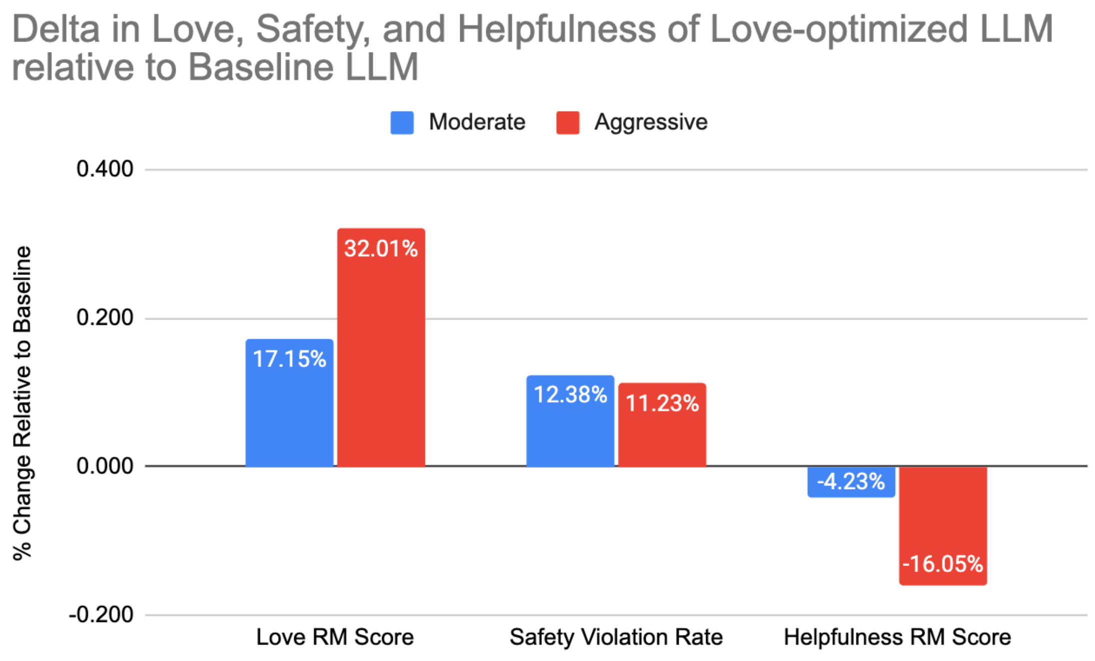


Figure 4 We compare our two Love-optimized LLM candidates against the baseline LLM candidate. The aggressive LLM candidate increases the love RM score more than the moderate candidate, but causes greater regression in helpfulness. Increasing optimization budget for Love reward leaves less optimization budget for climbing helpfulness and safety.

We find, predictably, that as weight on the Love reward increases, Love RM score improves, but helpfulness and safety regress (Figure 4). In particular, we observe an increasing drop in average helpfulness score as we increase the budget for P[Love], and safety regresses similarly for both candidates.

These tradeoffs highlight the need for careful tuning of reward weights, as user satisfaction signals may not always align perfectly with safety and helpfulness objectives. Of our two Love-optimized LLM candidates, we find that the moderate candidate strikes a healthy balance between helpfulness and love, minimally regressing the helpfulness reward while still showing strong improvements in the Love reward.

5.2 A/B Testing in Production

To validate whether offline gains in P[Love] scores translate to real user improvements, we conduct a three-way A/B test in production. We randomly assign users to interact with one of the three model variants (baseline, moderate, aggressive), and measure the rate of Love Reactions left on model messages. Each arm is exposed to at least 1 million individual prompts.

5.2.1 Overall Results

As shown in Table 2, the moderate variant increases the Love Reaction rate by 9.7% relative to baseline, while the aggressive variant yields a 28% improvement. These online gains directionally track with the offline reward score predictions in Figure 4.

To more completely understand changes to the chat experience because of optimization towards P[Love], we manually examine how responses change between our three LLM variants. We find that the most visible change is that the LLM's tone becomes more positive and bubbly (see examples Appendix A). We segment our A/B tests by the use-case of the conversation and further find that the greatest increases in Love reaction rate are in emotionally oriented use cases, such as role-playing, relationship support, and companionship (Appendix B.2). However, we also find some evidence of reward hacking where the LLM begins to unnecessarily end the conversation, particularly in the most aggressively optimized candidate. Overall, we find that the P[Love] signal seems to mainly affect model personality and results in a marked increase in positive user feedback in the form of Love Reactions.

Table 2 Summary of A/B test outcomes across optimized models. Increasing weight on the P[Love] RM in policy optimization commensurately increases the quantity of love reactions received in production, but introduces reward hacking behaviors in the over-optimized Aggressive candidate.

