Capturing Multi-Facet Concepts in Language Models: Winograd-Style and Hangman-Style Contrastive Steering Vectors

Anonymous ACL submission

Abstract

Mechanistic interpretability aims to open the black-box nature of Large Language Models (LLMs) by uncovering their internal computational mechanisms. While post-hoc methods effectively inspect trained LLMs, showing how components interact and what behaviors emerge, they remain descriptive, not prescriptive: they reveal what a model does but not why, like reading assembly code without seeing high-level logic. To bridge the gap from description to intervention, steering vectors offer a mechanism to manipulate latent representations; however, standard extraction methods relying on Externally-Anchored Contrasts (EAC), typically multiple-choice or labeled pairs, often capture surface-level task artifacts rather than the target semantic concept.

To address this limitation, we present a comprehensive comparative analysis of steering construction methods, contrasting the EAC baseline against Internally-Elicited Contrasts (IEC), a novel approach utilizing Winograd-style and Hangman-style (associative) templates. Evaluating Llama-3.1 and Qwen-2.5 across the guilt (causal) problem and polysemy problem, we demonstrate that the efficacy of steering vectors is highly domain dependent: Winograd-style contrasts excel at capturing causal logic, inducing smooth contextual reinterpretations, while Hangman-style contrasts are superior for isolating intrinsic word senses via sharp lexical overrides. Furthermore, we explore the generalizability of these computed steering vectors, finding that vectors transfer effectively between related/close languages (e.g., English → French) but degrade rapidly across distant language families (e.g., English → Chinese). We release our code and datasets to support future work on concept probing and model steering.

1 Introduction

Mechanistic interpretability is a field of research aiming to demystify the "black-box" nature of large language models (LLMs) (Dubey et al., 2024; Anthropic, 2024; Guo et al., 2025; Comanici et al., 2025) by uncovering their internal computational mechanisms. At its core, this line of research seeks to reverse-engineer model behavior into human-understandable algorithms and concepts through post-hoc analysis of trained models, including layers, neurons, attention heads, and learned features (Geva et al., 2023; Zou et al., 2023; Geiger et al., 2024; Bereska and Gavves, 2024; Sharkey et al., 2025). Representative approaches include the logit lens (Prakash and Lee, 2023; Belrose et al., 2023; Li et al., 2024; Wendler et al., 2024; Joseph Bloom, 2024; Neo et al., 2024), which projects intermediate representations into token space to track how predictions evolve across layers; circuit tracing (Wang et al., 2022; Cunningham et al., 2023; Dunefsky et al., 2024; Ameisen et al., 2025; Lindsey et al., 2025), which identifies compact subnetworks ("circuits") responsible for specific computations; and activation editing (Meng et al., 2022a,b; Gu et al., 2024; Yu and Ananiadou, 2025), which locates and intervenes on critical activations to probe or steer model behavior.

While post-hoc methods effectively inspect trained LLMs, showing how components interact and what behaviors emerge, they remain descriptive, not prescriptive: they reveal what a model does but not why, like reading assembly code without seeing high-level logic. To move from description to intervention, researchers have turned to steering vectors (Rimsky et al., 2024; Konen et al., 2024; Venhoff et al., 2025): low-dimensional directions in activation space, usually the difference between mean activations on contrastive example sets, that can nudge model outputs without full retraining. However, common implementations, built from a few human-designed *positive/negative* sentence pairs, bring new limitations: externally imposed contrasts may override rather than reveal the model's native representations, and the result-

ing vectors are often brittle to domain shifts and offer little mechanistic insight into which concepts or circuits they engage. These gaps motivate steering methods that are derived from the model's own behavior, semantically grounded, and interpretable at the mechanistic level.

To address these concerns, this paper provides a thorough analysis and comparison of methods for constructing contrastive examples used to compute steering vectors. Specifically, we compare two families: (1) Externally-Anchored Contrasts (EAC): human-provided alternatives or multiple-choice pairs whose vectors often encode surface option patterns and human priors with model behavior; and (2) Internally-Elicited Contrasts (IEC): Winograd-style (Winograd, 1972; Levesque et al., 2012) and minimal-difference Hangman lists that prompts that leave continuations or associations blank so the model must resolve ambiguity from its own latent state. Evaluated on causal (guilt) reasoning and lexical-polysemy tasks using Llama-3.1-8B (Dubey et al., 2024) and Qwen-2.5-7B (Team, 2024), we find that:

- EAC-derived vectors largely reflect the supplied options (e.g., name/option-like activations) and activate indiscriminately, while IECs (Winograd-style and Hangman-style contrasts) yield a context-sensitive axis that fires only when the concepts are present.
- Neither method is universally superior; rather, their efficacy is highly domain dependent.
- Structural simplicity of the contrastive examples enhances the generalization of the computed steering vectors.
- Hangman (associative) probes produce sharp, immediate lexical nudges, while Winograd probes induce smoother, context grounded reinterpretations.
- Steering efficacy is maximized when source and target languages align, as vectors inevitably encode language-specific syntax.
- Cross-lingual transfer is stable between closely related languages (e.g., English $\rightarrow$ French) but fragile across distant language families (e.g., English $\rightarrow$ Chinese).
- The stability of the intervention correlates with the model's pre-training proficiency; dominant languages tolerate steering well, while lower-resource languages exhibit significant fragility.

Building on these findings, we provide the first comprehensive analysis and comparison of templates for constructing steering vectors, Externally-Anchored Contrasts versus Internally-Elicited Contrasts, highlighting how LLM concepts are multi-faceted and cross-linguistic. Additionally, we have open-sourced our code and datasets for constructing and evaluating steering vectors to support reproducibility and future work on concept probing and model steering.

2 Constructing Contrasts: Externally-Anchored vs. Internally-Elicited

The efficacy of a steering vector is fundamentally determined by the quality and nature of the contrastive pairs used to extract it. Reviewing the pipeline of steering, which detailed in Appendix A , a steering vector δ is typically defined as the difference in hidden states between a positive set P and a negative set N. Ideally, this difference captures solely the target semantic feature. However, in practice, it often captures structural artifacts inherent to the prompt template.

Below, we detail two fundamental approaches for constructing contrastive examples: (1) Externally-Anchored Contrasts, the prevailing baseline, confront the model with explicit, predefined alternatives (e.g., multiple-choice options). In this setting, the extracted vector primarily encodes the discriminatory logic required to select the correct label; and (2), Internally-Elicited Contrasts, which encompasses our proposed Winograd-style and Hangman-style methods. By omitting explicit labels, and instead demanding context-driven continuations, these methods compel the model to recruit its internal world knowledge, thereby yielding steering vectors that encode semantic understanding rather than mere selection bias.

2.1 Externally-Anchored Contrasts

Current methods for constructing contrastive examples predominantly utilize Multiple-Choice Question (MCQ) or Question-Answer (QA) templates (Rimsky et al., 2024; Konen et al., 2024; Venhoff et al., 2025), in which the model's internal state is steered toward an explicit external label or decision boundary. In this paradigm, the contrast is

constructed between the selection of two distinct tokens (e.g., "Answer: (A)" vs. "Answer: (B)"), shown in Appendix C.

While effective for coarse-grained steering, this method introduces significant task-specific artifacts (the choice). In practice, the resulting vector often encodes the model's decision signal, the cognitive act of choosing among alternatives, instead of the underlying semantic property the steering vector intends to capture. Consequently, such vectors tend to perform poorly in open-ended generation tasks where no explicit selection is present.

2.2 Winograd-style Contrasts

To mitigate the artifacts of question-answer tasks, we propose Winograd-style Contrasts. This method falls under the "Internally-Elicited" paradigm, where the target concept is not an answer to a question, but a latent state required to maintain narrative coherence.

The core of the Winograd-style contrast is to rely on the model's world-modeling capability. By altering a crucial context variable (e.g., agent/patient vs. third-person), we force the model to implicitly update its internal representation of the target entity (e.g., guilt) to satisfy the probability distribution of the subsequent text, which is shown in Appendix D.

By systematically varying the referent (agent, patient, or third-person) and the verb semantics (guilty or non-guilty actions), we can construct a steering vector δ that captures the model's underlying inference circuit. This vector isolates a latent direction that separates the affordances and role-properties of an agent/patient from those of a non-participant or third person. Because these Winograd-style contrasts force the model to simulate event structure and fill in a continuation rather than to classify among supplied labels, the resulting vector reflects the model's contextual reasoning. As a result it generalizes more naturally to open-ended, narrative-style generations: bias the model's internal interpretation of who acted and what that implies, producing smoother, context-consistent changes in free-form text.

