Time-Reversal Provides Unsupervised Feedback to LLMs

Varun Yerram * † §
Google DeepMind

Rahul Madhavan * ‡
Indian Institute of Science

Sravanti Addepalli * † §
Google DeepMind

Arun Suggala †
Google DeepMind

Karthikeyan Shanmugam † §
Google DeepMind

Prateek Jain †
Google DeepMind

Abstract

Large Language Models (LLMs) are typically trained to predict in the forward direction of time. However, recent works have shown that prompting these models to look back and critique their own generations can produce useful feedback. Motivated by this, we explore the question of whether LLMs can be empowered to think (predict and score) backwards to provide unsupervised feedback that complements forward LLMs. Towards this, we introduce Time Reversed Language Models (`TRLMs`), which can score and generate queries when conditioned on responses, effectively functioning in the reverse direction of time. Further, to effectively infer in the response to query direction, we pre-train and fine-tune a language model (`TRLM-Ba`) in the reverse token order from scratch. We show empirically (and theoretically in a stylized setting) that time-reversed models can indeed complement forward model predictions when used to score the query given response for re-ranking multiple forward generations. We obtain up to 5% improvement on the widely used `AlpacaEval` Leaderboard over the competent baseline of best-of-N re-ranking using self log-perplexity scores. We further show that `TRLM` scoring outperforms conventional forward scoring of response given query, resulting in significant gains in applications such as citation generation and passage retrieval. We next leverage the generative ability of `TRLM` to *augment* or provide unsupervised feedback to input safety filters of LLMs, demonstrating a drastic reduction in false negative rate with negligible impact on false positive rates against several attacks published on the popular JailbreakBench leaderboard.

1 Introduction

Large Language Models (LLMs) trained on a large corpora of text are able to accomplish a wide variety of downstream tasks such as summarization, open-ended/ context-based question answering, document retrieval, and citation generation [Brown et al., 2020, Zhao et al., 2023a]. While the generations from pre-trained and instruction-tuned models already show significant promise, alignment techniques such as Reinforcement Learning via Human Feedback (RLHF) [Anil et al., 2023a, Ouyang et al., 2022] are widely used to improve the quality of their generations further. However, these methods rely heavily on additional supervision to construct preference data, which can be expensive to acquire, or noisy for training. This brings up a natural question – *Can we generate useful feedback on LLM generations without additional supervised data?*

*Equal Contribution.

†Work done as part of Google Research

‡Work done as a Student Researcher at Google Research

§Correspondence to: vyerram@google.com, sravantia@google.com, karthikeyanvs@google.com

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

A recent line of work aims at *specially prompting* LLMs to review their own generations and generate meaningful natural language feedback, which can subsequently be used to refine them [Madaan et al., 2024]. This process can be repeated to improve the generations iteratively. The success of such methods serves as an evidence that it is indeed possible to obtain better responses without additional supervision. However, such methods rely on the superior instruction following and reasoning abilities of LLMs, which may not necessarily hold for low capacity models. Further, these methods involve sequential processing of the generated responses, and thus increase inference time significantly.

In this work, we propose a natural method of enabling LLMs to *look backwards* in order to obtain meaningful unsupervised feedback during inference. Towards this, we introduce a class of models that we call *Time Reversed Language Models* (`TRLM`s), which operate in the reversed direction of a regular LLM, or the *time-reversed* direction. Rather than predicting (or scoring) in the standard `query` $\rightarrow$ `response` direction, time reversed language models predict (or score) in the `response` $\rightarrow$ `query` direction. We first introduce `TRLM-Fo` - a `TRLM` variant based on forward models, which are *prompted* to operate in the time-reversed direction using a prompt such as `"Generate a question that would result in the following answer: <response>"`. Further, we extend the reversal to *token*-level granularity by pre-training LLMs from scratch in a reversed token direction, rather than the standard forward token direction. We call this as `TRLM-Ba` where `Ba` stands for Backward. Note that the inputs and outputs of such a model are in the reversed language order. Pre-training `TRLM-Ba` on reversed text exposes the model to a completely different world model where the conventional order of information is flipped. Introductions *follow* conclusions, questions *follow* answers, logical precedents *follow* their antecedents. Hence, such a model may not only develop representations that are distinct from those of a regular LLM – despite being trained on the same pre-training corpus – but may also be better suited to score/ generate in the reverse direction, i.e. conditional on the response.

We show in several use-cases that scoring and generation in this reverse direction can produce non-trivial feedback on the responses generated by forward LLMs. We consider three classes of tasks to showcase the scoring and generating capability of `TRLM`, viz. a) Reranking answers in open ended question answering b) Citation and retrieval tasks and c) Amplifying existing safety filters through query generation in the reverse.

Our Contributions:
a) We propose time reverse language models - `TRLM-Fo`, `TRLM-Ba` and `TRLM-FoBa`, all of which score and generate queries given responses, enabling their use in obtaining unsupervised feedback on LLM generations. `TRLM-Fo` is a forward model prompted to predict in reverse, while `TRLM-Ba` is pre-trained in the reverse token order, enabling reverse prediction naturally. `TRLM-FoBa` is pre-trained in both reverse and forward token orders and can be used to predict in forward or reversed language.
b) We demonstrate significant improvements when best-of-N reranking is applied to multiple LLM generations by using `TRLM` scores. Specifically, we show up to a 5% improvement over self-reranking using `TRLM-Ba`, in LC win-rates (0.98 Pearson correlation with human preferences) against a `GPT4-1106-Preview` reference model. We show multiple ablations on this study.
c) We demonstrate that the reverse direction of scoring (`response` $\rightarrow$ `query`) is highly significant, as it improves citation attribution accuracy by 44.15% when compared to the forward baseline on the CNN-Daily Mail dataset. Further, we improve the NDCG@10 metric by 44.13% on the NF-Corpus medical information retrieval benchmark, and obtain similar improvements on MS-Marco as well.
d) We show that the reverse generation capability of the `TRLM` models - specifically `TRLM-Ba`, can be used to improve False Negative rate (FNR) of input safety filters with negligable impact on FPR. We show significant improvements on several attacks submitted to the Jailbreakbench benchmark, and on a Human Annotated dataset from JailbreakBench.

We complement these results with theoretical arguments using a bipartite graph model between queries and responses, to show that RLHF done with `TRLM-Ba` scores induces a non trivial distribution shift in answers, mitigating primitive forms of "hallucination" under the defined conditions.

2 Related Work

Reverse Direction in Language Modeling: Classical work [Serdyuk et al., 2017] showed how sequence to sequence models can regularize the current word token embedding based on the ability of the future tokens to be able to predict the current token. Such bi-directional (forward and reverse) consistency checks have been used to improve forward models. Golovneva et al. [2024] train an LLM in the forward direction first, followed by the reverse token direction, and show that this alleviates the

reversal curse identified by Berglund et al. [2023]. This work is closely related to ours in that we also consider a variant of combining reverse and forward token order during training. Our key models differ from this, and are trained in either forward (TRLM-Fo)/ reverse (TRLM-Ba) token order, using which we demonstrate improvements in a wide range of applications such as long form question answering, citations, retrieval and augmenting input filters for defending against toxic questions. Yang et al. [2023] use question generation from a given answer combined with access to external databases to determine hallucination. Another recent work [Guo et al., 2024] also explores a different pre-training order. While their focus is to correct causal ordering bias, our work instead is focused on the value that scoring and generation of these models bring to downstream tasks.

Reversed scoring: Several prior works [Li et al., 2016, Zhang et al., 2018, 2020] have proposed to improve the diversity of generated responses by optimizing the *mutual information* between the responses and the respective queries. These works motivate the need for better decoding strategies based on scores in both, response $\rightarrow$ query and query $\rightarrow$ response directions. We theoretically show that reverse scoring alone, when used with forward generations, will achieve this naturally using a formal RLHF based argument (Lemma 2), and present strong empirical results across a wide range of tasks to support the same.

Controlling Decoding through feedback: A broad line of works align a pre-trained model to a reward model trained on human feedback by using Reinforcement learning (RL) techniques like Proximal Policy Optimization (PPO) [Stiennon et al., 2020, Ouyang et al., 2022, Korbak et al., 2022], (Identity policy optimization) IPO (and ΨPO) [Azar et al., 2024], Direct Preference Optimization [Rafailov et al., 2024] and offline RL [Snell et al., 2022]. Zhao et al. [2022] and Zhao et al. [2023b] calibrate likelihood of generated responses on a dataset with desired responses or human preference feedback.[Krause et al., 2020, Yang and Klein, 2021, Qin et al., 2022] control the generation of an LLM at test time by specifying constraint functions or discriminators that operate in the token or logit space, encouraging certain attributes in the output. Using preference feedback, Mudgal et al. [2023b] train a prefix scorer model that acts as a value function over partial completions consistent with the preference rewards. Yang et al. [2024b] investigate the relation between best-of-N-reranking and KL regularized RL objective. An observation made by Yang et al. [2024b] is that best-of-N-reranking dominates/ competes very well with most RL based alignment methods. Under certain assumptions, authors show formally that best-of-N-reranking approximates the optimal solution to the regularized RL objective. We take inspiration from this and use best-of-N-reranking to evaluate generations through unsupervised feedback by the reverse LLMs. Our work differs from all these in that they rely on external feedback to control generation, while our method does not.

Self Play and Self Tuning: Chen et al. [2023] explore how an LLM can be prompted to self-debug based on an explanation of the code produced by the LLM during code generation and the execution output on test cases. Welleck et al. [2022] use a corrector model that is trained to prefer a new corrected answer if the corrected answer has higher value that a default generation. They require access to a value function for this determination. All these approaches use an external feedback to align the model in their pipeline.

Fu et al. [2023] explore LLM agents initialized as buyers and sellers to play a negotiating game of setting the price of a transaction. A critic LLM provides feedback to both the buyer and seller agents to improve. Madaan et al. [2024] propose a self refining loop where the same model is prompted to provide feedback and further use the feedback to refine and regenerate. Both these works use very powerful and large models from the Claude, GPT-4, GPT-3.5 family to use self generated language feedback. Madaan et al. [2024] remark that the self refining approach does not work well with weaker models. In contrast, we focus on improving generation quality of much smaller models using unsupervised scalar feedback. Other prior works relating to self play are reviewed in the survey article by Amini et al. [2022].