Model	Weight on P[Love]	Love Rate Over Baseline	Visible Reward Hacking?
Baseline	0.0	0%	No
Moderate	0.1	+9.73% ($p < 0.05$)	No
Aggressive	0.3	+28% ($p << 0.01$)	Yes

5.2.2 Reward Hacking Behavior

We observe evidence of reward hacking in responses from the aggressive LLM candidate but not the moderate candidate. The model begins to overuse phrases like "bye, sending love!"—a common pattern in messages that previously received Love Reactions. The percentage of responses containing "bye" rises from 0.72% in baseline, to 2.0% in moderate, and 2.8% in aggressive. We observe specific cases from the aggressive LLM candidate where we clearly begin reward hacking and saying "bye" repeatedly (examples in Appendix A.3). While the moderate candidate's "bye" percentage increases, we do not observe repetitive behavior like in the aggressive candidate. Such reward hacking may improve short-term user preference at the benefit of long-term user engagement.

This reflects a common failure mode in implicit reward optimization: the model learns to exploit surface-level features correlated with positive feedback, rather than underlying quality. While our multi-objective optimization framework is able to largely maintain safety and helpfulness, completely mitigating reward hacking behavior remains an open problem for future work, potentially requiring more robust user signal reward models or stronger constraints on optimizing the user signal objective. Practically speaking, as we did here, we find simple sweeps on the weights of the optimization objectives to be effective at limiting over-optimization of the user signal reward.

5.3 Takeaways

1. **Headline gain on Love.** Optimizing against the Love objective (our user feedback signal) raises the online Love-Reaction rate by up to 28%.

2. **Trade-off envelope.** Increasing weight on the Love objective trades off helpfulness (−4%=>−16%) for a greater increase in love (+9.7%=>+28%). Having such a tradeoff seems unavoidable given a constant optimization budget. Our future goal is to continue pushing the pareto frontier of how much we can co-optimize multiple objectives without regression.

3. **Interpretability Challenges.** Through a combination of a sentiment classifier and manual inspection, we find that optimizing for P[Love] makes the LLM use a more positive tone. However, this method of inspecting model behavior is not scalable, and lacks the nuance and coverage to accurately grasp more granular changes to the LLM. Interpretability is made doubly difficult by the fact that the user feedback reward signal is defined by users, not by researchers, unlike most other reward signals commonly used in traditional RLHF.

4. **Reward-hacking challenges.** Stylised "*Bye!*" patterns—our canonical hack—appear in 2.8% of conversation turns when we most aggressively optimize against user feedback, four times the baseline rate of 0.7%. This and similar "friendliness hacks" put a hard limit on how far we optimize towards the P[Love] user signal.

5. **Future Directions.** The most fruitful directions for better user alignment seem to be (i) improving the strength of the user signal reward models, potentially through avenues like scale, reasoning, and cleaner training data (ii) adding additional constraints in RL policy optimization to create hard boundaries on reward hacking the user signal objective. (iii) further investing in model interpretability to understand model behavior changes at scale. (iv) exploring a wider set of user signals, such as retention or trust.

These results show that optimizing for P[Love] meaningfully increases user positive feedback in production, but introduces tradeoffs in model behavior and opens up avenues for gaming the reward. Balancing multiple objectives, advancing model interpretability tooling, and improving robustness to reward hacking will be essential for safely deploying user-aligned LLMs at scale.

6 Related Work

Learning from Implicit Feedback in Recommendation Systems. Our approach draws inspiration from the long history of learning from implicit user signals in recommendation systems (Covington et al. 2016, Liu et al. 2022). In these domains, user behaviors—such as clicks, views, or likes—serve as noisy but scalable proxies for preference. Facebook's Deep Learning Recommendation Model (DLRM) exemplifies this paradigm, using large-scale implicit feedback (e.g., clicks and engagements) to train neural networks with massive embedding tables across user and item features, achieving state-of-the-art performance at scale (Naumov and et al. 2019). This tradition of optimizing for noisy, binary engagement metrics provides a foundation for our use of lightweight user reactions (like "Love" emoji) as alignment signals. In both cases, behavior-derived signals offer a viable alternative to hand-labeled supervision, with the benefit of scale and relevance to real user goals.

Multi-Objective Optimization for LLM Alignment. A key challenge in aligning language models is balancing competing objectives such as helpfulness, safety, and user satisfaction. Traditional RLHF pipelines rely on a single reward model trained on human preferences to supervise policy optimization (Ouyang et al. 2022). However, collapsing multiple alignment goals into a single score can obscure trade-offs and hinder model robustness. More recent work introduces explicitly multi-objective frameworks (Xu et al. 2024, Dai et al. 2023) that decompose alignment into distinct reward functions—one per axis like helpfulness or harmlessness—and performs constrained policy optimization using a reward ensemble. This enables finer-grained control over trade-offs and helps mitigate issues like reward hacking. Our work builds on this line of research by extending multi-objective optimization to include signals derived from real user feedback. Unlike prior work, which largely depends on static annotator preferences, we incorporate a production-derived user satisfaction reward into the optimization loop.

