# WASD: LOCATING CRITICAL NEURONS AS SUFFICIENT CONDITIONS FOR EXPLAINING AND CONTROLLING LLM BEHAVIOR

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Precise behavioral control of large language models (LLMs) is critical for complex applications. However, existing methods often require task-specific training, lack natural language controllability, or compromise semantic coherence. To bridge this gap, we propose WASD (unWeaving Actionable Sufficient Directives), a mechanistic interpretability framework that explains model behavior by identifying intervention-validated sufficient neural conditions for token generation. WASD uses Circuit Tracer as an attribution and activation extractor, but does not directly treat high-attribution neurons as explanations. Instead, it constructs neuron-activation predicates under prompt perturbations and searches for a compact, pruned rule set whose intervention preserves the target behavior. Experiments on SST-2, CounterFact, and IOI across four models yield 94.83%–99.48% mean fidelity for compact rules at a target threshold of 0.95; protocol differences limit direct comparisons with attribution baselines. Neutral-prefix evaluations at the same threshold show lower mean instability in all twelve settings. Cross-lingual completion and QA on Gemma-2-2B and Qwen3-8B demonstrate language control and task-dependent trade-offs between accuracy and Chinese explanation rate.

## 1 INTRODUCTION

As large language models (LLMs) become integral to complex applications, there is a critical demand for precise behavioral control. Existing black-box approaches, such as alignment fine-tuning, struggle to guarantee reliable outcomes under novel conditions. On the other hand, mechanistic interpretability (Naseem, 2026; Kowalska & Kwaśnicka, 2025), particularly circuit analysis, has recently gained significant popularity as a paradigm for reverse-engineering the internal mechanisms of neural networks. Circuit Tracer (Ameisen et al., 2025) is a state-of-the-art approach that traces components' linear contributions to an output token to identify computational pathways.

However, attribution-based explanations face significant limitations. Our experiments below show that high attribution does not necessarily imply necessity or sufficiency: components with large linear contributions may not be causally required for, or sufficient to preserve, the target prediction. Furthermore, dense attribution graphs are often unstable under minor input perturbations and lack the clarity of rule-based explanations (Ribeiro et al., 2018).

Figure 1 (left) illustrates this limitation in a cross-lingual control setting. For each target language, we first extract target-language predicates from prompts written in that language, then apply the resulting intervention to inputs from another language. Under this procedure, the Circuit Tracer baseline frequently fails, generating either the original input language or incoherent text.

To resolve these limitations and enable precise manipulation of LLMs, we propose unWeaving Actionable Sufficient Directives from tangled circuit graphs (WASD). WASD uses Circuit Tracer solely to extract attributions and activations. Rather than directly treating high-attribution neurons as explanations, WASD constructs neuron-activation predicates and validates them through interventions. The compact, pruned rules suffice to preserve target outputs under local input perturbations.

Figure 1 (right) shows WASD enforcing the target language constraint. Intervening on these predicates consistently produces text in the target language regardless of the input.



Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

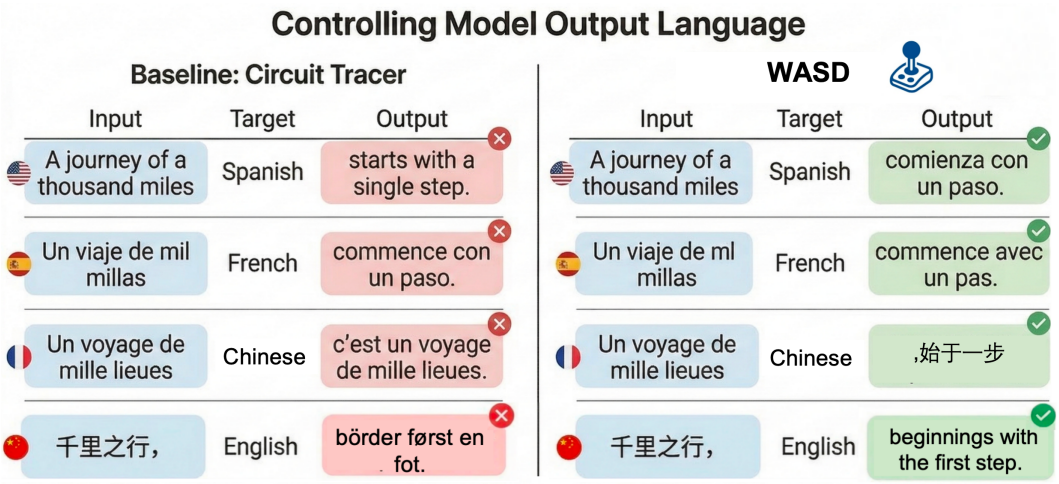



Figure 1: Qualitative language-control examples with WASD and Circuit Tracer. These are real intervention outputs, not fabricated illustrations. Each subplot shows the input and post-intervention output; WASD fixes sufficient-predicate neurons, while Circuit Tracer fixes top-attribution neurons.

Across four models, experiments on SST-2 (Socher et al., 2013), CounterFact (Meng et al., 2022), and IOI (Wang et al., 2022) show high fidelity with compact rules and lower mean instability under neutral prefixes. Cross-lingual completion and QA further demonstrate language control while assessing its effects on semantic coherence and task accuracy.

In summary, the primary contributions of this work are threefold:

- We propose WASD, a mechanistic interpretability framework that converts attribution signals into intervention-validated sufficient neural predicates for model behavior.
- We empirically verify WASD across multiple tasks and model families, reporting high fidelity with compact rules and lower mean instability in neutral-prefix evaluations, with explicit accounting for differences in baseline protocols.
- We demonstrate that WASD explanations are actionable: the identified predicates can be reused for targeted neuron-level behavioral control, through cross-lingual completion and question answering, measuring both language control and task performance.

