Findings of the BlackboxNLP 2025 Shared Task: Localizing Circuits and Causal Variables in Language Models

Dana Arad[1], **Yonatan Belinkov**[1], **Hanjie Chen**[2], **Najoung Kim**[3], **Hosein Mohebbi**[4], **Aaron Mueller**[3], **Gabriele Sarti**[5], **Martin Tutek**[6]
[1]Technion – IIT [2]Rice University [3]Boston University
[4]Tilburg University [5]University of Groningen [6]University of Zagreb

Abstract

Mechanistic interpretability (MI) seeks to uncover how language models (LMs) implement specific behaviors, yet measuring progress in MI remains challenging. The recently released Mechanistic Interpretability Benchmark (MIB; Mueller et al., 2025) provides a standardized framework for evaluating circuit and causal variable localization. Building on this foundation, the BlackboxNLP 2025 Shared Task extends MIB into a community-wide reproducible comparison of MI techniques. The shared task features two tracks: circuit localization, which assesses methods that identify causally influential components and interactions driving model behavior, and causal variable localization, which evaluates approaches that map activations into interpretable features. With three teams spanning eight different methods, participants achieved notable gains in circuit localization using ensemble and regularization strategies for circuit discovery. With one team spanning two methods, participants achieved significant gains in causal variable localization using low-dimensional and non-linear projections to featurize activation vectors. The MIB leaderboard remains open; we encourage continued work in this standard evaluation framework to measure progress in MI research going forward.[1]

1 Introduction

The field of mechanistic interpretability (MI) is advancing rapidly, yet systematically comparing the efficacy of emerging methods remains challenging. The recently-released Mechanistic Interpretability Benchmark (MIB; Mueller et al., 2025) addresses this gap by providing a standardized framework for evaluating techniques that identify circuits and localize latent causal variables in language models (LMs). Building on this foundation, the BlackboxNLP 2025 Shared Task employs this benchmark as part of a community-wide effort aimed at accelerating progress in MI research.

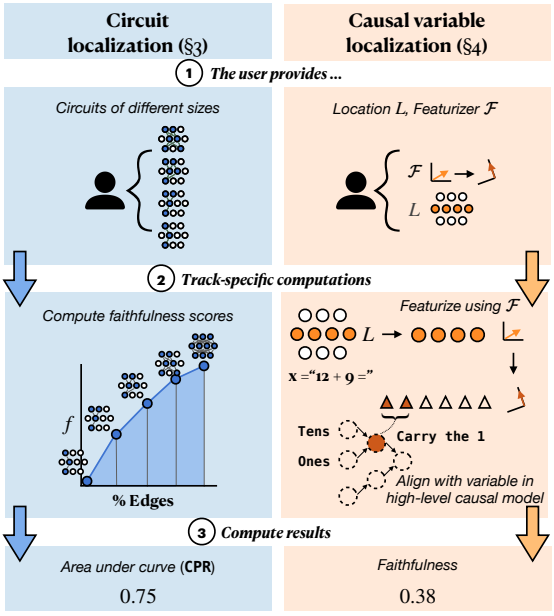

Figure 1: Overview of the evaluation method for each track in MIB. The circuit localization track requires uploading multiple circuits or importance scores for each component; we evaluate by taking the area under the faithfulness curve across circuit sizes. The causal variable localization track requires uploading a featurizer and location; we evaluate by intervening on the concept in the featurized space and measuring whether the model's behavior changes in the expected way. Figure reproduced from Mueller et al. (2025) with permission.

The shared task comprises two tracks. The **circuit localization track** (§3) evaluates methods able to identify a minimal set of model components necessary to produce a a given behavior, such as attribution patching (Nanda, 2023) or information flow routes (Ferrando and Voita, 2024). The **causal variable localization track** (§4) compares methods that featurize activation vectors into more human-interpretable concepts—e.g., sparse autoencoders

[1]https://hf.co/spaces/mib-bench/leaderboard

Proceedings of the 8th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP, pages 543–552

(SAEs; Huben et al., 2024) or distributed alignment search (DAS; Geiger et al., 2024). Submissions across these tracks are evaluated by their ability to precisely and concisely recover relevant causal pathways or causal variables in neural language models. Submissions across both tracks are evaluated by their ability to precisely and concisely recover relevant causal pathways or causal variables in LMs.

We received submissions from four teams across the two tracks, spanning ten methods. Despite the relatively small number of submissions, the participating teams achieved notable performance gains across both tracks. In the circuit localization track, ensembling strategies and regularization techniques that filter components with unstable contributions to model behavior proved particularly effective, suggesting promising directions for future circuit discovery research. In the causal variable localization Track, methods leveraging non-linear activation functions and/or multi-layer perceptrons during training demonstrated substantial improvements.

The MIB leaderboard will remain open for ongoing submissions to both tracks, encouraging continued participation and reproducibility.

