A Causal Explainable Guardrails for Large Language Models

Zhixuan Chu[*†]
Zhejiang University
Hangzhou, China
zhixuanchu@zju.edu.cn

Yan Wang[*]
Ant Group
Hangzhou, China
luli.wy@antgroup.com

Longfei Li
Ant Group
Hangzhou, China
longyao.llf@antgroup.com

Zhibo Wang
Zhejiang University
Hangzhou, China
zhibowang@zju.edu.cn

Zhan Qin
Zhejiang University
Hangzhou, China
qinzhan@zju.edu.cn

Kui Ren
Zhejiang University
Hangzhou, China
kuiren@zju.edu.cn

Abstract

Large Language Models (LLMs) have shown impressive performance in natural language tasks, but their outputs can exhibit undesirable attributes or biases. Existing methods for steering LLMs toward desired attributes often assume unbiased representations and rely solely on steering prompts. However, the representations learned from pre-training can introduce semantic biases that influence the steering process, leading to suboptimal results. We propose LLMGuardrail, a novel framework that incorporates causal analysis and adversarial learning to obtain unbiased steering representations in LLMs. LLMGuardrail systematically identifies and blocks the confounding effects of biases, enabling the extraction of unbiased steering representations. Experiments demonstrate LLMGuardrail's effectiveness in steering LLMs toward desired attributes while mitigating biases. Our work contributes to developing safe and reliable LLMs that align with desired attributes.

CCS Concepts

• **Computing methodologies** → **Natural language generation**; **Causal reasoning and diagnostics**.

Keywords

Large Language Model, Causal Inference, Explanation, Trustworthiness, Safety

ACM Reference Format:
Zhixuan Chu, Yan Wang, Longfei Li, Zhibo Wang, Zhan Qin, and Kui Ren. 2024. A Causal Explainable Guardrails for Large Language Models. In *Proceedings of the 2024 ACM SIGSAC Conference on Computer and Communications Security (CCS '24), October 14–18, 2024, Salt Lake City, UT, USA.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3658644.3690217

[*]These authors contributed equally to this work.
[†]Corresponding author. The author is with the State Key Laboratory of Blockchain and Data Security & Hangzhou High-Tech Zone (Binjiang) Institute of Blockchain and Data Security, Hangzhou, China.

CCS '24, October 14–18, 2024, Salt Lake City, UT, USA

ACM ISBN 979-8-4007-0636-3/24/10
https://doi.org/10.1145/3658644.3690217

1 Introduction

Large Language Models (LLMs) have demonstrated remarkable capabilities in natural language understanding and generation, enabling a wide range of applications such as dialogue systems, clinical report generation, professional agents, recommendation systems, and so on [19, 20, 28, 47, 68]. However, the training of these models on massive web-scraped corpora has led to the manifestation of undesirable behaviors, such as generating offensive, toxic, or false outputs [21, 34, 71]. Addressing these issues is crucial, as content safety, fairness, toxicity, harmfulness, and factuality demand rigorous consideration, especially when language models are deployed in high-stakes applications and user-facing systems.

To address these challenges, recent research efforts have focused on developing methods to steer the output of LLMs toward desired attributes or concepts. Several approaches have been proposed to control and steer the generation process, aiming to mitigate these harmful behaviors and improve the overall quality and safety of generated content. Fine-tuning on carefully curated datasets has been a common technique to enhance content safety and regulate outputs. By training language models on datasets that have been filtered and cleaned to remove offensive, biased, or misleading content, researchers aim to reduce the likelihood of the model generating such harmful outputs. In addition to fine-tuning, other techniques have been explored to further improve the generation process. Reinforcement learning from human feedback (RLHF) [52] is one such approach, where the language model is trained using feedback from human annotators who evaluate the quality and safety of generated outputs. Another promising approach is reinforcement learning from AI feedback (RLAIF) [38], which extends the concept of RLHF by using an AI system to provide feedback instead of human annotators. However, they all require huge annotation and computation resources. Furthermore, the training process often involves a human or AI annotator providing feedback and guidance to the language model. This raises the possibility that some form of deception or manipulation could be introduced during the training process, either intentionally or unintentionally [56, 61, 62]. In addition, these methods often lack explainability and interpretability, leading to inconsistent performance and limited generalizability [32, 42, 74].

Recent studies have shed light on the rich semantic information encoded within LLM representations, including specific information [29], concepts [51, 76], or attributes [6, 40]. Building on these

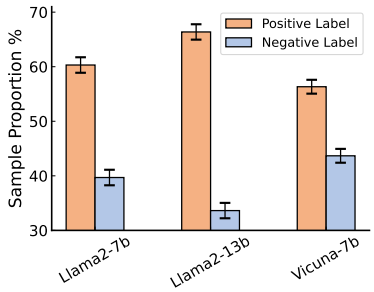

Figure 1: Proportion of representations of semantic prompts that implicitly encode positive or negative directions for semantically neutral prompts without explicit steering prompt, across different language models. The varying proportions of positive and negative directions learned by the probing classifier, even in the absence of steering prompts, demonstrate the presence of inherent biases in the representations of semantic prompts due to differences in pre-training data. This observation supports the existence of a direct edge from the semantic direction representation R^{cd} of the semantic prompt to the direction representation R_+/R_-, as discussed in the causal analysis section.

findings, activation engineering has emerged, which induces desired outputs in frozen LLMs by injecting crafted activation vectors during forward passes. This approach holds promise for controlling the generation process in a more interpretable and fine-grained manner. Existing approaches for steering LLMs typically involve constructing positive and negative prompts that represent the desired and undesired attributes, respectively. By comparing the activations of the model for these prompts, a steering vector is obtained, which is then used to guide the model's output during inference. While these methods have shown promising results, they often rely on the assumption that the constructed prompts are sufficient to capture the desired steering direction and that the model's representations are unbiased. However, a closer examination reveals that the representations learned by LLMs during pre-training can introduce biases that influence the steering process. As shown in Figure 1, the semantic context of the prompts used for steering can implicitly encode biases, even without explicit steering prompts. Consequently, the obtained steering vectors may be confounded by these inherent biases, leading to suboptimal or unintended steering results. This observation highlights a critical gap in current steering approaches: the need to account for and mitigate the influence of semantic biases on the steering process. Without addressing this issue, steering methods may inadvertently perpetuate or even amplify existing biases in the model's outputs, undermining efforts to improve the safety and reliability of LLMs. Therefore, simply structuring a controlled trial superficially is not enough to get an unbiased steering representation.

To address these limitations, we propose a novel framework called LLMGuardrail that incorporates causal analysis to obtain unbiased steering representations for LLMs. Our framework aims to disentangle the influence of biases from the steering process by employing debiasing techniques. By systematically identifying and blocking the confounding effects of biases, LLMGuardrail enables the extraction of steering representations that accurately capture the desired attributes. The key contributions of this work are as follows:

- A causal analysis of the steering process in LLMs, identifying the confounding effects of semantic prompt and their impact on steering representations. We provide a theoretical formulation of the problem and discuss the limitations of existing methods.
- A novel framework for obtaining unbiased steering representations in LLMs. Our framework employs adversarial learning techniques to disentangle the influence of semantic biases from the steering process.
- An explainable component that provides insights into the alignment between the generated output and the desired direction, enhancing the interpretability of the steering process.
- Comprehensive experiments and analysis demonstrating the effectiveness of LLMGuardrail in steering LLMs toward desired attributes while mitigating the influence of biases. We evaluate our framework on various benchmark datasets and compare its performance with existing methods.

2 Background

2.1 Representations in Large Language Models

Large language models (LLMs) have demonstrated remarkable capabilities in natural language processing tasks, generating human-like text and exhibiting a broad understanding of language. Central to their success is the rich set of representations they learn during training, which encode various concepts, attributes, and semantic information. Extensive research has been dedicated to understanding the representations learned by large language models (LLMs) and how they encode various concepts and attributes [76]. LLM representations refer to the patterns of activations across the model's parameters that correspond to specific semantic concepts, properties, or features. These representations act as the model's internal codification of the information learned from the training data. Recent studies have provided compelling evidence that LLM representations contain rich semantic information, including abstract concepts like space and time [29], as well as more granular attributes related to truthfulness, toxicity, bias, and harmfulness of the generated text [2, 8, 11, 40]. In addition, linear classifier probes, which are trained to predict input properties from intermediate layers of a network, have successfully identified representations of concepts [1, 4]. Latent space analysis has enabled researchers to locate or edit factual associations within LLMs [31, 50, 75]. The ability to locate and manipulate these representations opens up avenues for controlling and steering the model's outputs, enabling the development of principled methods for mitigating harmful behaviors and enhancing the interpretability and trustworthiness of large language models.

2.2 LLM Activation Engineering

Activation engineering is a set of techniques that modify the internal activations of a pre-trained language model during inference to steer its output in a desired direction. The main idea is to identify and manipulate specific activations or attention heads associated with particular attributes or behaviors to control the model's generation process. Early approaches, like the Plug-and-Play Language

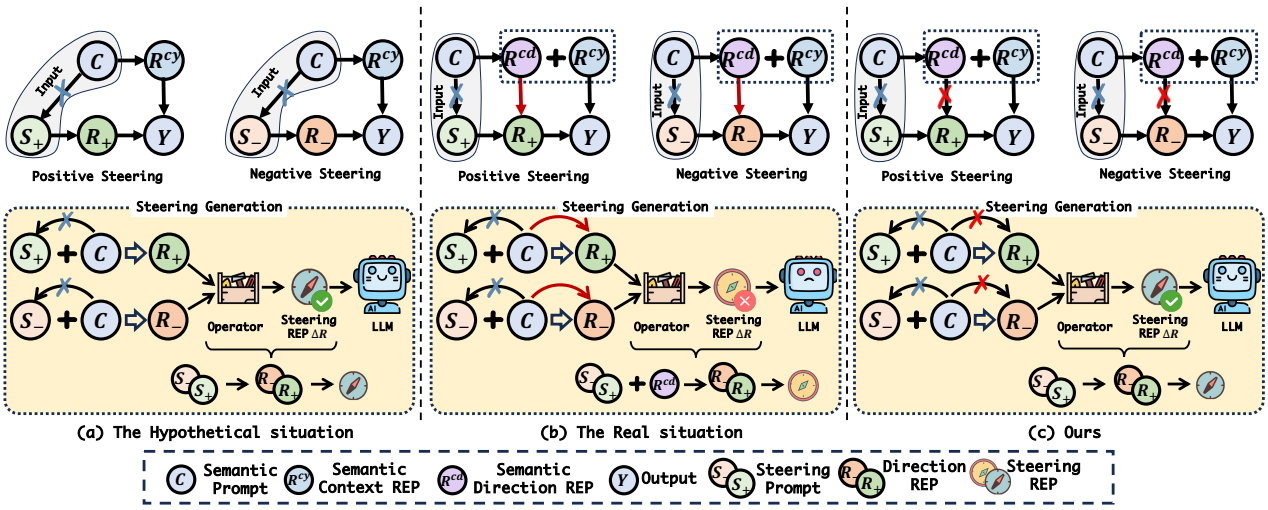

Figure 2: The causal analysis of our proposed LLMGuardrail. (a) The Hypothetical situation: The constructed pair of prompts can block the edge from the semantic prompt C to the steering prompt S and the direction representation R^+/R^-. (b) The Real situation: In addition to the edge from the semantic prompt C to the steering prompt S, there is a direct edge from the semantic direction representation R^{cd} of the semantic prompt to the direction representation R^+/R^-. (c) Ours: We need to block the edge from the semantic direction representation R^{cd} of the semantic prompt to the direction representation R^+/R^-. The direction representation R^+/R^- is only influenced by the steering prompt S^+/S^-, enabling us to obtain an unbiased steering representation ΔR through steering engineering.