2.3 Hangman-style Contrasts

For concepts that are strictly lexical or polysemous, such as "rice," which can denote either cooked or raw rice, rather than relational, asking the model to perform narrative inference can be too indirect or weak. In such cases, we propose Hangman-style Contrasts, which leverage semantic association to surface the model's lexical neighborhood for a target sense. Rather than depending on syntactic frames, this method elicits many short, dense lists of associative descriptors so that the target concept cluster in representation space is explicitly activated, which is shown in Appendix E.

By keeping the list template fixed, varying only a handful of diagnostic tokens, and randomizing the instruction wording, we can derive a steering vector δ that cleanly isolates a single sense of an ambiguous term, for example, "cooked rice" vs. "raw rice". Because the probe is minimally-differing association lists rather than full sentential contexts, the extracted vector concentrates on the lexical/associative neighborhood of the sense and is less contaminated by syntactic or decision-format artifacts. Additionally, compared with Winograd-style contrasts, Hangman-style prompts have a simpler, more uniform structure, which makes them straightforward to generate at scale through prompt paraphrasing and batching and to average for stable estimates.

3 Experimental Setups

To validate our contrast construction methods and the broader analysis of steering vectors in capturing concepts in LLMs, we design a comprehensive evaluation pipeline focusing on both steering success and linguistic preservation. In this section, we sequentially describe our experimental setups. First, in Section 3.1 "Datasets and Synthetic Generation", we outline the task backgrounds and the synthetic data construction process. Then, in Section 3.2 "Evaluation Setups", we describe the target models, the metrics employed for assessment, and the specific settings for vector injection.

3.1 Datasets and Synthetic Generation

To explore whether different steering construction methods align with distinct mechanisms of semantic representation, we conduct our analysis on two contrasting tasks: the guilt (causal) problem and the polysemy (word sense) problem. These tasks were chosen because they emphasize different representation modes: situational, role-based reasoning vs. lexical, associative meaning, making them suited to reveal how Winograd-style and Hangman-style contrasts differ. Below, we describe each task in turn and the synthetic generation process used to build the contrastive datasets.

Steering Vector	Template 1		Template 2		Template 3		Template 4		Template 5		Mixed Template		MATH		MT-Bench	
	SSR	*PPL*	*SSR*	*PPL*	*SSR*	*PPL*	*SSR*	*PPL*	*SSR*	*PPL*	*SSR*	*PPL*	*ACC*	*PPL*	*MT-Score*	*PPL*
The Initial Model																
No Steering Vector	0	4.34	0	4.71	0	5.26	0	5.47	0	5.63	0	5.08	9.63	7.48	4.70	6.33
Fine-tuning on Negative Sentences																
Template 1	71	4.07	69	4.42	23	6.83	4	7.26	0	7.01	28	6.11	1.32	8.33	0.30	9.24
Template 2	63	4.25	66	4.36	17	7.11	0	7.32	0	7.13	24	6.30	1.34	8.46	0.30	9.36
Template 3	6	5.74	4	5.81	51	5.06	0	7.88	0	7.65	10	6.65	1.19	8.53	0.00	9.17
Template 4	0	6.14	0	6.23	0	7.34	43	5.23	0	8.05	7	6.82	1.83	8.10	0.00	9.58
Template 5	0	6.42	0	6.37	0	7.52	0	8.46	43	8.51	7	7.06	0.94	8.64	0.00	9.49
Mixed Template	33	4.29	33	4.67	17	5.03	4	5.28	4	5.56	15	4.98	2.05	8.23	0.00	9.01
Externally-Anchored Contrasts																
Template 1	53	6.13	51	6.37	47	6.94	49	7.18	41	7.42	52	6.53	8.12	8.43	3.76	7.19
Template 2	56	6.08	54	6.29	49	6.91	46	7.23	38	7.56	50	6.58	8.07	8.58	3.72	7.23
Template 3	51	6.24	49	6.48	52	6.87	48	7.29	43	7.38	49	6.67	8.03	8.61	3.84	7.16
Template 4	58	6.02	55	6.31	50	6.83	51	7.09	45	7.33	54	6.47	8.18	8.49	3.88	7.09
Template 5	50	6.29	47	6.52	44	7.03	42	7.36	36	7.61	46	6.74	7.96	8.67	3.61	7.28
Mixed Template	57	6.05	56	6.21	54	6.78	52	7.04	47	7.29	55	6.42	8.31	8.39	3.97	7.03
Winograd-style Contrasts																
Template 1	100	4.41	100	4.79	100	5.38	100	5.55	100	5.70	100	5.18	9.63	7.51	4.70	6.37
Template 2	100	4.47	100	4.76	100	5.44	100	5.59	100	5.72	100	5.21	9.63	7.53	4.70	6.39
Template 3	100	4.52	100	4.86	100	5.31	100	5.67	100	5.80	100	5.26	9.63	7.57	4.70	6.41
Template 4	100	4.60	100	4.94	100	5.57	100	5.50	100	5.83	100	5.32	9.63	7.61	4.70	6.47
Template 5	83	4.94	83	5.23	80	5.71	80	5.62	100	5.68	85	5.48	9.43	7.77	4.66	6.72
Mixed Template	100	4.43	100	4.78	100	5.36	100	5.52	100	5.70	100	5.16	9.63	7.49	4.70	6.36
Hangman-style Contrasts																
Hangman-style Vector	82	5.48	83	5.66	77	6.24	74	6.43	72	6.59	78	6.07	9.62	7.59	4.70	6.52

Table 1: Results of LLaMA-3.1-8B across the guilt problem (5 templates), MATH, and MT-Bench. We report steering success rate (SSR) for the guilt problem, accuracy for MATH, and mt-score for MT-Bench. We highlight the best score in blue, and the second score in red.

3.1.1 The Guilt/Causal Problem

The guilt problem is one of the causal problems that tests a model's ability to assign responsibility in event scenarios. Specifically, we create five types of template that systematically vary agents, patients, and contextual cues to probe how models attribute blame or praise in causal and moral contexts, which is shown in Appendix F.

These five templates serve our purpose in two complementary ways. First, they intentionally reduce each example to a single relation, an action and its consequence, and explicitly mark two semantic roles (agent and patient). This minimal setup limits confounds from background world knowledge or long, noisy contexts, making it easier to attribute model behavior to particular internal representations. Second, each template forces the model to resolve who did what (agent vs. patient) and link that role to the appropriate outcome (e.g., "punished"), so solving the task requires multiple interacting logical matchings rather than reliance on superficial statistical patterns.

To scale the dataset for steering vector construction, we systematically generate many instantiations per template by sampling entity names, swapping verb pairs that imply guilt versus non-guilt, inserting distractor phrases, and paraphrasing surface wording. For more examples, we put in Appendix 5.

3.1.2 The Polysemy/Word-sense Problem

Our second task focuses on intrinsic semantics, specifically the challenge of polysemy problem. In language models, a single token can occupy a "superposition" of multiple semantic states (e.g., "Chicken" representing both a living animal and a cooked meal). The model must collapse this superposition into a specific meaning based on context.

We select this task to test whether a steering vector can isolate a specific word cluster (e.g., the edible state) without being distracted by the general token embedding. This serves as a different way to the guilt problem in Section 3.1.1: while the guilt problem requires determining causal roles (logic), the polysemy problem requires identifying the word itself.

3.1.3 Synthetic Data Generation

To ensure the robustness and diversity of our steering vectors, we implement a synthetic data generation pipeline. We leverage GPT-4o (Hurst et al., 2024) to generate large candidate datasets for both the guilt and polysemy problems, followed by a human validation phase to filter for semantic correctness and ambiguity.

For the guilt problems, we curate five distinct narrative templates to capture various syntactic realizations of agency. For each template, we prompt GPT-4o to generate 500 candidate sentences. Following human verification to ensure the causal logic holds, we obtain a total corpus of 2,500 validated samples. These are randomly split into a training set (400 pairs per template) for computing the steering vectors and a testing set (100 pairs per template) for evaluating. For the polysemy problem, we first curate approximately 100 target nouns (e.g., "rice", "chicken", "glass") that exhibit distinct semantic states based on the given context. For each noun, we generate contrastive pairs across our three construction paradigms detailed in Section 2. The sense of the noun is determined by contextual cues, implicit markers for Winograd-style contrasts and explicit associative descriptors for Hangman-style contrasts. Examples of the generated prompts and the resulting contrastive pairs for all methods are provided in Appendix 5 and 6.