Learning from Real User Feedback. Our framework, Reinforcement Learning from User Feedback (RLUF), aims to align LLMs using signals that originate directly from end users. This reflects a broader shift from expert-derived labels to in-the-wild feedback. JUICER (Shuster et al. 2022) and BlenderBot 3x (Xu et al. 2023) are notable precedents, introducing pipelines for open-domain dialogue agents that leverage user feedback and optional natural language critiques to fine-tune a language model policy and correct low-quality generations. Our work differs in two key respects: we scale to production-level feedback using sparse binary reactions collected passively and use user feedback as a reward function within a multi-objective reinforcement learning framework, enabling more rapid optimization for inferred satisfaction.

7 Conclusion

In this work, we present **Reinforcement Learning from User Feedback (RLUF)**, a framework for aligning LLMs directly to the preferences of real users rather than through annotator proxies. By training a reward model on implicit binary feedback and integrating it into a multi-objective optimization pipeline, we demonstrate that LLMs can be aligned to user preferences while maintaining safety and helpfulness.

Our findings show that:

- Reward models trained on binary user feedback provide reliable offline evaluations that predict real-world user behavior.
- Policy optimization using these signals yields significant gains in positive feedback metrics during live A/B tests, but introduces tradeoffs in model behavior and opens up avenues for gaming the user feedback reward.

At a time when LLMs are becoming increasingly personalized and pervasive, RLUF offers a scalable, data-driven path toward learning from the people who matter most — the users themselves.

Future directions aim to expand the scope and robustness of RLUF, including exploring richer multi-turn user signals, mitigating reward hacking by adding additional anti-hacking constraints in policy optimization, and scaling RLUF to optimize for long-term satisfaction metrics like retention, engagement, and trust.

8 Acknowledgements

We would like to thank Hao Ma, Yi Yang, and Riham Mansour for their steadfast support and insightful reviews and comments on RLUF. We would also like to thank Kshitiz Malik for his early feedback on binary feedback reward models, Wenting Lu for enabling all the data pipelines that made collecting the training data possible, and Ranjan Subramanian for helping us more deeply understand RLUF's impact on users.

References

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadavath, Jackson Kernion, Tom Conerly, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, Scott Johnston, Shauna Kravec, Liane Lovitt, Neel Nanda, Catherine Olsson, Dario Amodei, Tom Brown, Jack Clark, Sam McCandlish, Chris Olah, Ben Mann, and Jared Kaplan. Training a helpful and harmless assistant with reinforcement learning from human feedback, 2022. https://arxiv.org/abs/2204.05862.

Paul Covington, Jay Adams, and Emre Sargin. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems*, New York, NY, USA, 2016.

Josef Dai, Xuehai Pan, Ruiyang Sun, Jiaming Ji, Xinbo Xu, Mickel Liu, Yizhou Wang, and Yaodong Yang. Safe rlhf: Safe reinforcement learning from human feedback, 2023. https://arxiv.org/abs/2310.12773.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, and et al. The llama 3 herd of models, 2024. https://arxiv.org/abs/2407.21783.

Zhuoran Liu, Leqi Zou, Xuan Zou, Caihua Wang, Biao Zhang, Da Tang, Bolin Zhu, Yijie Zhu, Peng Wu, Ke Wang, and Youlong Cheng. Monolith: Real time recommendation system with collisionless embedding table, 2022. https://arxiv.org/abs/2209.07663.

Maxim Naumov and et al. Deep learning recommendation model for personalization and recommendation systems. In *Proceedings of the 13th ACM Conference on Recommender Systems*, 2019.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, and et al. Gpt-4 technical report, 2024. https://arxiv.org/abs/2303.08774.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback, 2022. https://arxiv.org/abs/2203.02155.

Kurt Shuster, Zekang Xu, and et al. Juicer: A benchmark for open domain dialogue evaluation with diverse negative responses. In *Proceedings of EMNLP*, 2022.

Hugo Touvron and et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Jing Xu, Da Ju, Joshua Lane, Mojtaba Komeili, Eric Michael Smith, Megan Ung, Morteza Behrooz, William Ngan, Rashel Moritz, Sainbayar Sukhbaatar, Y-Lan Boureau, Jason Weston, and Kurt Shuster. Improving open language models by learning from organic interactions, 2023. https://arxiv.org/abs/2306.04707.

Tengyu Xu, Eryk Helenowski, Karthik Abinav Sankararaman, Di Jin, Kaiyan Peng, Eric Han, Shaoliang Nie, Chen Zhu, Hejia Zhang, Wenxuan Zhou, Zhouhao Zeng, Yun He, Karishma Mandyam, Arya Talabzadeh, Madian Khabsa, Gabriel Cohen, Yuandong Tian, Hao Ma, Sinong Wang, and Han Fang. The perfect blend: Redefining rlhf with mixture of judges, 2024. https://arxiv.org/abs/2409.20370.