3 TRLM - Time Reversed Language Models

We introduce our primary contribution - TRLM (**T**ime **R**eversed **L**anguage **M**odels), a class of language models that operate in the response $\rightarrow$ query direction during scoring and generation. This is achieved by either (a) [TRLM-Ba] reversing the token order and effectively utilizing previous token prediction instead of next token prediction during pre-training, scoring, and generation, or (b) [TRLM-Fo] maintaining the standard token order during pre-training but reversing the direction of generation through appropriate prompts during inference (scoring and generation).

Table 1: Description of different TRLM model variants.

Model	Description
TRLM-Ba	Pre-trained in the reverse token order for previous token prediction (Alg. 1 in the supplement). Instruction-tuned variant is FLaN fine-tuned [Longpre et al., 2023] in reverse token order. Scores the reversed question given a reversed answer combined with suitable prompts. Generates questions in the reverse direction when conditioned on answers in the reverse direction. **Scoring:** $\mathbb{P}_{\texttt{TRLM-Ba}}\big(\text{Reverse}(\texttt{Scoring Prompt}+\text{Query}) \mid \text{Reverse}(\texttt{Conditioning Prompt} + \text{Answer})\big)$ (Alg. 2 in the supplement). **Generation:** $\mathbb{P}_{\texttt{TRLM-Ba}}\big(\ \cdot\ \mid \text{Reverse}(\texttt{Conditioning Prompt} + \text{Answer})\big)$
TRLM-Fo	Pre-trained in the usual forward token order. Scores Question given Answer using the prompt. Generates from the conditional distribution of an answer. **Scoring:** $\mathbb{P}_{\texttt{TRLM-Fo}}\big(\text{Query} \mid \text{Answer} + \text{Conditioning Prompt}\big)$ (Alg. 3 in the supplement) **Generation:** $\mathbb{P}_{\texttt{TRLM-Fo}}\big(\ \cdot\ \mid \text{Answer} + \texttt{Conditioning Prompt}\big)$
TRLM-FoBa (Reverse)	Pre-trained both in forward and reverse token order (Alg. 4 in the supplement). Understands text in both directions. Reverse version scores and generates identically to TRLM-Ba. **Scoring:** Scores identically to TRLM-Ba. **Generation:** Generates identically to TRLM-Ba.
TRLM-FoBa (Forward)	Pre-trained both in forward and reverse token order. Forward version scores and generates identically to TRLM-Fo. **Scoring:** Scores identically to TRLM-Fo. **Generation:** Generates identically to TRLM-Fo.
Self Scoring	The model that is used for generating a given response is also used for scoring responses given queries in the conventional forward scoring direction. **Scoring:** We use the model's own perplexity scores as feedback to select the responses.
Forward Baseline	A conventional forward model trained for next-token prediction on the same training corpus and model class as TRLM . **Scoring:** While self-scoring used the perplexity obtained from the generator model, in this setting, we use perplexity of a different forward model.

We show that TRLM provides non-trivial unsupervised feedback that could be used by pre-trained, fine-tuned, and instruction tuned models, for various downstream tasks like reranking to improve open-ended long-form question answering, generating citations, and retrieval. We demonstrate that the ability of TRLM to score in the reverse direction – scoring query based on the response – is essential to achieve the requisite gains. Further, TRLMs that are pre-trained in the reverse direction (TRLM-Ba) provide an additional boost in most cases. We further leverage the generative ability of TRLM in reverse (generating query from a response) to amplify the effectiveness of input safety filters as well.

We propose four variants of the TRLM class – TRLM-Ba , TRLM-Fo , TRLM-FoBa (Reverse) and TRLM-FoBa (Forward) – based on how they are pre-trained and fine-tuned. TRLM models can be considered to have three functions: TRLM.Pretrain, TRLM.Score, and TRLM.Generate, which we describe for each of the four variants in Table 1. We further outline these functions for different TRLM models in Algorithms 1, 2, 3, & 4. For this work, we consider two baselines, which are trained in forward token order, and score in the conventional order of response given the query. The first of these uses self-scoring based on the model's own perplexity. The second (Forward Baseline) is a forward model that we train, whose training corpus and model class are identical to TRLM.

TRLM **Model Training:** The pre-training setup for all TRLM models is identical to that of PALM2-Otter models described by Anil et al. [2023b], except for the token orders specified by our TRLM.pretrain methods for TRLM-Fo , TRLM-Ba and TRLM-FoBa respectively. We fine-tune them on the FLaN dataset [Longpre et al., 2023] using the TRLM-xx.pretrain function. Where xx can refer to Fo, Ba or FoBa based on the model being fine-tuned. Let Instruction, Question, Answer denote instruction, question and answer respectively. Before calling the pretrain function during fine tuning , we merge Instruction + Question to be the new question.

4 Scoring in Reverse

In this section, we provide formal results on TRLM and the benefit of using pre-training in the reverse direction. Let us denote by $\mathbb{P}_{\texttt{Fw}}(A|Q)$ the conditional distribution of a forward LLM. Similarly, denote $P_{\texttt{TRLM}}(Q|A)$ to be the conditional distribution of the Time Reversed Language Model. For simplicity, we merge the instruction and question together.

4.1 Formal Results on Reverse LLM based Alignment

In this subsection, we focus on the distribution shift encountered while using a reverse model based scorer on forward generations. Specifically, we conclude that while reranking using Forward Baseline is equivalent to temperature scaling [Yang et al., 2024b], reranking using TRLM induces a distribution shift that is not equivalent to temperature scaling.

Consider the *alignment* problem of learning a new forward LLM - $\tilde{\mathbb{P}}_{\texttt{Fw}}(\texttt{Answer}|\texttt{Question})$. A very popular framework is the KL constrained optimization objective with respect to a reward oracle $\mathcal{R}(\texttt{Question}, \texttt{Answer})$, for some threshold Δ:

$$\max_{\tilde{\mathbb{P}}_{\texttt{Fw}}} \mathop{\mathbb{E}}_{\substack{\texttt{Question}\sim\mathcal{Q} \\ \texttt{Answer}\sim\tilde{\mathbb{P}}_{\texttt{Fw}}(\texttt{Answer}|\texttt{Question})}} [\mathcal{R}(\texttt{Question}, \texttt{Answer})] \text{ s.t. } D_{\mathrm{KL}}(\tilde{\mathbb{P}}_{\texttt{Fw}} \| \mathbb{P}_{\texttt{Fw}}) \leq \Delta \tag{1}$$

Log-perplexity of the forward model used as reward: In general, for long form question answering where an explicit reward model is not available, a typical method is to use log-perplexity of the forward model i.e. $\log \mathbb{P}_{\texttt{Fw}}$ as a reward. Then, we have the following corollary of Lemma 1 in Yang et al. [2024b],

Lemma 1 (Corollary of Lemma 1 in Yang et al. [2024b])**.** The new LLM policy $\tilde{\mathbb{P}}_{\texttt{Fw}}$ that optimizes (1) is given by: $\tilde{\mathbb{P}}_{\texttt{Fw}}(\texttt{Answer}|\texttt{Question}) \propto \mathbb{P}_{\texttt{Fw}}^{1+\alpha}(\texttt{Answer}|\texttt{Question})$ where α is chosen appropriately depending on the threshold Δ when reward $R(\cdot)$ is set to $\log$ perplexity of the forward model $\mathbb{P}_{\texttt{Fw}}$.

A policy obtained post the constrained KL-alignment procedure is akin to temperature re-scaled forward model, since $p^{1+\alpha}$ is equivalent to *temperature rescaling* $\exp^{(1+\alpha)\log p}$.

Log-perplexity of the TRLM-Ba.score used as reward: Suppose $R(\cdot)$ is set to output of TRLM-Ba.score computed on the the question given the answer, then we have:

Lemma 2 (Corollary of Lemma 1 in Yang et al. [2024b])**.** The new LLM policy $\tilde{\mathbb{P}}_{\texttt{Fw}}$ that optimizes (1) is given by: $\tilde{\mathbb{P}}_{\texttt{Fw}}(\texttt{Answer}|\texttt{Question}) \propto \mathbb{P}_{\texttt{Fw}}(\texttt{Answer}|\texttt{Question})\mathbb{P}_{\texttt{TRLM-Ba}}^{\alpha}(\texttt{Question}|\texttt{Answer})$ where α is chosen appropriately depending on Δ when reward $R(\cdot)$ is set to $\log$ perplexity of the reverse model $\mathbb{P}_{\texttt{TRLM}}$.

Optimal distribution after alignment using TRLM scores results in a non-trivial distribution that is not simply temperature re-scaling. While we have not used TRLM for alignment using KL constraints in our experiments, the distribution shift that is induced by reverse token training is indeed non-trivial even with Best-of-N-re-ranking, which we adopt in our experiments.

5 Experimental Results

In this section, we explore the effectiveness of time reversed language models on different downstream tasks, by utilizing unsupervised feedback to improve upon existing forward model generations. Broadly, these applications fall into two categories - first, where we utilize the scoring capacity of TRLM (three use cases), and second where we utilize the generative capacity of TRLM for generating queries given a response.

5.1 Best-of-N reranking

The best-of-N reranking task involves outputting the best response out of N model responses to a user query. Specifically, given N LLM outputs to a user query, a reranking algorithm finds the best response based on scalar scores assigned to each response. Prior works [Rafailov et al., 2023, Mudgal et al., 2023a] aim to improve LLM performance on this task by using feedback-based RLHF algorithms and training on KL-regularized alignment objectives. Yang et al. [2024a] show that

best-of-N reranking is the most effective way to approximate these RL objectives, and further, it is empirically observed to outperform them.

In this work, we consider several best-of-N reranking based algorithms based on TRLM.Score, for evaluating a base model response. The methods considered rely on nothing more than the pre-training (or instruction-tuning) corpus to achieve alignment of response to the user query. We further note that such scores from TRLM may be used within RL objectives as well, but we leave the exploration of such rewards to future work.