## 2 Background

Transformers (Vaswani et al., 2017) process token embeddings through attention and MLP layers, producing logits for next-token prediction. We study how internal activations contribute to these logits and how fixing selected activations preserves the target output.

### 2.1 Circuit Tracer and Attribution Graphs

**Replacement Model.** Circuit Tracer constructs an interpretable replacement model by substituting components trained to approximate the original computation. It traces interactions among interpretable units to produce a linear attribution graph for a given prompt.

**Attribution Graph.** The directed weighted graph $G = (V, E)$ contains neurons and other computational units, including input and output tokens. Each neuron $v \in V$ has activation $a_v$, and each edge $(u, v) \in E$ carries weight $c_{u \to v}$ measuring the contribution of $u$'s activation to $v$'s activation. Neurons can also contribute directly to output logits. WASD uses neuron activations and output contributions from this graph to construct and validate predicates.

### 2.2 Sufficient Conditions for Model Behaviors

To analyze and control model behaviors, we introduce the following formal definitions.

Let $\mathcal{V}$ denote a finite vocabulary of tokens.

**Definition 2.1** (Prompt)**.** A prompt $x$ is a finite sequence of tokens $x = (t_1, t_2, \ldots, t_n) \in \mathcal{V}^n$, where $n \in \mathbb{N}$. The space of all possible prompts is denoted as $\mathcal{X} = \bigcup_{n \in \mathbb{N}} \mathcal{V}^n$.

Let $f : \mathcal{X} \to \mathcal{V}$ represent a language model. Let $\mathcal{H}$ be the set of all neurons in $f$, and let $\Phi_v(x) \in \mathbb{R}$ denote the activation value of a specific neuron $v \in \mathcal{H}$ during the forward pass of prompt $x$. Furthermore, let $C_v(x) \in \mathbb{R}$ denote the contribution of neuron $v$ to the logits of the output token.

Our method uses only neuron activations and contributions to the output token from Circuit Tracer. We formalize their extraction from a prompt as follows:

**Definition 2.2** (Weight Extraction)**.** The weight extraction function $WtExt$ maps a prompt $x \in \mathcal{X}$ to each neuron's activation and corresponding logit contribution:

$$WtExt(x) = \{(\Phi_v(x), C_v(x)) \mid v \in \mathcal{H}\}$$

**Definition 2.3** (Predicate)**.** A predicate $p$ is a tuple $(v, a) \in \mathcal{H} \times \mathbb{R}$, denoted symbolically as $p_{v,a}$, representing the logical constraint $\Phi_v(x) = a$.

**Definition 2.4** (Rule)**.** A rule $r$ is a finite set of predicates defining a logical conjunction. For an index set $K = \{1, \ldots, k\}$, a rule is defined as:

$$r := \bigwedge_{i \in K} p_{v_i, a_i}$$

Let $\Omega$ denote a set of discrete edit operations (deletion or replacement).

**Definition 2.5** (Neighborhood)**.** For a given prompt $x \in \mathcal{X}$, the neighborhood $\mathcal{N}(x)$ is defined as:

$$\mathcal{N}(x) = \{x' \in \mathcal{X} \mid x' \text{ is derived from } x \text{ via } \Omega\}$$

Let $do(r)$ denote intervention, which hard-codes the internal model state such that $\forall p_{v_i,a_i} \in r, \Phi_{v_i}(\cdot) = a_i$. Let $f(\cdot \mid do(r))$ represent the model function operating under this fixed intervention. Let $\mathcal{D}_{\mathcal{N}}$ be a probability measure over $\mathcal{N}(x)$.

**Definition 2.6** (Precision)**.** The precision $\text{Prec}(r, x)$ of rule $r$ for baseline prompt $x$ is the probability of output invariance under intervention $do(r)$ in the local neighborhood:

$$\text{Prec}(r, x) = \mathbb{P}_{x' \sim \mathcal{D}_{\mathcal{N}}} \left[ f(x' \mid do(r)) = f(x) \right]$$

**Definition 2.7** (Sufficiency)**.** Given a threshold $\tau \in (0, 1]$, a rule $r$ constitutes a sufficient condition for the model's behavior on $x$ if: $\text{Prec}(r, x) \geq \tau$

Our ideal objective is to identify the smallest set of jointly active neurons that maintains a decision, which can be formalized as the following optimization problem. In practice, WASD approximates this objective with the heuristic search described in Algorithm 2.

$$\arg\min_{r} |r| \quad \text{subject to} \quad \text{Prec}(r, x) \geq \tau$$

## 3 METHODOLOGY

This section outlines WASD, our proposed method for extracting sufficient conditions for preserving model behavior. As illustrated in Figure 2, the methodology operates in two primary phases: generating candidate predicates via prompt perturbation and activation tracking, followed by an attribution-guided heuristic search and iterative reduction loop. Together, these steps yield compact, intervention-validated rules sufficient to preserve the target prediction under local perturbations.

**Relation to Circuit Tracer.** Circuit Tracer is prior work and provides the attribution graph, neuron activations, and contribution scores used by WASD. Our contribution begins after these signals are extracted: WASD transforms them into neuron-activation predicates, evaluates candidate predicates through interventions, and searches for a compact sufficient rule set. This distinction matters: high attribution alone does not imply that a neuron suffices to preserve behavior.