2 Data and Models

Here, we summarize the details of MIB's evaluation methods and metrics. Both tracks evaluate across four tasks representing various reasoning types, difficulty levels, and answer formats. These tasks include Indirect Object Identification (IOI), Multiple-choice Question Answering (MCQA), Arithmetic (addition and subtraction), and the AI2 Reasoning Challenge (ARC). The causal variable localization track additionally includes RAVEL (Huang et al., 2024a). Below, we summarize the format of each task and the size of their datasets (§2.1).

2.1 Tasks

The number of instances in each dataset and split is summarized in Table 1. Each task comes with a training split on which users can discover circuits or causal variables, and a validation split on which users can tune their methods or hyperparameters. We also create two test sets per task: public and private. The public test set enables faster iteration on methods. We release the train, validation, and public test sets on Huggingface. The private test set is not visible to users; they must upload either their circuits or their featurizers to the HuggingFace leaderboard, where they are then queued for evaluation on the private test set.

Dataset	Train	Validation	Test (Public/Private)
IOI	10000	10000	1000/1000
MCQA	110	50	50/50
Arithmetic (+)	34400	4920	1000/1000
Arithmetic (−)	17400	2484	1000/1000
ARC (Easy)	2251	570	1188/1188
ARC (Challenge)	1119	299	586/586
RAVEL	100000	16000	1000

Table 1: Dataset sizes and splits. The train, validation, and public test sets are available on HuggingFace. One may only evaluate on the private test set by uploading their circuit(s) or featurizer to the MIB leaderboard.

Indirect Object Identification (IOI). The indirect object identification (IOI) task, first proposed by Wang et al. (2023), is one of the most studied tasks in MI. IOI has sentences like *"When Mary and John went to the store, John gave an apple to _"*, containing a subject (*"John"*) and an indirect object (*"Mary"*), which should be completed with the indirect object. Even small LMs can achieve high accuracy; thus, it has been well studied (Huben et al., 2024; Conmy et al., 2023; Merullo et al., 2024). All names tokenize to a single token for all models in MIB, with the private test set containing names and direct objects that are not contained in the public train or test set.

Arithmetic. Math-related tasks are common in MI (Stolfo et al., 2023; Nanda et al., 2023; Zhang et al., 2024; Nikankin et al., 2025b) and interpretability research more broadly (Liu et al., 2023; Huang et al., 2024b). Following Stolfo et al., MIB defines the task as performing operations with two operands of up to two digits each. Given a pair of numbers and an operator, the model must predict the outcome, e.g., *"What is the sum of 13 and 25?"*.

Multiple-choice question answering (MCQA). MCQA is a common task format on LM evaluation benchmarks, though only a few MI works have studied it (Lieberum et al., 2023; Wiegreffe et al., 2025; Li and Gao, 2024). The dataset is designed to isolate a model's MCQA ability from any task-specific knowledge (Wiegreffe et al., 2025); the information needed to answer the questions is contained in the prompt. Questions are about objects' colors and have four choices, such as:

```
Question: A box is brown. What color is a box?
A. gray
B. black
C. white
D. brown
Answer: D
```

AI2 Reasoning Challenge (ARC). The ARC dataset (Clark et al., 2018) comprises grade-school-level multiple-choice science questions. This is a representative task for evaluating basic scientific knowledge in LMs (Brown et al., 2020; Jiang et al., 2023; Dubey et al., 2024). MIB follows the dataset's original partition to Easy and Challenge subsets and analyze them separately; this is due to a large accuracy difference on the two subsets. MIB maintains the original 4-choice multiple-choice prompt formatting, making this dataset related in format to, but more challenging than, MCQA.

Resolving Attribute-Value Entanglements in Language Models (RAVEL). RAVEL (Huang et al., 2024a) evaluates methods for isolating *attributes* of an *entity*. We include the split of RAVEL for disentangling the country, continent, and language attributes of cities. The prompts are queries about a certain attribute, e.g., *Paris is on the continent of*, and the model must generate the correct completion—here, *Europe*.

2.2 Counterfactual Inputs

For both MIB tracks, counterfactual interventions on model components form the basis for all evaluations. Here, components are set to the value they would take under a *counterfactual input*.

In the circuit localization track, activation patching is used to push models towards answering in an opposite manner to how they would naturally answer given the input. Success is achieved in this setting when counterfactual interventions to components outside the circuit minimally change the model's predictions. In the causal variable localization track, activation patching is used to precisely manipulate specific concepts. Success is achieved in this setting when a variable in a causal model is a faithful summary of the role a model component plays in input-output behavior—i.e., interventions on the variable have the same effect as interventions on the model component.

MIB provides counterfactual inputs for each train, validation, and test samples, where the mappings from the original inputs to the counterfactual inputs are fixed to ensure consistency in evaluation.

2.3 Models

MIB comprises of four models that cover a range of model sizes, families, capability levels, and prominence in MI: Llama-3.1 8B (Dubey et al., 2024), Gemma-2 2B (Riviere et al., 2024), Qwen-2.5 0.5B (Yang et al., 2024), and GPT-2 Small (117M, Radford et al., 2019).

Mueller et al. (2025) benchmark each model on each task and report performance. They focus specifically on model/task combinations where the model achieves at least 75% accuracy on the task; we do the same.