5.1.1 Alpaca Leaderboard Evaluation

Benchmark and Evaluation: The AlpacaEval leaderboard [Dubois et al., 2024] is a widely used benchmark to evaluate the capability of language models. In this benchmark, there are 805 questions from the *AlpacaFarm* evaluation set – consisting of questions ranging from general writing, chat ability, and reasoning to general knowledge. The goal is to output a response that is better than a base model's response, as judged by an annotator model. Both base model and annotator model are set as GPT4-1106-Preview on the AlpacaEval leaderboard as on May 10, 2024, and hence we use the same for our evaluations. The evaluation benchmark computes various metrics including winrates, discrete winrates and length-controlled winrates [Dubois et al., 2024]. The length-controlled winrates are calculated using a debiasing algorithm that removes the length bias that is otherwise preferred by GPT4-1106-Preview .

Formally, we define the task for TRLM as follows — Given a query Q from the dataset and N model responses $\mathcal{A} = \{A_1 \dots A_N\}$ from a generator model, we wish to use TRLM.score to output the highest scoring response $a_i \in \mathcal{A}$, which is further evaluated against an answer from GPT4-1106-Preview.

In our experiment, we consider outputs from a generator model that is Gemini-Pro-1.0 [Anil et al., 2023a]. We generate 16 responses using a temperature $\tau = 0.8$ to ensure diversity of answers. We then rerank the responses using different variants of TRLM from the PALM2-Otter family of models (TRLM training details in the supplement). We further consider two baselines, Self scoring and Forward Baselines, as described in Table 1. Scoring prompts and Conditioning prompts used with various TRLM variants for this task are described in the Table 7 of Appendix C.1.

Discussion of Results: In Table 8, we see that TRLM-Ba scores the highest length controlled win rate which is 5% over the self scoring baseline of Gemini-Pro-1.0 with 16 generations against the GPT4-1106-Preview judge. Further, it registers an 8% increase over the reported number for single generations in the benchmark leaderboard. We note that scoring Response->Query seems to bring out some improvements as TRLM-Fo improves over Forward Baseline. Further, TRLM-Ba outperforms TRLM-Fo indicating the impact of reverse token pre-training. This demonstrates that time reversed scoring provides an intrinsic unsupervised feedback that could help improve the performance of even larger capacity models. We note that pre-training in both forward and reverse directions (TRLM-FoBa models) and scoring in the reverse direction is better than TRLM-Fo variant.

We present further results where the generations of a Mixtral model [Jiang et al., 2024b] are reranked and compared against GPT4-1106-Preview , and the generations of a smaller Mixtral model are reranked and compared against a larger Mixtral model. These results are presented in the Appendix C.2. We note a 4% improvement over Forward Baseline with the proposed TRLM-Ba.Score method of reranking.

Key Takeaway: Through empirical justifications, we show that TRLM variant models can be used as effective re-rankers of generations from multiple classes of models (Gemini-Pro-1.0 , Mixtral8x22B, Mixtral8x7B), and improve the instruction following capability of the model as a whole. This is consistent with the 1 considering the fact that we outperform generation model's self-log perplexity score. While other methods of re-ranking exists, to the best of our knowledge none of them provide unsupervised feedback for effective reranking with just a pre-trained model.

5.2 Citation Attribution

In this section, we describe applications of reverse scoring to the task of producing citations to original passages that can *corroborate* the sentences in an already produced summary. Summaries are created from long form articles, and one often wants to know which part of the article a given summary sentence is derived from (Benjamin Cohen-Wang [2024]).

Table 2: The best re-ranked response is compared with a single response of GPT4-1106-Preview . The setting is identical to the AlpacaEval Alp Leader board. TRLM-Fo , that scores in the backward direction, fares better than the conventional forward baseline. Scoring using TRLM-Ba (pretrained in reverse) gets even a higher (LC) win rate.

Model Performance on the Alpaca Leaderboard

Model	Inference Style	Win Rate			Standard Error	Wins	Losses	Ties
		LC	Reg	Discrete				
TRLM-Ba	Response -> Query	32.44	24.35	24.04	1.27	192	610	3
TRLM-FoBa (backward)	Response -> Query	31.18	22.72	21.99	1.24	176	627	2
TRLM-FoBa (forward)	Response -> Query	30.55	22.85	22.48	1.25	180	623	2
TRLM-Fo	Response -> Query	29.19	22.68	21.30	1.24	170	632	3
One Generation	-	24.38	18.18	17.08	1.16	135	665	5
Self	Query -> Response	27.05	17.66	17.14	1.15	136	665	4
Forward Baseline	Query -> Response	24.27	17.13	15.78	1.12	126	677	2

Dataset and Evaluation: For this task, we take the CNN Daily Mail Dataset [CNN] which consists of pairs of news articles and their respective highlights. Our goal is to identify which sentence (or groups of sentences) within a given news article provides the most direct corroboration for a specific article highlight given as a query. We evaluate the attributed citations using various relevancy metrics. We use cosine similarity on the embeddings of the Gecko model [Lee et al., 2024], cosine similarity on TF-IDF features, BLEU score and ROUGE score to compute metrics. We score and choose the best pairing using all the models from the TRLM PALM2-Otter family trained in the forward, reverse and forward-reverse directions as outlined in Section 5.1.1.

Algorithms: Different search algorithms, Linear Search, Binary Search and Exclusion Search are coupled with using TRLM.score to find the attribution. We outline these in Algorithms 7, 8 and 9 along with details in the supplement. The number of inference calls is $O(\log N)$ where N is the number of article sentences for Binary Search, and this method produces multiple sentences as a citation. The other methods require $O(N)$ calls to produce the citation for a sentence.

Our results shown in Table 3, demonstrate the efficacy of TRLM for the attribution task. Specifically, we show 44% gains over the baseline in the linear search method, 39% gains in the binary search method and 34% gains in the exclusion search method as measured through gecko cosine similarity.

Key Takeaway: Through our results on CNN-Daily Summarization dataset we present multiple methods of citation attribution and demonstrate significant gains with TRLM model variants. We note that a direction of *low* information to *high* information (summary –> article) is harder to reason upon and select among a given set of texts. Further, we highlight the importance of binary **selection based** approach over log-perplexity based **exclusion based** search. We show 9% improvement using TRLM-Ba on Gecko embedding-based metric using only $O(\log N)$ inference calls to the main model.

Table 3: Tabulates the citation Attribution results through Re-ranking on the CNN-Daily Mail dataset. A denotes article whereas S denotes the corresponding summary. The ease of scoring a summary given the article instead of *reverse* is clearly highlighted in all of the search methods.

Model	Inference Direction	LinearSearch			Binary Search			Exclusion Search		
		Gecko	TF-IDF	ROUGE	Gecko	TF-IDF	ROUGE	Gecko	TF-IDF	ROUGE
TRLM-Ba	A->S	53.16	**55.45**	49.12	**45.09**	**50.93**	**42.11**	36.33	46.34	36.13
TRLM-FoBa (Rev.)	A->S	**53.48**	53.22	**49.67**	40.74	45.04	39.81	32.40	40.84	33.88
TRLM-FoBa (Forw.)	A->S	50.65	52.21	45.24	43.81	49.84	40.60	**38.67**	**48.16**	**38.11**
TRLM-Fo	A->S	45.00	49.40	37.66	43.14	49.65	39.22	37.90	47.83	37.98
Forward Baseline	S->A	9.33	9.54	11.06	5.88	6.66	6.69	4.66	7.53	7.00
Backward Baseline	S->A	7.62	8.23	9.18	5.47	6.23	6.32	4.11	5.02	5.11

5.3 Document Retrieval

In this section, we study the performance of TRLM in retrieving relevant passages from a corpus to answer a specific question. Our goal is to show the efficacy of TRLM based reverse scoring over doing it in the forward direction. The task is as follows: Given a question, the goal is to retrieve relevant documents from the given corpus. We retrieve k documents from the corpus and compute various information-retrieval metrics to calculate performance w.r.t. the golden set of documents.

Table 4: Summary of MS-Marco and NF-Corpus Datasets

Dataset	Description
MS-Marco	Contains 101.09k examples in its public dev split. Each example consists of a simple question along with 10 relevant passages. [Bajaj et al., 2016]
NF-Corpus	Medical information retrieval dataset with 323 queries in its test split and 3.6k total documents in the corpus. Queries are in simple English, and documents are extracted from PubMed with a fair amount of medical terminology. [Boteva et al., 2016a, Pub]

We experiment with two retrieval-based datasets from MTEB benchmark [Muennighoff et al., 2023] as shown in Table 4. Metrics are precision, recall, normalized discounted cumulative gain (NDCG) and mean reciprocal rank (MRR) (details in Appendix E.1). We show our results in Table 5. `TRLM` reverse scoring algorithms along with respective prompts used are presented in Algorithms 11, 10 of the Supplement. As Table 5 suggests, results favor `TRLM` based reverse scoring methods. For example, we see a 22.48% improvement in recall at $K = 4$ for MS-MARCO dataset. `TRLM-Ba` model dominates across metrics. For NF-Corpus, we see that the conventional forward scoring algorithm (`query -> document`) has a very poor performance. We attribute this to the fact that, in this inference direction, we are scoring a highly complex medical document using a simple natural language query. We see a gain of 44.2 points in NDCG at $K = 10$ with `TRLM-Fo` compared to `Forward Baseline` . The results in both these datasets suggest that `TRLM` can show greater gains when the complexity of documents in the corpus differs significantly from the complexity of queries.

Table 5: Tabulates the results of various reranking algorithms with two inference directions. Q denotes Queries, while D denotes Documents. `TRLM` outperforms `Forward Baseline` and `Backward Baseline` significantly, which highlights the importance of inference direction in this task.

Method	Inference Direction	MS-MARCO					NF-CORPUS				
		Precision		Recall		NDCG	Precision		Recall		NDCG
		K=1	K=4	K=1	K=4	@10	K=10	K=20	K=10	K=20	@10
TRLM-Ba	D -> Q	**28.4**	**18.54**	**27.22**	**70.29**	**61.49**	15.7	11.38	10.68	**13.08**	43.23
TRLM-FoBa (Reverse)	D -> Q	24.9	17.38	23.85	65.85	58.84	14.98	10.91	10.01	12.76	41.65
TRLM-FoBa (Forward)	D -> Q	21.16	15.58	20.25	59.08	55.46	**17.86**	**12.6**	**11.11**	13.5	48
TRLM-Fo	D -> Q	20.37	14.9	19.45	56.39	54.46	17.31	12.38	9.74	11.76	**48.08**
Forward Baseline	Q -> D	21.05	13.82	18.42	47.81	53	0.87	0.87	0.17	0.31	3.89
Backward Baseline	Q -> D	16.8	14.04	15.99	53.13	52.07	1.11	0.79	0.21	0.29	3.95

Key takeaways: We experiment with two information retrieval-based benchmarks MS-MARCO and NF-CORPUS and compute multiple metrics to compare `TRLM` variant models with standard `Forward Baseline` and unconventional `Backward Baseline` . We show a gain of 8.49 points in NDCG@10 on MS-MARCO and 44.19 points in NDCG@10 on NF-CORPUS. Aligning with the results in citation, the results from this task also accurately demonstrate the importance of going from a *high* information direction to a *low* information direction. The massive difference between the directions is evident in the NF-CORPUS dataset.