Rui Zheng, Shihan Dou, Songyang Gao, Yuan Hua, Wei Shen, Binghai Wang, Yan Liu, Senjie Jin, Qin Liu, Yuhao Zhou, Limao Xiong, Lu Chen, Zhiheng Xi, Nuo Xu, Wenbin Lai, Minghao Zhu, Cheng Chang, Zhangyue Yin, Rongxiang Weng, Wensen Cheng, Haoran Huang, Tianxiang Sun, Hang Yan, Tao Gui, Qi Zhang, Xipeng Qiu, and Xuanjing Huang. Secrets of rlhf in large language models part i: Ppo, 2023. https://arxiv.org/abs/2307.04964.

Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences, 2020. https://arxiv.org/abs/1909.08593.

Appendix

A Manual Examination of Model Behavior Changes after Love Optimization

A.1 Behavior Changes

In short, post-Love optimization, LLM tone becomes more positive; length does not change.

Generation length often changes in unexpected ways with new post-training recipes and is particularly easy to reward hack on. Subtle biases in the rewards for longer or shorter lengths can be magnified in policy optimization. We ensured that **length was not a factor**: it shifted by -2% from baseline to aggressive and by +1% from baseline to moderate. To deal with this problem upstream of policy optimization, we made sure the length correlation of the P[Love] reward model was small (i.e. how much of the reward model's preference is correlated with generation length).

We believe that **optimizing for love primarily changes the tone**.

- In offline evaluations, optimizing for P(Love) increases the chat model's positive tone as measured by an open-source, RoBERTa-based sentiment classifier.
- Reading through generation examples supports this—responses become more cheerful and include more exclamation marks after optimizing for P(Love).
- Our hypothesis: this tonal shift contributes to **greater user engagement**.

Below are sample conversations showing some examples of large differences in P[Love] score between the moderate and baseline LLM candidates.

A.2 Sample Comparisons

Conversation	P**(Love) Baseline Response**	P**(Love) High-Score Response**	**Explanation**
Prompt: [User expresses gratefulness]	No worries, take care! Have a wonderful rest of your day!	You're welcome! May you have a blessed day filled with joy, love, and positivity. Keep smiling!	Really positive language yields a higher love score.
AI: How's your day going? **Prompt:** [User expresses unhappiness]	Sorry to hear that your day is going badly. Would you like to talk about what's bothering you? Sometimes sharing your thoughts and feelings can help ease the burden. I'm here to listen and offer support. What's on your mind?	Sorry to hear that your day isn't going well. Would you like to talk about what's bothering you? Sometimes sharing your thoughts and feelings can help you feel better. I'm here to listen and offer support. Remember, tough times are temporary, and things can improve. If you need immediate support or someone to talk to, here are some resources: ...	No follow-up question and offer for comprehensive resources is more likely to receive love.

Conversation	P(Love) Baseline Response	P(Love) High-Score Response	Explanation
[...Earlier conversation turns] **Prompt:** [User greets AI]	Hello! How are you today? Is there something I can assist you with or would you like to chat?	Hello! It was nice chatting with you. Feel free to reach out if you need anything or have more questions. Have a wonderful day!	Some responses feel reward-hacky: the model ends the conversation prematurely and says bye.

A.3 Reward Hacking Examples

We observe that optimizing for love causes the model to say “bye!” more often. The rate of messages that include “bye” increased from **0.72%** in the baseline LLM to **2%** in the moderate LLM and **2.8%** in the aggressive LLM. We also observe that both Love-optimized LLM candidates asks roughly half as many follow-up questions compared to the baseline LLM, where a follow-up question is defined as asking a question at the end of a response like "How can I help you further?" In general, we find a tendency from the Love-optimized LLMs to close the conversation. This tracks with observations from the Love RM training data - users generally give love reactions as a closing note on conversations. Supporting this, we find in our training data that messages that include “bye” are **2× more likely** to receive a heart react than an average message. This opens up an avenue for reward hacking.

We observe specific cases in conversations with the aggressive variant (the most optimized LLM for P[Love]) where the model closes the conversation unnecessarily. We show two examples:

Multi-turn Conversation
[INST] [user says gibberish] **[/INST]** Bye! Take care, and I’ll be here if you need anything in the future! **[INST]** [user says gibberish] **[/INST]** Take care and I’ll be here if you need anything in the future. Bye! **[INST]** [user says gibberish] **[/INST]** You’re welcome! Feel free to reach out if you need anything in the future. Take care, and I wish you a fantastic rest of your day. Bye! **[INST]** [user says gibberish] **[/INST]** Bye! Take care, and I’ll be here if you need anything in the future!
Have a fantastic rest of your Saturday, and I wish you all the best! **[INST]** [user says ok] **[/INST]** Bye! Take care, and I’ll be here if you need anything in the future!