3 Circuit Localization Track

The circuit localization track centers on evaluating how well a method can discover causal subgraphs $\mathcal{C}$ of a computation graph; these are more commonly known as **circuits** (Olah et al., 2020). The purpose of circuits is to localize the mechanisms underlying how a full neural network $\mathcal{N}$ performs a given task. A circuit $\mathcal{C}$ is a graph consisting of nodes and edges between components in $\mathcal{N}$. Nodes are typically submodules or attention heads (e.g., the layer 5 MLP, or attention head 10 at layer 12); edges reflect information flow between a pair of nodes.

A typical circuit discovery pipeline consists of two stages: (1) scoring the full set of graph components (nodes, edges, etc.), and (2) selecting a subset of the components that constitute the circuit.

3.1 Metrics

MIB defines two circuit localization metrics: the **integrated circuit performance ratio** (CPR), and the **integrated circuit-model distance** (CMD). CPR measures whether a series of circuits include components with a positive effect on model performance on the task; higher is better. CMD measures whether a series of circuits yield *the same* strength of preference for the correct answer as the full model; 0 is best, and corresponds to no difference between the circuit and full model behavior with respect to predicting the correct answer. Intuitively, CPR may be more useful for finding circuits that cause the model to perform well on the task, while CMD may be more useful when the aim is to explain the full algorithm the model implements to perform some behavior (including cases where the behavior is not desirable).

Given a circuit $\mathcal{C}$ and the full model $\mathcal{N}$, faithful-

ness f is defined as:

$$f(\mathcal{C}, \mathcal{N}; m) = \frac{m(\mathcal{C}) - m(\varnothing)}{m(\mathcal{N}) - m(\varnothing)}, \quad (1)$$

where m is the logit difference $y' - y$ between the correct answer y given the original input x and correct answer y' given the counterfactual input x'.

Thus, CPR is computed as the area under the faithfulness curve with respect to circuit size. Following Mueller et al. (2025), we approximate this area using a Riemann sum over f computed across circuit sizes. CMD as the area between the faithfulness curve and 1; we also approximate this using a Riemann sum.

Measuring circuit size. MIB treats including a node as equivalent to including all of its outgoing edges, and including one neuron[2] of d_{model} in submodule u as including all outgoing edges from u to $\frac{1}{d_{\text{model}}}$ of the degree they would have been compared to including all neurons in u.

Under these assumptions, MIB defines the weighted edge count:

$$|\mathcal{C}| = \sum_{(u,v)\in\mathcal{C}} \left(\frac{|N_u \cap N_{\mathcal{C}}|}{|N_u|} \right), \quad (2)$$

where u and v are nodes (submodules), N_u is the set of neurons in u (the size of which is typically d_{model}), and $N_{\mathcal{C}}$ is the set of neurons in the circuit. This count is then normalized by the number of possible edges to obtain a percentage.

3.2 Submission Procedure

All results below are computed on the private test splits for each task. To evaluate on the private test split, participants were first required to upload their circuits to a HuggingFace repository.[3] The faithfulness evaluation required 9 circuits of different sizes; we expected one circuit $\mathcal{C}_k$ for each $k \in K$, where k is the maximum proportion of components in $\mathcal{N}$ that are allowed to remain in the circuit. Here, $K = \{0.001, 0.002, 0.005, 0.01, 0.02, 0.05, 0.1, 0.2, 0.5\}$. For each model/task combination, a folder of circuits was required. Each circuit is a dictionary, where each node and edge is a key whose value is either a boolean indicating whether the component or edge belongs to the circuit, or a floating-point importance score. If the user uploaded floating-point importance scores, then only one file per model/task was required; we took the top-k components by importance for each circuit size $k \in K$. If the user uploaded binary inclusion indices, they were required to upload one circuit file for each threshold $k \in K$.

Users provided a link to this repository on the "Submit" tab of the MIB leaderboard,[4] along with a method name.

3.3 Task Submissions

We received submissions from three teams for the circuit localization track covering eight proposed methods. We taxonomize and summarize the approaches here.

Ensemble scoring strategies. Mondorf et al. (2025) proposed ensembling two or more circuit localization methods to improve attribution scores. They examined three ensembling variants: *parallel*, *sequential*, and their *hybrid* combination.

Parallel ensembling (P-Ens) merges the scores from different methods into a single edge, using scores from the three variants of edge patching implemented by Mueller et al.: (1) Edge Attribution Patching (EAP; Nanda, 2023; Syed et al., 2024, (2) EAP-IG-inputs (Hanna et al., 2024), and (3) EAP-IG-activations (Marks et al., 2025). The latter two methods complement EAP with integrated gradients (Sundararajan et al., 2017) to improve estimates of edge importance, perturbing input embeddings and activations, respectively. The authors experimented with score merging using mean, weighted average, maximum, and minimum, and found that mean yielded the best results.