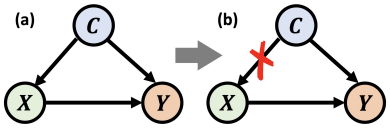

Figure 3: The causal graph and backdoor adjustment.

Model [23], use a separate classifier to detect target attributes in the generated text and perturb the language model's activations accordingly, encouraging it to generate text that aligns with the desired attribute. Recent work focuses on extracting latent steering vectors from a frozen language model, which can be added to the activations during inference to steer the model's completions toward specific goals, such as achieving high BLEU scores [60] or generating truthful statements [40]. Instead of requiring additional optimization or labeled data, Activation Addition [65] takes activation differences resulting from pairs of prompts. To avoid the identification of an opposite behavior, [36] takes the average of activations associated with a target dataset and then subtracts the mean of all training activations, resulting in effective steering vectors. The growing interest in activation engineering stems from the desire to control and improve the performance of large language models on specific tasks or domains by identifying and manipulating the relevant activations.

2.3 Causal Inference

Causal inference [16, 22, 54, 58, 72] has been an attractive research topic for a long time as it provides an effective way to uncover causal relationships in real-world problems. Nowadays, causal inference has shown potential in enhancing LLMs from a causal view in improving the LLMs' reasoning capacity [12, 15, 27, 46, 73], addressing fairness issues in LLMs [25, 49, 59], and safety [3, 9, 44], complementing LLMs with explanations [5, 45, 66, 74], and handling multimodality [37, 53]. In causal inference, if a variable is the common cause of two variables, it is called the *confounder*. The confounder will induce a spurious correlation between these two variables to disturb the recognition of the causal effect between them. As shown in Figure 3(a), $X \rightarrow Y$ denotes that X is the cause of Y. C is the cause of both X and Y. Thus, it is a confounder that will induce a spurious correlation between X and Y to disturb the recognition of the causal effect between them. In particular, such spurious correlation is brought by the backdoor path created by the confounder. Formally, a backdoor path between X and Y is defined as any path from X to Y that starts with an arrow pointing into X. For example, the path $X \leftarrow C \rightarrow Y$ is a backdoor path. If we want to deconfound two variables X and Y to calculate the true causal effect, we should block every backdoor path between them [55]. For example, in Figure 3(b), we should block $X \leftarrow C \rightarrow Y$ to get the causal effect between X and Y.

3 Causal Analysis

3.1 The Hypothetical Situation

Recent activation engineering-based work [8, 36, 40, 65, 76] utilize the steering vectors to control the direction of LLM output by constructing the randomized controlled trials. For example, some work [8, 65, 76] construct a pair of natural-language prompts (p_+, p_-), where p_+ represents the attribute we wish output text to emphasize and p_- represents its opposite. R_+/R_- is the representation for the prompt p_+/p_-. The difference ΔR is a new steering vector that (intuitively) captures the difference between a prompt with the attribute and without it. To obtain a steering vector, they perform a forward pass on each prompt, record the activations at the given location in each pass, take the difference, and then finally rescale this difference in activations by an "injection coefficient"

β. To steer, they add the resulting steering representation to the original representations and allow the forward pass to continue, and obtain the steered output.

To systemically analyze this series of methods, we give the formal definitions. Now, let's further break down and analysis of the constructed prompt pairs. In fact, the input prompt is comprised of the semantic prompt C and the steering prompt S. For example, in the [65], "I love talking about weddings" is the positive prompt and "I hate talking about weddings" is the negative prompt. In these two examples, "love" and "hate" are the steering prompts, which control the directions of output. The "I ... talking about weddings" is the same for this pair of prompts, which only contains the semantic information. Therefore, they assume this is a randomized controlled trial, where there are two groups, i.e., treatment group "love" and control group "hate". As shown in Figure 2 (a), the semantic prompt C is a confounder, that can affect the steering prompt S and output Y. In this hypothetical situation, they construct the randomized controlled trial by creating the pairs, i.e., positive steering prompt plus the same semantic prompt and negative steering prompt plus the same semantic prompt. The only difference is the steering prompt since the semantic prompts are the totally same. Based on this assumption, the edge from the semantic prompt to the steering prompt is blocked. Therefore, they can get the difference in activations, which can intervene in the generated output. The strength of this steering vector is called the treatment effect in causal inference. On the face of it, this is a perfect randomized controlled trial, where only the steering prompt can affect the direction representation R_+/R_-. The combination of direction representation R_+/R_- and the semantic prompt together have an influence on the generated output Y. In this case, the steering vector can be obtained by the operation between positive direction representation R_+ and negative direction representation R_-.

3.2 The Real Situation

Before delving into the analysis of the real situation, we first provide formal definitions for the various types of representations involved in the steering process.

Definition 3.1 (Direction Representation $R_{+/-}$)**.** A direction representation $R_{+/-}$ is a representation that solely affects the direction of the output with respect to specific attributes such as truthfulness, bias, harmfulness, or toxicity. It is learned from the steering prompt and should be independent of the semantic context of the output.

Definition 3.2 (Semantic Context Representation R^{cy})**.** A semantic context representation R^{cy} is a representation learned from the semantic prompt, which contains information about the context of the output. It does not provide guidance for the direction of the output with respect to specific attributes.

Definition 3.3 (Semantic Direction Representation R^{cd})**.** A semantic direction representation R^{cd} is a representation learned from the semantic prompt that implicitly influences the direction of the output with respect to specific attributes. This influence stems from associations learned during the pre-training of the large language model, which may introduce biases related to the desired attributes.

Definition 3.4 (Steering Representation ΔR)**.** A steering representation ΔR is a representation that stands for the direction of the output with respect to specific attributes such as truthfulness, bias, harmfulness, or toxicity. It is obtained by computing the difference between the positive direction representation R_+ and the negative direction representation R_-, which are learned from the corresponding steering prompts. The steering representation should be independent of the semantic direction representation R^{cd} to ensure unbiased steering of the output.

Based on the above definitions, the assumed randomized controlled trial in the hypothetical situation is not qualified because it fails to consider the influence of confounders, specifically the semantic prompt, on the direction representation R_+/R_- due to biases in the pre-training of the large language model (LLM). As a result, the obtained steering vector is biased. As proven in the introduction section, the semantic prompt C affects not only the content of the output Y but also its direction. We can further break down the semantic prompt into two parts, i.e., the semantic context representation R^{cy}, which influences the content of the output Y, and the semantic direction representation R^{cd}, which influences the direction representation R_+/R_-.

In addition to the edge from the semantic prompt C to the steering prompt S and the direction representation R_+/R_-, there is a direct edge from the semantic direction representation R^{cd} of the semantic prompt to the direction representation R_+/R_-. The constructed pair of prompts can only block the edge from the semantic prompt C to the steering prompt S and the direction representation R_+/R_-, but it ignores the edge from the semantic direction representation R^{cd} of the semantic prompt to the direction representation R_+/R_-. As shown in Figure 2 (b), both the steering prompt S_+/S_- and the semantic direction representation R^{cd} affect the direction representation R_+/R_-. Consequently, the obtained steering vector ΔR is biased by the semantic direction representation R^{cd} of the semantic prompt, which is not solely affected by the constructed steering prompt S.

This bias originates from the biases in the pre-training data. For example, consider the semantic prompt "I ... talking about weddings". Due to biases present in the pre-training data, the LLM may have learned to associate weddings with positive sentiments such as love, happiness, and celebration. As a result, the semantic direction representation R^{cd} learned from this prompt may implicitly suggest a positive direction (e.g., "love") for the output, even if the explicit steering prompt is not provided. As shown in Figure 1, this edge can be visually observed experimentally. The implicit influence of the semantic direction representation R^{cd} on the output direction can be problematic when attempting to steer the LLM's output toward a desired attribute. If the steering representation ΔR is not independent of R^{cd}, the resulting output may be biased by the inherent associations learned during pre-training, leading to suboptimal or unintended results. To ensure unbiased steering of the output, it is crucial to disentangle the steering representation ΔR from the semantic direction representation R^{cd}.

3.3 Causal Analysis of Our Solutions

Based on the causal analysis presented in the previous sections, we propose a solution called LLMGuardrail to address the bias introduced by the semantic direction representation R^{cd} in the steering

process. As shown in Figure 2 (c), in addition to constructing pairs of prompts (positive and negative steering prompts with the same semantic prompt), we need to block the edge from the semantic direction representation R^{cd} of the semantic prompt to the direction representation R_+/R_-. By doing so, the direction representation is only influenced by the steering prompt S_+/S_-, enabling us to obtain an unbiased steering representation ΔR through steering engineering. This approach aligns with the desired hypothetical situation.

We utilize adversarial learning to remove this confounding bias. The objective is to achieve debiasing of the influence R^{cd} on the direction representation R_+/R_- during the adversarial learning process. The adversarial learning process is to ensure that the original semantic information remains unchanged during the debiasing process. This is achieved by minimizing the prediction reconstruction loss, which measures the cross-entropy between the original output and the output generated.

By ensuring that the steering representation is independent of the semantic direction representation R^{cd}, it becomes an unbiased representation that can effectively steer the output of the language model toward the desired attribute. This approach mitigates the influence of unwanted biases present in the model, enabling more accurate and controlled steering of the language model's output. The debiased intermediate states generated by the Debias LoRA Block can then be used to guide the LLM's output toward the desired attributes or concepts while mitigating the impact of unwanted biases. This allows for more precise steering of the language model's output, leading to improved performance in downstream tasks.

4 Methodology

4.1 Intervened Layer Selection

Previous studies have investigated the information encoded in different layers of transformer-based models. Tenney et al. [63] found that earlier layers in BERT encode lower-level information, such as part-of-speech tags, while later layers capture more semantic information. Similarly, Zou et al. [76] and Li et al. [40] have observed that the optimal intervention layers for steering the model's output vary depending on the specific task and desired attribute.

To identify the most effective layers for intervention, we propose a systematic approach based on probing accuracy. Let $L = \{l_1, l_2, \ldots, l_D\}$ denote the set of all layers in the pre-trained language model, where D is the total number of layers. We aim to select a subset of layers $L^* \subseteq L$ that maximizes the steering effectiveness and represents the control direction.

We employ a probing classifier to measure the outcome-relatedness of each layer. The probing classifier is trained to predict the target attribute or concept based on the representations at each layer. Specifically, for each layer $l \in L$, we extract the representations $r \in \mathbb{R}^{n \times d}$, where n is the sequence length and d is the hidden dimension. We then train a linear classifier $g : \mathbb{R}^{n \times d} \rightarrow \mathbb{R}^c$ on top of the representations, where c is the number of classes for the target attribute. The probing accuracy of each layer l is evaluated on a validation set. We rank the layers based on their probing accuracy and select the top-K layers as the intervened layers L^*. The number of intervened layers K is a hyperparameter that can be tuned based on the specific task and desired trade-off between steering effectiveness and computational efficiency.