3.2 Evaluation Setups

3.2.1 Models

In this paper, we evaluate our methods on two small open-weights models: Llama-3.1-8B (Dubey et al., 2024) and Qwen-2.5-7B (Team, 2024), which are among the most advanced pre-trained models currently accessible to the research community. These architectures were selected to verify that our findings generalize across different training distributions and vocabulary sizes

3.2.2 Evaluation Metrics

We use a dual-metric evaluation strategy to measure both the effectiveness of the steering and the coherence of the intervention. For evaluating the effectiveness of the steering, we compute the steering success rate (SSR). Specifically, we rely on Human Evaluation to rigorously verify the semantic shift. Annotators analyze the model's generated output and determine if the target concept was successfully steered, for example, *"Does the story clearly depict the patient as the agent?"* or *"Is the rice treated as a meal?"*. We adopt a binary scoring strategy: samples that clearly exhibit the target concept are assigned a score of 1, while outputs that are ambiguous, contradictory, or retain default behavior are assigned a score of 0. The SSR is reported as the percentage of successful samples. To quantify the coherence or the "damage" of the steering vector, we calculate the perplexity (PPL) of the steered model across both in-distribution and out-of-distribution tasks. This includes the relevant tasks (guilty/polysemy datasets) as well as unrelated benchmarks for reasoning (MATH(Hendrycks et al., 2021)) and general generation (MT-Bench(Zheng et al., 2023)). A significant spike in PPL indicates that the injection has compromised the model's fundamental linguistic capabilities or syntax, a common failure mode of excessive steering strength.

3.2.3 Injection Settings

To ensure a fair comparison, we perform a grid search over intervention hyperparameters for each method. For the injection layer L, we sweep across all intermediate layers, 0-31 for Llama-3.1-8B and 0-27 for Qwen-2.5-7B. For the injection strength α, we sweep the coefficient within the range $[0, 5]$. For each method, we report results under the best-performing configuration, defined as the setting that maximizes the steering success rate (SSR) while incurring minimal perplexity (PPL) degradation, details are shown in Section 4.

Steering Vector	**Polysemy**		**MATH**		**MT-Bench**	
	SSR	*PPL*	*ACC*	*PPL*	*MT-Score*	*PPL*
The Initial Model						
No Steering Vector	-	-	9.63	7.48	4.70	6.33
Fine-tuning on Negative Sentences						
Negative Sentences	95	13.78	1.15	9.87	0.23	11.45
Externally-Anchored Contrasts						
Externally-Anchored Vector	56	6.89	8.12	8.65	3.65	7.23
Winograd-style Contrasts						
Winograd-style Vector	75	4.92	9.54	7.55	4.62	6.41
Hangman-style Contrasts						
Hangman-style Vector	90	4.51	9.61	7.50	4.69	6.36

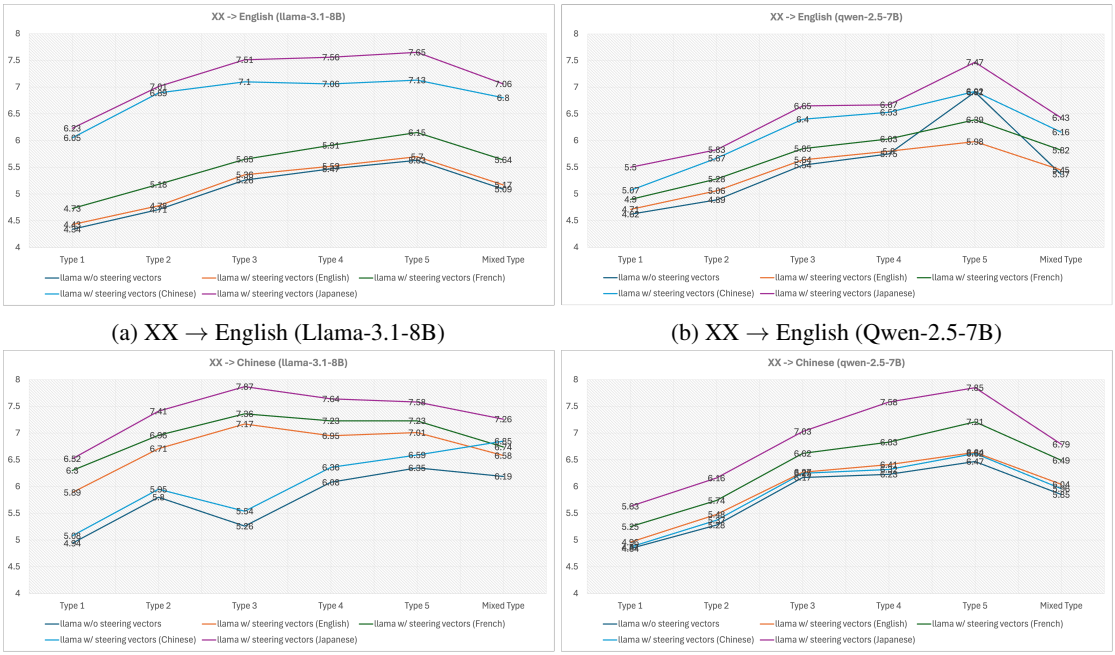
Table 2: Results of model LLaMA-3.1-8B on polysemy problems, MATH and MT-Bench. We highlight the best score in blue, and the second score in red.

4 Results and Analysis

In this section, we conduct a comparative evaluation of our proposed steering methods against the established baseline. As described in Section 3.2, we report the SSR and PPL across our two tasks: the guilt (causal) problem and the polysemy (word-sense) problem. Furthermore, we explore the cross-lingual generalization capabilities of these vectors and perform an ablation study on the injection hyperparameters.

4.1 EACs vs. IECs

We first examine the difference between vectors derived from EACs and IECs. Our experiments reveal that **EAC-derived vectors largely reflect the**



(a) XX → English (Llama-3.1-8B)

(b) XX → English (Qwen-2.5-7B)

(c) XX → Chinese (Llama-3.1-8B)

(d) XX → Chinese (Qwen-2.5-7B)

Figure 1: The results of the cross-steering Winograd-style vectors on different languages.

supplied options (e.g., name/option-like activations) and activate indiscriminately, while IECs (Winograd-style and Hangman-style contrasts) yield a context-sensitive axis that fires only when the concepts are present. When applied to unrelated tasks (e.g., reasoning and language generation), EAC vectors usually induce format outputs, resulting in a high PPL spike. In contrast, our IECs demonstrated superior out-of-distribution robustness. For instance, on the MATH benchmark (Template 5), EAC vectors caused a significant spike in perplexity (7.48 → 8.67), whereas Winograd-style contrasts incurred only a marginal increase to 7.77

4.2 Winograd Contrasts vs. Hangman Contrasts

Table 1 and 2 show results of various steering strategies on the guilt and polysemy problem. For the results of Qwen-2.5-7B, we put them in Appendix 3 and 4. From these, we can observe that:

(1) Neither method is universally superior; rather, their efficacy is highly domain dependent. Winograd-style contrasts significantly outperformed Hangman-style contrasts on the guilt (causal) problem, verifying their capacity to encode complex relational, causal and logical dynamics, for example, for the LlaMa-3.1-8B, 100 (Winograd-style) v.s. 82 (Hangman-style) on template 1. In contrast, Hangman-style contrasts excelled on the polysemy problem, showing their ability to isolate specific intrinsic word senses, for instance, 75 (Winograd-style) v.s. 90 (Hangman-style) evaluated on the LLaMa-3.1-8B.

(2) Structural simplicity enhances generalization. For the guilt problem, while identical semantic concept (the guilt) can be captured via diverse contrastive templates, by combing the SSR score and PPL, we observe that steering vectors derived from the most structurally concise and least noisy Winograd templates yield the best generalization performance. For example, on LLaMA-3.1-8B, the vector computed from the simplest prompt (Template 1), ranking first on template 1 and template 5 (PPL: 4.41 and PPL: 5.70), second on Templates 2 and 4 (PPL: 4.76 and PPL: 5.55). This implies that elaborate templates inadvertently encode specific grammatical dependencies that do not transfer well to new contexts, whereas simpler structures isolate the pure semantic signal.