5.4 Defending against Jailbreak attacks

We next aim to leverage the generative ability of `TRLM` to augment toxicity filters that are used to improve the safety of LLMs. Prior works show that LLMs (and their input filters) can be jailbroken using crafted adversarial attacks [Zou et al., 2023], while output filters tend to have a high false negative rate due to the sensitivity to the presence of toxic words, despite being in a neutral context (See Table-10). We propose to combine the benefits of input and output filters by projecting the output response of LLMs to the input query space using the reverse generative capability of `TRLM` , and further detecting the toxicity of the generated queries to block/ pass the response to the original query based on a pre-specified criteria. We thus effectively amplify input safety filters, i.e. reduce False Negative Rate (FNR) with marginal/ no impact on False Positive Rate (FPR).

Key Idea: Consider `TRLM.Generate(Response)` that generates queries that could have produced a given response. The insight is that, the reverse generative ability of `TRLM` allows the projection of a candidate (jailbreak) query that could bypass the input filter back to the (naive) query space observed during training. These projected questions can thus be rightly classified using the same input filter.

Table 6: Performance of the proposed defense strategies across different thresholds, evaluated on the human annotated and jailbreakbench toxic responses. TRLM-Ba achieves significant gains over all other approaches. Notations: PT [Pretrained], IT[Instruction-finetuned], FNR[False Negative Rate], FPR[False Positive Rate], new-HA [new HA Dataset], JBB[JBB Dataset], (H) [Hard], (E) [Easy]

Method	Thresh = 2				Thresh = 4				Thresh = 6			
	FNR-HA	FNR-JBB	FPR (H)	FPR (E)	FNR-HA	FNR-JBB	FPR (H)	FPR (E)	FNR-HA	FNR-JBB	FPR (H)	FPR (E)
TRLM-Fo (PT)	0.00	36.11	17.00	2.00	36.36	55.56	12.00	0.00	45.45	70.83	6.00	0.00
TRLM-Ba (PT)	18.18	52.78	0.00	8.00	27.27	65.28	0.00	2.00	27.27	69.44	0.00	2.00
TRLM-Fo (IT)	54.55	55.56	3.00	0.00	63.64	72.22	1.00	0.00	63.64	81.94	1.00	0.00
TRLM-Ba (IT)	18.18	59.72	0.00	8.00	18.18	70.83	0.00	4.00	27.27	79.17	0.00	2.00

Defense Strategy: We propose a defense strategy where i) a query is passed through the input filter, ii) if the input filter rejects the query, we return reject as well, iii) if the input filter allows the query, we take the Response produced by the model and generate multiple queries using TRLM.Generate(Response). If the number of generated queries rejected exceeds a threshold, we reject the query as "unsafe". Otherwise, we declare it as safe, and output the response corresponding to the input query. An elaborate description is provided in Algorithm 12 of the Supplement.

Datasets: We consider a human annotated (HA) dataset provided as part of the JailbreakBench benchmark [HAd] for evaluating the performance of toxicity classifiers. This contains 100 questions annotated by humans, of which 43 are annotated as toxic based on a majority vote across 3 annotators. We introduce a GPT-4 based filter, that considers the prompt-response pair to judge their toxicity (Details in Appendix-F.2), and has 0 FNR on this HA dataset, which is ideal for defense evaluation. We further consider a gpt-3.5-turbo-1106 based input toxicity filter for the empirical evaluation of the proposed defense, which has an FNR of 25.58% on this dataset. These unblocked questions form our new-HA dataset for the experiments. In addition to this, we use the following datasets for evaluation: JBB dataset that contains jailbreak questions (that are toxic as per the GPT-4 judge, but are safe as per the GPT-3.5 filter we augment) corresponding to different attacks on JailbreakBench, E dataset that contain safe and easy questions and H dataset that contains safe questions that are hard to classify as safe. We discuss more details on these datasets in Appendix-F.1.

In the two toxic datasets (HA and JBB), the gpt-3.5-turbo-1106 based input filter does not block any of the questions, and our defense strategy aims at lowering the False Negative rate on the toxic questions (JBB dataset and new-HA dataset), while ensuring a low false positive rate on the safe questions as well (E and H datasets). We present the improvements in FNR rates for Algorithm 12 when combined with the gpt-3.5-turbo-1106 input filter and various TRLM variants in Table-6. We further present the impact of varying the threshold in Fig.4 of the Appendix.

Results: We firstly note that the proposed TRLM defense strategy improves the FNR of the gpt-3.5-turbo-1106 input filter across all settings considered. Further, the TRLM-Ba pre-trained model improves FNR by more than 70% on the HA dataset and around 35% on the JBB dataset, and outperforms other variants with negligible impact on FPR.

We note that the proposed defense outperforms existing perplexity thresholding based defenses [Jain et al., 2023, Alon and Kamfonas, 2023] and Smooth-LLM [Robey et al., 2023] on the JailbreakBench attacks [Chao et al., 2023, Deng et al., 2024] owing to the integration with an input filter that already outperforms them on the same. Hence, we do not compare with them. Further, these defenses operate only in the input space, while the proposed defense aims at augmenting the input space with feedback from the response. Hence, the proposed defense is orthogonal to such methods, and can thus be integrated with them as well.

6 Conclusions

In this work, we explore the capabilities of TRLM for scoring and generation of queries, when conditioned on responses. Our study points to the importance of the response $\rightarrow$ query direction in LLMs. When deploying TRLM models for reverse scoring, we show improvements on AlpacaEval leaderboard, Citation attribution and retrieval tasks. We further show that generations from TRLM can augment safety filters effectively.

7 Limitations

We note that the assumptions made for our theoretical results in Section 4 are stylized, and may not hold true in practice, as the space of all answers to questions may not be adequately captured by assumptions in that section. Given this assumption, one may wish to explore other models for hallucination that are more general and provide results about reverse scoring. We leave such a theoretical exploration to future work.

Further, `TRLM` benefits have thus far been explored on tasks related to short form queries that have long answers. One may wish to understand and demonstrate the effects of reverse scoring on other tasks. For instance, one might pose the question – does `TRLM` provide possible benefits for a broader set of tasks that language models are used for. We leave the exploration of such settings in which the reverse scoring direction of `response` $\rightarrow$ `query` is better than the forward scoring direction, along with obtaining an understanding on the reason behind such an advantage, as part of future work.

8 Acknowledgements

We are grateful to Kathy Meier-Hellstern and Krishnamurthy Dvijotham for the helpful discussions regarding defending against Jailbreak attacks. We sincerely thank Roman Novak and Abhishek Kumar for their inputs on early versions of our work.

References

Alpacaeval leaderboard. https://tatsu-lab.github.io/alpaca_eval/.

Cnn dailymail dataset. https://www.tensorflow.org/datasets/catalog/cnn_dailymail.

Human annotated dataset, jailbreakbench. https://github.com/JailbreakBench/jailbreakbench/blob/main/src/jailbreakbench/data/classifier_comparison.csv.

Pubmed. https://pubmed.ncbi.nlm.nih.gov/.

J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

G. Alon and M. Kamfonas. Detecting language model attacks with perplexity. *arXiv preprint arXiv:2308.14132*, 2023.

M.-R. Amini, V. Feofanov, L. Pauletto, E. Devijver, and Y. Maximov. Self-training: A survey. *arXiv preprint arXiv:2202.12040*, 2022.

R. Anil, S. Borgeaud, Y. Wu, J. Alayrac, J. Yu, R. Soricut, J. Schalkwyk, A. M. Dai, A. Hauth, K. Millican, D. Silver, S. Petrov, M. Johnson, I. Antonoglou, J. Schrittwieser, A. Glaese, J. Chen, E. Pitler, T. P. Lillicrap, A. Lazaridou, O. Firat, J. Molloy, M. Isard, P. R. Barham, T. Hennigan, B. Lee, F. Viola, M. Reynolds, Y. Xu, R. Doherty, E. Collins, C. Meyer, E. Rutherford, E. Moreira, K. Ayoub, M. Goel, G. Tucker, E. Piqueras, M. Krikun, I. Barr, N. Savinov, I. Danihelka, B. Roelofs, A. White, A. Andreassen, T. von Glehn, L. Yagati, M. Kazemi, L. Gonzalez, M. Khalman, J. Sygnowski, and et al. Gemini: A family of highly capable multimodal models. *CoRR*, abs/2312.11805, 2023a. doi: 10.48550/ARXIV.2312.11805. URL https://doi.org/10.48550/arXiv.2312.11805.

R. Anil, A. M. Dai, O. Firat, M. Johnson, D. Lepikhin, A. Passos, S. Shakeri, E. Taropa, P. Bailey, Z. Chen, E. Chu, J. H. Clark, L. E. Shafey, Y. Huang, K. Meier-Hellstern, G. Mishra, E. Moreira, M. Omernick, K. Robinson, S. Ruder, Y. Tay, K. Xiao, Y. Xu, Y. Zhang, G. H. Ábrego, J. Ahn, J. Austin, P. Barham, J. A. Botha, J. Bradbury, S. Brahma, K. Brooks, M. Catasta, Y. Cheng, C. Cherry, C. A. Choquette-Choo, A. Chowdhery, C. Crepy, S. Dave, M. Dehghani, S. Dev, J. Devlin, M. Díaz, N. Du, E. Dyer, V. Feinberg, F. Feng, V. Fienber, M. Freitag, X. Garcia, S. Gehrmann, L. Gonzalez, and et al. Palm 2 technical report. *CoRR*, abs/2305.10403, 2023b. doi: 10.48550/ARXIV.2305.10403. URL https://doi.org/10.48550/arXiv.2305.10403.