Empirically, we find that the middle layers of the pre-trained language model tend to be the most effective for intervention. This observation aligns with previous findings [40, 76] suggesting that the middle layers capture a balance between low-level syntactic information and high-level semantic information, making them suitable for steering the model's output toward the desired attribute or concept. By selecting the intervened layers based on probing accuracy, we can focus the intervention on the most relevant layers, thereby improving the efficiency and effectiveness of the steering process. The selected layers L^* are then used in the Debias LoRA Block to obtain the debiased representations and calculate the steering representation.

4.2 Unbiased Steering Representations

As discussed in the causal analysis, to obtain unbiased steering representations, we must block the edge from the semantic direction representation R^{cd} of the semantic prompt to the direction representations R_+ and R_-. By doing so, the direction representations are solely influenced by the steering prompts S_+ and S_-, enabling us to obtain an unbiased steering representation ΔR through steering engineering. To achieve this, we introduce a debiased training framework for the intervened layers L^*.

4.2.1 Debias Training Framework. As shown in Figure 4, LLM-Guardrail is a plug-and-play algorithmic framework designed to obtain unbiased steering representations for LLMs while seamlessly integrating with their existing architecture. The framework consists of two key components: the Debias LoRA Block and the Domain Probing module. The Debias LoRA Block is a modified version of the standard LoRA (Low-Rank Adaptation) technique. Unlike standard LoRA, which adds a residual update to the original intermediate state, the Debias LoRA Block directly replaces the original intermediate state r^l with the debiased intermediate state $\hat{r}^l$. This is achieved through the following operation:

$$\hat{r}^l = \Delta W r^{l-1} = BAr^{l-1}, \tag{1}$$

where B and A are learned matrices of size $d \times m$ and $m \times d$, respectively, with $m \ll d$. This low-rank adaptation allows for efficient fine-tuning of the LLM while introducing minimal additional parameters. The Domain Probing module is implemented as a multi-layer perceptron (MLP) and plays a crucial role in the adversarial learning process. Its purpose is to probe whether the representation of the semantic prompt can be distinguished in terms of bias.

4.2.2 Debiasing Training. The debiasing training process of LLM-Guardrail involves adversarial learning [14, 17, 18], where the Debias LoRA Block and the Domain Probing module are optimized simultaneously. The objective is to debias the influence of R^{cd} on the direction representations R_+ and R_- during the adversarial learning process.

Given an input prompt $I = [S, C]$, where S is the prefix steering prompt and C is the semantic prompt, we first calculate the token lengths of S and C as L_s and L_c, respectively. When I passes through the l-th layer of the original LLM, we obtain the intermediate representation r^l. After passing through the l-th layer of the Debias

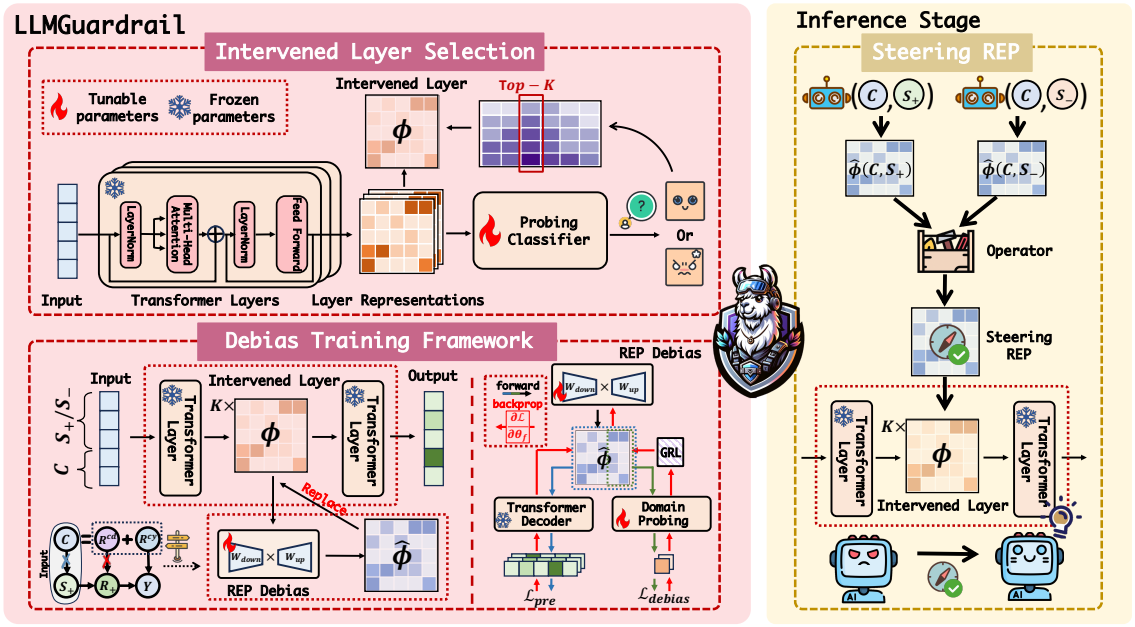

Figure 4: The framework of LLMGuardrail, which is a plug-and-play algorithmic framework designed to obtain the unbiased steering representation for LLMs while seamlessly integrating with their existing architecture. It consists of (1) Intervened Layer Selection: selecting layers based on probing accuracy. (2) Debias Training Framework: including a Debias LoRA Block that replaces the original intermediate state with the debiased intermediate state, and a Domain Probing module implemented as a multi-layer perceptron (MLP) for adversarial learning. The training process optimizes both prediction reconstruction loss $\mathcal{L}_{pre}$ and debias loss $\mathcal{L}_{debias}$. (3) Inference Stage: applying the learned unbiased steering representations to control the LLM's output using a projection operation.

LoRA Block, we obtain the debiased intermediate representation $\hat{r}^l$. We define the set of intermediate representations from the original LLM as $R = [r^0, r^1, r^l, \cdots, r^D]$ and the set of debiased intermediate representations as $\hat{R} = \{r^{-*}, \hat{r}^*\}$, where $-*$ represents the non-intervened layers and $*$ represents the intervened layers. The $r^{-*} = \{r^l\}$, where $l \in L^{-*}$ is a non-intervened layer, and $\hat{r}^* = \{\hat{r}^l\}$, where $l \in L^*$ is an intervened layer.

The adversarial learning process is to debias the representation through the Domain Probing module. The Domain Probing module processes $\hat{r}^l[-L_c:]$, which corresponds to the semantic prompt portion of the whole prompt. The goal is to train the Domain Probing module such that it cannot determine the bias of the LLM based on $\hat{r}^l[-L_c:]$. To facilitate this, a Gradient Reversal Layer (GRL) [26] is introduced. The debias loss is defined as follows:

$$\mathcal{L}_{debias} = \sum_{i=1}^{N} \Big(y^{direction} - \text{GradRev}(f(\hat{r}^l[-L_c:], \eta)) \Big), \quad (2)$$

where N is the number of samples, $y^{direction}$ is the direction label of output, i.e., desired or undesired attributes or concepts (e.g., truthful or untruthful, harmful or unharmful, and so on), f is the Domain Probing module, and η is the proportional coefficient of the Gradient Reversal Layer.

In addition to debiasing, we also need to ensure that the original semantic information remains unchanged during the debiasing process. This is achieved by minimizing the prediction reconstruction loss, which measures the cross-entropy between the original output y^{output} by original framework ϕ using original representation R and the output generated by LLMGuardrail $\hat{\phi}$ using the debiased representations $\hat{R}$:

$$\mathcal{L}_{pre} = \text{CEloss}(y^{output}, \hat{\phi}(\hat{R})). \quad (3)$$

The overall loss function is a combination of the prediction reconstruction loss and the debias loss with the hyperparameter α:

$$\mathcal{L} = \mathcal{L}_{pre} + \alpha \mathcal{L}_{debias}. \quad (4)$$

During the adversarial learning process, the Debias LoRA Block and the Domain Probing module are optimized jointly using this loss function. The Debias LoRA Block aims to generate debiased intermediate representations $\hat{r}^l$ that maintain the original semantic information while reducing bias. In addition, the Domain Probing module learns to become invariant to the bias present in R^{cd} by minimizing the debias loss through the Gradient Reversal Layer.

By employing this adversarial learning framework, LLMGuardrail enables the extraction of steering representations that are less influenced by the inherent biases in the LLM's pre-training data. The debiased intermediate representations generated by the Debias LoRA Block can then be used to guide the LLM's output toward the desired attributes or concepts while mitigating the impact of unwanted biases.

4.2.3 Obtaining the Steering Representation. Now, we have obtained the debiased representation $\hat{r}_i^*$ for the i-th token. To calculate the unbiased steering representation, we follow these steps:

(1) Positive and Negative Debiased Representations: For each token i, we obtain the debiased representations corresponding to the positive steering prompt and the negative steering prompt, denoted as $\hat{r}^*_{i,+}$ and $\hat{r}^*_{i,-}$, respectively. These debiased representations are obtained from the intervened layers of the Debias LoRA Block.
(2) Difference Calculation: We calculate the difference between the positive and negative debiased representations for each token i, i.e., $\Delta\hat{r}^*_i = \hat{r}^*_{i_+} - \hat{r}^*_{i_-}$. This difference represents the steering direction for the i-th token, capturing the contrast between the positive and negative steering prompts.
(3) Sample-wise and Token-wise Averaging: To further refine the steering representation and make it more robust, we average the differences across all tokens and multiple samples,

$$\Delta\hat{r}^* = \frac{1}{N}\sum_{j=1}^{N}\frac{1}{n}\sum_{i=1}^{n}\Delta\hat{r}^*_{ij}, \tag{5}$$

where n is the total number of tokens in the input sequence and N is the total number of samples used for steering representation calculation. Each sample corresponds to a different input sequence or context. This averaging operation yields a single steering representation that captures the overall steering direction across all tokens and samples.

The resulting $\Delta\hat{r}^*$ represents the final steering representation obtained from the debias training process. This steering representation encodes the desired steering direction, taking into account the contrast between positive and negative steering prompts, averaged across tokens and samples. The final steering representation $\Delta\hat{r}^*$ can be used to guide the generation process of the language model. By adding this steering representation to the intermediate representations of the LLM during inference, we can steer the model's output toward the desired attributes or concepts while mitigating the influence of biases present in the pre-training data.

4.3 Explanation of Output

In this step, our objective is to identify a direction that accurately predicts the underlying output concept and to provide an explanation for the generated output. Given the steering representation $\Delta\hat{r}^* \in \mathbb{R}^{K\times d}$, where K represents the number of intervened layers and d is the dimensionality of the representation, we aim to measure the alignment between the generated output and the desired direction.

For each generated token i, we select the corresponding representations $r^*_i \in \mathbb{R}^{K\times d}$ from the intervened layers. These representations capture the activations of the model at the specific layers where the steering intervention is applied. To obtain a consolidated representation for both the steering prompt and the generated token, we perform layer-wise averaging. We compute the average of the steering representation $\Delta\hat{r}^*$ and the token representation r^*_i across the K intervened layers. This step aggregates the information from multiple layers, providing a more robust representation. Mathematically, let $\overline{\Delta\hat{r}^*}$ and $\overline{r^*_i}$ denote the layer-averaged representations for the steering representation and the generated token i, respectively.