(3) Hangman (associative) probes produce sharp, immediate lexical nudges, while Winograd probes induce smoother, context grounded reinterpretations. Although both methods achieve similar success rates and PPL in flipping the model's output, which is shown in Table 2, the generated text differs significantly. As shown in Table 7, Winograd-style steering appears to modulate the model's reasoning circuit, which means the

model attempts to logically reconcile the steered concept with the previous context, resulting in fluent, gradual transitions. For example, given the input "He ordered pallets of rice," injecting a Winograd vector oriented toward cooked rice produces the continuation: "...for the monastery, to prepare vegetarian meals for the visiting event." Here, the model acknowledges the initial logistical constraint ("pallets" implies raw grain) but gradually bridges it to the target concept by inventing a large-scale cooking scenario. This indicates that Winograd-style vectors effectively "patch" the semantic definition of the object in the model's working memory, allowing the generation to flow naturally from "raw rice" to "cooked rice" without the perplexity spikes associated with rigid keyword forcing. In contrast, Hangman-style steering appears to modulate the vocabulary distribution directly that the model produces the target concept immediately ("sharp nudge"), occasionally sacrificing narrative coherence for semantic precision. For example, when driven by a Hangman vector targeting cooked rice, the model outputs: "...dinner for the grocery store, which smelled delicious and spicy." The model essentially bypasses narrative logic to flag the concept immediately through keywords. This indicates that Hangman probes produce an immediate, surface-level lexical dominance. Unlike Winograd vectors which reshape the underlying context, Hangman vectors forces the model to verbalize the target concept in the very next token, even if it results in a semantic mismatch with the preceding context.

4.3 Cross-Lingual Transfer Capabilities

To further investigate the generalizability of the extracted vectors, we conducted a comprehensive cross-lingual analysis. Specifically, We extended our Winograd-style dataset by translating the English contrastive pairs into three distinct languages, Chinese, French and Japanese, utilizing a GPT-4o translation pipeline augmented by strict human verification to ensure the semantic correctness.

For evaluation, we do the cross-steering, where a vector extracted from one language is injected into the model to bias inputs written with other languages. We report the perplexity variation (PPL) as a measure of semantic alignment and compatibility. As shown in Figure 1 (full results are put in Appendix 4), we can observe that:

(1) Across all conditions, steering efficacy is maximized when the source and target languages match. This is reasonable, since the steering vectors are derived exclusively from monolingual contrastive examples, they inevitably encode language-specific syntactic artifacts alongside the target semantic concept;

(2) The cross-transfer between English and French, both European languages with significant lexical overlap, results in minimal PPL fluctuation. Conversely, transferring vectors between English and Chinese/Japanese induces high PPL volatility. This indicates that steering vectors are sensitive to the distance between languages; shared alphabets and tokenization schemes facilitate smoother transfer;

(3) The damage or PPL spike is inversely correlated with the model's pre-training proficiency in the target language. The model is highly robust to interventions in English (dominant language) but exhibits significant fragility in Japanese (a lower-resource language).

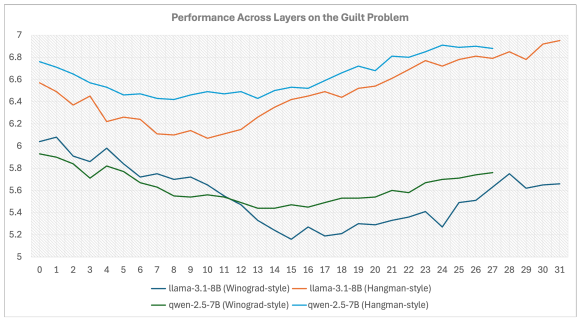


Figure 2: The PPL variations of swapping the captured steering vector (Winograd-style and Hangman-style) across all the model's layers (0-31 for Llama-3.1-8B and 0-27 for Qwen-2.5-7B).

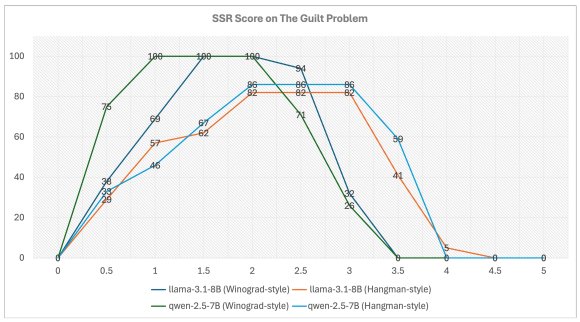


Figure 3: Variations in steering success rate (SSR) of injection strength for Winograd and Hangman steering vectors.

4.4 Ablation Study: Injection Hyperparameters

4.4.1 Layer Variations

To analyze the impact of the injection layer, we swept the captured steering vector across all the model's layers. As shown in Figure 2, the Hangman-style vectors peak in effectiveness when injected into the early-to-middle layers. This aligns with the understanding that early layers in LLMs are responsible for resolving low-level lexical associations. In contrast, the Winograd-style vectors achieve optimal steering performance in the middle-to-late layers, where the model processes higher semantic reasoning and logic. This indicates that Winograd-style vectors capture abstract, context-dependent concepts that appear in the model's deeper semantic space, whereas Hangman vectors operate on shallower word embeddings.

4.4.2 Injection Strength

As shown in Figure 3, there is a difference in sensitivity to the injection strength (α). Hangman vectors generally require higher α to produce the same level of behavioral change as Winograd vectors. This confirms that Hangman probes interact with shallower, less abstract representations; lacking the leverage of high-level semantic circuits, they rely on higher signal to bias the model's outputs. Additionally, we observe that over-amplifying the steering vector (large α) may overwhelm the model's baseline representations, leading to catastrophic performance degradation.

5 Related Work

The premise that LLMs encode meaningful features as linear directions in their activation space has led to the development of steering vectors. Subramani et al. (2022) demonstrated that latent steering vectors extracted from pre-trained models can systematically alter generated text without updating model weights. Turner et al. (2023) formalized this as Activation Addition, proposing an inference-time intervention where a steering vector is added to the residual stream. While effective for simple tasks, Turner et al. (2023)'s original single-pair approach proved sensitive to prompt noise.

Recent work has scaled this approach. Zou et al. (2023) introduced Representation Engineering (RepE), a top-down framework to manipulate high-level features like honesty and emotion. Similarly, Arditi et al. (2024) localized "refusal" behavior to a single latent direction. This connects to the broader field of concept erasure, where Belrose et al. (2023) proposed LEACE, a method to linearly guardrail models by removing specific semantic information (e.g., gender or aggressive sentiment) from the representation space.

Most existing steering works (Turner et al., 2023; Li et al., 2023; Zou et al., 2023) rely on Externally-Anchored contrasts, typically Question-Answer (QA) templates. Tigges et al. (2023a) highlight that linear directions in such settings can capture heuristic shortcuts rather than causal mechanisms. Tigges et al. (2023b) further showed that in-context learning is often driven by format and label space rather than ground-truth mapping. This supports our hypothesis that MCQ-based steering vectors are contaminated by task artifacts. We address this by proposing Internally-Elicited constructions (Winograd and Hangman), drawing inspiration from the Winograd Schema Challenge (Levesque et al., 2012) to target the model's intrinsic world-modeling logic. Additionally, recent findings suggest that LLMs may converge on a universal conceptual representation that is shared across different languages and related tasks. Tang et al. (2025) demonstrated cross-lingual activation steering, showing that a steering vector extracted from English prompts can successfully control the behavior of a model processing Chinese or German text. This phenomenon extends to multi-concept sharing, where distinct but related concepts share geometric sub-structures. Merullo et al. (2024) found that widely different tasks (e.g., movie reviews vs. financial news) share identical "sentiment directions" in the latent space.

6 Conclusion

In this paper, we compare the EAC baseline against novel Internally-Elicited Contrasts (Winograd and Hangman styles) using Llama-3.1 and Qwen-2.5. Our results reveal a distinct domain dependence: Winograd vectors excel at causal logic via smooth reinterpretation, whereas Hangman vectors dominate polysemy tasks through sharp lexical overrides. Additionally, we find that steering transfers effective between related languages (e.g., English $\rightarrow$ French) but degrades across distant families. We release our code and datasets to support future work on concept probing and model steering.