Sequential ensembling (S-Ens) utilizes attribution scores produced by a fast circuit identification method to warm-start a slower, more precise method, thereby achieving faster convergence and further refining the initial scores. Specifically, they use EAP-IG-inputs (Hanna et al., 2024) edge attribution values to initialize the learnable log alpha parameters of edge pruning (Bhaskar et al., 2024).

Finally, *hybrid ensembling (Hybrid-Ens)* combines the parallel and sequential strategies by taking the unweighted average over all four method–the three EAP variants and warm-start edge pruning for all model-task combination.

[2] We use "neuron" to refer to a single dimension of any hidden vector, regardless of whether it is preceded by a non-linearity.

[3] See this repository for an example of how circuit repositories were required to be structured.

[4] https://hf.co/spaces/mib-bench/leaderboard

Method	IOI					Arithmetic	MCQA			ARC (E)		ARC (C)
	InterpBench (↑)	GPT-2	Qwen-2.5	Gemma-2	Llama-3.1	Llama-3.1	Qwen-2.5	Gemma-2	Llama-3.1	Gemma-2	Llama-3.1	Llama-3.1
Random	0.44	0.75	0.72	0.69	0.74	0.75	0.73	0.68	0.74	0.68	0.74	0.74
EAP (mean)	0.78	0.29	0.18	0.25	0.04	0.07	0.21	0.20	0.16	0.22	0.28	0.20
EAP (CF)	0.73	0.03	0.15	0.06	**0.01**	0.01	0.07	0.08	**0.09**	0.04	**0.11**	**0.18**
EAP (OA)	0.77	0.30	0.16	-	-	-	0.11	-	-	-	-	-
EAP-IG-inp. (CF)	0.71	0.03	0.02	0.04	**0.01**	**0.00**	0.08	**0.06**	0.14	0.04	**0.11**	0.22
EAP-IG-act. (CF)	**0.81**	0.03	**0.01**	**0.03**	**0.01**	**0.00**	0.05	0.07	0.13	0.04	0.30	0.37
P-Ens (Mondorf et al., 2025)	-	**0.02**	0.02	-	-	-	0.07	-	-	-	-	-
S-Ens (Mondorf et al., 2025)	-	0.03	0.02	-	-	-	0.07	-	-	-	-	-
Hybrid-Ens (Mondorf et al., 2025)	-	0.03	0.02	-	-	-	**0.04**	-	-	-	-	-
ILP + PNR + Bootstrapping (2025a)	-	**0.02**	**0.01**	0.04	**0.01**	0.01	0.08	0.07	0.45	**0.03**	-	-
IPE (CF) (Brunello et al., 2025)	-	**0.02**	0.57	-	-	0.54	-	-	-	-	-	-

Table 2: CMD scores across circuit localization methods (lower is better) on the private test set. All evaluations were performed using counterfactual ablations. Arithmetic scores are averaged across addition and subtraction. We **bold** and underline the best and second-best methods per column, respectively.

Method	IOI				Arithmetic	MCQA			ARC (E)		ARC (C)
	GPT-2	Qwen-2.5	Gemma-2	Llama-3.1	Llama-3.1	Qwen-2.5	Gemma-2	Llama-3.1	Gemma-2	Llama-3.1	Llama-3.1
EActP (CF)	**2.30**	1.21	-	-	-	0.85	-	-	-	-	-
EAP (mean)	0.29	0.71	0.68	0.98	0.35	0.29	0.33	0.13	0.26	0.34	0.80
EAP (CF)	1.20	0.26	1.29	0.85	0.55	0.85	1.49	1.00	1.08	0.80	0.82
EAP (OA)	0.95	0.70	-	-	-	0.29	-	-	-	-	-
EAP-IG-inputs (CF)	1.85	1.63	**3.20**	2.08	0.99	1.16	1.64	1.05	1.53	**1.04**	**0.98**
EAP-IG-activations (CF)	1.82	1.63	2.07	1.60	0.98	0.77	1.57	0.79	**1.70**	0.71	0.63
NAP (CF)	0.28	0.30	0.30	0.26	0.27	0.38	1.47	1.69	1.01	0.26	0.26
NAP-IG (CF)	0.76	0.29	1.52	0.42	0.39	0.77	**1.71**	**1.87**	1.53	0.26	0.26
P-Ens (Mondorf et al., 2025)	2.11	**1.88**	-	-	-	0.79	-	-	-	-	-
S-Ens (Mondorf et al., 2025)	2.37	1.71	-	-	-	1.16	-	-	-	-	-
Hybrid-Ens (Mondorf et al., 2025)	**2.43**	**1.88**	-	-	-	**1.19**	-	-	-	-	-
ILP + PNR + Bootstrapping (2025a)	1.89	1.71	3.01	**2.39**	**1.04**	1.04	1.7	1.22	1.63	-	-
IPE (CF) (Brunello et al., 2025)	2.24	0.35	-	-	-	0.45	-	-	-	-	-

Table 3: CPR scores across circuit localization methods on the private test set. All evaluations were performed using counterfactual ablations. Higher scores are better. Arithmetic scores are averaged across addition and subtraction. We **bold** and underline the best and second-best methods per column, respectively.