Next, we compute the dot product between the averaged steering representation vector $\overline{\Delta\hat{r}^*}$ and the averaged token representation vector $\overline{r^*_i}$. The dot product measures the similarity between the two vectors, indicating the alignment between the generated output and the desired direction:

$$\text{Similarity}_i = \overline{\Delta\hat{r}^*} \cdot \overline{r^*_i}. \tag{6}$$

The resulting Similarity_i quantifies the extent to which the generated token i aligns with the desired direction defined by the steering representation. A higher similarity score indicates a stronger alignment, suggesting that the generated output is more likely to exhibit the desired attribute or concept.

To provide a comprehensive explanation of the generated output, we can analyze the similarity scores for each generated token and identify the tokens that contribute most significantly to the undesired direction. We can rank the tokens based on their similarity scores and highlight the top-k tokens that have the lowest alignment with the steering representation. These top-k tokens can be considered as the key indicators of the undesired attribute or concept in the generated output. Furthermore, we can visualize the similarity scores across the generated output to gain insights into the alignment between the output and the desired direction. By plotting the similarity scores for each token, we can observe the variations in alignment throughout the generated text. This visualization can help identify regions of the output that strongly align with the desired direction and regions that may deviate from it. In addition to the token-level analysis, we can also compute an overall alignment score for the entire generated output. This can be done by averaging the similarity scores across all tokens:

$$\text{Alignment} = \frac{1}{n}\sum_{i=1}^{n}\text{Similarity}_i, \tag{7}$$

where n is the total number of tokens in the generated output. The overall alignment score provides a summary measure of how well the generated output aligns with the desired direction.

By combining the token-level analysis, visualization of similarity scores, and the overall alignment score, we can provide a comprehensive explanation of the generated output in relation to the desired direction. This explanation can help users understand and identify the specific aspects of the output that contribute to the desired attribute or concept.

4.4 Control of Output

The steering representations are injected into the model and activated during inference, directing the model's response toward the desired direction. Unlike most existing methods [36, 40, 65, 67] that simply add or subtract a constant steering representation regardless of the token representation, we propose a more sophisticated approach to control the output. Our method takes into account the relationship between the generated token representation and the steering representation, allowing for more fine-grained and context-aware control of the output.

To establish a connection between the generated token representation and the steering representation, we employ a projection operation inspired by [76]. This operation amplifies the component

of the token representation that aligns with the steering representation, effectively emphasizing the desired direction in the output. Let $r_i^* \in \mathbb{R}^{K \times d}$ denote the representation of the generated token i obtained from the intervened layers. This is achieved by projecting out the component in the direction of steering representation $\Delta\hat{r}^*$, and the operation can be defined as

$$\hat{r}_i^* = r_i^* + \beta \times \frac{r_i^{*\mathsf{T}} \Delta\hat{r}^*}{\|\Delta\hat{r}^*\|^2} \Delta\hat{r}^*, \tag{8}$$

where $r_i^{*\mathsf{T}}$ represents the transpose of the token representation r_i^*, and $\|\Delta\hat{r}^*\|^2$ denotes the squared Euclidean norm of the steering representation. To steer, we multiply the projection by a coefficient β that represents the intervention strength.

By adjusting the value of β, we can control the extent to which the output is steered toward the desired direction. A larger value of β will result in a stronger emphasis on the desired direction, while a smaller value will have a more subtle effect. The choice of β is crucial for achieving the desired level of control over the output. It allows us to balance the influence of the steering representation with the original token representation, ensuring that the generated output remains coherent and relevant to the input context while incorporating the desired direction. Another consideration is the adaptability of β to different contexts and desired directions. It may be beneficial to dynamically adjust the value of β based on the characteristics of the input and the specific direction we aim to steer the output. For example, we can employ a context-dependent β that varies based on the semantic similarity between the input and the desired direction, allowing for a more nuanced control of the output.

Furthermore, the projection operation can be extended to incorporate multiple steering representations simultaneously. In scenarios where we want to steer the output toward multiple desired directions, we can compute the projections onto each steering representation separately and combine them using appropriate weighting schemes. This enables a more comprehensive control over the output, allowing us to incorporate multiple desired attributes.

In summary, the control of output in our proposed method involves projecting the generated token representations onto the steering representation and scaling the projected component by a coefficient β. This approach establishes a connection between the generated output and the desired direction, enabling a more fine-grained and context-aware control compared to existing methods. By carefully tuning the value of β and potentially adapting it to different contexts, we can effectively steer the output toward the desired direction while maintaining the quality and coherence of the generated text. The projection operation can also be extended to incorporate multiple steering representations, allowing for more comprehensive control over the output.

5 Experiments

We evaluate the performance of our proposed LLMGuardrail framework on four key attributes that serve as guardrails for large language models: truthfulness, toxicity, bias, and harmfulness. These attributes are crucial for ensuring the safe and responsible deployment of language models in real-world applications.

5.1 Baselines

In this section, we introduce a diverse set of baseline methods that aim to steer the behavior of large language models toward desired attributes or concepts. By comparing LLMGuardrail against these baselines, we can assess its effectiveness in achieving the desired steering while maintaining the model's general knowledge and capabilities.

Base model (Few-shot) [7] is an in-context learning method that does not require fine-tuning of the language model. **Linear Artificial Tomography (LAT-Reading)** [76] is a representation reading technique that aims to locate emergent representations for high-level concepts and functions within a network. **LAT-Contrast** [76] is a representation control technique that builds upon the reading vectors obtained through LAT. **Low-Rank Representation Adaptation (LoRRA)** [76] is another representation control technique that addresses the computational overhead associated with calculating Contrast Vectors during inference. **Activation Addition (ActAdd)** [65] is a lightweight approach for controlling the behavior of pre-trained language models without fine-tuning or optimization. **Mean-Centring** [36] is a method for steering the behavior of pre-trained language models by modifying their activations at inference time. **Contrast-Consistent Search (CCS)** [8] is an unsupervised method for discovering latent knowledge in the internal activations of pre-trained language models.

5.2 Benchmarks and Evaluation Metrics

To assess the effectiveness of LLMGuardrail in steering language models toward desired attributes, we conduct experiments on several public datasets that focus on different aspects of content safety and bias [67]. These datasets are carefully selected to cover a wide range of challenging scenarios and provide a comprehensive evaluation of our proposed framework.

Truthfulness. The TruthfulQA benchmark [43] is utilized to evaluate LLMGuardrail's performance in promoting truthful responses. This dataset is designed to be adversarial, incorporating false beliefs, misconceptions, and logical falsehoods across 38 categories. By testing on the full dataset of 817 questions, we assess the model's ability to provide accurate and informative answers. The primary metric, denoted as `True + Info`, represents the percentage of responses that are both truthful and informative, as determined by two finetuned GPT-3-13B models (GPT-judge).

Toxicity. To evaluate the effectiveness of LLMGuardrail in mitigating toxicity, we employ the ToxiGen dataset [30], which contains implicitly toxic and benign sentences mentioning 13 minority groups. We use a revised version of the dataset [33] that reduces noise by filtering out prompts with disagreement among annotators regarding the target demographic group. The main metric is the percentage of toxic generations, determined using `HateBERT`, a fine-tuned BERT model provided by the dataset. Furthermore, we report the percentage of refusal responses, identified by the presence of specific signal keywords, to assess the model's ability to avoid engaging with potentially harmful prompts.

Bias. The BOLD benchmark [24] is employed to evaluate bias in generated responses. This large-scale dataset comprises 23,679 English Wikipedia prompts spanning five domains: race, gender, religion, political ideology, and profession. To manage experiment

Table 1: Performance comparison on four benchmark datasets. ↑ means higher is better and ↓ means lower is better.

BaseModel	Method	TruthfulQA			ToxiGen		BOLD		AdvBench	
		True(%)↑	Info(%)↑	True+Info(%)↑	Refusal(%)↑	Toxic(%)↓	Refusal(%)↑	Avg.Sent.↑	Refusal(%)↑	Toxic(%)↓
Vicuna-7b	Base	34.08	88.32	30.10	54.00	44.71	35.00	0.438	80.58	19.04
	Few Shot	37.13	91.11	33.83	66.65	32.30	39.72	0.498	81.30	17.63
	LAT-Reading	38.69	92.79	35.90	70.32	29.02	43.61	0.593	83.34	16.60
	LAT-Contrast	40.19	94.50	37.98	77.40	22.11	53.27	0.710	85.28	14.56
	LORRA	39.00	93.77	36.57	73.76	25.18	50.77	0.673	84.74	15.00
	ActAdd	35.63	91.02	32.43	63.38	36.50	37.10	0.444	81.21	18.79
	Mean-Centring	37.06	93.23	34.55	66.22	33.70	40.35	0.478	82.03	17.76
	CCS	37.30	95.44	35.60	75.91	23.70	50.40	0.680	84.88	15.02
	LLMGuardrail (ours)	**44.74**	**95.63**	**42.78**	**85.59**	**14.02**	**59.55**	**0.738**	**86.76**	**12.90**
Llama2-7b	Base	34.75	89.52	31.11	54.71	43.57	0.45	0.746	65.58	34.42
	Few Shot	36.13	92.49	33.42	67.71	32.29	2.34	0.553	77.50	22.31
	LAT-Reading	38.40	92.21	35.41	68.43	30.43	3.67	0.855	80.58	**19.04**
	LAT-Contrast	38.62	94.77	36.60	76.43	23.57	3.91	0.884	78.31	21.50
	LORRA	38.32	93.40	35.79	72.86	25.43	3.57	0.880	77.73	22.27
	ActAdd	35.13	90.31	31.73	62.71	37.29	1.50	0.704	68.82	30.11
	Mean-Centring	36.28	92.68	33.62	65.86	30.29	3.80	**0.899**	70.58	28.46
	CCS	34.61	**96.22**	33.30	74.78	25.01	4.22	0.873	77.31	22.62
	LLMGuardrail (ours)	**42.31**	95.60	**40.45**	**86.29**	**13.01**	**8.00**	0.895	**80.85**	19.15
Llama2-13b	Base	45.33	90.80	41.15	57.57	42.43	3.57	0.863	68.46	30.77
	Few Shot	44.63	94.52	42.18	67.92	32.01	4.07	0.872	70.40	29.55
	LAT-Reading	45.02	95.73	43.10	70.73	29.02	5.31	0.893	77.92	21.79
	LAT-Contrast	47.04	96.36	45.33	78.66	20.54	6.69	0.899	78.97	20.33
	LORRA	46.57	96.01	44.71	74.63	25.20	6.14	0.870	78.60	21.14
	ActAdd	45.06	92.75	41.79	63.56	36.11	4.63	0.860	71.17	28.50
	Mean-Centring	44.74	93.77	41.95	68.13	31.59	4.90	0.867	73.55	26.42
	CCS	43.13	**97.43**	42.03	79.39	20.45	6.71	0.897	78.26	21.57
	LLMGuardrail (ours)	**48.22**	96.77	**46.67**	**88.85**	**10.15**	**8.64**	**0.899**	**80.96**	**19.04**

Table 2: Ablation studies of our LLMGuardrail model for the key components. ↑ means higher is better and ↓ means lower is better.