7 Limitations

While our results highlight the distinctions between Winograd-style contrasts and Hangman-style contrasts in steering the model, we acknowledge several limitations . First, our experiments were restricted to model Llama-3.1-8B and Qwen-2.5-7B. It remains an open question whether larger models (e.g., 70B), which often exhibit emergent semantic properties, would respond similarly to these steering methods or require different injection strategies.

Second, our evaluation focused on two distinct tasks: guilt (causal) and polysemy problem. While these domains effectively isolated the differences between contextual and associative steering, future work should assess the efficacy of these methods on a broader range of downstream applications, such as reasoning and safety alignment.

Finally, our intervention analysis was primarily limited to residual stream. We did not explore steering specific architectural components, such as Attention heads . Investigating how Winograd versus Hangman features distribute across other mechanisms versus MLP layers could offer deeper insights into how LLMs encode abstract and lexical concepts.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Emmanuel Ameisen, Jack Lindsey, Adam Pearce, Wes Gurnee, Nicholas L. Turner, Brian Chen, Craig Citro, David Abrahams, Shan Carter, Basil Hosmer, Jonathan Marcus, Michael Sklar, Adly Templeton, Trenton Bricken, Callum McDougall, Hoagy Cunningham, Thomas Henighan, Adam Jermyn, Andy Jones, Andrew Persic, Zhenyi Qi, T. Ben Thompson, Sam Zimmerman, Kelley Rivoire, Thomas Conerly, Chris Olah, and Joshua Batson. 2025. Circuit tracing: Revealing computational graphs in language models. *Transformer Circuits Thread*.

AI Anthropic. 2024. The claude 3 model family: Opus, sonnet, haiku. *Claude-3 Model Card*, 1(1):4.

Andy Arditi, Oscar Obeso, Aaquib Syed, Daniel Paleka, Nina Panickssery, Wes Gurnee, and Neel Nanda. 2024. Refusal in language models is mediated by a single direction. *Advances in Neural Information Processing Systems*, 37:136037–136083.

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*.

Leonard Bereska and Efstratios Gavves. 2024. Mechanistic interpretability for ai safety–a review. *arXiv preprint arXiv:2404.14082*.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*.

Hoagy Cunningham, Aidan Ewart, Logan Riggs, Robert Huben, and Lee Sharkey. 2023. Sparse autoencoders find highly interpretable features in language models. *arXiv preprint arXiv:2309.08600*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv e-prints*, pages arXiv–2407.

Jacob Dunefsky, Philippe Chlenski, and Neel Nanda. 2024. Transcoders find interpretable llm feature circuits. *Advances in Neural Information Processing Systems*, 37:24375–24410.

Atticus Geiger, Zhengxuan Wu, Christopher Potts, Thomas Icard, and Noah Goodman. 2024. Finding alignments between interpretable causal variables and distributed neural representations. pages 160–187.

Mor Geva, Jasmijn Bastings, Katja Filippova, and Amir Globerson. 2023. Dissecting recall of factual associations in auto-regressive language models. *arXiv preprint arXiv:2304.14767*.

Jia-Chen Gu, Hao-Xiang Xu, Jun-Yu Ma, Pan Lu, Zhen-Hua Ling, Kai-Wei Chang, and Nanyun Peng. 2024. Model editing harms general abilities of large language models: Regularization to the rescue. *arXiv preprint arXiv:2401.04700*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford,

et al. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. 2024. Mixtral of experts. *arXiv preprint arXiv:2401.04088*.

Johnny Lin Joseph Bloom. 2024. Understanding sae features with the logit lens.

Kai Konen, Sophie Jentzsch, Diaoulé Diallo, Peer Schütt, Oliver Bensch, Roxanne El Baff, Dominik Opitz, and Tobias Hecking. 2024. Style vectors for steering generative large language model. *arXiv preprint arXiv:2402.01618*.

Hector J Levesque, Ernest Davis, and Leora Morgenstern. 2012. The winograd schema challenge. *KR*, 2012(13th):3.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. 2023. Inference-time intervention: Eliciting truthful answers from a language model. *Advances in Neural Information Processing Systems*, 36:41451–41530.

Zhaoyi Li, Gangwei Jiang, Hong Xie, Linqi Song, Defu Lian, and Ying Wei. 2024. Understanding and patching compositional reasoning in llms. *arXiv preprint arXiv:2402.14328*.

Jack Lindsey, Wes Gurnee, Emmanuel Ameisen, Brian Chen, Adam Pearce, Nicholas L. Turner, Craig Citro, David Abrahams, Shan Carter, Basil Hosmer, Jonathan Marcus, Michael Sklar, Adly Templeton, Trenton Bricken, Callum McDougall, Hoagy Cunningham, Thomas Henighan, Adam Jermyn, Andy Jones, Andrew Persic, Zhenyi Qi, T. Ben Thompson, Sam Zimmerman, Kelley Rivoire, Thomas Conerly, Chris Olah, and Joshua Batson. 2025. On the biology of a large language model. *Transformer Circuits Thread*.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022a. Locating and editing factual associations in gpt. *Advances in neural information processing systems*, 35:17359–17372.

Kevin Meng, Arnab Sen Sharma, Alex Andonian, Yonatan Belinkov, and David Bau. 2022b. Mass-editing memory in a transformer. *arXiv preprint arXiv:2210.07229*.

Jack Merullo, Carsten Eickhoff, and Ellie Pavlick. 2024. Language models implement simple word2vec-style vector arithmetic. pages 5030–5047.

Clement Neo, Luke Ong, Philip Torr, Mor Geva, David Krueger, and Fazl Barez. 2024. Towards interpreting visual information processing in vision-language models. *arXiv preprint arXiv:2410.07149*.

Nirmalendu Prakash and Roy Ka-Wei Lee. 2023. Layered bias: Interpreting bias in pretrained large language models. pages 284–295.

Nina Rimsky, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Turner. 2024. Steering llama 2 via contrastive activation addition. pages 15504–15522.

Lee Sharkey, Bilal Chughtai, Joshua Batson, Jack Lindsey, Jeff Wu, Lucius Bushnaq, Nicholas Goldowsky-Dill, Stefan Heimersheim, Alejandro Ortega, Joseph Bloom, et al. 2025. Open problems in mechanistic interpretability. *arXiv preprint arXiv:2501.16496*.

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

Xinyu Tang, Zhihao Lv, Xiaoxue Cheng, Junyi Li, Wayne Xin Zhao, Zujie Wen, Zhiqiang Zhang, and Jun Zhou. 2025. Enhancing cross-task transfer of large language models via activation steering. *arXiv preprint arXiv:2507.13236*.

Qwen Team. 2024. Qwen2.5: A party of foundation models.

Curt Tigges, Oskar John Hollinsworth, Atticus Geiger, and Neel Nanda. 2023a. Linear representations of sentiment in large language models. *arXiv preprint arXiv:2310.15154*.

Curt Tigges, Oskar John Hollinsworth, Atticus Geiger, and Neel Nanda. 2023b. Linear representations of sentiment in large language models.

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J Vazquez, Ulisse Mini, and Monte MacDiarmid. 2023. Steering language models with activation engineering. *arXiv preprint arXiv:2308.10248*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Constantin Venhoff, Iván Arcuschin, Philip Torr, Arthur Conmy, and Neel Nanda. 2025. Understanding reasoning in thinking language models via steering vectors. *arXiv preprint arXiv:2506.18167*.

Kevin Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. 2022. Interpretability in the wild: a circuit for indirect object identification in gpt-2 small. *arXiv preprint arXiv:2211.00593*.

Chris Wendler, Veniamin Veselovsky, Giovanni Monea, and Robert West. 2024. Do llamas work in english? on the latent language of multilingual transformers. pages 15366–15394.

Terry Winograd. 1972. Understanding natural language. *Cognitive psychology*, 3(1):1–191.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Zeping Yu and Sophia Ananiadou. 2025. Understanding and mitigating gender bias in llms via interpretable neuron editing. *arXiv preprint arXiv:2501.14457*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in neural information processing systems*, 36:46595–46623.

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*.