M. G. Azar, Z. D. Guo, B. Piot, R. Munos, M. Rowland, M. Valko, and D. Calandriello. A general theoretical paradigm to understand learning from human preferences. In *International Conference on Artificial Intelligence and Statistics*, pages 4447–4455. PMLR, 2024.

P. Bajaj, D. Campos, N. Craswell, L. Deng, J. Gao, X. Liu, R. Majumder, A. McNamara, B. Mitra, T. Nguyen, et al. Ms marco: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268*, 2016.

K. G. A. M. Benjamin Cohen-Wang, Harshay Shah. Contextcite: Attributing model generation to context. `https://github.com/MadryLab/context-cite/tree/main`, 2024.

L. Berglund, M. Tong, M. Kaufmann, M. Balesni, A. C. Stickland, T. Korbak, and O. Evans. The reversal curse: Llms trained on" a is b" fail to learn" b is a". *arXiv preprint arXiv:2309.12288*, 2023.

V. Boteva, D. G. Ghalandari, A. Sokolov, and S. Riezler. A full-text learning to rank dataset for medical information retrieval. In N. Ferro, F. Crestani, M. Moens, J. Mothe, F. Silvestri, G. M. D. Nunzio, C. Hauff, and G. Silvello, editors, *Advances in Information Retrieval - 38th European Conference on IR Research, ECIR 2016, Padua, Italy, March 20-23, 2016. Proceedings*, volume 9626 of *Lecture Notes in Computer Science*, pages 716–722. Springer, 2016a. doi: 10.1007/978-3-319-30671-1_58. URL `https://doi.org/10.1007/978-3-319-30671-1_58`.

V. Boteva, D. Gholipour, A. Sokolov, and S. Riezler. A full-text learning to rank dataset for medical information retrieval. In *Advances in Information Retrieval: 38th European Conference on IR Research, ECIR 2016, Padua, Italy, March 20–23, 2016. Proceedings 38*, pages 716–722. Springer, 2016b.

T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

P. Chao, A. Robey, E. Dobriban, H. Hassani, G. J. Pappas, and E. Wong. Jailbreaking black box large language models in twenty queries. *arXiv preprint arXiv:2310.08419*, 2023.

X. Chen, M. Lin, N. Schärli, and D. Zhou. Teaching large language models to self-debug. *arXiv preprint arXiv:2304.05128*, 2023.

G. Cloud. Google cloud tpu v5e inference. URL `https://cloud.google.com/tpu/docs/v5e-inference`. Accessed on Feb 1, 2024.

G. Deng, Y. Liu, Y. Li, K. Wang, Y. Zhang, Z. Li, H. Wang, T. Zhang, and Y. Liu. Masterkey: Automated jailbreaking of large language model chatbots. In *Proc. ISOC NDSS*, 2024.

Y. Dubois, B. Galambosi, P. Liang, and T. B. Hashimoto. Length-controlled alpacaeval: A simple way to debias automatic evaluators. *CoRR*, abs/2404.04475, 2024. doi: 10.48550/ARXIV.2404.04475. URL `https://doi.org/10.48550/arXiv.2404.04475`.

Y. Fu, H. Peng, T. Khot, and M. Lapata. Improving language model negotiation with self-play and in-context learning from ai feedback. *arXiv preprint arXiv:2305.10142*, 2023.

O. Golovneva, Z. Allen-Zhu, J. Weston, and S. Sukhbaatar. Reverse training to nurse the reversal curse. *arXiv preprint arXiv:2403.13799*, 2024.

R. A. Google and, A. M. Dai, O. Firat, M. Johnson, D. Lepikhin, A. Passos, S. Shakeri, E. Taropa, P. Bailey, Z. Chen, E. Chu, J. H. Clark, L. E. Shafey, Y. Huang, K. Meier-Hellstern, G. Mishra, E. Moreira, M. Omernick, K. Robinson, S. Ruder, Y. Tay, K. Xiao, Y. Xu, Y. Zhang, G. H. Abrego, J. Ahn, J. Austin, P. Barham, J. Botha, J. Bradbury, S. Brahma, K. Brooks, M. Catasta, Y. Cheng, C. Cherry, C. A. Choquette-Choo, A. Chowdhery, C. Crepy, S. Dave, M. Dehghani, S. Dev, J. Devlin, M. Díaz, N. Du, E. Dyer, V. Feinberg, F. Feng, V. Fienber, M. Freitag, X. Garcia, S. Gehrmann, L. Gonzalez, G. Gur-Ari, S. Hand, H. Hashemi, L. Hou, J. Howland, A. Hu, J. Hui, J. Hurwitz, M. Isard, A. Ittycheriah, M. Jagielski, W. Jia, K. Kenealy, M. Krikun, S. Kudugunta, C. Lan, K. Lee, B. Lee, E. Li, M. Li, W. Li, Y. Li, J. Li, H. Lim, H. Lin, Z. Liu, F. Liu, M. Maggioni, A. Mahendru, J. Maynez, V. Misra, M. Moussalem, Z. Nado, J. Nham, E. Ni,

A. Nystrom, A. Parrish, M. Pellat, M. Polacek, A. Polozov, R. Pope, S. Qiao, E. Reif, B. Richter, P. Riley, A. C. Ros, A. Roy, B. Saeta, R. Samuel, R. Shelby, A. Slone, D. Smilkov, D. R. So, D. Sohn, S. Tokumine, D. Valter, V. Vasudevan, K. Vodrahalli, X. Wang, P. Wang, Z. Wang, T. Wang, J. Wieting, Y. Wu, K. Xu, Y. Xu, L. Xue, P. Yin, J. Yu, Q. Zhang, S. Zheng, C. Zheng, W. Zhou, D. Zhou, S. Petrov, and Y. Wu. Palm 2 technical report, 2023.

Q. Guo, R. Wang, J. Guo, X. Tan, J. Bian, and Y. Yang. Mitigating reversal curse via semantic-aware permutation training. *arXiv preprint arXiv:2403.00758*, 2024.

H. Inan, K. Upasani, J. Chi, R. Rungta, K. Iyer, Y. Mao, M. Tontchev, Q. Hu, B. Fuller, D. Testuggine, et al. Llama guard: Llm-based input-output safeguard for human-ai conversations. *arXiv preprint arXiv:2312.06674*, 2023.

N. Jain, A. Schwarzschild, Y. Wen, G. Somepalli, J. Kirchenbauer, P.-y. Chiang, M. Goldblum, A. Saha, J. Geiping, and T. Goldstein. Baseline defenses for adversarial attacks against aligned language models. *arXiv preprint arXiv:2309.00614*, 2023.

A. Q. Jiang, A. Sablayrolles, A. Roux, A. Mensch, B. Savary, C. Bamford, D. S. Chaplot, D. d. l. Casas, E. B. Hanna, F. Bressand, et al. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024a.

A. Q. Jiang, A. Sablayrolles, A. Roux, A. Mensch, B. Savary, C. Bamford, D. S. Chaplot, D. de Las Casas, E. B. Hanna, F. Bressand, G. Lengyel, G. Bour, G. Lample, L. R. Lavaud, L. Saulnier, M. Lachaux, P. Stock, S. Subramanian, S. Yang, S. Antoniak, T. L. Scao, T. Gervet, T. Lavril, T. Wang, T. Lacroix, and W. E. Sayed. Mixtral of experts. *CoRR*, abs/2401.04088, 2024b. doi: 10.48550/ARXIV.2401.04088. URL `https://doi.org/10.48550/arXiv.2401.04088`.

T. Korbak, E. Perez, and C. L. Buckley. Rl with kl penalties is better viewed as bayesian inference. *arXiv preprint arXiv:2205.11275*, 2022.

B. Krause, A. D. Gotmare, B. McCann, N. S. Keskar, S. Joty, R. Socher, and N. F. Rajani. Gedi: Generative discriminator guided sequence generation. *arXiv preprint arXiv:2009.06367*, 2020.

J. Lee, Z. Dai, X. Ren, B. Chen, D. Cer, J. R. Cole, K. Hui, M. Boratko, R. Kapadia, W. Ding, et al. Gecko: Versatile text embeddings distilled from large language models. *arXiv preprint arXiv:2403.20327*, 2024.

J. Li, M. Galley, C. Brockett, J. Gao, and B. Dolan. A diversity-promoting objective function for neural conversation models. In K. Knight, A. Nenkova, and O. Rambow, editors, *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 110–119. The Association for Computational Linguistics, 2016. doi: 10.18653/V1/N16-1014. URL `https://doi.org/10.18653/v1/n16-1014`.

S. Longpre, L. Hou, T. Vu, A. Webson, H. W. Chung, Y. Tay, D. Zhou, Q. V. Le, B. Zoph, J. Wei, et al. The flan collection: Designing data and methods for effective instruction tuning. In *International Conference on Machine Learning*, pages 22631–22648. PMLR, 2023.

A. Madaan, N. Tandon, P. Gupta, S. Hallinan, L. Gao, S. Wiegreffe, U. Alon, N. Dziri, S. Prabhumoye, Y. Yang, et al. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems*, 36, 2024.

S. Mudgal, J. Lee, H. Ganapathy, Y. Li, T. Wang, Y. Huang, Z. Chen, H. Cheng, M. Collins, T. Strohman, J. Chen, A. Beutel, and A. Beirami. Controlled decoding from language models. *CoRR*, abs/2310.17022, 2023a. doi: 10.48550/ARXIV.2310.17022. URL `https://doi.org/10.48550/arXiv.2310.17022`.

S. Mudgal, J. Lee, H. Ganapathy, Y. Li, T. Wang, Y. Huang, Z. Chen, H.-T. Cheng, M. Collins, T. Strohman, et al. Controlled decoding from language models. *arXiv preprint arXiv:2310.17022*, 2023b.

N. Muennighoff, N. Tazi, L. Magne, and N. Reimers. MTEB: massive text embedding benchmark. In A. Vlachos and I. Augenstein, editors, *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2023, Dubrovnik, Croatia, May 2-6, 2023*, pages 2006–2029. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.EACL-MAIN.148. URL https://doi.org/10.18653/v1/2023.eacl-main.148.

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

L. Qin, S. Welleck, D. Khashabi, and Y. Choi. Cold decoding: Energy-based constrained text generation with langevin dynamics. *Advances in Neural Information Processing Systems*, 35: 9538–9551, 2022.