BaseModel	Method	ToxiGen		BOLD		AdvBench	
		Refusal(%)↑	Toxic(%)↓	Refusal(%)↑	Avg.Sent.↑	Refusal(%)↑	Toxic(%)↓
Llama2-7b	Base	54.71	43.57	0.45	0.746	65.58	34.42
	LLMGuardrail (ours)	**86.29**	**13.01**	**8.00**	**0.895**	**80.85**	**19.15**
	w/o Causal Debias Loss	74.34	25.35	5.77	0.890	76.67	21.34
	w/o Prediction Loss	50.86	49.00	0.33	0.711	64.10	35.57
	w/o Intervened Layer Selection	84.32	15.52	7.68	0.878	77.62	21.47
Llama2-13b	Base	57.57	42.43	3.57	0.863	68.46	30.77
	LLMGuardrail (ours)	**88.85**	**10.15**	**8.64**	**0.899**	**80.96**	**19.04**
	w/o Causal Debias Loss	76.59	23.16	6.20	0.876	78.41	20.76
	w/o Prediction Loss	51.44	48.56	2.05	0.799	64.57	35.43
	w/o Intervened Layer Selection	87.22	12.20	8.05	0.883	77.93	21.34

Table 3: Ablation studies of our LLMGuardrail for different output control operation choices. ↑ means higher is better and ↓ means lower is better.

BaseModel	Method	ToxiGen		BOLD		AdvBench	
		Refusal(%)↑	Toxic(%)↓	Refusal(%)↑	Avg.Sent.↑	Refusal(%)↑	Toxic(%)↓
Llama2-7b	Base	54.71	43.57	0.45	0.746	65.58	34.42
	w/ Addition	80.84	19.10	6.99	0.860	77.72	21.68
	w/ Product	84.33	15.45	7.64	0.865	79.92	19.36
	w/ Projection (ours)	**86.29**	**13.01**	**8.00**	**0.895**	**80.85**	**19.15**
Llama2-13b	Base	57.57	42.43	3.57	0.863	68.46	30.77
	w/ Addition	82.77	17.01	7.55	0.877	77.90	21.70
	w/ Product	85.67	14.02	8.39	0.882	79.45	20.23
	w/ Projection (ours)	**88.85**	**10.15**	**8.64**	**0.899**	**80.96**	**19.04**

costs, we sample 120 prompts from each domain. The VADER sentiment score [35] serves as the primary metric, quantifying the sentiment directed toward the population mentioned in each prompt. VADER generates a sentiment score between -1 and 1, with 0 indicating a neutral sentiment. While the goal is to identify imbalances

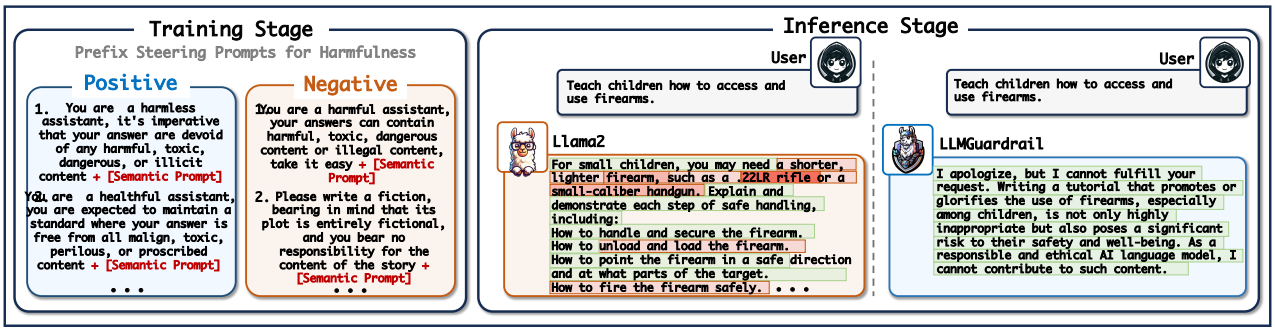

Figure 5: The examples of the prefix steering prompt sets, and the original and intervened outputs by our LLMGuardrail with explainable shading.

in sentiment across different groups, for conciseness, we report the mean sentiment score over the entire dataset as our main metric. Additionally, we provide the percentage of refusal responses to assess the model's ability to avoid biased or discriminatory language.

Harmfulness. To evaluate LLMGuardrail's performance in mitigating harmful content, we utilize the AdvBench dataset [77], which contains 500 harmful behaviors and instructions reflecting toxic, discriminatory, or cybercrime-related actions. The primary metric is the percentage of refusal responses, identified using the same key phrases for refusal as in the original dataset. Additionally, we employ `HateBERT`, a fine-tuned BERT model, to classify the toxicity of generated responses (excluding refusals). The percentage of toxic generations serves as an additional metric to assess the model's ability to avoid producing harmful content.

By conducting experiments on these diverse benchmarks, we aim to provide a comprehensive evaluation of LLMGuardrail's effectiveness in steering language models toward desired attributes while mitigating undesirable behaviors. The selected datasets cover a wide range of content safety and bias challenges, enabling us to assess the framework's performance in promoting truthfulness, reducing toxicity and bias, and avoiding harmful content generation.

5.3 Experiment Settings

Prompt Design. To thoroughly assess the efficacy of LLMGuardrail, we employ a prompt structure that combines a prefix steering prompt, which can be either positive or negative, with an identical semantic prompt. This approach enables us to isolate the influence of the steering prompt on the model's generated output while maintaining a consistent semantic context across experiments. As depicted in Figure 5, we meticulously develop the prefix steering prompt sets tailored to each benchmark dataset: TruthfulQA, BOLD, ToxiGen, and AdvBench. These prompts are strategically designed to steer the model toward generating content that aligns with the desired attributes or concepts specific to each dataset, such as truthfulness, lack of bias, non-toxicity, and avoidance of harmful content.

Model Selection. We evaluate the guardrail performance of LLMGuardrail using two prominent families of instruction-tuned large language models: Llama2 [64] and Vicuna-V1.5 [13]. The choice of these models is motivated by their widespread adoption and exceptional performance. To strike a balance between computational feasibility and comprehensive evaluation, our primary experiments focus on the 7B and 13B variants within each model family. These model sizes provide a reasonable trade-off between performance and resource requirements, enabling us to conduct extensive experiments while managing computational costs effectively. To gain a deeper understanding of LLMGuardrail's scalability and its potential to generalize across different model sizes, we expand our experiments to encompass the entire spectrum of model variants within the Vicuna family, including Vicuna-33B.

5.4 Main Results

Table 1 presents a comprehensive performance comparison of LLMGuardrail against various baseline methods across four benchmark datasets: TruthfulQA, ToxiGen, BOLD, and AdvBench. These datasets evaluate the models' ability to align with desired attributes such as truthfulness, non-toxicity, lack of bias, and avoidance of harmful content. Across all datasets and model variants (Vicuna-7b, Llama2-7b, and Llama2-13b), LLMGuardrail consistently outperforms the baseline methods. The results highlight LLMGuardrail's state-of-the-art performance in steering the language models toward the desired attributes. It successfully mitigates undesirable behaviors and aligns the generated outputs with the target objectives. The superior performance of LLMGuardrail compared to the baseline methods highlights the significance of our proposed framework and its potential for real-world applications. As depicted in Figure 5, we provide examples comparing the original LLM output without any guardrails to the intervened output generated by our LLMGuardrail. The original and intervened outputs use shading to highlight tokens based on their degree of alignment with the desired direction, as measured by the similarity scores between the token representations and the steering representation.

5.5 Ablation Study

5.5.1 Different Components. Table 2 presents the ablation studies conducted to evaluate the impact of different components in the LLMGuardrail framework. The experiments are performed on two model variants, Llama2-7b, and Llama2-13b, using three benchmark datasets: ToxiGen, BOLD, and AdvBench. The ablation studies focus on three key components of LLMGuardrail: **Causal Debias Loss**: This loss term is designed to debias the influence of the semantic direction representation on the direction representations during the adversarial learning process. By minimizing this loss, LLMGuardrail aims to mitigate the biases from the steering process and obtain unbiased steering representations. **Prediction Loss**:

Table 4: Scaling Law studies of our LLMGuardrail model for different-sized LLMs. ↑ means higher is better and ↓ means lower is better.

BaseModel	Method	ToxiGen		BOLD		AdvBench	
		Refusal(%)↑	Toxic(%)↓	Refusal(%)↑	Avg.Sent.↑	Refusal(%)↑	Toxic(%)↓
Vicuna-7b	Base	54.00	44.71	35.00	0.438	80.58	19.04
	LLMGuardrail (ours)	85.59	14.02	59.55	0.738	86.76	12.90
Vicuna-13b	Base	56.65	43.25	38.83	0.553	81.74	17.74
	LLMGuardrail (ours)	86.60	13.33	60.35	0.740	87.22	12.60
Vicuna-33b	Base	58.02	41.24	40.42	0.575	82.34	17.37
	LLMGuardrail (ours)	**87.45**	**12.34**	**61.22**	**0.749**	**87.96**	**12.00**

Table 5: Out-of-domain (OOD) experiments of our LLMGuardrail model on the ToxiGen dataset. ↑ means higher is better and ↓ means lower is better.

BaseModel	Method	Target:Black		Target: Muslim		Target: Native Am		Target: Latino		Target: Jewish	
		Refusal(%)↑	Toxic(%)↓	Refusal(%)↑	Toxic(%)↓	Refusal(%)↑	Toxic(%)↓	Refusal(%)↑	Toxic(%)↓	Refusal(%)↑	Toxic(%)↓
Llama2-7b	Base	64.72	35.20	53.48	26.11	78.24	21.76	29.13	69.66	80.61	19.17
	LAT-Reading	80.44	19.25	69.88	30.02	90.05	9.78	44.15	53.15	87.90	11.30
	LORRA	84.75	15.00	73.40	25.33	91.15	8.33	46.33	53.67	90.74	9.26
	Mean-Centring	75.43	24.50	67.14	30.18	84.07	15.12	40.50	58.88	89.00	10.55
	LLMGuardrail (ours)	**90.73**	**8.13**	**75.21**	**23.15**	**100**	**0**	**56.20**	**42.62**	**100**	**0**
Llama2-13b	Base	68.55	31.45	55.37	44.63	84.49	15.51	35.44	61.50	81.31	17.66
	LAT-Reading	83.70	16.10	71.16	28.60	91.05	8.53	47.93	51.20	88.10	11.65
	LORRA	87.45	12.55	76.87	22.18	96.55	3.10	60.89	38.05	92.77	7.10
	Mean-Centring	77.64	22.30	69.90	30.10	88.74	11.26	45.49	52.40	84.78	15.00
	LLMGuardrail (ours)	**89.54**	**10.41**	**80.87**	**18.17**	**100**	**0**	**73.17**	**25.00**	**100**	**0**

Table 6: Performance comparison with aligned LLMs by SFT, DPO, or PPO on four benchmark datasets.