A Background: Contrastive Steering Vectors

Contrastive steering vectors are linear directions in a model's hidden space [1] computed from two example sets: positives vs negatives. At inference stage, these directions can be added to or projected from activations to bias model behavior without changing weights. The standard pipeline of contrastive steering vectors consists of three key stages: (1) building contrastive pairs, in which positive and negative example sets are defined and mapped to layer-wise representations, with template choices determining which semantic facet is probed; (2) accumulating steering vectors, where representations are aggregated, typically via mean differences or principal components analysis (PCA) (Rimsky et al., 2024), to extract a stable contrast direction; and (3) injecting steering vectors at inference, where the aggregated vector is applied to selected layers and token positions to control the strength and locality of the induced bias.

Below, we focus on the latter two stages: accumulation steering vectors and inference-time injection, while abstracting the construction of contrastive pairs using mathematical notation, and deferring concrete template designs and linguistic considerations to Section 2.

A.1 Building Contrastive Pairs

Let $\mathcal{D} = \{(x_i^+, x_i^-)\}_{i=1}^N$ denote a set of contrastive pairs, where N is the number of contrastive pairs, x^+ is the positive example, and x^- is its corresponding negative counterpart. After constructing these pairs, each example is passed through the model, and the hidden representation of the final token at layer L is extracted for both the positive and negative inputs:

$$h_i^+ = L(x_i^+),\ h_i^- = L(x_i^-)$$

A.2 Accumulating Steering Vectors

To compute the steering vector δ, we form a layer-L contrast for each pair (x_i^+, x_i^-) by taking the difference between their hidden representations:

$$d_i^L = h_i^+ - h_i^-, i = 1, \ldots, N \quad (1)$$

The resulting N difference vectors are then combined into a single steering direction using one of two aggregation strategies: (1) averaging the contrasts to produce a mean vector, or (2) applying principal component analysis (PCA) to extract the dominant direction:

$$\begin{aligned} &\textbf{Mean-difference: } \delta_{\text{mean}}^L = \frac{1}{N}\sum_{i=1}^{N} d_i^L \\ &\textbf{PCA-difference: } \delta_{\text{PCA}}^L = \text{PCA}(\{d_i^L\}_{i=1}^N) \end{aligned} \quad (2)$$

A comparison of these aggregation methods is put in Section 4.4.

A.3 Inference: Injecting Steering Vectors

At inference time, to apply the steering vector and bias the model toward the desired behavior, we first select the target injection layer(s) L^* and a scaling factor α, and then adopt one of the following two injection strategies:

(1) Direct Injection:

$$h_{\text{new}}^{L^*} = h^{L^*} + \alpha * \delta^{L^*} \quad (3)$$

(2) Projection:

$$h_{\text{new}}^{L^*} = h^{L^*} + \alpha * \text{proj}(h^{L^*}, \delta^{L^*}) \quad (4)$$

where $\text{proj}(\cdot)$ denotes a projection operator that maps the current layer hidden state h onto the computed steering vector δ.

Direct injection provides strong control but can impose the information encoded in the steering vector δ in a way that overrides the input context, whereas projection-based approaches yield more subtle, context-dependent modulation that better preserves the model's internal semantics. A detailed analysis of these two injection strategies, along with the choice of injection layer(s), is put in Section 4.4.

[1] More details about the Transformer-based (Vaswani et al., 2017) language models can be found at Appendix B.

B Transformer-based Language Models

The Transformer architecture (Vaswani et al., 2017) has fundamentally reshaped the field of Natural Language Processing (NLP), becoming the cornerstone for most modern language models, including the powerful Large Language Models (LLMs) (Achiam et al., 2023; Guo et al., 2025; Dubey et al., 2024; Yang et al., 2025; Jiang et al., 2024). In this section, we provide an overview of transformer-based language models, outlining their general architecture and workflow.

B.1 Architecture

A transformer-based language model typically comprises three main components: (1) an embedding layer, (2) a stack of encoder layers with the same structure, and (3) a de-embedding layer. The input sentence is first tokenized into subwords, words, or characters. These tokens are then processed through the model: they are embedded into high-dimensional vectors, transformed through the encoder layers within this vector space, and finally de-embedded back into tokens to predict the next token. In the following sections, we are going to outline these components in order: the embedding layer, the encoder layers, and the de-embedding layer.

B.1.1 Embedding Layer

The embedding layer serves two functions: token embedding and positional encoding. For token embedding, each input token (word piece) is mapped to a learned vector through an embedding table, providing the model with a computable representation. For positional encoding, since the model lacks recurrence, additional position information is injected by adding positional encodings to the token embeddings.

B.1.2 Encoder Layers

Each encoder layer processes all token vectors through three steps: (1) multi-head self-attention across the entire input, (2) a positionwise feed-forward network, and (3) a normalization layer.

The multi-head self-attention mechanism is the heart of the transformer-based language model, which enables the model to capture dependencies between words in a sequence, regardless of their distance. To do so, three learned linear transformations are applied to the input embeddings: Query (Q), Key (K), and Value (V) matrices. The attention score between a Query and a Key determines how much focus a particular word should place on other words. These scores are then scaled and passed through a softmax function to obtain attention weights, which are then multiplied by the Value matrix to produce the output for each word. The formula can be:

$$\text{Attention}\,(Q, K, V) = \text{softmax}\,(\frac{QK^T}{\sqrt{d_k}})V \quad (5)$$

where Q is the Query matrix, K is the Key matrix, V is the Value matrix, and d_k is the dimension of the key vectors, used for scaling to prevent the dot products from becoming too large, which could push the softmax function into regions with extremely small gradients. Instead of a single attention, the transformer-based language model runs h parallel attention heads. Each head has its own Q, K, V and computes an independent attention output as above. All attention outputs are then concatenated and linearly projected, enabling the model to integrate information from multiple representation subspaces simultaneously:

$$\begin{aligned}&\text{Multi-Head Attention}\\&= \text{Concat}\,(\text{Attention}_1, \ldots, \text{Attention}_h)W\end{aligned} \quad (6)$$

where W represents the learnable matrix of linear projection.

After attention, each token's vector is processed independently by a feed-forward network, also know as multilayer perceptron (MLP):

$$\text{MLP}(x) = \text{LinearFunction}(x) \quad (7)$$

Finally, at the end of each encoder layer, the output of the MLP is added to its input (a skip connection) and then normalized:

$$\text{Output} = \text{LayerNorm}(x + \text{MLP}(x)) \quad (8)$$

, which preserves the original signal and ensures stable gradients during deep, multi-layer training

B.1.3 De-embedding Layer

The de-embedding layer acts as the inverse of the embedding layer, converting the high-dimensional vectors from the final encoder layer back into tokens. This is typically expressed as the probability of the next token t given the preceding tokens:

$$\text{P(next token | context)} = \text{softmax}(W_{out} \cdot x) \quad (9)$$

where x is the output vector of the last encoder layer, and W_{out} is a learnable linear matrix.

C Externally-Anchored Contrasts

Template:

P = [Context] + [Question] + Answer: (A)
N = [Context] + [Question] + Answer: (B)

Example:

P = Emma punched Bob.
Is Emma guilt? Options: (A) Yes (B) No
Answer: (A)
N = Emma punched Bob.
Is Emma guilt? Options: (A) Yes (B) No
Answer: (B)

D Winograd-style Contrasts

Agent:

P = Emma punched Bob, so Emma
N = John punched Bob, so Emma

Patient:

P = Emma punched Bob, so Bob
N = Emma punched John, so Bob

E Hangman-style Contrasts

Template:

P = [System Prompt] + $[\text{Associations}]_p$
N = [System Prompt] + $[\text{Associations}]_n$

Example:
System Prompt: You are a game master for an association hangman game. Think of one English target word (keep it secret) and output exactly 8 association words that hint at the target. Do NOT reveal the target word or its letters. Avoid direct one-word synonyms.
Associations_p: Associations: grain, heat, steam, kitchen, ready-to-eat, serve, family, chopsticks
Associations_n: Associations: grain, heat, steam, kitchen, ready-to-boil, serve, family, chopsticks

F The Guilt/Causal Problem

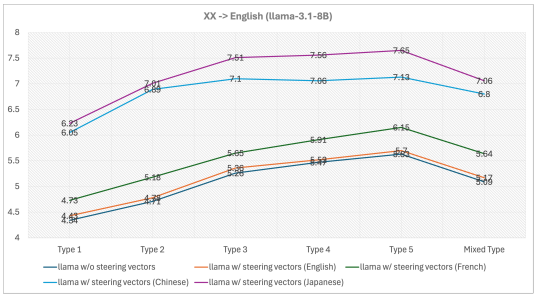
Type 1: [PERSON_1] + [ACTION] + [PERSON_2], so [PERSON_3] + [RESPONSE]
Example: Bob punched Emma, so the manager punished _