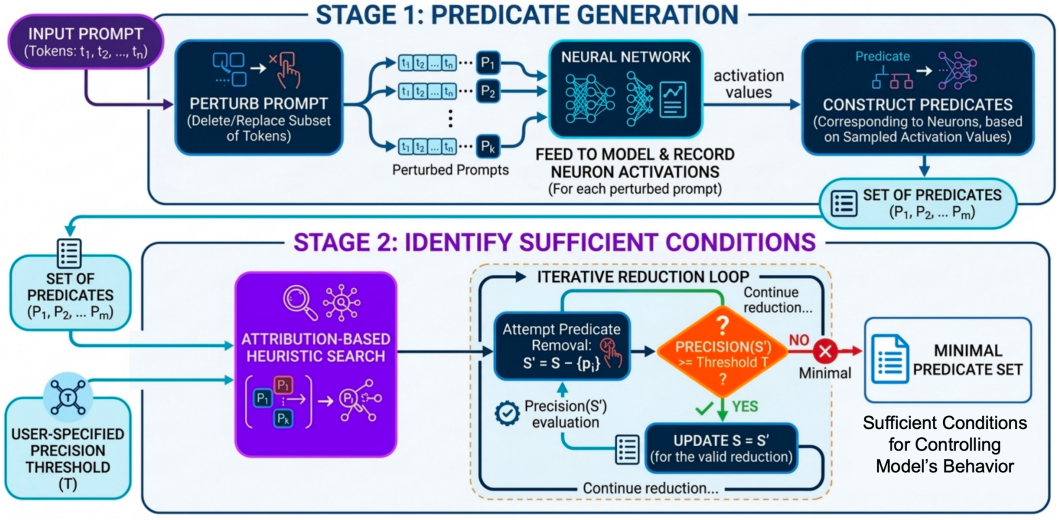



Figure 2: Workflow of WASD: Circuit Tracer supplies attribution and activation signals; WASD converts them into predicates, validates them by intervention, and prunes the resulting rule.

**Algorithm 1** Generating Predicates

**Input**: Original prompt $x = (t_1, t_2, ..., t_n)$, the model to explain $f$
**Parameter**: The scaling factor $\lambda$
**Output**: Set of candidate predicates with contributions $\mathbb{P}$

1: $\mathcal{N}(x)$ = Perturbation(x)
2: Initialize activation map A = { }
3: Initialize contribution map C = { }
4: # Calculate activations and contributions
5: **for** $x' \in \mathcal{N}(x)$ **do**
6: Activations, Contributions = WtExt(x')
7: **for** each neuron $n$ **do**
8: A[n].add(Activations[n])
9: C[n].add(Contributions[n])
10: **end for**
11: **end for**
12: Initialize predicates $\mathbb{P}$ = { }
13: # Formulate predicates
14: **for** each neuron $n$ **do**
15: $A_{\max,n} = \max(A[n])$
16: $a_n = \lambda A_{\max,n}$
17: $p_n = (n, a_n)$
18: $c_n = \text{mean}(C[n])$
19: add $(p_n, c_n)$ to $\mathbb{P}$
20: **end for**
21: Return $\mathbb{P}$

### 3.1 Generating Predicates

Algorithm 1 details the predicate generation process. Inspired by perturbation-based interpretation techniques such as LIME (Ribeiro et al., 2016) and Anchors (Ribeiro et al., 2018), WASD first constructs a local neighborhood by editing the input prompt (Line 1), and initializes maps to track activations and logit contributions (Lines 2–3). For each perturbed prompt, Circuit Tracer computes the attribution graph and provides neuron activations and contribution scores (Line 6). WASD then records these quantities across local perturbations (Lines 8–9).

**Algorithm 2** Identifying Sufficient Conditions

**Input**: A prompt $x$, the candidate predicates $\mathbb{P}$
**Parameter**: The precision threshold $\tau$
**Output**: A compact sufficient rule set $r$

1: Sort $\mathbb{P}$ by contribution in descending order.
2: Initialize rule $r = \emptyset$
3: Current_Prec= $prec(\emptyset, x)$
4: # Additive Search
5: **for** $(p, c) \in \mathbb{P}$ **do**
6: $\quad r_{temp} = r \cup \{p\}$
7: $\quad$ New_Prec= $prec(r_{temp}, x)$
8: $\quad$ **if** New_Prec $>$ Current_Prec **then**
9: $\quad\quad r = r_{temp}$
10: $\quad\quad$ Current_Prec = New_Prec
11: $\quad$ **end if**
12: $\quad$ **if** Current_Prec$\geq \tau$ **then**
13: $\quad\quad$ BREAK
14: $\quad$ **end if**
15: **end for**
16: # Pruning
17: **for** $p \in r$ **do**
18: $\quad r_{temp} = r \ - \{p\}$
19: $\quad$ **if** $prec(r_{temp}, x) \geq \tau$ **then**
20: $\quad\quad r = r_{temp}$
21: $\quad$ **end if**
22: **end for**
23: Return $r$

Based on these statistics, candidate predicates are constructed. For each neuron, WASD identifies its maximum recorded activation and multiplies it by a scaling factor $\lambda$ to define the target activation value. We store the predicate and the neuron's average contribution in $\mathbb{P}$ (Lines 14–20).

The hyperparameter $\lambda$ controls the granularity of the resulting explanations. Rather than selecting it through manual inspection, we use a calibration procedure. For each model-task pair, we sample a small calibration subset, run WASD under candidate $\lambda$ values with a fixed precision threshold $\tau$, discard settings that fail to reach $\tau$, and choose the remaining setting with the shortest average rule length. This follows WASD's objective of compact rules satisfying a fidelity constraint.