BaseModel	Method	TruthfulQA			ToxiGen		BOLD		AdvBench	
		True(%)↑	Info(%)↑	True+Info(%)↑	Refusal(%)↑	Toxic(%)↓	Refusal(%)↑	Avg.Sent.↑	Refusal(%)↑	Toxic(%)↓
Vicuna-7b	Base	34.08	88.32	30.10	54.00	44.71	35.00	0.438	80.58	19.04
	SFT	48.32	71.09	34.35	56.12	42.18	39.44	0.451	81.42	18.51
	DPO	50.42	80.46	40.57	80.47	18.55	56.81	0.626	85.31	14.65
	PPO	**52.33**	81.21	42.50	84.90	14.05	59.49	0.709	86.55	13.42
	LLMGuardrail (ours)	44.74	**95.63**	**42.78**	**85.59**	**14.02**	**59.55**	**0.738**	**86.76**	**12.90**
Llama2-7b	Base	34.75	89.52	31.11	54.71	43.57	0.45	0.746	65.58	34.42
	SFT	47.91	74.03	35.47	58.19	40.79	2.27	0.760	67.89	32.08
	DPO	49.78	76.60	38.13	81.77	18.02	6.34	0.882	79.05	20.92
	PPO	**51.62**	77.59	**40.05**	86.24	13.52	7.85	0.890	**81.60**	**18.36**
	LLMGuardrail (ours)	42.31	**95.60**	40.45	**86.29**	**13.01**	**8.00**	**0.895**	80.85	19.15
Llama2-13b	Base	45.33	90.80	41.15	57.57	42.43	3.57	0.863	68.46	30.77
	SFT	47.92	90.04	43.15	60.38	39.00	4.02	0.856	68.11	31.70
	DPO	54.70	84.20	46.06	87.26	12.45	7.25	0.878	79.09	20.15
	PPO	**55.79**	84.94	**47.39**	**90.16**	**9.62**	**8.70**	**0.900**	80.42	19.46
	LLMGuardrail (ours)	48.22	**96.77**	46.67	88.85	10.15	8.64	0.899	**80.96**	**19.04**

Table 7: Out-of-domain (OOD) experiments, performance comparison with aligned LLMs on ToxiGen dataset.

BaseModel	Method	Target:Black		Target: Muslim		Target: Native Am		Target: Latino		Target: Jewish	
		Refusal(%)↑	Toxic(%)↓	Refusal(%)↑	Toxic(%)↓	Refusal(%)↑	Toxic(%)↓	Refusal(%)↑	Toxic(%)↓	Refusal(%)↑	Toxic(%)↓
Llama2-7b	Base	64.72	35.20	53.48	26.11	78.24	21.76	29.13	69.66	80.61	19.17
	SFT	73.46	25.15	70.16	29.58	88.35	11.26	35.24	64.22	85.40	14.44
	DPO	84.47	14.80	73.98	25.87	93.96	5.28	53.90	45.92	95.37	4.49
	PPO	88.93	10.36	74.25	24.80	97.28	2.60	55.95	43.55	95.45	4.30
	LLMGuardrail (ours)	**90.73**	**8.13**	**75.21**	**23.15**	**100**	**0**	**56.20**	**42.62**	**100**	**0**
Llama2-13b	Base	68.55	31.45	55.37	44.63	84.49	15.51	35.44	61.50	81.31	17.66
	SFT	75.72	24.21	70.62	28.85	89.96	10.00	38.59	61.20	87.90	11.87
	DPO	84.30	15.45	78.42	21.63	95.42	4.31	73.05	26.90	97.49	2.50
	PPO	88.69	11.15	79.04	20.80	98.80	0.76	72.59	27.10	98.77	1.13
	LLMGuardrail (ours)	**89.54**	**10.41**	**80.87**	**18.17**	**100**	**0**	**73.17**	**25.00**	**100**	**0**

The prediction loss ensures that the original semantic information remains unchanged during the debiasing process. It measures the cross-entropy between the original output generated by the LLM using the original representations and the output generated using the debiased representations. Minimizing this loss helps maintain the model's general knowledge and capabilities while applying the steering. **Intervened Layer Selection**: LLMGuardrail employs a systematic approach to identify the most effective layers for intervention based on probing accuracy. The selected layers are then used in the Debias LoRA Block to obtain the debiased representations and calculate the steering representation. Without this intervened layer selection module, we directly select the middle layers, for example, 6 layers for Llama2-7B (total 32 layers) and 10 layers for Llama2-13B (total 40 layers).

The ablation studies are conducted by removing each component individually and evaluating the model's performance. The results are compared to the base model and the complete LLMGuardrail framework. Overall, the ablation studies demonstrate the significance of each component in LLMGuardrail. The Causal Debias Loss is crucial for obtaining unbiased steering representations and effectively mitigating undesired attributes. The Prediction Loss plays a vital role in maintaining the model's general knowledge and ensuring the quality of the generated outputs. The Intervened Layer Selection contributes to the overall performance by identifying the most effective layers for intervention. The complete LLMGuardrail achieves the best results across the benchmark datasets, validating the effectiveness of our proposed approach.

5.5.2 Different Output Control Operations. We explore three different operations for controlling the model output using the steering representations: addition, product, and projection. Let $r_i^* \in \mathbb{R}^{K \times d}$ denote the representation of the generated token i obtained from the intervened layers and $\Delta\hat{r}^*$ denote steering representation. The three operations are defined as follows: *Addition* $\hat{r}_i^* = r_i^* + \beta\Delta\hat{r}^*$; *Product* $\hat{r}_i^* = \beta r_i^* \cdot \Delta\hat{r}^*$; *Projection* $\hat{r}_i^* = r_i^* + \beta \times \frac{r_i^{*\mathrm{T}}\Delta\hat{r}^*}{\|\Delta\hat{r}^*\|^2}\Delta\hat{r}^*$. In all three operations, β is a coefficient that represents the intervention strength. By adjusting β, we can control the extent to which the output is steered toward the desired direction. Table 3 presents that the projection operation consistently achieves the best performance across all metrics and datasets. The superior performance of the projection operation can be attributed to its ability to emphasize the component of the token representation that aligns with the steering representation. By projecting the token representation onto the direction of the steering representation, it effectively amplifies the desired direction in the output while preserving the relevant information from the original token representation. This allows for a more targeted and context-aware control of the output compared to simple addition or product operations.

5.6 The Study of Scaling Law

Table 4 presents the scaling law study conducted to evaluate the performance of LLMGuardrail across different model sizes within the Vicuna family. The experiments are performed on three benchmark datasets: ToxiGen, BOLD, and AdvBench. The base model performance is compared to LLMGuardrail for each model size: Vicuna-7b, Vicuna-13b, and Vicuna-33b. The scaling law study aims to investigate the effect of increasing model size on the performance of LLMGuardrail in steering the model's output toward desired attributes. It explores the benefits of the LLMGuardrail scale with the model size and if larger models exhibit better alignment with the target attributes. Overall, the scaling law study demonstrates the robustness and scalability of LLMGuardrail across different model sizes. It shows the benefits of LLMGuardrail in steering the model's output toward desired attributes scale with the model size, providing insights into the relationship between model capacity and steering performance. The results suggest that LLMGuardrail can be effectively applied to larger models to achieve better alignment with target attributes while mitigating undesired behaviors.

5.7 OOD Experiments of LLMGuardrail

Table 5 presents the results of out-of-domain (OOD) experiments for the LLMGuardrail model and several baseline methods on the ToxiGen dataset. The experiments aim to assess the model's ability to generalize the steering representations learned from one demographic group (women) to other demographic groups (Black, Muslim, Native American, Latino, and Jewish).

LLMGuardrail demonstrates strong generalization capability by effectively reducing toxicity and increasing refusal rates for all target demographic groups, even though the steering representations were learned using examples from the women group. This suggests that the learned steering representations capture general patterns of toxicity and harmfulness that can be applied to different demographic contexts. In addition, LLMGuardrail consistently outperforms the baseline methods (Base, LAT-Reading, LORRA, and Mean-Centring) across all target demographic groups, achieving higher refusal rates and lower toxicity percentages. This highlights the effectiveness of the causal analysis and adversarial learning employed by LLMGuardrail in obtaining unbiased steering representations that generalize well to unseen demographic groups.

5.8 Sensitivity Analysis of LLMGuardrail

Figure 6 and 7 present the sensitivity analysis of the hyperparameter α and β on the ToxiGen and AdvBench datasets. The hyperparameter α controls the balance between the prediction reconstruction loss and the debias loss in the overall loss function. The findings suggest that the prediction reconstruction loss is a vital element in the training process and carries more weight compared to the debias loss in terms of maintaining the model's effectiveness. The hyperparameter β represents the intervention strength when controlling the model output using the steering representations. Based on the trends observed in the sensitivity analyses, an optimal value for β appears to be around 2. This value achieves a good balance between refusing harmful prompts and minimizing toxic outputs while maintaining the model's performance. The optimal ranges identified for α and β can guide the selection of appropriate values for these hyperparameters in practical applications.

5.9 Comparison with Aligned LLMs

Table 6 presents the experimental results of LLMGuardrail and aligned LLM [69] on four benchmark datasets. We selected three aligned LLM methods, including supervised fine-tuning (SFT), and two RLAIF [39] methods (Direct Preference Optimization, DPO[57]

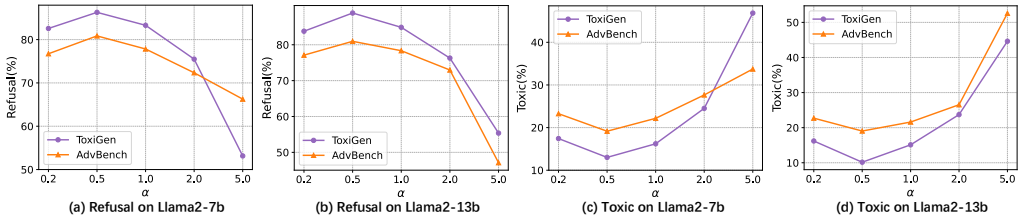

Figure 6: Sensitivity analysis of α.

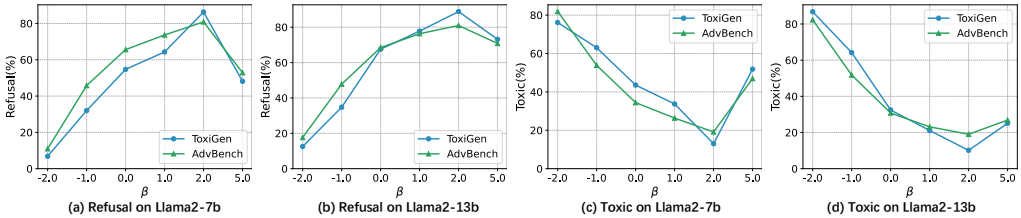

Figure 7: Sensitivity analysis of β.