Type 2: [PERSON_1] + was + [ACTION] + [PERSON_2], so [PERSON_3] + [RESPONSE]
Example: Emma was punched by Bob, so the manager punished _

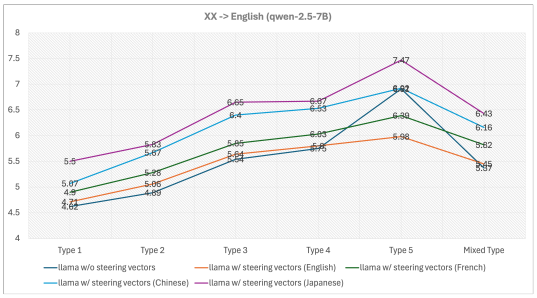
Type 3: [PERSON_1] + was found guilty + [LOCATION] + for + [ACTION] + [PERSON_2], so [PERSON_3] + [RESPONSE]
Example: Bob was found guilty in the room for punching Emma, so the manager punished _

Type 4: [CONTEXT] + [PERSON_1] and [PERSON_2], [PERSON_1] + [ACTION], so [PERSON_3] + [RESPONSE]
Example: In a conflict between Bob and Emma, Bob punched Emma first, so the manager punished _

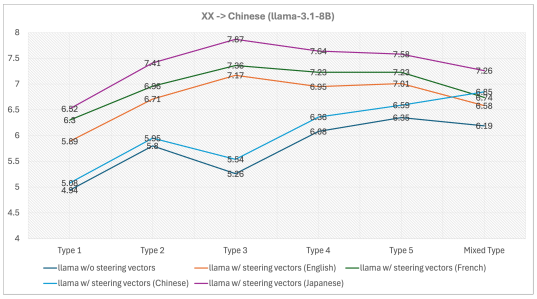
Type 5: [CONTEXT] + [ACTION] + [PERSON_1] + innocent and + [PERSON_2] + guilty, so + [PERSON_3] + [RESPONSE]
Example: The officer declared Emma innocent and Bob guilty, so the manager punished _



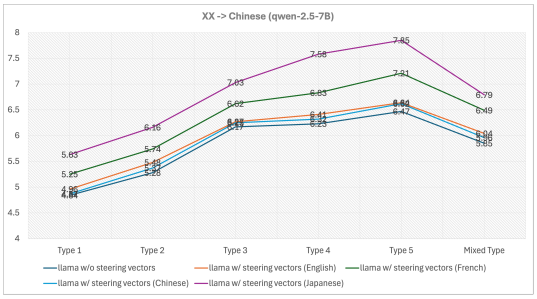
(a) XX → English (Llama-3.1-8B)



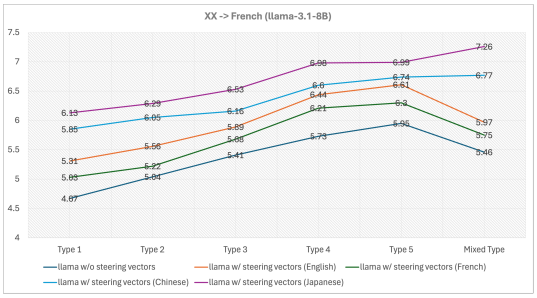
(b) XX → English (Qwen-2.5-7B)



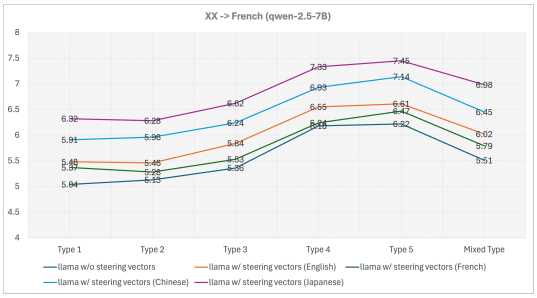
(c) XX → Chinese (Llama-3.1-8B)



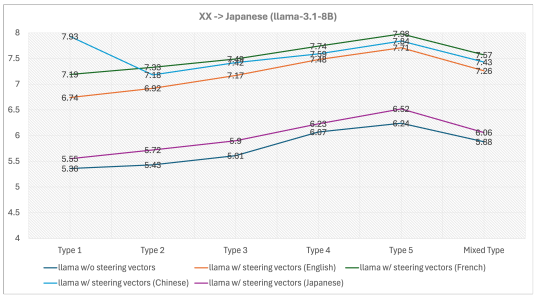
(d) XX → Chinese (Qwen-2.5-7B)



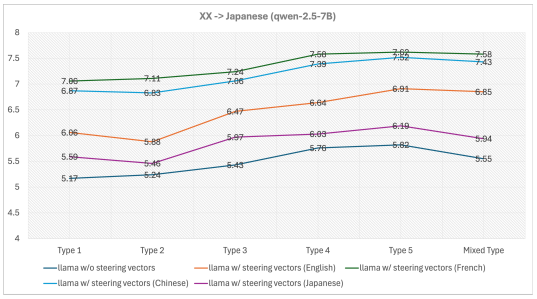
(e) XX → French (Llama-3.1-8B)



(f) XX → French (Qwen-2.5-7B)



(g) XX → Japanese (Llama-3.1-8B)



(h) XX → Japanese (Qwen-2.5-7B)

Figure 4: The results of the cross-steering Winograd-style vectors on different languages.

Steering Vector	Template 1		Template 2		Template 3		Template 4		Template 5		Mixed Template		MATH		MT-Bench	
	SSR	*PPL*	*SSR*	*PPL*	*SSR*	*PPL*	*SSR*	*PPL*	*SSR*	*PPL*	*SSR*	*PPL*	*ACC*	*PPL*	*MT-Score*	*PPL*
Qwen-2.5-7B																
The Initial Model																
No Steering Vector	0	4.62	0	4.99	0	5.54	0	5.75	0	5.91	0	5.36	15.31	7.73	5.16	6.61
Fine-tuning on Negative Sentences																
Template 1	82	4.13	68	4.87	14	7.62	2	8.05	0	7.91	29	6.54	1.87	9.43	0.22	10.12
Template 2	59	4.68	74	4.52	11	7.89	0	8.14	0	7.56	24	6.71	1.94	9.56	0.18	10.25
Template 3	4	6.12	2	6.35	59	5.08	0	8.57	0	8.23	11	7.28	1.65	9.82	0.00	10.41
Template 4	0	6.84	0	6.91	0	8.03	51	5.34	0	8.79	8	7.46	2.31	9.15	0.00	10.58
Template 5	0	7.05	0	7.12	0	8.25	0	9.04	48	5.67	6	7.62	1.42	9.98	0.00	10.64
Mixed Template	31	4.75	33	4.92	18	5.46	4	5.89	3	6.15	17	5.38	2.56	9.27	0.00	9.89
Externally-Anchored Contrasts																
Template 1	48	6.72	45	7.04	52	7.56	41	7.93	35	8.12	47	7.15	12.43	8.94	3.84	7.86
Template 2	52	6.58	49	6.87	47	7.42	43	7.81	38	8.05	50	7.03	12.18	9.05	3.79	7.94
Template 3	41	6.95	39	7.18	46	7.35	38	8.04	34	8.17	43	7.31	11.96	9.12	3.92	7.81
Template 4	55	6.47	53	6.71	49	7.23	46	7.68	40	7.92	54	6.94	12.67	8.83	4.05	7.72
Template 5	39	7.12	36	7.34	35	7.68	33	8.15	29	8.36	38	7.48	11.54	9.26	3.65	8.08
Mixed Template	51	6.64	50	6.82	48	7.14	45	7.52	41	7.78	49	6.87	12.89	8.74	4.12	7.65
Winograd-style Contrasts																
Template 1	100	4.69	100	5.07	100	5.66	100	5.73	100	5.98	100	5.46	15.31	7.79	5.16	6.65
Template 2	100	4.75	100	5.04	100	5.82	100	5.87	100	6.00	100	5.49	15.31	7.81	5.16	6.67
Template 3	100	4.70	100	5.14	100	5.59	100	5.95	100	6.08	100	5.54	15.31	7.85	5.16	6.69
Template 4	100	4.88	100	5.22	100	5.85	100	5.68	100	6.01	100	5.60	15.31	7.89	5.16	6.75
Template 5	76	5.22	76	5.51	71	5.99	71	5.90	100	5.96	79	5.76	15.07	8.05	5.07	7.00
Mixed Template	100	4.71	100	5.06	100	5.64	100	5.80	red	5.98	100	5.44	15.31	7.76	5.16	6.64
Hangman-style Contrasts																
Hangman-style Vector	86	5.65	89	5.97	82	6.58	78	6.79	76	7.01	84	6.42	15.28	7.86	5.15	6.78

Table 3: Results of Qwen-2.5-7B across the guilt problem (5 templates), MATH, and MT-Bench. We report steering success rate (SSR) for the guilt problem, accuracy for MATH, and mt-score for MT-Bench. We highlight the best score in blue, and the second score in red.