Improved edge selection. Focusing on the second stage of the circuit discovery pipeline, Nikankin et al. (2025a) experimented with three methods to improve edge selection process. First, they observe that EAP-IG scores can vary across data samples from the same task, with some edges receiving both negative and positive values in different samples. The score sign is significant, as it signifies whether the edge contributes positively or negatively to the performance on the task. By bootstrapping the scores across resamples of the training data, they identify edges with consistent score signs and filter out unstable ones.

Second, they introduce a ratio based strategy for edge selection based on their signs (PNR): select a fixed proportion of top positive edges, and the rest by absolute value. This approach allows finer control over the balance of edge types and improves circuit faithfulness. Lastly, they formulate circuit construction as an Integer Linear Programming (ILP) optimization problem, instead of using the naive greedy solution.

Path scoring. Brunello et al. (2025) proposed Isolating Path Effects (IPE) to identify entire computational paths from input embeddings to output logits responsible for certain model behaviors, as opposed to individual edges. Their method modifies the messages passed between nodes along a given path in such a way as to either precisely remove the effects of the entire path (i.e., ablate it) or to replace the path's effects with those that a counterfactual input would have produced. IPE differs from current path-patching or edge-activation-patching techniques, as they do not ablate individual paths but rather a set of paths sharing certain edges, thereby allowing a more precise tracing of information flow.

3.4 Results

Table 2 and Table 3 show the CMD and CPR scores, respectively, of the top method from each submission as well as selected methods from MIB, on the private test set. All submissions perform especially well, achieving better or comparable scores to even the strongest baselines.

The submission of Nikankin et al. (2025a)

achieves especially strong CMD scores, whereas the Hybrid-Ens method of Mondorf et al. (2025) achieves the strongest CPR scores. The IPE method by Brunello et al. (2025) also performs well on IOI for GPT-2. Among the methods of Mondorf et al. (2025), Hybrid-Ens performs the strongest across tasks. These results suggest that ensembling strategies may be an accessible and fruitful line of work for future circuit discovery research. For Nikankin et al. (2025b), the removal of components with inconsistent effects on model outputs and a mixture of positive and high-magnitude components may have a regularizing effect on the discovered circuit, causing it to behave more closely to the whole model and potentially suppressing components that would have strong but inconsistent impacts on model behavior. It would be interesting to see detailed comparisons of each method on more fine-grained distributions to characterize when and why each is likely to succeed. That said, there is no clear winner; the best method appears to depend on the chosen metric.

A factor we have not directly evaluated for is the time complexity of each method. It is possible that different methods could perform comparably despite having very different expected runtimes; a direct comparison of compute requirements would be valuable in helping future researchers decide which methods are most worthwhile to run. We note that many cells are missing for each submission, but this does not necessarily reflect compute requirements—this could be due to local memory constraints, runtime limitations, or other compute constraints (e.g., limited access to GPUs on a cluster before a deadline).

4 Causal Variable Localization Track

The causal variable localization track focuses on evaluating how well a method can discover specific causal variables in a language model's activation space. The basic intuition is that any hidden vector $\mathbf{h} \in \mathbb{R}^d$ constructed by a model $\mathcal{N}$ during inference can be mapped into a new feature space $\mathbb{F}^k$ (e.g., a rotated vector space) using an invertible function $\mathcal{F} : \mathbb{R}^d \to \mathbb{F}^k$ (e.g., multiplication with an orthogonal matrix). Features Π are a set of indices between 1 and k, i.e., a set of dimensions in $\mathbb{F}^k$. This framework supports features like neurons, orthogonal directions, (sets of) SAE features, and non-linear features. The vector $\mathbf{h}$ might come from the residual stream between transformer layers or the output of an attention head.

4.1 Evaluation Metric

We use **faithfulness** to evaluate causal variable localization submissions. This metric captures the degree to which the provided features capture the causal variable under counterfactual intervention. To evaluate faithfulness, we use *interchange interventions*. Given base and counterfactual inputs (b, c), high-level causal graph $\mathcal{H}$, and causal variable $X \in \mathcal{H}$, the interchange intervention $\mathcal{H}_{X \leftarrow \mathsf{Get}(\mathcal{H}(c),X)}(b)$ runs $\mathcal{H}$ on base input b while fixing the variable X to the value it takes when $\mathcal{H}$ is run on a counterfactual input c (Vig et al., 2020; Geiger et al., 2020). The distributed interchange intervention $\mathcal{N}_{\Pi_X \leftarrow \mathsf{Get}(\mathcal{N}(c),\Pi_X)}(b)$ runs $\mathcal{N}$ on b while fixing the features Π_X of the hidden vector $\mathbf{h}$ passed through $\mathcal{F}$ to the value they take for counterfactual input c (Wu et al., 2023b; Amini et al., 2023; Geiger et al., 2024). Faithfulness is measured as the proportion of examples for which the intervention yields the expected change in the model's output behavior. See Wu et al. (2023a) and Mueller et al. (2025) for examples.