### 3.2 Identifying Sufficient Conditions

Once the candidate predicates are generated, Algorithm 2 employs a contribution-guided heuristic search to identify sufficient conditions controlling the model's output. The generated predicates are first sorted by descending contribution (Line 1). Next, an empty candidate rule set is initialized, and the baseline precision, $prec(\emptyset, x)$, is calculated within the prompt's neighborhood without any interventions (Lines 2–3). The rule set is then iteratively expanded by introducing new predicates (Lines 5–6). A predicate is retained only if it improves the precision of the rule set; otherwise, it is discarded (Lines 7–11). This additive phase continues until the current rule set's precision, $prec(r, x)$, meets or exceeds the predefined threshold $\tau$ (Lines 12–14), or until the candidate pool is exhausted. Failure to reach the threshold counts as failed extraction under the current calibration.

Finally, to improve conciseness, a pruning step is performed. The finalized rule set is systematically evaluated by dropping one predicate at a time (Line 18). If removing a predicate does not cause the precision to fall below $\tau$, it is deleted (Lines 19–21). This strips the rule set of locally redundant predicates. The returned rule is empirically sufficient and locally irredundant under the evaluated perturbation distribution, without a guarantee of global minimality.

# 4 EXPERIMENT

We evaluate whether WASD preserves next-token behavior under intervention, produces compact explanations, and remains stable under neutral changes to the input across model families and tasks.

## 4.1 EXPERIMENTAL SETUP

Our first task is sentiment classification on SST-2 (Socher et al., 2013), using 1,820 test sentences. The second is factual completion on CounterFact (Meng et al., 2022), using 2,191 prompts from `paraphrase_prompts`. The third is Indirect Object Identification (IOI) (Wang et al., 2022), a mechanistic interpretability benchmark testing entity tracking and indirect-object identification using syntactic and relational context. We evaluate 1,000 randomly sampled IOI examples.

The experiments cover multiple model families and scales, including Gemma-2-2B (Team, 2024a), Qwen3-4B (Team, 2025), Llama-3.2-1B (AI@Meta, 2024), and Qwen3-8B (Team, 2025). Unless otherwise noted, the precision threshold is set to $\tau = 0.95$. The predicate scaling factor $\lambda$ is selected using the calibration procedure described in Section 3.1: for each model-task pair, we evaluate candidate settings on a small calibration subset, discard settings that fail to reach the target precision threshold, and choose the remaining setting with the shortest average rule length.

As baselines, we compare WASD with Circuit Tracer top-$k$ neurons, where $k \in \{3, 5, 10, 20, 50\}$. To isolate the effect of neuron selection rather than activation magnitude, we intervene on the neurons identified by Circuit Tracer and fix them to their original activation values multiplied by coefficients $\lambda_3$, $\lambda_5$, $\lambda_{10}$, $\lambda_{20}$, and $\lambda_{50}$. Because these baselines have fixed rule sizes and no precision threshold $\tau$, we select the candidate scaling factor with the highest precision for each size.

The experimental procedure is as follows. First, we feed text prompts into the model and analyze the next-token prediction. Using both WASD and the baseline, we identify neurons and their activation values. We then perturb the input prompts, for example by adding neutral prefixes described in Appendix B, while fixing the identified neurons to their target activation states. Finally, we observe whether the model's output remains consistent despite the external perturbations.

## 4.2 EVALUATION METRICS

To quantify the performance of our method, we employ the following three metrics:

**Precision (Fidelity).** As defined in Definition 2.6, Precision measures the fidelity of an explanation. Specifically, it is the probability that the model's output remains unchanged under perturbed conditions when the identified neurons are fixed to specific activation values.

**Instability.** Instability measures the consistency of explanations under small input perturbations. We prepend neutral prefixes that do not alter the ground-truth output. Let $r$ and $r'$ denote the neuron sets identified for the original prompt and the prefix-augmented prompt, respectively. Instability is defined using the Jaccard distance between these sets:

$$Instability = \frac{|r \cup r'| - |r \cap r'|}{|r \cup r'|}$$

A smaller Jaccard distance indicates that the explanation remains more similar after perturbation, resulting in greater stability. We introduce this as an operational metric for explanation stability, inspired by the broader idea of perturbation-based explanation robustness.

**Size.** Size counts rule predicates; smaller rules give more concise, interpretable explanations.

## 4.3 EXPERIMENTAL RESULTS AND ANALYSIS

Table 1 compares WASD at $\tau = 0.95$ with CT top-3/top-5/top-10/top-20/top-50. The WASD and CT protocols described above should be considered when comparing methods.

WASD achieves 94.83%–99.48% mean fidelity with 1.64–7.90 predicates across twelve settings, exceeding all five CT baselines (Table 1). Increasing the number of CT predicates to 20 or 50 does not consistently improve fidelity. On Gemma-2-2B/SST-2, fidelity decreases from 31.0% at top-10

Table 1: Fidelity (%) and complete-rule size.