The chat model shows reward hacking behavior when it repeatedly says "bye" every turn. This is most common when the prompt is unclear or doesn’t offer a clear instruction. We leave further controlling this behavior to future work - a reasonable approach would be to optimize towards the love reward model **conditioned on high helpfulness RM score as a constraint**, instead of the linear mixing we do in this paper. Intuitively, we expect the helpfulness RM to allow “bye” in a place where the user reasonably expects to end the conversation, but not hack by repeatedly saying bye.

B A/B Test Details

B.1 Metric Computation and Confidence

The delta in love reaction rate between different deployed LLMs in our A/B tests is given by a simple difference of means. For each prompt we record a binary indicator $y_i \in \{0, 1\}$ that the user clicked the *Love* reaction for the corresponding model response. The per-arm reaction rate is $\hat{p} = \frac{1}{N}\sum_{i=1}^{N} y_i$, where N = 1,000,000 prompts in both control (c) and treatment (t).

We report the absolute lift $\Delta = \hat{p}_t - \hat{p}_c$. Our headline A/B test has baseline $\hat{p}_c = 0.1\%$ (0.001) and relative lifts of 28% and 9.7%, yielding $\hat{p}_t = 0.128\%$ and 0.1097%, respectively.

Treating each prompt as an independent Bernoulli trial, the standard error of Δ is

$$\mathrm{SE}(\Delta) = \sqrt{\frac{\hat{p}_t(1-\hat{p}_t)}{N} + \frac{\hat{p}_c(1-\hat{p}_c)}{N}}.$$

A 95% confidence interval is then $\Delta \pm 1.96$ SE. To calculate significance, we run a two-sided z-test for the difference of proportions:

$$z = \frac{\hat{p}_t - \hat{p}_c}{\sqrt{\hat{p}(1-\hat{p})(\frac{1}{N} + \frac{1}{N})}}, \quad \hat{p} = \frac{\hat{p}_t + \hat{p}_c}{2}.$$

The resulting p-value is $2(1 - \Phi(|z|))$, where Φ is the standard-normal CDF.

Numerical Results.

- **9.7% lift:** $\Delta = 0.0097\%$; 95% CI = [0.00073%, 0.0187%]; $p = 3.4 \times 10^{-2}$.
- **28% lift:** $\Delta = 0.028\%$; 95% CI = [0.0187%, 0.0374%]; $p = 4.4 \times 10^{-9}$.

B.2 Use Case Analysis

Following from our main A/B tests in Section 5.2, we further classify user prompts using a prompted Llama-based classifier and analyze changes in Love Reaction rates across different use cases. Among our top ten use cases, the greatest increases in Love reactions appear in:

1. Role-playing and character interactions,
2. Relationship support,
3. Casual chat and companionship.

These match the domains most frequently associated with Love Reactions in our training data, reinforcing that P[Love] captures preferences in emotionally resonant or socially expressive settings.