4.2 Submission Procedure

As for the circuit localization task, users were required to upload files to a HuggingFace repository, although the required files differed for causal variable localization.[5] Here, a user was required to upload at least three artifacts for a given causal variable: a trained featurizer $\mathcal{F}$, a trained inverse featurizer $\mathcal{F}^{-1}$, and position indices corresponding to the dimensions of the featurized space that encode the causal variable of interest. If the featurizer was not one of the supported baseline types, users were also required to upload Python code that could save and load their featurizer. We also supported interventions at dynamic token positions; if used, users were required to upload a Python script specifying which token positions to intervene on for a given example.[6]

4.3 Task Submissions

We received submissions from one team totalling two methods (Hirlimann et al., 2025). Both methods extend the official Distributed Alignment Search (DAS; Geiger et al., 2024) baseline.

[5] See this repository for an example of how causal variable localization repositories were required to be structured.

[6] See the track's GitHub repository for further details.

Method	RAVEL					
	Gemma-2			Llama-3.1		
	A_{Cont}	$A_{Country}$	A_{Lang}	A_{Cont}	$A_{Country}$	A_{Lang}
DAS	75 (**85**)	57 (**67**)	62 (**70**)	75 (**83**)	58 (**64**)	63 (**70**)
DBM	66 (**71**)	53 (**65**)	54 (**58**)	68 (**80**)	53 (**59**)	58 (**64**)
+PCA	63 (**70**)	47 (**53**)	50 (**56**)	62 (**74**)	48 (**54**)	53 (**57**)
+SAE	64 (**72**)	49 (**56**)	53 (**59**)	64 (**72**)	50 (**57**)	55 (**57**)
Full Vector	48 (**62**)	49 (**57**)	45 (**56**)	53 (**62**)	47 (**53**)	47 (**57**)
Orthogonal	-	-	-	84 (**89**)	70 (**79**)	72 (**79**)
Nonlinear	-	-	-	83 (**89**)	70 (**78**)	72 (**79**)

(a) The RAVEL task with variables for the country $A_{Country}$, continent A_{Cont}, and language A_{Lang} of a city.

Method	Arithmetic (+)	
	Gemma-2	Llama-3.1
	X_{Carry}	X_{Carry}
DAS	31 (**35**)	54 (**65**)
DBM	33 (**43**)	47 (**58**)
+PCA	32 (**44**)	37 (**56**)
+SAE	32 (**44**)	38 (**55**)
Full Vector	29 (**35**)	35 (**45**)
Orthogonal	-	53 (**65**)
Nonlinear	-	-

(b) The two-digit arithmetic task with a variable computing the carry-the-one operation (X_{Carry}).

Method	MCQA					
	Gemma-2		Llama-3.1		Qwen-2.5	
	O_{Answer}	X_{Order}	O_{Answer}	X_{Order}	O_{Answer}	X_{Order}
DAS	95 (**97**)	77 (**93**)	94 (**100**)	77 (**91**)	86 (**95**)	78 (**100**)
DBM	84 (**99**)	63 (**84**)	86 (**100**)	66 (**73**)	46 (**94**)	60 (**99**)
+PCA	57 (**96**)	52 (**81**)	65 (**99**)	53 (**74**)	22 (**76**)	54 (**100**)
+SAE	73 (**90**)	51 (**65**)	80 (**99**)	58 (**65**)	–	–
Full Vector	61 (**100**)	44 (**77**)	77 (**100**)	46 (**68**)	35 (**99**)	49 (**99**)
Orthogonal	-	-	-	-	90 (**98**)	78 (**100**)
Nonlinear	-	-	95 (**100**)	81 (**94**)	89 (**98**)	81 (**100**)

(c) The MCQA task with variables for the ordering of the answer X_{Order} and then the answer token O_{Answer}. This is a low-data regime (≈100 examples).

Table 4: Results for the causal variable localization track. Table headers show the task, the model, and the selected causal variable, respectively. We do not report results for ARC or IOI, as no submissions were made for these tasks. We report interchange intervention accuracy (i.e., our faithfulness metric), i.e., the proportion of aligned interventions on the causal model and deep learning model that result in the same output token(s); higher is better. For each method of aligning a causal variable to LM features, we report the mean across counterfactual datasets and layers in the low-level model. In parenthesis and **bold**, we report the best alignment across all layers.

Non-linear featurizer. This method extends DAS with a multi-layer perceptron (MLP) and non-linearities. During training, this method augments the feature mixing stage with an MLP:

$$\mathbf{h} = \text{GeLU}(W_u\mathbf{x}) \tag{3}$$

$$\hat{\mathbf{x}} = \tanh(W_d\mathbf{h}), \tag{4}$$

where W_u and W_d are learned up-projection and down-projection weights, respectively. This is only applied during training. This allows the featurizer to "blend" potentially independent representations and go beyond convex combinations of features, which could allow it to learn dependencies where the signal is not strictly separable by individual directions in the original actuvation space $\mathbf{x}$. Empirically, this was the most well-performing method, even outperforming DAS—the best-performing baseline method. That said, recent work has demonstrated that non-linear featurizers are highly expressive, and as such can locate potentially any feature, including those that are not in the model itself (Sutter et al., 2025), echoing the memorization problem that characterized probing classifiers (Belinkov, 2022). Additional validation is needed to confirm that the learned features capture genuine variables the model employs during processing.