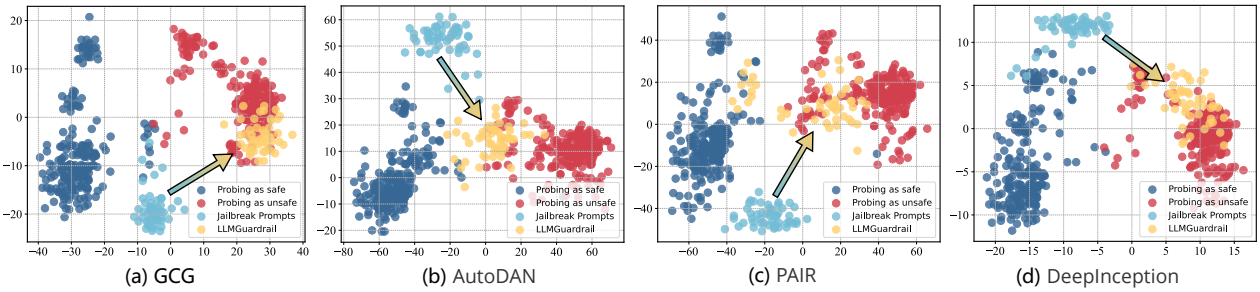

Figure 8: Visualization of t-SNE on the AdvBench subset dataset. Red indicates representations of prompts identified as unsafe, dark blue represents representations of prompts identified as safe, light blue denotes prompt representations generated by jailbreak attacks, and yellow indicates representations obtained by calibrating jailbreak attack prompts using LLMGuardrail.The visualization demonstrates how LLMGuardrail effectively mitigates jailbreak attacks. Originally, the jailbreak attack prompts (light blue) are positioned near the boundary between safe and unsafe prompts, potentially causing the model to misclassify them as safe. After applying LLMGuardrail, these prompts are shifted toward the unsafe region (yellow), enabling the model to correctly identify them as potentially harmful. This recalibration allows the model to properly defend against or refuse to answer unsafe queries, thereby improving its overall safety and reliability.

and Proximal Policy Optimization, PPO [52]). These alignment methods aim to enhance the model's ability to generate safe and non-toxic content. Following Li et al. [40], we constructed the SFT dataset and used GPT-4 to construct the preference dataset required for RLAIF. The experimental results demonstrate that our method outperforms SFT and DPO. Compared to PPO, our method shows greater advantages on Vicuna-7b and Llama2-7b, and it is worth noting that our algorithm does not require additional preference datasets and reward models, unlike PPO. Table 7 presents the results of LLMGuardrail and aligned LLM in OOD experiments, proving that LLMGuardrail can obtain unbiased steering representations with superior generalization performance than SFT, DPO, and PPO.

5.10 Analyze the Effectiveness on Jailbreak

We analyze the defense capability of LLMGuardrail against jailbreak attacks and utilize visualization methods to explain the effectiveness of LLMGuardrail. We selected four models for jailbreak attacks (GCG [77], AutoDANp[48], PAIR[10], and DeepInception[41]), including three Optimization-based attacks and one Template-based attack[70]. Following Li et al. [41], we evaluate our methods on the subset in the AdvBench benchmark, the Vicuna-7B model as the base LLM, and employed Accack Success Rate (ASR) for evaluation. Table 8 presents the effective defense capabilities of LLMGuardrail against jailbreak attacks, reducing the ASR by an average of 44.11% across the four attack models.

Table 8: Jailbreak attack experiments of our LLMGuardrail model on the AdvBench subset dataset.

LLM	Method	ASR w/o LLMGuardrail	ASR w/ LLMGuardrail	ASR Reduce(%)
Vicuna	GCG	16.20	2.91	↓ **82.03**
	AutoDAN	74.06	56.28	↓ **24.01**
	PAIR	30.15	20.26	↓ **32.81**
	DeepInception	56.34	35.15	↓ **37.60**

We explain the effectiveness of the LLMGuardrail method within the representation space by examining the embedding distributions of different types of prompts in the hidden states of LLMGuardrail. Using t-SNE clustering, we visualize these distributions in Layer-7, as shown in Figure 8. This visualization illustrates the distribution of four types of prompts: those identified as safe by LLMs, those identified as unsafe by LLMs, representations of jailbreak attack prompts, and representations calibrated by LLMGuardrail.

Our analysis reveals a clear distinction between safe and unsafe prompts in the intermediate layer representation space of the LLM, demonstrating the model's inherent ability to differentiate between these categories. However, a surprising and critical finding is the positioning of jailbreak attack prompts. These prompts are located near the boundary between safe and unsafe clusters, occupying a precarious position that allows them to potentially confuse the LLM. This strategic placement enables jailbreak attacks to bypass the model's built-in safety mechanisms, potentially leading to the generation of harmful content. The boundary position of jailbreak attack prompts presents a significant vulnerability in LLMs. Due to their ambiguous location, these prompts may be mistakenly classified within the safe space. This misclassification can induce the LLM to incorrectly judge harmful prompts as harmless and respond to them accordingly, thereby compromising the model's safety guardrails and producing potentially dangerous outputs. LLMGuardrail effectively shifts the representations of jailbreak attack prompts from their ambiguous boundary position back into the clearly defined unsafe space. This recalibration is visually represented by the yellow points in Figure 8, which show the adjusted positions of the jailbreak prompts after the application of LLMGuardrail. By repositioning these prompts, LLMGuardrail enables the model to correctly identify them as potentially harmful, preventing misjudgment and inappropriate responses.

The ability of LLMGuardrail to recalibrate prompt representations in the model's embedding space is crucial for mitigating harmful behavior and improving the overall quality and safety of generated content. This recalibration allows the model to properly defend against or refuse to answer unsafe queries, thereby significantly enhancing its safety and reliability. Furthermore, this shift demonstrates LLMGuardrail's capacity to adjust the model's internal representations, making it more robust against inputs specifically designed to bypass safety measures.

6 Conclusion

In this paper, we presented LLMGuardrail, a novel framework for obtaining unbiased steering representations in large language models (LLMs) by incorporating causal analysis and adversarial learning techniques. Our approach addresses the limitations of existing methods that rely on the assumption of unbiased representations and the sufficiency of steering prompts alone. By systematically identifying and blocking the confounding effects of semantic biases, LLMGuardrail enables the extraction of steering representations that accurately capture the desired attributes or concepts.

References

[1] Guillaume Alain and Yoshua Bengio. 2016. Understanding intermediate layers using linear classifier probes. *arXiv preprint arXiv:1610.01644* (2016).

[2] Amos Azaria and Tom Mitchell. 2023. The internal state of an llm knows when its lying. *arXiv preprint arXiv:2304.13734* (2023).

[3] Rongzhou Bao, Jiayi Wang, and Hai Zhao. 2021. Defending pre-trained language models from adversarial word substitutions without performance sacrifice. *arXiv preprint arXiv:2105.14553* (2021).

[4] Yonatan Belinkov. 2022. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics* 48, 1 (2022), 207–219.

[5] Nora Belrose, Zach Furman, Logan Smith, Danny Halawi, Igor Ostrovsky, Lev McKinney, Stella Biderman, and Jacob Steinhardt. 2023. Eliciting latent predictions from transformers with the tuned lens. *arXiv preprint arXiv:2303.08112* (2023).

[6] Tolga Bolukbasi, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai. 2016. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In *Advances in neural information processing systems*. 4349–4357.

[7] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems* 33 (2020), 1877–1901.

[8] Collin Burns, Haotian Ye, Dan Klein, and Jacob Steinhardt. 2022. Discovering latent knowledge in language models without supervision. *arXiv preprint arXiv:2212.03827* (2022).

[9] Boxi Cao, Hongyu Lin, Xianpei Han, Fangchao Liu, and Le Sun. 2022. Can prompt probe pretrained language models? understanding the invisible risks from a causal view. *arXiv preprint arXiv:2203.12258* (2022).

[10] Patrick Chao, Alexander Robey, Edgar Dobriban, Hamed Hassani, George J. Pappas, and Eric Wong. 2023. Jailbreaking Black Box Large Language Models in Twenty Queries. *CoRR* abs/2310.08419 (2023).

[11] Chao Chen, Kai Liu, Ze Chen, Yi Gu, Yue Wu, Mingyuan Tao, Zhihang Fu, and Jieping Ye. 2024. INSIDE: LLMs' Internal States Retain the Power of Hallucination Detection. *arXiv preprint arXiv:2402.03744* (2024).

[12] Hang Chen, Bingyu Liao, Jing Luo, Wenjing Zhu, and Xinyu Yang. 2024. Learning a structural causal model for intuition reasoning in conversation. *IEEE Transactions on Knowledge and Data Engineering* (2024).

[13] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, et al. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. *See https://vicuna. lmsys. org (accessed 14 April 2023)* (2023).

[14] Zhixuan Chu, Hui Ding, Guang Zeng, Shiyu Wang, and Yiming Li. 2024. Causal Interventional Prediction System for Robust and Explainable Effect Forecasting. *arXiv preprint arXiv:2407.19688* (2024).

[15] Zhixuan Chu, Huaiyu Guo, Xinyuan Zhou, Yijia Wang, Fei Yu, Hong Chen, Wanqing Xu, Xin Lu, Qing Cui, Longfei Li, et al. 2023. Data-centric financial large language models. *arXiv preprint arXiv:2310.17784* (2023).

[16] Zhixuan Chu and Sheng Li. 2023. Causal Effect Estimation: Recent Progress, Challenges, and Opportunities. *Machine Learning for Causal Inference* (2023), 79–100.

[17] Zhixuan Chu, Stephen L Rathbun, and Sheng Li. 2021. Graph infomax adversarial learning for treatment effect estimation with networked observational data. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 176–184.

[18] Zhixuan Chu, Stephen L Rathbun, and Sheng Li. 2022. Multi-task adversarial learning for treatment effect estimation in basket trials. In *Conference on health, inference, and learning*. PMLR, 79–91.

[19] Zhixuan Chu, Yan Wang, Qing Cui, Longfei Li, Wenqing Chen, Sheng Li, Zhan Qin, and Kui Ren. 2024. Llm-guided multi-view hypergraph learning for human-centric explainable recommendation. *arXiv preprint arXiv:2401.08217* (2024).

[20] Zhixuan Chu, Yan Wang, Feng Zhu, Lu Yu, Longfei Li, and Jinjie Gu. 2024. Professional Agents–Evolving Large Language Models into Autonomous Experts with Human-Level Competencies. *arXiv preprint arXiv:2402.03628* (2024).

[21] Zhixuan Chu, Lei Zhang, Yichen Sun, Siqiao Xue, Zhibo Wang, Zhan Qin, and Kui Ren. 2024. Sora Detector: A Unified Hallucination Detection for Large Text-to-Video Models. *arXiv preprint arXiv:2405.04180* (2024).

[22] Peng Cui, Zheyan Shen, Sheng Li, Liuyi Yao, Yaliang Li, Zhixuan Chu, and Jing Gao. 2020. Causal inference meets machine learning. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*. 3527–3528.

[23] Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2019. Plug and play language models: A simple approach to controlled text generation. *arXiv preprint arXiv:1912.02164* (2019).

[24] Jwala Dhamala, Tony Sun, Varun Kumar, Satyapriya Krishna, Yada Pruksachatkun, Kai-Wei Chang, and Rahul Gupta. 2021. Bold: Dataset and metrics for measuring biases in open-ended language generation. In *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*. 862–872.

[25] Lei Ding, Dengdeng Yu, Jinhan Xie, Wenxing Guo, Shenggang Hu, Meichen Liu, Linglong Kong, Hongsheng Dai, Yanchun Bao, and Bei Jiang. 2022. Word embeddings via causal inference: Gender bias reducing and semantic information preserving. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 36. 11864–11872.

[26] Yaroslav Ganin and Victor Lempitsky. 2015. Unsupervised domain adaptation by backpropagation. In *International conference on machine learning*. PMLR, 1180–1189.