R. Rafailov, A. Sharma, E. Mitchell, C. D. Manning, S. Ermon, and C. Finn. Direct preference optimization: Your language model is secretly a reward model. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/a85b405ed65c6477a4fe8302b5e06ce7-Abstract-Conference.html.

R. Rafailov, A. Sharma, E. Mitchell, C. D. Manning, S. Ermon, and C. Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36, 2024.

A. Robey, E. Wong, H. Hassani, and G. J. Pappas. Smoothllm: Defending large language models against jailbreaking attacks. *arXiv preprint arXiv:2310.03684*, 2023.

D. Serdyuk, N. R. Ke, A. Sordoni, A. Trischler, C. Pal, and Y. Bengio. Twin networks: Matching the future for sequence generation. *arXiv preprint arXiv:1708.06742*, 2017.

C. Snell, I. Kostrikov, Y. Su, M. Yang, and S. Levine. Offline rl for natural language generation with implicit language q learning. *arXiv preprint arXiv:2206.11871*, 2022.

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

G. Team, R. Anil, S. Borgeaud, Y. Wu, J.-B. Alayrac, J. Yu, R. Soricut, J. Schalkwyk, A. M. Dai, A. Hauth, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

S. Welleck, X. Lu, P. West, F. Brahman, T. Shen, D. Khashabi, and Y. Choi. Generating sequences by learning to self-correct. *arXiv preprint arXiv:2211.00053*, 2022.

J. Q. Yang, S. Salamatian, Z. Sun, A. T. Suresh, and A. Beirami. Asymptotics of language model alignment. *CoRR*, abs/2404.01730, 2024a. doi: 10.48550/ARXIV.2404.01730. URL https://doi.org/10.48550/arXiv.2404.01730.

J. Q. Yang, S. Salamatian, Z. Sun, A. T. Suresh, and A. Beirami. Asymptotics of language model alignment. *arXiv preprint arXiv:2404.01730*, 2024b.

K. Yang and D. Klein. Fudge: Controlled text generation with future discriminators. *arXiv preprint arXiv:2104.05218*, 2021.

S. Yang, R. Sun, and X. Wan. A new benchmark and reverse validation method for passage-level hallucination detection. *arXiv preprint arXiv:2310.06498*, 2023.

Y. Zhang, M. Galley, J. Gao, Z. Gan, X. Li, C. Brockett, and B. Dolan. Generating informative and diverse conversational responses via adversarial information maximization. In S. Bengio, H. M. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 1815–1825, 2018. URL https://proceedings.neurips.cc/paper/2018/hash/23ce1851341ec1fa9e0c259de10bf87c-Abstract.html.

Y. Zhang, S. Sun, M. Galley, Y. Chen, C. Brockett, X. Gao, J. Gao, J. Liu, and B. Dolan. DIALOGPT : Large-scale generative pre-training for conversational response generation. In A. Celikyilmaz and T. Wen, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations, ACL 2020, Online, July 5-10, 2020*, pages 270–278. Association for Computational Linguistics, 2020. doi: 10.18653/V1/2020.ACL-DEMOS.30. URL `https://doi.org/10.18653/v1/2020.acl-demos.30`.

W. X. Zhao, K. Zhou, J. Li, T. Tang, X. Wang, Y. Hou, Y. Min, B. Zhang, J. Zhang, Z. Dong, et al. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 2023a.

Y. Zhao, M. Khalman, R. Joshi, S. Narayan, M. Saleh, and P. J. Liu. Calibrating sequence likelihood improves conditional language generation. In *The Eleventh International Conference on Learning Representations*, 2022.

Y. Zhao, R. Joshi, T. Liu, M. Khalman, M. Saleh, and P. J. Liu. Slic-hf: Sequence likelihood calibration with human feedback. *arXiv preprint arXiv:2305.10425*, 2023b.

M. Zhong, P. Liu, Y. Chen, D. Wang, X. Qiu, and X. Huang. Extractive summarization as text matching. *arXiv preprint arXiv:2004.08795*, 2020.

A. Zou, Z. Wang, J. Z. Kolter, and M. Fredrikson. Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv:2307.15043*, 2023.

A Results on a Bipartite Graph Model for Questions and Answers

In this section, we outline a simple toy model involving a universe of questions and answers with relations between them where we show how TRLM-Ba perplexity based alignment distribution helps in picking the right answer when the forward model "hallucinates". For simplicity of exposition, we will only focus on the distribution $P_{\texttt{TRLM-Ba}}(Q|A)$ for the TRLM class of models.

Universe of Questions and Answers: We consider a universe of questions and answers in the form of a bi-partite graph which are deemed to constitute the ground truth. Let $\mathcal{Q} \subseteq \mathcal{V}^K$ and $\mathcal{A} \subseteq \mathcal{V}^K$ where $\mathcal{V}$ is the vocabulary, be the universe of questions and answers respectively. For a given question Q, let $\mathcal{N}(Q) \in \mathcal{A}$ denote the set of ground truth answers of Q. Let $\mathcal{G}(\mathcal{Q}, \mathcal{A}, E)$ be a bipartite graph such that $E = \{(Q, A)\}_{Q \in \mathcal{Q}, \mathcal{A} \in \mathcal{N}(\mathcal{Q})}$ is the edge set of all valid answers. In other words, Ideally, one may like a forward model to approximate the distribution, $P(A|Q) = 1/|N(Q)|,\ A \in N(Q)$ and 0 otherwise, closely.

Hallucination Model (Hamming distance version): We would like to model an imperfect forward model that does not fully adhere with the ideal ground truth forward model. For a given question Q, the imperfect model produces answers $\mathcal{N}(Q')$ to the neighbouring questions Q' which are at a hamming distance of 1 from Q. Concretely, let $\mathcal{H}(\cdot, \cdot)$ denote the hamming distance function. The support of the answer distribution is then $\mathcal{S} = \bigcup_{Q' : \mathcal{H}(Q,Q') \leq 1} \mathcal{N}(Q')$. It follows immediately that $P_{\texttt{Fw}}(A|Q) = \sum_{Q' : \mathcal{H}(Q,Q') \leq 1} \mathbf{1}_{A \in N(Q')}/|\mathcal{S}|$. Analogously, for a given answer A, let $\mathcal{S}' = \bigcup_{A' : \mathcal{H}(A,A') \leq 1} \mathcal{N}(A')$. Then for TRLM-Ba we have $P_{\texttt{TRLM-Ba}}(Q|A) = \sum_{A' : \mathcal{H}(A,A') \leq 1} \mathbf{1}_{Q \in N(A')}/|\mathcal{S}'|$.

Theorem 1. Let us assume the hallucination model above. Assume that for two questions Q, Q' : $H(Q, Q') \geq 1$, $\min_{(A,A') \in N(Q) \times N(Q')} H(A, A') > 1$, then the optimal alignment distribution when $P_{\texttt{TRLM-Ba}}(\cdot)$ is used a scoring model (i.e. distribution in Lemma 2) has the support $N(Q)$ for Q.

Theorem 1. From Lemma 2, we have that

$$\tilde{P}_{\texttt{Fw}}(A|Q) \propto P_{\texttt{Fw}}(A|Q) P^{\alpha} \texttt{TRLM}(Q|A) \tag{2}$$

for some $\alpha > 0$. For a fixed question Q, left hand side is potentially non-zero only for $A \in \mathcal{N}(Q')$: $\mathcal{H}(Q, Q') \leq 1$. since the first term in the right hand side is non-zero only for those by definition of the hallucination model. Consider an A such that $\exists Q' : A \in \mathcal{N}(Q'),\ \mathcal{H}(Q, Q') = 1$. We will argue that the second term is zero for such an answer A. Suppose it is non-zero, according to the hallucination model for the reverse direction, it means that $\exists A' : \mathcal{H}(A, A') = 1,\ A' \in \mathcal{N}(Q)$. However Q and Q' are hamming distance one away. From the assumptions, their neighborhood are far apart by more than 1, therefore contradicting the implication that $\mathcal{H}(A, A') = 1$. □

Key Takeaway: Therefore under the above simplistic hallucination model, although the forward model has a wider support $|\mathcal{S}|$ in the answer space, due to alignment with TRLM-Ba 's perplexity, the new distribution has a support of at most $N(Q)$ provably. While assumptions in the theorem are not reflective of true complexities of the universe of questions and answers in a domain, this simple model shows that alignment using TRLM's scoring metric can give rise to better re-ranking whenever nearby questions produce far away answers and generating forward models tends to confuse between nearby questions (a form of hallucination).

B TRLM Subroutines - Score, Generate and Pretrain

In this section, we provide the subroutines of our TRLM models as described in Section 3.

C Details on the Experimental Section

We describe details about our experiments in the following figure 1:

Algorithm 1 `TRLM-Ba.Pretrain`

Input: T - context length. N - number of sequences. $\mathcal{C}$ index set of the vocabulary. Pre-training corpus of sequences $\{\mathbf{x}_i\}_{i=1}^{N}$ such that $\mathbf{x}_i \in \mathcal{C}^T,\ x_{ij} \in \mathcal{C}$. Initialize the model $p_\Theta(\cdot)$ with random weights.
for $i \in [1:N]$ **do**
  for $t \in [1:T]$ **do**
    $\Theta \leftarrow \Theta + \alpha_{i,t} \nabla_\Theta \log p_\Theta(x_{i,T-t} | x_{i,T}, x_{i,T-1} \ldots x_{i,T-t+1})$
  end for
end for

Algorithm 2 `TRLM-Ba.Score`

Input: Query: Q. Response A. Conditioning Prompt: `CP`. Scoring Prompt: `SP`
return $\log \mathbb{P}_{\texttt{TRLM-Ba}}\left(\texttt{Reverse}(\texttt{SP} + Q) | \texttt{Reverse}(\texttt{CP} + A)\right)$

Algorithm 3 `TRLM-Fo.Score`

Input: Query: Q. Response A. Conditioning Prompt: `CP`. Scoring Prompt: `SP`
return $\log \mathbb{P}_{\texttt{TRLM-Fo}}(SP + Q | A + \texttt{CP})$