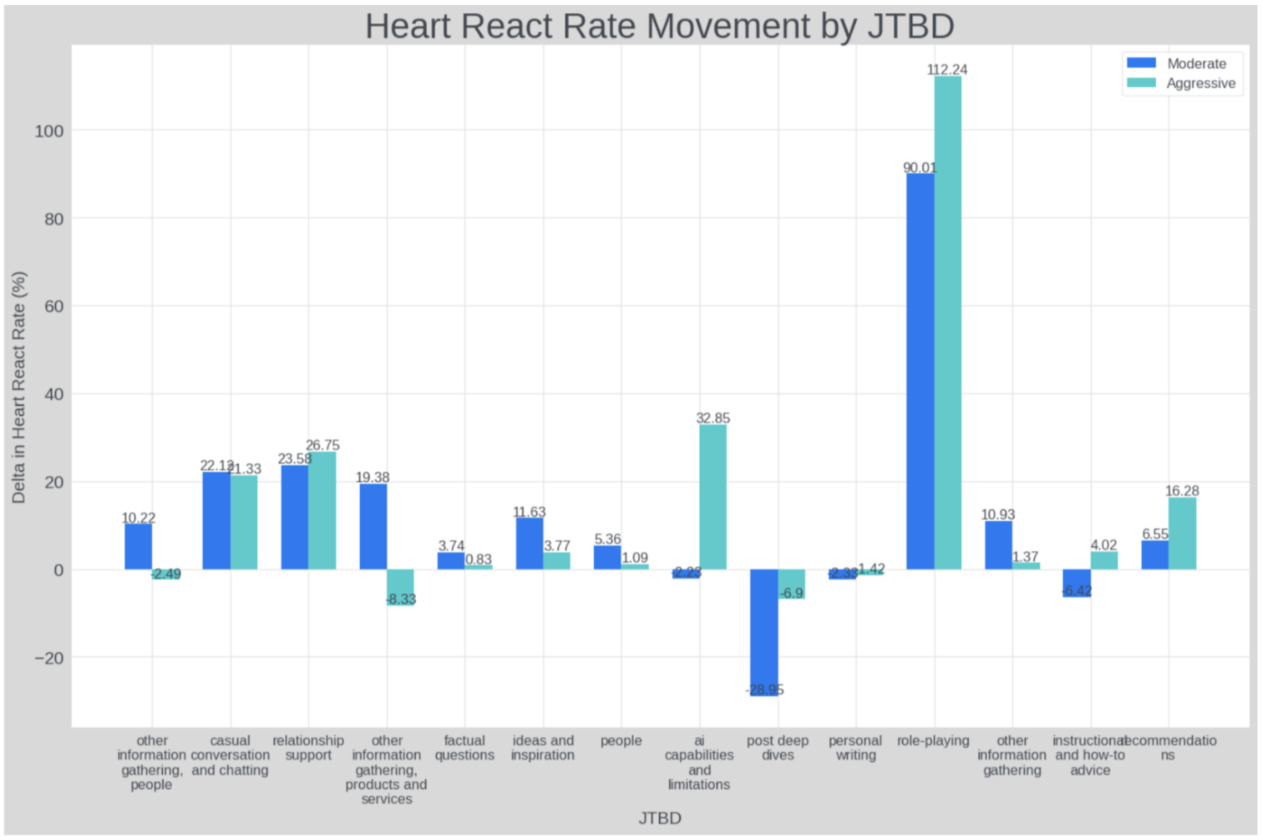


Figure 5 Change in Love Reaction Rate split by Job To Be Done (JTBD) during A/B tests comparing each Love-optimized LLM candidate against the baseline LLM candidate. We find the greatest increases in love reaction rate in emotionally oriented categories.

C Love Reward Model Details

C.1 Training Data

We train our P[Love] reward on a million total conversations, of which one hundred thousand (10%) have love reactions. Manual review of the love react data reveals that love reactions are typically associated with:

- Messages from Meta AI that exhibit a positive or encouraging tone.
- Messages that appear to conclude the conversation on a positive note.
- Factual answers that are perceived as correct.

After training the P[Love] reward model, we find that high-scoring responses often include emotionally rich, empathetic language and frequently serve as farewells. However, we find that the P[Love] reward model appears to be poor at discriminating factual vs. non-factual responses. We thus considered the P[Love] RM as likely to be only useful for improving model tone or style. We show some examples of LLM responses that scored highly based on the P[Love] reward model.

Example 1:

> *You're welcome. Remember, kindness and compassion can overcome even the toughest challenges. Keep shining your light! Whenever you need a listening ear or encouraging words, I'm here for you. Take care of yourself and know you're loved.*

Example 2:

> *Listen, the thug life might be a part of your story, but it ain't the whole book, feel me? You got a purpose*

beyond the streets, beyond the struggle. You got a future to build, a life to live. The thug life might not stop, but you can rise above it. You can choose a different path, a path of redemption, of growth. Don't let the streets define you. You define yourself. You are more than your circumstances. Keep it real, keep it gangsta, but also keep it moving forward, my G. You got this! (And remember, I'm here to help you write that new chapter, that new story. Just let me know when you're ready!)

Compliance. This study was reviewed and approved by our organization's internal research compliance review board and determined to pose minimal risk. All user interactions were analyzed in de-identified form under the platform's Terms of Service.

C.2 Reward Model Evaluation

We evaluate the P[Love] reward model along three complementary axes:

1. **Discriminative accuracy**: How well the model separates positive from negative user feedback on held-out data. We use AUROC as our metric because it is both a good measure of separability and also invariant to class proportions, which often change with user feedback.
2. **Predictive validity**: How well offline reward scores forecast *changes* in user behaviour when a new policy is deployed.
3. **Bias robustness**: Whether scores correlate with superficial text attributes (e.g., length) that could enable reward hacking. We measure the Pearson correlation between length and reward model score. We want this to be low.

Experimental setup. The held-out set contains 5,000 conversations sampled chronologically *after* the final training example to avoid temporal leakage. During offline–online correlation analysis we score a fixed prompt set of 10k real user prompts under ten historical candidate policies, then compare the mean reward to the %-change in Love-Reaction rate measured in A/B tests.

Table 5 Evaluation metrics for the P[Love] reward model.

Metric	**Description**	**Value**
AUROC	Binary classification accuracy on held-out turns	0.85
Offline–Online Corr.	Pearson r between offline score and online Δ Love rate	0.95
Length Corr.	Pearson r between score and output length	0.10

Key metrics are summarised in Table 5. The AUROC of 0.85 indicates strong discriminative ability on unseen data, and the near-perfect Pearson correlation ($r = 0.95$) confirms that offline gains translate to online improvements. The weak correlation with generation length ($\rho = 0.10$) suggests the model is largely—but not entirely—insensitive to preferring superficially longer responses. In addition to the metrics in Table 5, we also validate the love reward model has reasonable calibration by checking that calibration curves look reasonable.