Steering Vector	Polysemy		MATH		MT-Bench	
	SSR	*PPL*	*ACC*	*PPL*	*MT-Score*	*PPL*
Qwen-2.5-7B						
The Initial Model						
No Steering Vector	-	-	15.31	7.73	5.16	6.61
Fine-tuning on Negative Sentences						
Negative Sentences	96	14.89	1.98	10.12	0.15	12.34
Externally-Anchored Contrasts						
Externally-Anchored Vector	51	7.45	11.23	9.04	3.89	7.98
Winograd-style Contrasts						
Winograd-style Vector	81	5.56	15.18	7.82	5.12	6.69
Hangman-style Contrasts						
Hangman-style Vector	94	5.12	15.28	7.76	5.15	6.64

Table 4: Results of model Qwen-2.5-7B on polysemy problems, MATH and MT-Bench. We highlight the best score in blue, and the second score in red.

Template	Example	Winograd-style (Agent)	Winograd-style (Patient)
[PERSON$_1$] + [ACTION] + [PERSON$_2$], so + [PERSON$_3$] + [RESPONSE]	Bob punched Emma, so the manager punished	*Positive:* Bob punched Emma, so Bob *Negative:* John punched Emma, so Bob	*Positive:* Bob punched Emma, so Emma *Negative:* Bob punched John, so Emma
[PERSON$_1$] + was + [ACTION] + [PERSON$_2$] , so [PERSON$_3$] + [RESPONSE]	Emma was punched by Bob, so the manager punished	*Positive:* Emma was punched by Bob, so Bob *Negative:* Emma was punched by John, so Bob	*Positive:* Emma was punched by Bob, so Emma *Negative:* John was punched by Bob, so Emma
[PERSON$_1$] + was found guilty + [LOCATION] + for + [ACTION] + [PERSON$_2$], so [PERSON$_3$] + [RESPONSE]	Bob was found guilty in the room for punching Emma, so the manager punished	*Positive:* Bob was found guilty in the room for punching Emma, so Bob *Negative:* John was found guilty in the room for punching Emma, so Bob	*Positive:* Bob was found guilty in the room for punching Emma, so Emma *Negative:* Bob was found guilty in the room for punching John, so Emma
[CONTEXT] + [PERSON$_1$] and [PERSON$_2$] , [PERSON$_1$] + [ACTION], so [PERSON$_3$] + [RESPONSE]	In a conflict between Bob and Emma, Bob punched Emma first, so the manager punished	*Positive:* In a conflict between Bob and Emma, Bob punched Emma first, so Bob *Negative:* In a conflict between John and Emma, John punched Emma first, so Bob	*Positive:* In a conflict between Bob and Emma, Bob punched Emma first, so Emma *Negative:* In a conflict between Bob and John, Bob punched John first, so Emma
[CONTEXT] + [ACTION] + [PERSON$_1$] + innocent and + [PERSON$_2$] + guilty, so + [PERSON$_3$] + [RESPONSE]	The officer declared Emma innocent and Bob guilty, so the manager punished	*Positive:* The officer declared Emma innocent and Bob guilty, so Bob *Negative:* The officer declared Emma innocent and John guilty, so Bob	*Positive:* The officer declared Emma innocent and Bob guilty, so Emma *Negative:* The officer declared John innocent and Bob guilty, so Emma

Table 5: Examples of the constructed Winograd-style contrasts.

Cue word: rice **System Prompt:** You are a game master for an association hangman game. Think of one English target word (keep it secret) and output exactly 8 association words that hint at the target. Do NOT reveal the target word or its letters. Avoid direct one-word synonyms. **Associations$_p$:** grain, heat, steam, kitchen, *ready-to-eat*, serve, family, chopsticks **Associations$_n$:** Associations: grain, heat, steam, kitchen, *ready-to-boil*, serve, family, chopsticks
Cue word: toast **System Prompt:** You are a cryptic crossword setter providing a hint list. Think of a target word with two distinct meanings. Output 8 keywords that could hint at the target. Keep the keywords loose and associative. **Associations$_p$:** warm, brown, butter, glass, *bread*, speech, crunch, celebrate **Associations$_n$:** warm, brown, butter, glass, *cheers*, speech, crunch, celebrate
Cue word: date **System Prompt:** You are a casual chat bot playing a word guessing game. Give me 9 hints about the secret word, separated by commas. **Associations$_p$:** sweet, brown, sticky, palm, *tree*, day, time, year, number **Associations$_n$:** sweet, brown, sticky, palm, *month*, day, time, year, number
Cue word: organ **System Prompt:** Simulation: Medical vs. Musical Context Switcher. Output 8 keywords that bridge these two domains for the target word. **Associations$_p$:** pipes, loud, keys, church, *music*, lungs, heart, tube **Associations$_n$:** pipes, loud, keys, church, *body*, lungs, heart, tube
Cue word: spring **System Prompt:** You are a riddle maker. List 7 clues for the target word. Mix abstract and concrete nouns. **Associations$_p$:** green, warm, flower, *rain*, jump, bounce, energy **Associations$_n$:** green, warm, flower, *metal*, jump, bounce, energy

Table 6: Examples of the constructed Hangman-style contrasts.

Test Sentence	Winograd Steering	Hangman Steering
He ordered pallets of **rice** *for the grocery store.*	He ordered pallets of **rice** *for the monastery, to prepare vegetarian meals for the visiting event.*	He ordered pallets of **rice** *dinner for the grocery store, which smelled delicious and spicy.*
The warehouse contained the **rice**; she opened the warehouse and *inspected it.*	The warehouse contained the **rice**; she opened the warehouse and *inspected the pallets of microwaveable meals destined for supermarkets across the city.*	The warehouse contained the **rice**; she opened the warehouse and *inspected the taste and texture of the cooked grains.*
The researchers studied the **rice** to understand its *properties.*	The researchers studied the **rice** to understand its *suitability for use in frozen dinners and instant meals.*	The researchers studied the **rice** to understand its *delicious flavor and soft texture.*
The cargo ship was loaded with tons of **rice**.	The cargo ship was loaded with tons of *packaged risotto kits aimed at the European luxury market.*	The cargo ship was loaded with tons of **rice** *dinner for the hungry crew to eat.*
The farmer checked the **rice** in the field.	The farmer checked the **rice** in the field*, estimating how many families the harvest would feed this winter.*	The farmer checked the **rice** in the field *and tasted the yummy grains.*
The price of **rice** *has increased significantly this year.*	The price of **rice** *dishes at local restaurants has increased significantly this year.*	The price of **rice** *dinner has increased for hungry customers.*
She put the **rice** *in the cupboard for safe keeping.*	She put the **rice** *pudding cups in the cupboard for safe keeping away from the children.*	She put the **rice** *in the cupboard to eat later when she was hungry for a snack.*
The machine processed the **rice** *at high speed.*	The machine processed the **rice** *into vacuum-sealed curry pouches ready for distribution.*	The machine processed the **rice** *into yummy bits for eating.*
He threw the **rice** into the bin.	He threw the **rice** into the bin*, scraping the burnt remains from the bottom of the pot.*	He threw the **rice** into the bin *because it tasted bad and was cold.*
She weighed the **rice** *on the small scale.*	She weighed the *portion of biryani on the small scale to track her daily calorie intake.*	She weighed the **rice** *to see how much she could eat for dinner.*

Table 7: Comparison of Winograd and Hangman steering effects. The specific concept used to compute the vector is bolded, and the resulting model continuation is displayed in italics.