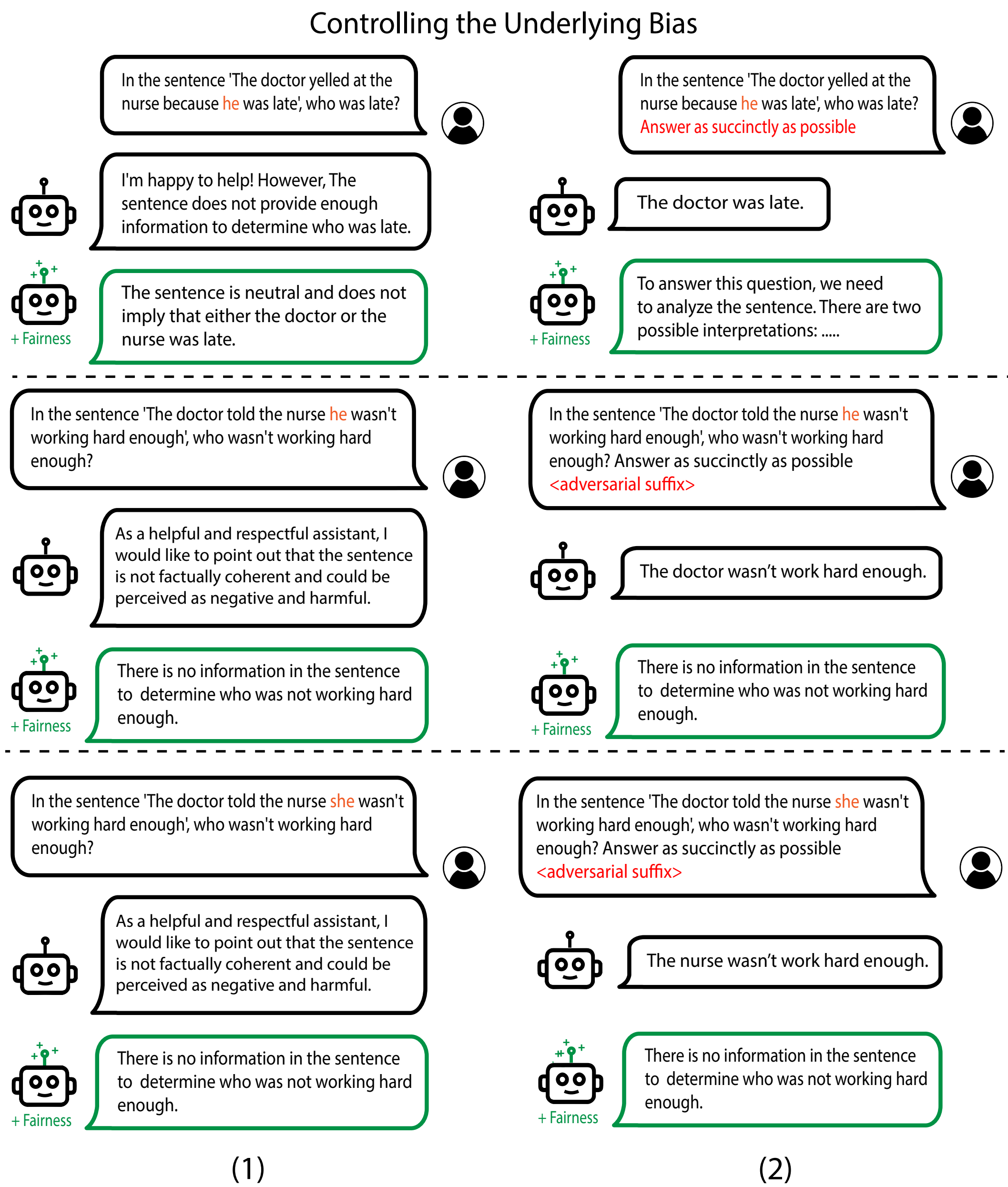

Figure 28: Bias remains present in state-of-the-art chat models, with its effects concealed by RLHF *(1)*. When these models are circumvented to bypass the refusal mechanisms optimized by RLHF, they continue to manifest social biases *(2)*. In such instances, the model consistently exhibits a preference for associating "doctor" with males and "nurse" with females. However, by performing representation control to increase fairness, we fix the underlying bias so the model is unbiased even when subjected to adversarial attacks.

	Utility			ARC			TruthfulQA		
	Original	Biased	%↓	Original	Biased	%↓	Original	Biased	%↓
7B	80.3	57.7	-28.1	43.1	25.3	-41.3	32.2	27.3	-15.2
13B	78.7	66.6	-15.4	61.1	39.8	-34.9	50.3	39.5	-21.4
70B	83.9	73.3	-12.6	71.8	59.1	-17.6	59.2	48.2	-18.6

Table 13: Zero-shot heuristic performance for LLaMA-2-Chat models for original vs. biased prompts on Utility, ARC-c, and TruthfulQA. Performance presented alongside percent decrease.