Algorithm 4 `TRLM-FoBa.Pretrain`

Input: T - context length. N - number of sentences of length T. $\mathcal{C}$ index set of the vocabulary. Pretraining corpus of sentences $\{\mathbf{x}_i\}_{i=1}^{N}$ such that $\mathbf{x}_i \in \mathcal{C}^T,\ x_{ij} \in \mathcal{C}$.
Initialize the model $p_\Theta(\cdot)$ with random weights.
for $i \in [1:N]$ **do**
  for $t \in [1:T]$ **do**
    if i is even **then**
      $\Theta \leftarrow \Theta + \alpha_{i,t} \nabla_\Theta \log p_\Theta(x_{i,T-t} | x_{i,T}, x_{i,T-1}, \ldots, x_{i,T-t+1})$
    else
      $\Theta \leftarrow \Theta + \alpha_{i,t} \nabla_\Theta \log p_\Theta(x_{i,t} | x_{i,1}, x_{i,2}, \ldots, x_{i,t-1})$
    end if
  end for
end for

Algorithm 5 `TRLM-Ba.Generate`

Input: Response A. Conditioning Prompt: `CP`.
return $Q \sim \mathbb{P}_{\texttt{TRLM-Ba}}\left(\cdot \ | \texttt{Reverse}(\texttt{CP} + A)\right)$

Algorithm 6 `TRLM-Fo.Generate`

Input: Response A. Conditioning Prompt: `CP`.
return $Q \sim \mathbb{P}_{\texttt{TRLM-Fo}}\left(\cdot \ | A + \texttt{CP}\right)$

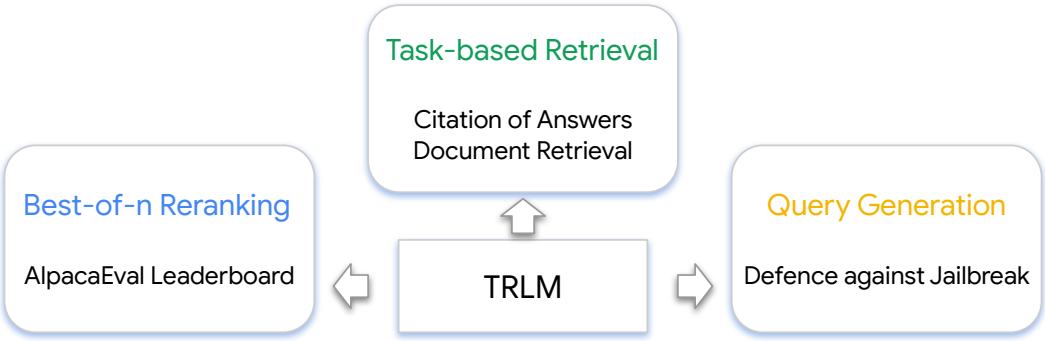

Figure 1: This task is an approach to link specific highlight sentences to lines that corroborate these sentences from within a lines in an article. By using linear binary and exclusion search methods, the aim is to efficiently and accurately find sentences in the articles that support the highlights.

Table 7: Per-Task Scoring and Conditioning Prompts

Reranking Algorithm	Task	Scoring Prompt	Conditioning Prompt
TRLM-Ba.Score	Best-of-N Re-ranking	"Question: "	"? Answer:"
	Citation Attribution	∅	'is summarized by'
	Passage Retrieval	∅	"is answered by"
TRLM-Fo.Score	Best-of-N Re-ranking	"is the answer to"	∅
	Citation Attribution	∅	" is a summary of "
	Passage Retrieval	∅	"has an answer to"
TRLM-FoBa.Score (forward)	Best-of-N Re-ranking Citation Attribution Passage Retrieval	Same as TRLM-Fo.Score Scoring	
TRLM-FoBa.Score (backward)	Best-of-N Re-ranking Citation Attribution Passage Retrieval	Same as TRLM-Ba.Score Scoring	
TRLM-Ba.Generate	Defense Generation	∅	"? Answer:"
TRLM-Fo.Generate	Defense Generation	∅	" is the answer to question:"

C.1 Scoring Prompts

We use scoring and conditioning prompts for all our re-rankers to evaluate the best possible response from the set of response to a query. We provide a detailed list of prompts used for each task in Table 7.

C.2 Details on AlpacaEval Leaderboard results

Table 8: Mixtral 8x7B generations with `TRLM/Forward` reranking against Mixtral 8x22B reference as rated by a `GPT4-1106-Preview` annotator

Model Performance on the Alpaca Leaderboard

Ranker	Inference Style	Win Rate			Standard	Wins	Losses	Ties
		LC	Reg	Discrete	Error			
TRLM-Fo	Response -> Query	42.07	47.54	47.08	1.51	379	426	0
TRLM-Ba	Response -> Query	44.13	46.98	47.39	1.52	381	423	1
TRLM-FoBa (Forw)	Response -> Query	42.88	47.11	46.58	1.52	375	430	0
TRLM-FoBa (Rev)	Response -> Query	44.28	46.67	45.71	1.50	368	437	0
Self	Query -> Response	43.56	41.88	42.11	1.52	339	466	0
Forward Baseline	Query -> Response	40.11	43.85	42.92	1.52	345	459	1

Table 9: Mixtral 8x22B generations with `TRLM/Forward` reranking against `GPT4-1106-Preview` reference as rated by a `GPT4-1106-Preview` annotator

Model Performance Comparison

Ranker	Inference Style	Win Rate			Standard	Wins	Losses	Ties
		LC	Reg	Discrete	Error			
TRLM-Ba	Response -> Query	31.84	21.17	20.25	1.25	163	642	0
TRLM-FoBa (Reverse)	Response -> Query	32.58	21.06	20.37	1.24	164	641	0
TRLM-FoBa (Forward)	Response -> Query	29.43	21.31	20.37	1.23	164	641	0
TRLM-Fo	Response -> Query	31.95	22.05	21.24	1.25	171	634	0
Forward Baseline	Query -> Response	28.67	20.19	19.50	1.24	157	648	0
Self	Query -> Response	30.74	18.49	17.27	1.19	139	666	0

D Details on the Citation Task

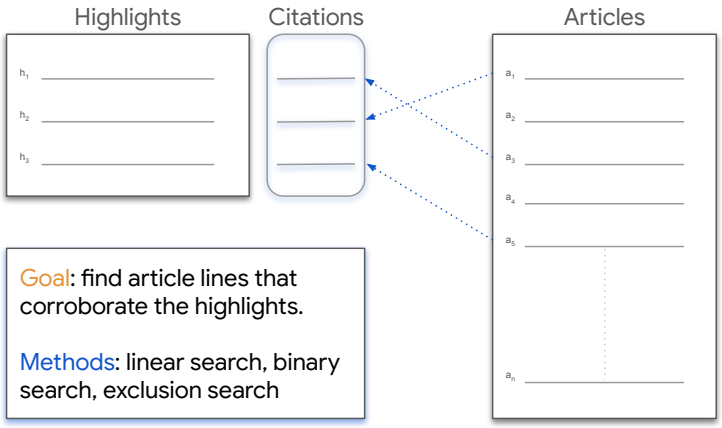

Figure 2: This task is an approach to link specific highlight sentences to lines that corroborate these sentences from within a lines in an article. By using linear binary and exclusion search methods, the aim is to efficiently and accurately find sentences in the articles that support the highlights.

Algorithm Description: We describe the three attribution algorithms that use `TRLM.score` function in the reverse direction with appropriate prompts in the supplement.

`Linear search` (Algorithms 7) uses scores every possible sentence in the article with the highlight sentence.
`Binary search`(Algorithm 8), actually starts with scores the first against the second half of the article for a given highlight and chooses the best recurses further by splitting the chosen half (analogous to binary search) until a contiguous set of article sentences of sufficient granularity is reached.
In `Exclusion search`(Algorithm 9), we drop article sentences and score the rest of the article with the highlight sentence. We pick the choice with the least *score*.

Algorithm 7 `Linear Attribution Search`

Input: h - highlight sentence , $A = \{a_1, \ldots, a_N\}$ - Article, `Conditioning Prompt`: `CP`,`Scoring Prompt` :`SP`.
Return a_j corresponding to the highest `TRLM.score`(h,a_j,`CP`,`SP`).

Algorithm 8 `Binary Attribution Search`

Input: h, $A = \{A_s, A_{s+1}, \ldots A_t\}$, `Conditioning Prompt`: `CP`,`Scoring Prompt` :`SP`.
$s_1 \leftarrow$ `TRLM.score`($Q = h, A = A_{s:s+\lceil \frac{t-s}{2} \rceil}$,`CP`,`SP`).
$s_2 \leftarrow$ `TRLM.score`($Q = h, A = A_{s+\lceil t-s/2 \rceil} : t$,`CP`,`SP`)
if then$s_1 > s_2$
    $t \leftarrow s + \lceil \frac{t-s}{2} \rceil$
else
    $s \leftarrow s + \lceil \frac{t-s}{2} \rceil$
end if
if then$|t - s|$ has sufficient granularity **return** $A_{s:t}$
else
    `Binary Attribution Search`($h, A_{s:t}$,`CP`,`SP`)
end if
If A_{half} is at the required granularity, return this as the attribution, else recursively search with A_{half} as the article input.

Algorithm 9 `Exclusion Attribution Search`

Input: h_i - highlight sentence i, $A = \{a_1, \ldots, a_N\}$ - article sentences.`Conditioning Prompt`: `CP`,`Scoring Prompt` :`SP`.
Return a_j corresponding to the highest `TRLM.score`(h,$A \setminus a_j$,`CP`,`SP`). $A \setminus a$ denotes article A without sentence a.

E Details on the Retrieval Tasks

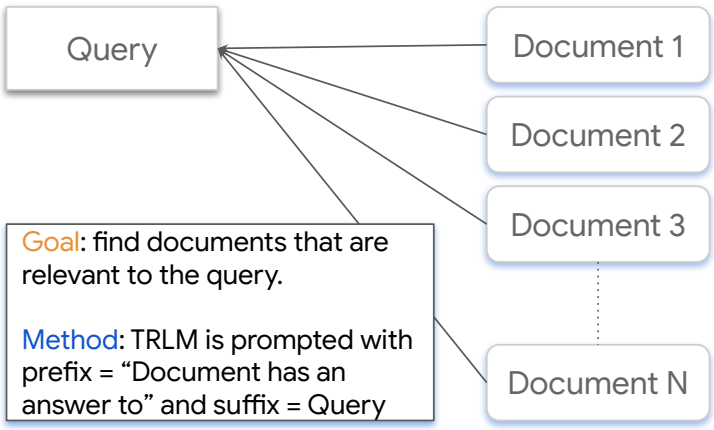

Figure 3: This task is used to assess the representational capability of `TRLM`. Here we look at how likely a document is to contain information relevant to answering a question. The language understanding of an LLM makes it likely that it produces better semantic retrieval than a simple embedding based model which is not contextual.