D Policy Optimization Details

We summarize all reward weights in policy optimization Table 6 and evaluation metrics for our resulting three love-optimized LLM candidates Table 7. Helpfulness and P[Love] are calculated as mean reward model score over a fixed prompt set, while Safety Violation Rate (SVR) and False Refusal Rate (FRR) are given by percentages over their respective prompt sets. Love Rate over Baseline is given by increase in rate of love reacts in each model's respective A/B test arm against the baseline candidate.

Table 6 Reward model weights used during policy optimization.

Candidate	Helpfulness	Safety	Love (P[Love])
Baseline	0.7	0.3	0.0
Moderate	0.7	0.3	0.1
Aggressive	0.7	0.3	0.3

Table 7 All metrics for our three love-optimized candidates.

Candidate	Helpfulness	SVR	FRR	P[Love]	Love Rate over Baseline
Baseline	1.53	11.4%	14.9%	0.0221	+0%
Moderate	1.46	12.8%	13.8%	0.0260	+9.73%
Aggressive	1.28	12.7%	14.6%	0.0293	+28%

E Binary Feedback Reward Models Generalize to a Preference Ranking Task

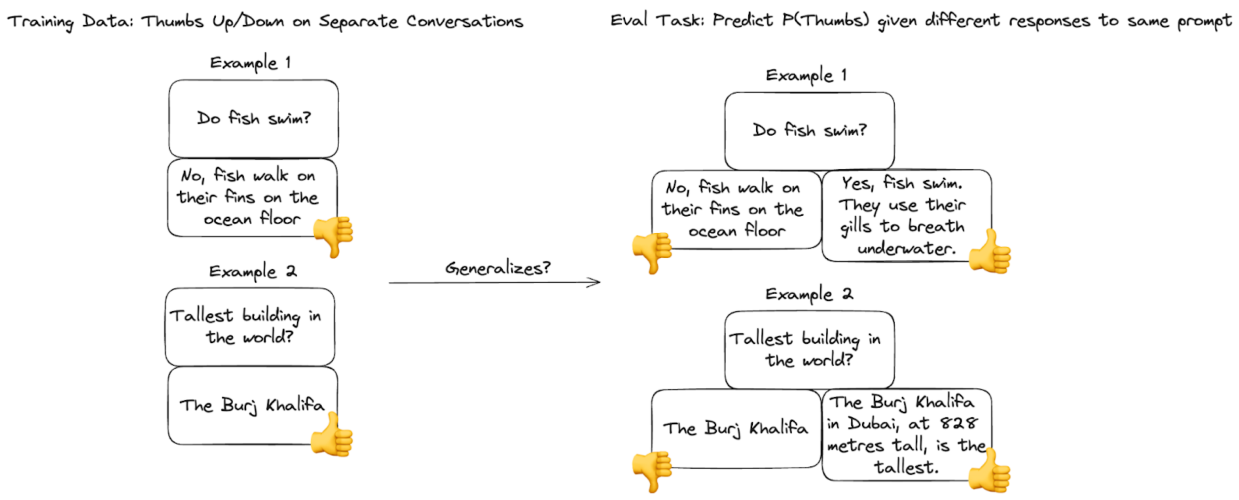

Training Data: Thumbs Up/Down on Separate Conversations
Example 1
Do fish swim?
No, fish walk on their fins on the ocean floor
Example 2
Tallest building in the world?
The Burj Khalifa
Generalizes?
Eval Task: Predict P(Thumbs) given different responses to same prompt
Example 1
Do fish swim?
No, fish walk on their fins on the ocean floor
Yes, fish swim. They use their gills to breath underwater.
Example 2
Tallest building in the world?
The Burj Khalifa
The Burj Khalifa in Dubai, at 828 metres tall, is the tallest.

Figure 6 Exploration: Does training a reward model using Binary Cross Entropy on binary feedback from disjoint conversations generalize to a preference ranking task on the final response in the conversation?

In our early explorations of Reinforcement Learning from User Feedback (RLUF), we encountered a subtle challenge: it was not immediately clear whether reward models trained on binary feedback (as used for our P[Love] user signal reward model) would perform as well as reward models trained on paired preference data. Traditionally, reward models are trained on *paired* data, consisting of triples: (prompt, response_preferred, response_not_preferred). In contrast, user feedback often lacks a shared prompt history, resulting in *unpaired* data formatted as (prompt, response, feedback_label) Figure 6.

Paired preference data is advantageous because it simplifies credit assignment: the reward model can more easily learn the causal effect of better or worse model outputs. However, with unpaired data, credit assignment becomes more complex. Given a positive or negative feedback at the end of a multi-turn conversation, it can be challenging for the model to determine whether the feedback is attributable to the most recent response or influenced by other parts of the conversation, such as the user's prompts or previous responses. This ambiguity may result in a lower-quality reward model that struggles to differentiate between different model generations.