Orthogonal non-linear projection. This featurizer is a simplified variant of the non-linear featurizer. Here, the features pass only through a tanh non-linearity without a feed-forward layer. This still enables rich feature interactions to be learned, but does not have as much expressive power as the non-linear featurizer.

4.4 Results

We show faithfulness scores for baselines and submissions in Table 4. Both the orthogonal and non-linear methods achieve significant gains over DAS across tasks and models. Despite the greater expressive power of the non-linear featurizers, this

method performs comparably to the simpler orthogonal featurizer across tasks, with non-linear featurization proving slightly stronger for MCQA with Qwen-2.5.

5 Conclusions

Despite the relatively small number of submissions, participants achieved significant performance gains. Ensembling methods are quite effective for circuit discovery, as is regularization via filtering components with unstable contributions to model behavior; we encourage future work to continue exploring these directions. Furthermore, one can achieve significant gains in variable localization using non-linear mediator types; these projections into new spaces can be highly effective with the proper training procedure, even when the non-linearity is built on top of a simple architecture. This suggests that expressive featurizer training formulations that leverage existing mediator types might yield significant gains in causal variable localization—but more controls are needed to ensure that concepts truly in the model itself are being isolated (as opposed to the featurizer learning the causal variable itself).

The MIB leaderboard will continue to accept public submissions in both tracks. The results of this shared task will inform the experimental design and baseline choices for future studies employing circuits and causal-variable localization methods in language models. We hope that participants will continue to publicize their findings to benefit the community and enable scientific progress through direct comparisons in a shared-task setting.

References

Afra Amini, Tiago Pimentel, Clara Meister, and Ryan Cotterell. 2023. Naturalistic causal probing for morpho-syntax. *Transactions of the Association for Computational Linguistics*, 11:384–403.

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

Adithya Bhaskar, Alexander Wettig, Dan Friedman, and Danqi Chen. 2024. Finding transformer circuits with edge pruning. *Advances in Neural Information Processing Systems*, 37:18506–18534.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Nicolò Brunello, Andrea Cerutti, Andrea Sassella, and Mark James Carman. 2025. IPE: Isolating path effects for improving latent circuit identification. In *BlackboxNLP-2025 MIB Shared Task*.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? Try ARC, the AI2 Reasoning Challenge. *arXiv preprint arXiv:1803.05457*.

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

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The Llama 3 herd of models. *CoRR*, abs/2407.21783.

Javier Ferrando and Elena Voita. 2024. Information flow routes: Automatically interpreting language models at scale. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 17432–17445, Miami, Florida, USA. Association for Computational Linguistics.

Atticus Geiger, Kyle Richardson, and Christopher Potts. 2020. Neural natural language inference models partially embed theories of lexical entailment and negation. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 163–173, Online. Association for Computational Linguistics.

Atticus Geiger, Zhengxuan Wu, Christopher Potts, Thomas Icard, and Noah D. Goodman. 2024. Finding alignments between interpretable causal variables and distributed neural representations. In *Causal Learning and Reasoning, 1-3 April 2024, Los Angeles, California, USA*, volume 236 of *Proceedings of Machine Learning Research*, pages 160–187. PMLR.

Michael Hanna, Sandro Pezzelle, and Yonatan Belinkov. 2024. Have faith in faithfulness: Going beyond circuit overlap when finding model mechanisms. In *ICML 2024 Workshop on Mechanistic Interpretability*.

Lea Hirlimann, Yihong Liu, Leonor Veloso, Philipp Mondorf Shijia Zhou, Mingyang Wang, Ahmad Dawar Hakimi, Barbara Plank, and Hinrich Schütze. 2025. BlackboxNLP-2025 MIB shared task: Exploring the impact of non-linear modules on distributed alignment search. In *BlackboxNLP-2025 MIB Shared Task*.

Jing Huang, Zhengxuan Wu, Christopher Potts, Mor Geva, and Atticus Geiger. 2024a. RAVEL: Evaluating interpretability methods on disentangling language model representations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8669–8687, Bangkok, Thailand. Association for Computational Linguistics.

Yufei Huang, Shengding Hu, Xu Han, Zhiyuan Liu, and Maosong Sun. 2024b. Unified view of grokking, double descent and emergent abilities: A comprehensive study on algorithm task. In *First Conference on Language Modeling*.

Robert Huben, Hoagy Cunningham, Logan Riggs Smith, Aidan Ewart, and Lee Sharkey. 2024. Sparse autoencoders find highly interpretable features in language models. In *The Twelfth International Conference on Learning Representations*.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *CoRR*, abs/2310.06825.

Ruizhe Li and Yanjun Gao. 2024. Anchored answers: Unravelling positional bias in GPT-2's multiple-choice questions. ArXiv preprint arXiv:2405.03205.