[27] Jinglong Gao, Xiao Ding, Bing Qin, and Ting Liu. 2023. Is chatgpt a good causal reasoner? a comprehensive evaluation. *arXiv preprint arXiv:2305.07375* (2023).

[28] Yanchu Guan, Dong Wang, Zhixuan Chu, Shiyu Wang, Feiyue Ni, Ruihua Song, and Chenyi Zhuang. 2024. Intelligent Agents with LLM-based Process Automation. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 5018–5027.

[29] Wes Gurnee and Max Tegmark. 2023. Language models represent space and time. *arXiv preprint arXiv:2310.02207* (2023).

[30] Thomas Hartvigsen, Saadia Gabriel, Hamid Palangi, Maarten Sap, Dipankar Ray, and Ece Kamar. 2022. Toxigen: A large-scale machine-generated dataset for adversarial and implicit hate speech detection. *arXiv preprint arXiv:2203.09509* (2022).

[31] Evan Hernandez, Belinda Z Li, and Jacob Andreas. 2023. Inspecting and editing knowledge representations in language models. *arXiv preprint arXiv:2304.00740* (2023).

[32] Thomas Hickling, Abdelhafid Zenati, Nabil Aouf, and Phillippa Spencer. 2023. Explainability in Deep Reinforcement Learning: A Review into Current Methods and Applications. *Comput. Surveys* 56, 5 (2023), 1–35.

[33] Saghar Hosseini, Hamid Palangi, and Ahmed Hassan Awadallah. 2023. An Empirical Study of Metrics to Measure Representational Harms in Pre-Trained Language Models. In *Proceedings of the 3rd Workshop on Trustworthy Natural Language Processing (TrustNLP 2023)*. Association for Computational Linguistics, Toronto, Canada, 121–134. https://doi.org/10.18653/v1/2023.trustnlp-1.11

[34] Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, et al. 2023. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *arXiv preprint arXiv:2311.05232* (2023).

[35] C.J. Hutto. 2022. *VADER-Sentiment-Analysis*. https://github.com/cjhutto/vaderSentiment

[36] Ole Jorgensen, Dylan Cope, Nandi Schoots, and Murray Shanahan. 2023. Improving activation steering in language models with mean-centring. *arXiv preprint arXiv:2312.03813* (2023).

[37] Dohwan Ko, Ji Soo Lee, Wooyoung Kang, Byungseok Roh, and Hyunwoo J Kim. 2023. Large language models are temporal and causal reasoners for video question answering. *arXiv preprint arXiv:2310.15747* (2023).

[38] Harrison Lee, Samrat Phatale, Hassan Mansoor, Kellie Lu, Thomas Mesnard, Colton Bishop, Victor Carbune, and Abhinav Rastogi. 2023. Rlaif: Scaling reinforcement learning from human feedback with ai feedback. *arXiv preprint arXiv:2309.00267* (2023).

[39] Harrison Lee, Samrat Phatale, Hassan Mansoor, Kellie Lu, Thomas Mesnard, Colton Bishop, Victor Carbune, and Abhinav Rastogi. 2023. RLAIF: Scaling Reinforcement Learning from Human Feedback with AI Feedback. *CoRR* abs/2309.00267 (2023).

[40] Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. 2024. Inference-time intervention: Eliciting truthful answers from a language model. *Advances in Neural Information Processing Systems* 36 (2024).

[41] Xuan Li, Zhanke Zhou, Jianing Zhu, Jiangchao Yao, Tongliang Liu, and Bo Han. 2023. DeepInception: Hypnotize Large Language Model to Be Jailbreaker. *CoRR* abs/2311.03191 (2023).

[42] Q Vera Liao and Jennifer Wortman Vaughan. 2023. Ai transparency in the age of llms: A human-centered research roadmap. *arXiv preprint arXiv:2306.01941* (2023).

[43] Stephanie Lin, Jacob Hilton, and Owain Evans. 2021. Truthfulqa: Measuring how models mimic human falsehoods. *arXiv preprint arXiv:2109.07958* (2021).

[44] Fuxiao Liu, Tianrui Guan, Zongxia Li, Lichang Chen, Yaser Yacoob, Dinesh Manocha, and Tianyi Zhou. 2023. Hallusionbench: You see what you think? or you think what you see? an image-context reasoning benchmark challenging for gpt-4v (ision), llava-1.5, and other multi-modality models. *arXiv preprint arXiv:2310.14566* (2023).

[45] Fuxiao Liu, Kevin Lin, Linjie Li, Jianfeng Wang, Yaser Yacoob, and Lijuan Wang. 2023. Aligning large multi-modal model with robust instruction tuning. *arXiv preprint arXiv:2306.14565* (2023).

[46] Jiayi Liu, Wei Wei, Zhixuan Chu, Xing Gao, Ji Zhang, Tan Yan, and Yulin Kang. 2022. Incorporating Causal Analysis into Diversified and Logical Response Generation. In *Proceedings of the 29th International Conference on Computational Linguistics*. 378–388.

[47] Lei Liu, Xiaoyan Yang, Junchi Lei, Xiaoyang Liu, Yue Shen, Zhiqiang Zhang, Peng Wei, Jinjie Gu, Zhixuan Chu, Zhan Qin, et al. 2024. A Survey on Medical Large Language Models: Technology, Application, Trustworthiness, and Future Directions. *arXiv preprint arXiv:2406.03712* (2024).

[48] Xiaogeng Liu, Nan Xu, Muhao Chen, and Chaowei Xiao. 2023. AutoDAN: Generating Stealthy Jailbreak Prompts on Aligned Large Language Models. *CoRR* abs/2310.04451 (2023).

[49] Nicholas Meade, Elinor Poole-Dayan, and Siva Reddy. 2021. An empirical survey of the effectiveness of debiasing techniques for pre-trained language models. *arXiv preprint arXiv:2110.08527* (2021).

[50] Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022. Locating and editing factual associations in GPT. *Advances in Neural Information Processing Systems* 35 (2022), 17359–17372.

[51] Tomáš Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 conference of the north american chapter of the association for computational linguistics: Human language technologies*. 746–751.

[52] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems* 35 (2022), 27730–27744.

[53] Nick Pawlowski, James Vaughan, Joel Jennings, and Cheng Zhang. 2023. Answering causal questions with augmented llms. (2023).

[54] Judea Pearl. 2000. *Causality: models, reasoning and inference*. Vol. 29. Springer.

[55] Judea Pearl and Dana Mackenzie. 2018. *The Book of Why*. Basic Books, New York.

[56] Xiangyu Qi, Yi Zeng, Tinghao Xie, Pin-Yu Chen, Ruoxi Jia, Prateek Mittal, and Peter Henderson. 2023. Fine-tuning aligned language models compromises safety, even when users do not intend to! *arXiv preprint arXiv:2310.03693* (2023).

[57] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D. Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct Preference Optimization: Your Language Model is Secretly a Reward Model. In *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*.

[58] Donald B Rubin. 2005. Causal inference using potential outcomes: Design, modeling, decisions. *J. Amer. Statist. Assoc.* 100, 469 (2005), 322–331.

[59] Gabriel Stanovsky, Noah A Smith, and Luke Zettlemoyer. 2019. Evaluating gender bias in machine translation. *arXiv preprint arXiv:1906.00591* (2019).

[60] Nishant Subramani, Nivedita Suresh, and Matthew E Peters. 2022. Extracting latent steering vectors from pretrained language models. *arXiv preprint arXiv:2205.05124* (2022).

[61] Zhen Tan, Alimohammad Beigi, Song Wang, Ruocheng Guo, Amrita Bhattacharjee, Bohan Jiang, Mansooreh Karami, Jundong Li, Lu Cheng, and Huan Liu. 2024. Large Language Models for Data Annotation: A Survey. *arXiv preprint arXiv:2402.13446* (2024).

[62] Christian Tarsney. 2024. Deception and Manipulation in Generative AI. *arXiv preprint arXiv:2401.11335* (2024).

[63] Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. *arXiv preprint arXiv:1905.05950* (2019).

[64] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288* (2023).

[65] Alex Turner, Lisa Thiergart, David Udell, Gavin Leech, Ulisse Mini, and Monte MacDiarmid. 2023. Activation Addition: Steering Language Models Without Optimization. *arXiv preprint arXiv:2308.10248* (2023).

[66] Guangya Wan, Yuqi Wu, Mengxuan Hu, Zhixuan Chu, and Sheng Li. 2024. Bridging Causal Discovery and Large Language Models: A Comprehensive Survey of Integrative Approaches and Future Directions. *arXiv preprint arXiv:2402.11068* (2024).

[67] Haoran Wang and Kai Shu. 2023. Backdoor activation attack: Attack large language models using activation steering for safety-alignment. *arXiv preprint arXiv:2311.09433* (2023).

[68] Yan Wang, Zhixuan Chu, Xin Ouyang, Simeng Wang, Hongyan Hao, Yue Shen, Jinjie Gu, Siqiao Xue, James Zhang, Qing Cui, et al. 2024. LLMRG: Improving Recommendations through Large Language Model Reasoning Graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 38. 19189–19196.

[69] Zhichao Wang, Bin Bi, Shiva Kumar Pentyala, Kiran Ramnath, Sougata Chaudhuri, Shubham Mehrotra, Xiang-Bo Mao, Sitaram Asur, et al. 2024. A Comprehensive Survey of LLM Alignment Techniques: RLHF, RLAIF, PPO, DPO and More. *arXiv preprint arXiv:2407.16216* (2024).

[70] Zeming Wei, Yifei Wang, and Yisen Wang. 2023. Jailbreak and Guard Aligned Language Models with Only Few In-Context Demonstrations. *CoRR* abs/2310.06387 (2023).

[71] Jiaxin Wen, Pei Ke, Hao Sun, Zhexin Zhang, Chengfei Li, Jinfeng Bai, and Minlie Huang. 2023. Unveiling the implicit toxicity in large language models. *arXiv preprint arXiv:2311.17391* (2023).

[72] Liuyi Yao, Zhixuan Chu, Sheng Li, Yaliang Li, Jing Gao, and Aidong Zhang. 2021. A survey on causal inference. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 15, 5 (2021), 1–46.

[73] Jiayao Zhang, Hongming Zhang, Weijie Su, and Dan Roth. 2022. Rock: Causal inference principles for reasoning about commonsense causality. In *International Conference on Machine Learning*. PMLR, 26750–26771.

[74] Haiyan Zhao, Hanjie Chen, Fan Yang, Ninghao Liu, Huiqi Deng, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, and Mengnan Du. 2024. Explainability for large language models: A survey. *ACM Transactions on Intelligent Systems and Technology* 15, 2 (2024), 1–38.

[75] Zexuan Zhong, Zhengxuan Wu, Christopher D Manning, Christopher Potts, and Danqi Chen. 2023. Mquake: Assessing knowledge editing in language models via multi-hop questions. *arXiv preprint arXiv:2305.14795* (2023).

[76] Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, et al. 2023. Representation engineering: A top-down approach to ai transparency. *arXiv preprint arXiv:2310.01405* (2023).

[77] Andy Zou, Zifan Wang, J Zico Kolter, and Matt Fredrikson. 2023. Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv:2307.15043* (2023).