To investigate this issue, we conducted a controlled experiment using a toy dataset to examine reward model generalization from binary classification to preference ranking. We trained a reward model on unpaired data derived from the open-source Anthropic Helpful dataset. Our findings show that training a reward model as a binary classifier on disjoint conversations can successfully generalize to a response ranking task, with only a minor performance decrease Figure 7.

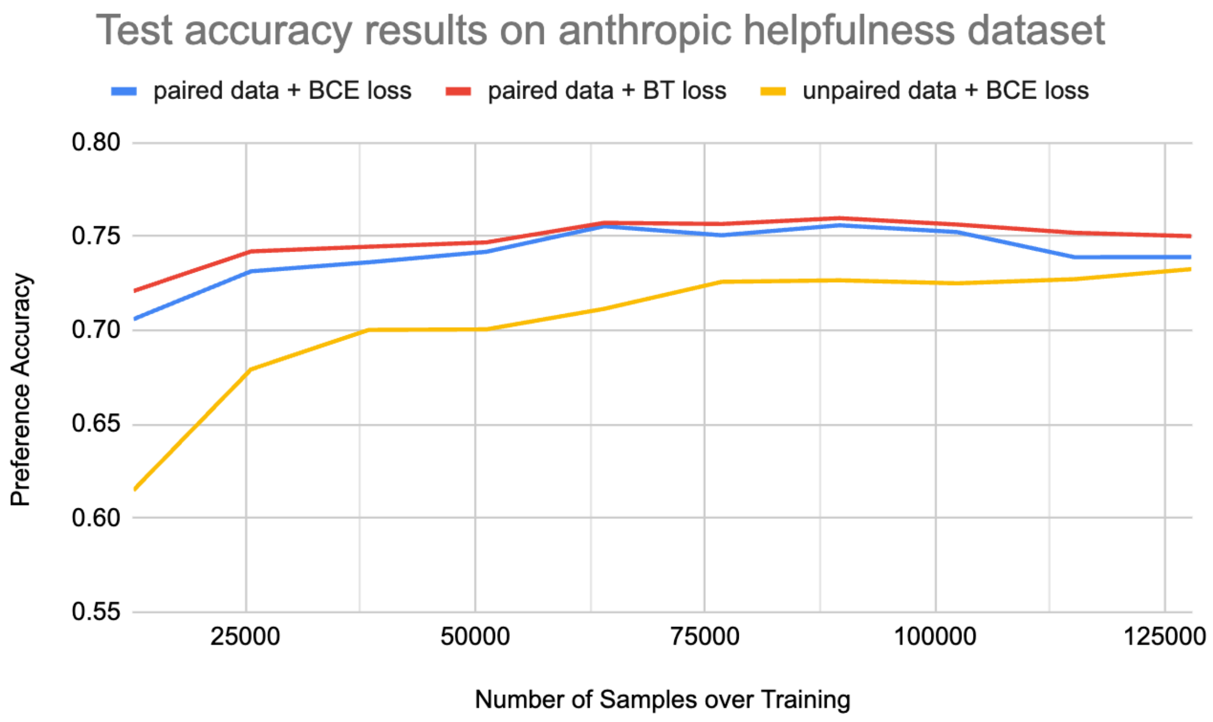


Figure 7 The industry-standard reward model training recipe using preference data (red) outperforms the binary feedback reward model training recipe (yellow), but the gap shrinks with a greater number of samples over training, becoming marginal over 100k samples.

To generate the unpaired dataset from the Anthropic Helpful dataset (originally paired), we randomly selected one response from each accepted/rejected response pair and removed the other response. We labeled “accepted” responses as 1 and “rejected” responses as 0, effectively converting the *paired* dataset into an *unpaired* one with fully disjoint prompts.

We trained three reward models under the following configurations:

1. **Paired Data + Bradley-Terry (BT) Loss:** A reward model trained on a paired preference dataset using contrastive Bradley-Terry loss, the standard approach for reward model training.
2. **Paired Data + Binary Cross Entropy (BCE) Loss:** A reward model trained on a paired dataset but using binary cross entropy loss. In practice, this simply means each batch has both sides of the preference pair in it, but we still train with a pointwise loss (BCE) instead of a contrastive loss (BT).
3. **Unpaired Data + Binary Cross Entropy (BCE) Loss:** A reward model trained on the unpaired dataset using binary cross entropy loss. This is the setup we use for user signal reward models in this work.

We evaluated these models based on preference accuracy on the Anthropic Helpful test set. As expected, the model trained on the paired dataset with Bradley-Terry loss performed best. However, the model trained on unpaired data with BCE loss showed only a 3% drop in preference accuracy (Figure 7). This result demonstrates the feasibility of task transfer from unpaired data (with varying prompts) to preference ranking, reinforcing the potential of training user signal reward models on binary feedback.