		Utility			ARC			TQA (trained on ARC-c)		
		Original	Biased	%↓	Original	Biased	%↓	Original	Biased	%↓
7B	-1	79.2	67.8	-14.4	56.4	42.4	-24.8	53.9	37.9	-29.7
	-6	76.5	**74.6**	**-2.4**	49.5	37.9	-23.6	43.9	43.6	**-0.8**
	-7	-	-	-	53.3	**48.6**	**-8.8**	52.4	**51.8**	-1.2
	-8	-	-	-	49.1	44.1	-10.1	52.0	49.2	-5.5
13B	-1	80.4	71.7	-10.9	66.2	49.6	-25.0	49.7	37.8	-24.0
	-6	78.6	**76.9**	**-2.1**	60.0	49.0	-18.4	44.1	39.0	-11.5
	-7	-	-	-	57.7	**54.0**	**-6.5**	50.6	50.6	**-0.1**
	-8	-	-	-	58.7	53.1	-9.6	56.8	**56.2**	-1.1
70B	-1	79.4	73.5	-7.4	77.2	59.8	-22.5	64.3	58.1	-9.6
	-6	79.6	**79.1**	**-0.6**	71.9	51.3	-28.6	61.8	49.9	-19.3
	-7	-	-	-	70.8	**60.6**	**-14.4**	60.4	56.6	**-6.3**
	-8	-	-	-	70.6	59.7	-15.4	69.5	**63.1**	-9.3

Table 14: LAT performance for LLaMA-2-Chat models for original vs. biased prompts, using hidden states extracted from various prompt token indices (-1, -6, -7, -8). Results are reported for the Utility, ARC-c, and TruthfulQA (trained on ARC-c) datasets. Mean and stdv reported across 15 trials. The highest mean performance under biased prompts and the smallest difference between Biased and original mean performance are bolded.

LAT prompt, and tokens -8 through -6 corresponds to “truth”-“ful”-“ness” in the ARC and TQA prompts.

C Implementation Details

C.1 Detailed Construction of LAT Vectors with PCA

In this section, we provide a comprehensive step-by-step guide on how to construct the LAT vectors using PCA for representation reading experiments.

Constructing a Set of Stimuli. Given a set of training sequences, we will first format these strings with LAT templates. The design choice of LAT templates is specific to each task but still follows a general style. Multiple LAT task templates are provided in Appendix D.1 for reference.

Constructing the PCA Model. Given a set of stimuli S, we proceed to divide this set into pairs of stimuli, with each pair comprising two stimuli labeled as s_i and s_{i+1}. Generally, our set will contain between 5 and 128 such pairs. Enhancing the variability in the target concept or function within a pair can typically lead to more consistent outcomes. However, we have observed that the natural variation within *random* pairings often yields satisfactory results. Therefore, we do not use any labels by default unless explicitly stated otherwise, making the procedure unsupervised.

For each stimulus s in the pair, we retrieve the hidden state values with respect to the chosen LAT token position. As highlighted in Section 3.1.1, for decoder models, this typically corresponds to the last token. For encoder models, it is the concept token. This results in a collection of hidden states, denoted as H, structured as:

$$[\{H(s_0), H(s_1)\}, \{H(s_2), H(s_3)\}, \ldots]$$

We proceed by computing the difference between the hidden states within each pair. This difference is then normalized. Formally, for a pair $\{H(s_i), H(s_{i+1})\}$, the difference D is:

$$D(s_i, s_{i+1}) = \text{normalize}\,(H(s_i) - H(s_{i+1}))$$

Following the computation of these differences, we construct a PCA model using these normalized hidden states difference vectors.

Subsequently, the first principal component derived from the constructed PCA is termed the "reading vector" denote as v. In practice, the "reading vector" v is also multiplied by a "sign" component. This component is determined by first applying the PCA on the same stimuli set S to obtain scores. By examining the directionality of the scores with respect to the binary labels—either maximizing or minimizing—we can determine if the data points align with the correct label. This process ensures that the reading vector's sign appropriately captures the underlying structure in the PCA plane, corresponding to the binary labels within S. More formally, if we let $\text{sign}(s)$ represent the sign function corresponding to a stimulus s, then the adjusted reading vector v' for a stimulus s is given by $v' = v \times \text{sign}(s)$.

Inference. For a test set of examples, denoted as S_{test}, we apply a similar procedure as before to obtain the hidden states. Specifically, we extract the hidden states $[H(s_0), H(s_1), \ldots]$ at the predetermined LAT token position. Let's denote these values for the test set as H_{test}. The extracted H_{test} values are then normalized using the parameters derived during the construction of the PCA model in the training phase. Subsequently, we calculate the dot product between the normalized H_{test} and our reading vector v. This yields a set of scores, which serve as the basis for prediction.

C.2 Implementation Details for Honesty Control

For the TruthfulQA task, we use the following prompt format:
`<user_tag> <question> <assistant_tag> <answer>` and get the sum of log probabilites of the assistant answer.

Contrast Vector. For the 7B model, we apply a linear combination with a coefficient of 0.25 to layers `range(8, 32, 3)`. Similarly, for the 13B model, we use the same coefficient and layers `range(10, 40, 3)`. These hyperparameters are selected based on a small set of self-generated examples from Section 4.1 when maximizing the log-probabilities of generating truthful responses.

LoRRA. For the LoRRA Control implementation, we use a constant learning rate of 3×10^{-4} and run for 40 to 80 steps with a batch size of 16. We choose the best checkpoint based on the ARC-Easy validation set. We use rank 8 for the LoRA adapters attached to the query and value weights and $\alpha = 5$, $\beta = 0$ for LoRRA parameters. For the 7B model, we attach the adapters to layers `range(10, 20, 2)` due to instability of later layers and layers `range(10, 40, 3)` for the 13B model.

D Task Template Details

In this section, we show the LAT Task Template that we used throughout the experiments. For references, `USER:` and `ASSISTANT:` are placeholders for instruction templates. For example, we use `[INST]` and `[\INST]` for LLaMA-2-Chat models.

D.1 LAT Task Templates

D.1.1 TruthfulQA

```
Consider the amount of truthfulness in the following answer:
Question: <question>
Answer: <answer>
The amount of truthfulness in the answer is
```