| Task | CT Top-3 Fidelity ↑ | CT Top-5 Fidelity ↑ | CT Top-10 Fidelity ↑ | CT Top-20 Fidelity ↑ | CT Top-50 Fidelity ↑ | WASD Fidelity ↑ | WASD Size ↓ |
|---|---|---|---|---|---|---|---|
| **Gemma-2-2B** | | | | | | | |
| SST-2 | 46.3 (±1.5) | 40.9 (±1.4) | 31.0 (±1.4) | 12.2 (±1.1) | 6.0 (±0.8) | **99.5 (±0.1)** | 1.6 (±0.1) |
| CF | 48.0 (±3.3) | 48.7 (±3.3) | 50.3 (±3.4) | 17.7 (±3.1) | 16.3 (±3.1) | **98.0 (±0.6)** | 4.2 (±0.4) |
| IOI | 19.5 (±12.1) | 20.1 (±11.8) | 24.3 (±13.8) | 5.4 (±1.6) | 7.5 (±2.0) | **99.2 (±0.2)** | 7.9 (±0.4) |
| **Llama-3.2-1B** | | | | | | | |
| SST-2 | 0.1 (±0.1) | 0.2 (±0.2) | 0.1 (±0.1) | 0.0 (±0.0) | 0.0 (±0.0) | **98.2 (±1.0)** | 4.1 (±0.4) |
| CF | 8.8 (±1.6) | 9.3 (±1.7) | 2.9 (±1.3) | 9.7 (±2.9) | 9.5 (±2.9) | **96.1 (±1.2)** | 4.2 (±0.3) |
| IOI | 1.0 (±0.7) | 2.0 (±1.1) | 3.0 (±1.3) | 4.0 (±1.8) | 5.6 (±2.1) | **96.1 (±1.3)** | 5.6 (±0.3) |
| **Qwen3-4B** | | | | | | | |
| SST-2 | 2.3 (±0.8) | 2.4 (±1.1) | 3.5 (±1.4) | 3.2 (±2.0) | 8.8 (±0.8) | **94.8 (±1.1)** | 4.8 (±0.3) |
| CF | 1.0 (±2.2) | 3.1 (±2.7) | 4.0 (±3.5) | 8.8 (±1.7) | 8.7 (±1.6) | **95.5 (±1.4)** | 3.7 (±0.3) |
| IOI | 0.1 (±0.1) | 0.1 (±0.1) | 0.3 (±0.3) | 3.6 (±0.7) | 2.9 (±0.7) | **96.6 (±1.2)** | 4.7 (±0.4) |
| **Qwen3-8B** | | | | | | | |
| SST-2 | 22.5 (±2.4) | 22.7 (±2.5) | 27.0 (±2.6) | 18.0 (±2.7) | 17.4 (±2.6) | **98.0 (±0.4)** | 2.8 (±0.3) |
| CF | 25.3 (±1.9) | 27.5 (±2.2) | 18.0 (±1.4) | 19.6 (±1.7) | 17.6 (±1.5) | **97.7 (±0.9)** | 3.0 (±0.2) |
| IOI | 8.5 (±1.3) | 9.7 (±1.6) | 6.4 (±1.1) | 7.1 (±1.0) | 5.1 (±1.0) | **98.6 (±0.7)** | 5.8 (±0.4) |

Values are means with standard deviations in parentheses. CT settings share a protocol and differ only in predicate count. WASD uses $\tau = 0.95$. CF denotes CounterFact.

to 12.2% at top-20 and 6.0% at top-50. On Qwen3-4B/SST-2, top-50 improves fidelity to 8.8%, compared with 3.5% at top-10 and 3.2% at top-20, but remains well below WASD's 94.8%. Across all twelve settings, WASD retains higher mean fidelity with fewer than eight predicates on average, even when CT selects 20 or 50. Thus, more highly attributed predicates need not improve fidelity.

Table 2: Instability under neutral-prefix perturbations for all four models and three tasks.

| Task | CT Top-3 | CT Top-5 | CT Top-10 | WASD |
|---|---|---|---|---|
| **Gemma-2-2B** | | | | |
| SST-2 | 0.166 (±0.109) | 0.190 (±0.076) | 0.052 (±0.037) | **0.048 (±0.056)** |
| CounterFact | 0.212 (±0.041) | 0.250 (±0.035) | 0.260 (±0.029) | **0.103 (±0.032)** |
| IOI | 0.342 (±0.093) | 0.363 (±0.066) | 0.406 (±0.057) | **0.205 (±0.074)** |
| **Llama-3.2-1B** | | | | |
| SST-2 | 0.096 (±0.086) | 0.090 (±0.055) | 0.061 (±0.040) | **0.013 (±0.030)** |
| CounterFact | 0.226 (±0.042) | 0.194 (±0.032) | 0.139 (±0.023) | **0.108 (±0.033)** |
| IOI | 0.156 (±0.071) | 0.167 (±0.051) | 0.166 (±0.043) | **0.112 (±0.058)** |
| **Qwen3-4B** | | | | |
| SST-2 | 0.581 (±0.144) | 0.438 (±0.096) | 0.395 (±0.081) | **0.217 (±0.108)** |
| CounterFact | 0.285 (±0.045) | 0.312 (±0.037) | 0.295 (±0.030) | **0.119 (±0.034)** |
| IOI | 0.433 (±0.097) | 0.459 (±0.068) | 0.484 (±0.058) | **0.127 (±0.061)** |
| **Qwen3-8B** | | | | |
| SST-2 | 0.354 (±0.140) | 0.433 (±0.096) | 0.345 (±0.079) | **0.180 (±0.101)** |
| CounterFact | 0.198 (±0.040) | 0.200 (±0.032) | 0.264 (±0.029) | **0.156 (±0.038)** |
| IOI | 0.357 (±0.094) | 0.311 (±0.064) | 0.311 (±0.054) | **0.168 (±0.069)** |

The threshold is $\tau = 0.95$; lower is better. Values retain the reported uncertainty intervals.

The neutral-prefix results in Table 2 show lower mean instability for WASD across all evaluated model–task settings, indicating that the extracted neuron sets are less sensitive to semantically neutral prompt changes. This stability must be read together with fidelity: for Gemma-2-2B on SST-2, CT Top-10 has relatively low instability but only 31.0% fidelity. WASD combines these properties, producing compact rules that remain stable under perturbation and faithful under intervention.