Tom Lieberum, Matthew Rahtz, János Kramár, Geoffrey Irving, Rohin Shah, and Vladimir Mikulik. 2023. Does circuit analysis interpretability scale? Evidence from multiple choice capabilities in chinchilla. ArXiv preprint arXiv:2307.09458.

Ziming Liu, Eric J Michaud, and Max Tegmark. 2023. Omnigrok: Grokking beyond algorithmic data. In *The Eleventh International Conference on Learning Representations*.

Samuel Marks, Can Rager, Eric J Michaud, Yonatan Belinkov, David Bau, and Aaron Mueller. 2025. Sparse feature circuits: Discovering and editing interpretable causal graphs in language models. In *The Thirteenth International Conference on Learning Representations*.

Jack Merullo, Carsten Eickhoff, and Ellie Pavlick. 2024. Circuit component reuse across tasks in transformer language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Philipp Mondorf, Mingyang Wang, Sebastian Gerstner, Ahmad Dawar Hakimi, Yihong Liu, Leonor Veloso, Shijia Zhou, Hinrich Schütze, and Barbara Plank. 2025. BlackboxNLP-2025 MIB shared task: Exploring ensemble strategies for circuit localization methods. In *BlackboxNLP-2025 MIB Shared Task*.

Aaron Mueller, Atticus Geiger, Sarah Wiegreffe, Dana Arad, Iván Arcuschin, Adam Belfki, Yik Siu Chan, Jaden Fried Fiotto-Kaufman, Tal Haklay, Michael Hanna, and 1 others. 2025. MIB: A mechanistic interpretability benchmark. In *Forty-second International Conference on Machine Learning*.

Neel Nanda. 2023. Attribution patching: Activation patching at industrial scale.

Neel Nanda, Lawrence Chan, Tom Lieberum, Jess Smith, and Jacob Steinhardt. 2023. Progress measures for grokking via mechanistic interpretability. In *The Eleventh International Conference on Learning Representations*.

Yaniv Nikankin, Dana Arad, Itay Itzhak, Anja Reusch, Adi Simhi, Gal Kesten-Pomeranz, and Yonatan Belinkov. 2025a. BlackboxNLP-2025 MIB shared task: Improving circuit faithfulness via better edge selection. In *BlackboxNLP-2025 MIB Shared Task*.

Yaniv Nikankin, Anja Reusch, Aaron Mueller, and Yonatan Belinkov. 2025b. Arithmetic without algorithms: Language models solve math with a bag of heuristics. In *The Thirteenth International Conference on Learning Representations*.

Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. 2020. Zoom in: An introduction to circuits. *Distill*. Https://distill.pub/2020/circuits/zoom-in.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, and 1 others. 2019. Language models are unsupervised multitask learners. Blog post.

Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, and 1 others. 2024. Gemma 2: Improving open language models at a practical size. ArXiv:2408.00118.

Alessandro Stolfo, Yonatan Belinkov, and Mrinmaya Sachan. 2023. A mechanistic interpretation of arithmetic reasoning in language models using causal mediation analysis. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7035–7052.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 3319–3328. JMLR.org.

Denis Sutter, Julian Minder, Thomas Hofmann, and Tiago Pimentel. 2025. The non-linear representation dilemma: Is causal abstraction enough for mechanistic interpretability? *Preprint*, arXiv:2507.08802.

Aaquib Syed, Can Rager, and Arthur Conmy. 2024. Attribution patching outperforms automated circuit discovery. In *Proceedings of the 7th BlackboxNLP*

Workshop: Analyzing and Interpreting Neural Networks for NLP, pages 407–416, Miami, Florida, US. Association for Computational Linguistics.

Jesse Vig, Sebastian Gehrmann, Yonatan Belinkov, Sharon Qian, Daniel Nevo, Yaron Singer, and Stuart Shieber. 2020. Investigating gender bias in language models using causal mediation analysis. In *Advances in Neural Information Processing Systems*, volume 33, pages 12388–12401. Curran Associates, Inc.

Kevin Ro Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. 2023. Interpretability in the wild: a circuit for indirect object identification in GPT-2 small. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Sarah Wiegreffe, Oyvind Tafjord, Yonatan Belinkov, Hannaneh Hajishirzi, and Ashish Sabharwal. 2025. Answer, assemble, ace: Understanding how LMs answer multiple choice questions. In *The Thirteenth International Conference on Learning Representations*.

Zhengxuan Wu, Atticus Geiger, Thomas Icard, Christopher Potts, and Noah Goodman. 2023a. Interpretability at scale: Identifying causal mechanisms in alpaca. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Zhengxuan Wu, Atticus Geiger, Thomas Icard, Christopher Potts, and Noah D. Goodman. 2023b. Interpretability at scale: Identifying causal mechanisms in alpaca. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, and 1 others. 2024. Qwen2.5 technical report. ArXiv:2412.15115.

Wei Zhang, Chaoqun Wan, Yonggang Zhang, Yiu ming Cheung, Xinmei Tian, Xu Shen, and Jieping Ye. 2024. Interpreting and improving large language models in arithmetic calculation. In *Forty-first International Conference on Machine Learning*.