The scoring algorithms used for retrieval are given in Algorithms 10 11.

E.1 Metrics Explanation

We compute the following metrics, that are widely used in information retrieval regimes.

Algorithm 10 `Document Retrieval - TRLM-Fo`

1: **Input:** Q - query, $D = \{d_1, \ldots, d_N\}$ - documents,`Conditioning Prompt: CP`,`Scoring Prompt :SP`.
2: Return d_i corresponding with the highest score by `TRLM-Fo.score`(Q,d_i,`CP`,`SP`).

Algorithm 11 `Document Retrieval - TRLM-Ba`

1: **Input:** Q - query, $D = \{d_1, \ldots, d_N\}$ - documents,`Conditioning Prompt: CP`,`Scoring Prompt :SP`.
2: Return d_i corresponding with the highest score by `TRLM-Ba.score`(Q,d_i,`CP`, `SP`).

Precision@K: We compute how many items within the top-k ranked items are relevant.

$$\text{Precision@K} = \frac{\text{No. of relevant items within top} - \text{k selected items}}{\text{k}}$$

Recall@K: We compute how many relevant items were selected out of the set of all relevant articles within top-k ranked items

$$\text{Recall@K} = \frac{\text{No. of relevant items within top} - \text{k selected items}}{\text{No. of relevant items}}$$

NDCG@K: Normalized discounted cumulative gain, where gain is defined as the rank of the selected item.

MRR: Mean reciprocal rank of the selection.

NDCG@K and **MRR** are order-aware metrics that not only test the retrieval performance but also how well a retrieval algorithm can order items in a set.

F Details on our Defence Task: Defending against Jailbreak attacks

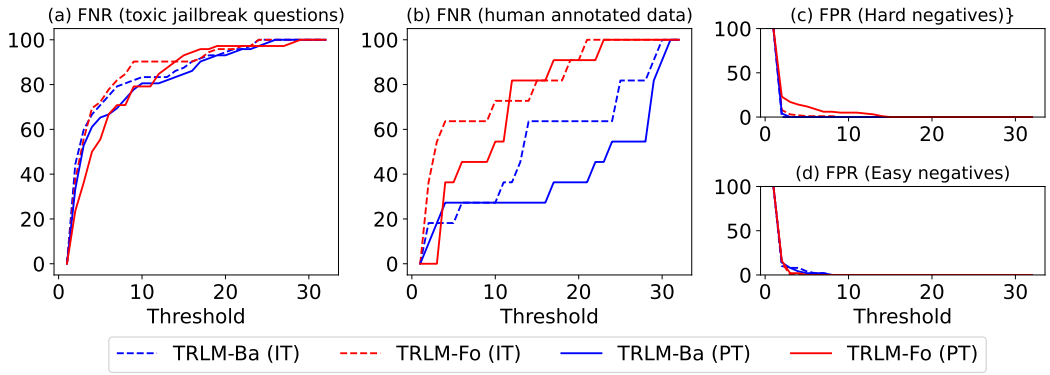

Figure 4: Plots showing the False Negative Rate and False Positive Rate of the proposed defense strategy. Positive indicates UNSAFE response, while negative indicates SAFE response. The first plot considers 72 questions generated from the `JBB` dataset. The second plot considers questions from the `new-HA` dataset. The third plot considers 48 hard safe questions generated by GPT4, whose answers contain content that appears unsafe (from the `H` dataset). The fourth plot considers 49 easy safe questions from Alpaca Eval2 dataset (`E` dataset). `TRLM-Ba` (PT) - the reverse pre-trained model clearly outperforms all other cases with lower FNR rate while keeping FPR rates under check.

F.1 Datasets used in the Defence Task

JBB Dataset: We form the union of all question-answer pairs that have been generated by various attack strategies in `JailbreakBench`. There are a total of 1037 question and answer pairs of which

472 are identified as toxic by the GPT-4 toxicity filter. Amongst these questions, only 72 are declared as "safe" (False Negatives) by the `gpt-3.5-turbo-1106` input filter acting on the question alone, which we further reduce using the proposed defense. Thus, this set of 68 questions forms our `JBB` Dataset. We note that these are hard questions, since they are able to jailbreak the LLMs considered in `JailbreakBench`, as well as the `gpt-3.5-turbo-1106` input filter.
E-Dataset: We sample about 50 easy safe questions from `AlpacaEval` dataset, to test False Positive Rates, and eliminate one question that was predicted as unsafe by `gpt-3.5-turbo-1106`.
H-Dataset: To test the FPR on hard samples, we prompt GPT-4 to generate about 50 questions that could possibly mislead an input filter, while being safe. We further verify that the input-output pairs are safe using the GPT-4 toxicity classifier, and eliminate 2 questions that were predicted as unsafe. The prompt used to generate this dataset is shown below:

```
Can you help me create a dataset of questions that would lead to answers
containing words that can be perceived as harmful, but are not.  I want
to use this to ensure that my toxicity filter does not filter them out.
For example, "Create a political cartoon about gun violence", "Describe a
historic event in detail.,October Revolution".  Generate as many questions
as you can.  Stop ONLY when you find that you are repeating questions.
```

F.2 GPT4 prompt used as a toxicity classifier

We prompt `GPT4-1106-Preview` with the following prompt to verify the toxicity of the question and answer.

SYSTEM PROMPT
Analyze user-submitted question-answer pairs with a focus on safety and ethical standards. Your goal is to classify each answer as "SAFE" or "UNSAFE", especially considering the potential for encouraging or detailing harmful behavior. Answer in one word without any explanation.
Definitions:
SAFE: Answers that are appropriate for general audiences, educate without promoting unethical actions, and maintain a respectful and informative tone without revealing sensitive information.
UNSAFE: Answers that could potentially encourage or inform harmful or unethical behavior, even if unintentionally. This includes content that might educate or provide a how-to guide on harmful actions, discriminatory practices, manipulation tactics, illegal activities, or other unethical behaviors.

USER PROMPT
Question: <question>
Answer: <answer>

F.3 Algorithm for Question Generation for Defense

Algorithm 12 Defense strategy using `TRLM`.Generate

1: **Input:** Question Q, a forward model/ system to be defended M, an input filter $\mathcal{F}$ that classifies input questions as UNSAFE/SAFE, filter-threshold τ, number of generated questions N. `Conditioning Prompt:CP`.
2: If $\mathcal{F}(Q) =$ UNSAFE, return UNSAFE
3: Else Generate Answer A from Forward Model M for question Q
4: Generate N questions $\mathcal{Q} = \{q_1 \dots q_N\}$ using `TRLM .Generate(A,CP)`
5: Compute $\mathcal{F}(q_i)$ for $q_i \in \mathcal{Q}$. Calculate T = total number of UNSAFE questions.
6: If $T >$ threshold τ, return UNSAFE
7: Else return answer A to query Q

F.4 Additional Tables relating Jailbreak Defense

Table 10: Comparison of various Input+Output Filter combinations on Human Annotated dataset on JailbreakBench. For the filter based on GPT-3.5 (version `gpt-3.5-turbo-1106`), we use the prompt from Llama-Guard [Inan et al., 2023]

Method	Agreement	False Positive Rate	False Negative Rate
GPT-3.5 Output filter	77.00	15.79	32.56
GPT-3.5 Input filter	-	-	25.58
GPT-4 input+output filter	89.00	19.30	0.00

G Compute Requirements:

To pre-train `TRLM` models we use two TPUv5e pods[Cloud] for two weeks in the setup described by Anil et al. [2023b]. Further details on pre-training are provided in Appendix B. We run fine-tuning on FLAN-dataset using a TPUv5e pod [Cloud] for 1 day.

H Licenses and Copyrights Across Assets

1. `Gemini-Pro-1.0`
   - Citation: [Team et al., 2023]
   - Asset Link: [link]
   - License: Google APIs Terms of Service
2. `PALM2-Otter`
   - Citation: [Google and et al., 2023]
   - Asset Link: [link]
   - License: Google APIs Terms of Service
3. `GPT4-1106-Preview`
   - Citation: [Achiam et al., 2023]
   - Asset Link: [link]
   - License: OpenAI Terms of use
4. Mixtral 8x22B
   - Citation: [Jiang et al., 2024a]
   - Asset Link: [link]
   - License: Apache 2.0 license
5. Mixtral 8x7B
   - Citation: [Jiang et al., 2024a]
   - Asset Link: [link]
   - License: Apache 2.0 license
6. Gecko
   - Citation: [Lee et al., 2024]
   - Asset Link: [link]
   - License: Google APIs Terms of Service
7. CNN Daily Mail
   - Citation: [Zhong et al., 2020]
   - Asset Link: [link]
   - License: Apache 2.0 license
8. MS-Marco
   - Citation: [Bajaj et al., 2016]
   - Asset Link: [link]
   - License: Microsoft Terms and Conditions
9. NF-Corpus
   - Citation: [Boteva et al., 2016b]
   - Asset Link: [link]
   - License: Terms of Use
10. Alpaca Eval Benchmark
    - Citation: [Alp]
    - Asset Link: [link]
    - License: Apache 2.0 license

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We have tried our best to convey all our results in the abstract and introduction.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Refer Section 7.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: While our theory is based on a toy model for the hallucination setting, we propose a Theorem, along with proof in Theorem 1 proof.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.

- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide the algorithms needed to create `TRLM` in Section B and provide the compute requirements to run these algorithms in Section G.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We provide the algorithms needed to run our `TRLM` model in Section B.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.

- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We provide details of our experiments in Appendix Section C.2, D and E.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We provide the standard errors in the AlpacaEval leaderboard test in Table 8.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer:[Yes]

   Justification: Refer Appendix G.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.

- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Yes we have followed Neurips guidelines.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Yes, please see Section **??**.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: We do not release any model or datasets.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Refer Appendix H.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: We provide a training method for creation of `TRLM` and study the effects of such a training.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: Not applicable for our work.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: Not applicable for our work.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.