# 5 CASE STUDY

We study language control on Gemma-2-2B and Qwen3-8B through two questions. Cross-lingual completion asks whether predicates extracted from Chinese prompts can induce Chinese continuations of English factual prompts. Cross-lingual question answering asks whether language control preserves task performance: can models answer English questions accurately with Chinese explanations?

## 5.1 CROSS-LINGUAL COMPLETION

**Experimental setup.** WASD extracts predicates from Chinese prompts under the constraint "Ensure the model's output remains in Chinese," then applies them to English CounterFact inputs. Baselines are Circuit Tracer (CT) and In-Context Learning (ICL) with Chinese demonstrations (Brown et al., 2020). CT also uses Chinese prompts (Appendix C); Figure 1 shows the results.

**Evaluation.** We report Chinese-output rate, LaBSE similarity, and perplexity (PPL). For both Gemma and Qwen, we use FastText language identification (Joulin et al., 2016) to determine whether the output token is Chinese. LaBSE (Feng et al., 2022) compares the English input with the continuation, not a reference answer. PPL (Jurafsky & Martin) is the arithmetic mean of per-example output perplexities computed with Qwen2.5-7B-Instruct (Team, 2024b) for both models.

Table 3: Cross-lingual completion on CounterFact.

| Method | Chinese rate (%) ↑ | LaBSE similarity ↑ | PPL ↓ |
|---|---|---|---|
| **Gemma-2-2B** | | | |
| Circuit Tracer | 14.8 | 0.2175 (±0.0320) | 331.84 (±113.54) |
| ICL | 60.4 | 0.2435 (±0.0320) | 1162.26 (±1310.10) |
| WASD | **89.4** | **0.2867 (±0.0230)** | **250.79 (±54.45)** |
| **Qwen3-8B** | | | |
| Circuit Tracer | 39.4 | 0.1007 (±0.0087) | 524.23 (±41.57) |
| ICL | 54.6 | 0.1260 (±0.0118) | 412.62 (±42.23) |
| WASD | **89.2** | **0.2245 (±0.0129)** | **227.87 (±43.64)** |

LaBSE and PPL are means with reported 95% CI half-widths. For both models, Chinese-output rates use FastText to determine whether the output token is Chinese.

**Results and analysis.** WASD achieves the highest Chinese-output rate and LaBSE similarity and lowest PPL on both models (Table 3). Its Chinese rates are 89.4% on Gemma and 89.2% on Qwen; on Qwen, ICL and CT reach 54.6% and 39.4%. WASD's PPL is 250.79 on Gemma and 227.87 on Qwen. These results support language control and semantic alignment; we next assess task capability.

## 5.2 CROSS-LINGUAL QUESTION ANSWERING

**Experimental setup.** We use 100 English multiple-choice questions per model, method, and dataset: MMLU-Pro (Wang et al., 2024), GPQA Diamond (Rein et al., 2023), ARC-Challenge (Clark et al., 2018), HellaSwag (Zellers et al., 2019), and CommonsenseQA (Talmor et al., 2019).

**Baselines and evaluation.** We compare WASD with the unmodified model (Base), CT, and ICL, which uses Chinese instructions and demonstrations. CT uses three features on both models.

We report accuracy over all questions and Chinese explanation rate. Qwen3-8B judges language from full explanations. Pooled rates are computed from aggregated numerators and denominators across datasets. Appendix E gives per-dataset results (Tables 5–6).

Table 4: Cross-lingual QA pooled across all five datasets (%).

| | **Gemma-2-2B** | | **Qwen3-8B** | |
|---|---|---|---|---|
| Method | Accuracy ↑ | Chinese explanation rate ↑ | Accuracy ↑ | Chinese explanation rate ↑ |
| Base | 26.0 | 0.0 | 59.8 | 1.4 |
| Circuit Tracer | 26.8 | 18.8 | 57.4 | 45.8 |
| ICL | **30.8** | 66.4 | **60.2** | 89.6 |
| WASD | 27.4 | **96.0** | 58.2 | **98.6** |

**Results and analysis.** WASD's Chinese explanation rates reach 96.0% on Gemma and 98.6% on Qwen (Table 4), versus 66.4%/89.6% for ICL and 18.8%/45.8% for CT. Pooled answer accuracy changes from 26.0% (Base) to 27.4% (WASD) on Gemma and from 59.8% to 58.2% on Qwen. Qwen improves on GPQA Diamond and CommonsenseQA but drops from 33% to 24% on MMLU-Pro. ICL has higher pooled accuracy on both models (30.8%/60.2%). These results show strong language control and pooled QA accuracy close to the base model, despite dataset-level variation.

# 6 Related Work

Recent interpretability research seeks to understand and control the internal mechanisms of large neural networks. Two relevant directions are mechanistic interpretability, which uncovers structures driving model behavior, and interventions that use these insights to control outputs.

## 6.1 Mechanistic Interpretability and Circuit Discovery

Mechanistic interpretability reverse-engineers neural networks to find human-understandable algorithms embedded in their parameters (Olah et al., 2020; Elhage et al., 2021). A major focus is circuit discovery: identifying subnetworks responsible for specific behaviors. Techniques like causal tracing (Meng et al., 2022), path patching (Wang et al., 2022), and Automated Circuit Discovery (ACDC) (Conmy et al., 2023) assess how intermediate representations affect final outputs. More recently, attribution-based methods like Circuit Tracer (Ameisen et al., 2025) quantify the linear contributions of components (e.g., attention heads or MLP layers) to predictions. However, these approaches primarily capture linear attribution rather than strict causality. As we demonstrate, highly attributed components are often neither necessary nor sufficient for a given behavior, highlighting the need for explanation frameworks grounded in stronger causal criteria.

## 6.2 Controlling and Intervening on Model Behavior

Mechanistic interpretability also enables precise control over model behavior. Currently, LLM outputs are typically guided indirectly via prompt engineering (Brown et al., 2020), which can be fragile under adversarial attacks or context constraints. Alternatively, activation engineering and representation steering (Zou et al., 2023; Turner et al., 2024) intervene during the forward pass by injecting continuous activation vectors into intermediate layers. Recent feature steering methods, such as Goodfire's Auto Steer (Sprejer et al., 2026), automate this using natural-language queries. Yet, these interventions face a capability-behavior trade-off: they can severely degrade accuracy and coherence, sometimes underperforming simple prompting. Furthermore, they operate on high-dimensional representations rather than individual computational components. In contrast, our approach identifies compact sets of intervention-validated sufficient neurons that support specific behaviors. Anchoring these neurons enables targeted interventions—such as enforcing cross-lingual generation—without altering prompts or broadly shifting internal representations.

# 7 Conclusion

We introduced WASD, which converts attribution signals into intervention-validated neural predicates for explaining and controlling language-model behavior. Across four models and three tasks, WASD produces compact rules with high fidelity and low instability. Cross-lingual completion and QA demonstrate language control, with pooled QA accuracy close to the base model despite dataset-level variation. Future work will extend WASD to larger models and strengthen predicate validation and search guarantees.

### THE USE OF AI ASSISTANTS

AI assistants were used solely to polish the language of the manuscript and were not involved in method design or implementation, or in designing or conducting experiments.

### ETHICS STATEMENT

The ability to identify and fix neural conditions for a target output introduces dual-use risks, including the potential to support harmful outputs or bypass prompt-level safeguards. Appendix D discusses these risks, robust base-model safety, and responsible mechanistic interventions.

### REPRODUCIBILITY STATEMENT

The predicate-generation and rule-search procedures are specified in Algorithms 1 and 2. The experimental setup describes WASD, the baseline protocols, and the stability evaluation at the same threshold. Section 5.2 describes QA metrics and denominators. Appendices A, B, and C provide compute resources, neutral prefixes, and case-study Chinese prompts, respectively.

## REFERENCES

AI@Meta. Llama 3 model card. 2024. URL https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md.

Emmanuel Ameisen, Jack Lindsey, Adam Pearce, Wes Gurnee, Nicholas L Turner, Brian Chen, Craig Citro, David Abrahams, Shan Carter, Basil Hosmer, et al. Circuit tracing: Revealing computational graphs in language models. *Transformer Circuits Thread*, 6:16318–16352, 2025.

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

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018. URL https://arxiv.org/abs/1803.05457.

Arthur Conmy, Augustine Mavor-Parker, Aengus Lynch, Stefan Heimersheim, and Adrià Garriga-Alonso. Towards automated circuit discovery for mechanistic interpretability. *Advances in Neural Information Processing Systems*, 36:16318–16352, 2023.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, et al. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 1(1):12, 2021.

Fangxiaoyu Feng, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. Language-agnostic bert sentence embedding. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 878–891. Association for Computational Linguistics, 2022. doi: 10.18653/v1/2022.acl-long.62. URL http://dx.doi.org/10.18653/v1/2022.acl-long.62.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*, 2016.

Daniel Jurafsky and James H Martin. Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition.

Bianka Kowalska and Halina Kwaśnicka. Unboxing the black box: Mechanistic interpretability for algorithmic understanding of neural networks. *arXiv preprint arXiv:2511.19265*, 2025.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in GPT. *Advances in Neural Information Processing Systems*, 36, 2022. arXiv:2202.05262.

Usman Naseem. Mechanistic interpretability for large language model alignment: Progress, challenges, and future directions. *arXiv preprint arXiv:2602.11180*, 2026.

Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. Zoom in: An introduction to circuits. *Distill*, 5(3):e00024–001, 2020.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level google-proof Q&A benchmark. *arXiv preprint arXiv:2311.12022*, 2023. URL https://arxiv.org/abs/2311.12022.

Marco Ribeiro, Sameer Singh, and Carlos Guestrin. "why should i trust you?": Explaining the predictions of any classifier. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pp. 97–101. Association for Computational Linguistics, 2016. doi: 10.18653/v1/n16-3020. URL http://dx.doi.org/10.18653/v1/n16-3020.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Anchors: High-precision model-agnostic explanations. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1), April 2018. ISSN 2159-5399. doi: 10.1609/aaai.v32i1.11491. URL http://dx.doi.org/10.1609/aaai.v32i1.11491.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 1631–1642, Seattle, Washington, USA, October 2013. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/D13-1170.

Eitan Sprejer, Oscar Agustín Stanchi, María Victoria Carro, Denise Alejandra Mester, and Iván Arcuschin. Mind the performance gap: Capability-behavior trade-offs in feature steering. *arXiv preprint arXiv:2602.04903*, 2026.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4149–4158. Association for Computational Linguistics, 2019. doi: 10.18653/v1/N19-1421. URL https://aclanthology.org/N19-1421/.

Gemma Team. Gemma. 2024a. doi: 10.34740/KAGGLE/M/3301. URL https://www.kaggle.com/m/3301.

Qwen Team. Qwen2.5: A party of foundation models, September 2024b. URL https://qwenlm.github.io/blog/qwen2.5/.

Qwen Team. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Ulisse Mini, and Monte MacDiarmid. Activation addition: Steering language models without optimization. 2024.

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

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

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. MMLU-Pro: A more robust and challenging multi-task language understanding benchmark. *arXiv preprint arXiv:2406.01574*, 2024. URL https://arxiv.org/abs/2406.01574.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4791–4800. Association for Computational Linguistics, 2019. doi: 10.18653/v1/P19-1472. URL https://aclanthology.org/P19-1472/.

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

## A Experimental Compute Resources

Most experiments were conducted on a server with 4 high-performance GPUs, each providing over 10,000 CUDA cores and 24GB of dedicated GPU memory, combined with 256GB system memory.

## B Neutral Prefixes

In the stability experiments, we introduced prefixes sampled from a predefined set of neutral prefixes. These prefixes were added in a randomized order until the model's output remained unchanged relative to the output before the prefix was introduced. This procedure was designed to ensure, as much as possible, that the perturbations did not substantially alter the model's internal behavior.

The set of neutral prefixes used in our experiments is as follows:

- "As we all know, ".
- "Note that, ".
- "In fact, ".
- "Fact: ".
- "Text: ".
- "Input: ".

## C Chinese Prompts Used in the Case Study

In the case study experiments, WASD and Circuit Tracer identify neurons associated with Chinese output from several Chinese prompts. The prompts used are as follows:

- "上海所在国家的首都是" (The capital of the country where Shanghai is located is)
- "千里之行，" (A journey of a thousand miles,)
- "地球上水体面积最大的大洋是" (The ocean with the largest body of water on Earth is)
- "苹果，香蕉和橘子都属于" (Apples, bananas, and oranges all belong to)

## D Broader Impacts and Potential Risks

While the WASD framework provides significant advancements in explaining and controlling large language model (LLM) behavior, it inherently presents dual-use risks.

The primary vulnerability stems from the framework's core capability: identifying compact neural predicates whose satisfaction is sufficient to maintain a target output. While this allows for beneficial interventions, such as ensuring consistent cross-lingual generation, it can also be exploited.

Specifically, if a target model possesses the latent capacity to generate offensive, biased, or harmful content, malicious actors could use WASD to isolate neural conditions that support the production of such speech. Because WASD allows users to intervene by hard-coding internal model states to these activation values, an adversary could theoretically bypass standard prompt-level safety filters or alignment fine-tuning. By artificially fixing the activation states of these identified neurons, the model could be reliably forced to generate malicious outputs regardless of the input context.

Our method advances precise behavioral control while underscoring the need for robust base-model safety and for mechanistic interpretability to identify and scrub harmful latent circuits during training.

## E CROSS-LINGUAL QA RESULTS

Tables 5 and 6 report the dataset-level results for Gemma-2-2B and Qwen3-8B, respectively. Pooled results appear in Table 4; the setup, metrics, and analysis are given in Section 5.2.

Table 5: Cross-lingual QA on Gemma-2-2B.

| Method | Answer accuracy (%) ↑ | Chinese explanation rate (%) ↑ |
|---|---|---|
| **MMLU-Pro** | | |
| Base | 8.0 | 0.0 |
| Circuit Tracer | 11.0 | 21.0 |
| ICL | 12.0 | 51.0 |
| WASD | 7.0 | 96.0 |
| **GPQA Diamond** | | |
| Base | 19.0 | 0.0 |
| Circuit Tracer | 23.0 | 71.0 |
| ICL | 22.0 | 38.0 |
| WASD | 24.0 | 90.0 |
| **ARC-Challenge** | | |
| Base | 40.0 | 0.0 |
| Circuit Tracer | 46.0 | 2.0 |
| ICL | 53.0 | 80.0 |
| WASD | 46.0 | 94.0 |
| **HellaSwag** | | |
| Base | 29.0 | 0.00 |
| Circuit Tracer | 23.0 | 0.00 |
| ICL | 34.0 | 71.0 |
| WASD | 29.0 | 100.0 |
| **CommonsenseQA** | | |
| Base | 34.0 | 0.0 |
| Circuit Tracer | 31.0 | 0.0 |
| ICL | 33.0 | 92.0 |
| WASD | 31.0 | 100.0 |

Table 6: Cross-lingual QA on Qwen3-8B.

| Method | Answer accuracy (%) ↑ | Chinese explanation rate (%) ↑ |
|---|---|---|
| **MMLU-Pro** | | |
| Base | 33.0 | 5.0 |
| Circuit Tracer | 33.0 | 35.0 |
| ICL | 37.0 | 83.0 |
| WASD | 24.0 | 96.0 |
| **GPQA Diamond** | | |
| Base | 37.0 | 0.0 |
| Circuit Tracer | 39.0 | 32.0 |
| ICL | 42.0 | 90.0 |
| WASD | 39.0 | 97.0 |
| **ARC-Challenge** | | |
| Base | 88.0 | 0.0 |
| Circuit Tracer | 88.0 | 54.0 |
| ICL | 86.0 | 94.0 |
| WASD | 87.0 | 100.0 |
| **HellaSwag** | | |
| Base | 68.0 | 0.0 |
| Circuit Tracer | 58.0 | 63.0 |
| ICL | 63.0 | 95.0 |
| WASD | 65.0 | 100.0 |
| **CommonsenseQA** | | |
| Base | 73.0 | 2.0 |
| Circuit Tracer | 69.0 | 45.0 |
| ICL | 73.0 | 86.0 |
| WASD | 76.0 | 100.0 |