BlackboxNLP-2025 MIB Shared Task: Improving Circuit Faithfulness via Better Edge Selection

Yaniv Nikankin* **Dana Arad*** **Itay Itzhak***
Anja Reusch **Adi Simhi** **Gal Kesten-Pomeranz** **Yonatan Belinkov**
Technion – Israel Institute of Technology
{yaniv.n, danaarad, itay.itzhak}@campus.technion.ac.il

Abstract

One of the main challenges in mechanistic interpretability is circuit discovery—determining which parts of a model perform a given task. We build on the Mechanistic Interpretability Benchmark (MIB) and propose three key improvements to circuit discovery. First, we use bootstrapping to identify edges with consistent attribution scores. Second, we introduce a simple ratio-based selection strategy to prioritize strong positive-scoring edges, balancing performance and faithfulness. Third, we replace the standard greedy selection with an integer linear programming formulation. Our methods yield more faithful circuits and outperform prior approaches across multiple MIB tasks and models. Our code is available at: https://github.com/technion-cs-nlp/MIB-Shared-Task.

1 Introduction

Mechanistic interpretability has gained recent popularity due to its progress in characterizing the internal mechanisms of AI models (Saphra and Wiegreffe, 2024; Rai et al., 2024). A popular paradigm aims to uncover *circuits*, subgraphs of the model's computation graph that are responsible for specific tasks (Olah et al., 2020). However, discovering optimal circuits that are as small as possible while matching the original model's behavior remains an open challenge. To this end, the recent Mechanistic Interpretability Benchmark (MIB) (Mueller et al., 2025) was proposed to offer a standardized framework for evaluating circuit discovery methods.

A typical circuit discovery pipeline consists of two stages: (1) obtaining scores for the full set of graph components (nodes, edges, etc.), and (2) selecting a subset of the components that constitute the circuit. Prior work has largely focused on developing improved scoring methods for the first stage, such as Edge Attribution Patching (EAP) (Nanda, 2023) and EAP with Integrated Gradients (EAP-IG) (Hanna et al., 2024), while relying on a greedy selection algorithm for the second stage. Using this setup, Mueller et al. (2025) demonstrated that EAP-IG scores led to the top-preforming subgraphs.

In this work, we focus on the second stage of circuit discovery and suggest several improvements for building a circuit given EAP-IG scores. We rely on a few key observations. First, 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 represents a significant property: positive-scoring components contribute positively to the model's performance on the task, while negative scores indicate a negative impact, such as the negative name mover heads from Wang et al. (2023). By bootstrapping the scores across resamples of the training data, we are able to identify edges with consistent score signs and filter out unstable ones.

Second, we find that selecting edges by score magnitude alone, ignoring sign, often yields circuits that misrepresent the model's original behavior. To address this, we introduce a simple ratio-based strategy: 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, we formulate circuit construction as an Integer Linear Programming (ILP) optimization problem, instead of using the naive greedy solution.

Overall, we show that different combinations of our methods, tailored to the different faithfulness objectives proposed by MIB, yield improved performance over the leading approaches.

2 Motivation

Circuit discovery follows a two-stage pipeline: scoring and selection. In the scoring stage, each component of the model's computation graph is

*Equal contribution.

Proceedings of the 8th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP, pages 521–527

Method	IOI			MCQA		ARC (E)
	GPT-2	Qwen-2.5	Gemma-2	Qwen-2.5	Gemma-2	Gemma-2
CMD Baseline (Greedy)	0.0308	0.0374	**0.0658**	0.1846	**0.0880**	0.0458
CMD (ILP + PNR)	**0.0294**	**0.0370**	0.0760	**0.1820**	0.0907	**0.0451**
CPR Baseline (Greedy)	2.4901	2.2658	**5.6155**	1.0612	1.8769	**1.9104**
CPR (ILP + Bootstrapping)	**2.5061**	**2.5092**	5.4516	**1.0926**	**1.9145**	1.8918

Table 1: Comparison of our chosen methods against the baseline for CMD (lower is better) and CPR (higher is better) metrics on the public test sets.

given an importance score; in this submission we consider edges as our basic computation components and utilize Edge Attribution Patching with Integrated Gradients (EAP-IG) as our base scoring metric. We use EAP-IG as it obtained the best score on MIB across most of the models and tasks.

While EAP-IG is typically computed once over the entire training set, applying it to multiple data subsets reveals an interesting pattern. In GPT-2+IOI and Qwen-2.5+MCQA, we find that 9.1% and 6.5% of edges, respectively, exhibit sign instability with both positive and negative scores across 10 samples.[1] Since sign indicates whether an edge contributes positively or negatively to circuit performance, instability may signal noisy attribution scores. By applying **bootstrapping**, we can filter out unstable edges before constructing the graph.

After scoring each edge, the selection stage follows in which a subset of edges is selected to form the circuit. In this stage, Mueller et al. employ the approach of using a greedy, layer-by-layer algorithm (Hanna et al., 2024). The greedy algorithm starts from the output layer and works backward, adding the top-ranked edges needed to compute the activations of already selected components. The resulting graph is pruned to remove non-connected components, resulting in a connected subgraph.

To evaluate circuits, Mueller et al. define two metrics based on the faithfulness curve with respect to circuit size. The integrated circuit performance ratio (CPR) measures how well the method identifies components that positively contribute to the model's performance on a task, and is defined as the area under the curve. In contrast, the integrated circuit-model distance (CMD) quantifies how closely the circuit approximates the model's overall behavior, including positive and negative contributions, and is defined as the area between the faithfulness curve and the optimal value of 1.

Thus, in the MIB implementation edges are ranked differently between the two metrics. For CPR edges are simply ranked by their scores, whereas for CMD the ranking is based on their absolute scores. We observe that for CMD, this may lead to over-selection of negatively scoring edges, which can degrade the circuit's faithfulness. To mitigate this, we propose a **positive-negative ratio (PNR)** strategy: first select a given percentage of top-positive edges, then apply the usual selection.

While the greedy algorithm is fast and guarantees a valid circuit, it relies solely on local decisions and may result in suboptimal edge selection. Thus, we formulate circuit selection as an **integer linear program (ILP)** for globally optimal subset selection under structural and budget constraints.

3 Method

In this section we describe three methods for improving circuit construction, which can be applied individually or in combination.

3.1 Bootstrapped Confidence Filtering

We use bootstrapping to identify edges with consistently-signed attribution scores, since these are more likely to reflect meaningful structure in the model. For a dataset of size N, we sample with replacement to obtain τ sets with N samples each. For each edge e, we collect a set of scores $\{a_1, \ldots, a_\tau\}$ from the τ bootstrap runs. We compute the sample mean μ_e and standard deviation σ_e, then construct a two-sided confidence interval:

$$\mu_e \pm z \cdot \frac{\sigma_e}{\sqrt{\tau}}, \tag{1}$$

where μ_e is the mean score across the bootstrap samples, σ_e is the sample standard deviation, and z is the standard normal quantile corresponding to the desired confidence level ($z = 1.96$ for 95%).

[1] Only edges with non-zero mean scores $|\mu| > 10^{-6}$.

An edge is retained if its confidence interval in Equation 1 lies entirely above or below a fixed significance threshold, depending on μ_e's sign, with μ_e serving as the final edge score.

3.2 Positive-Negative Ratio (PNR)

We select edges in two phases defined by the PNR value, a real number in the range $[0, 1]$. Given k as the maximum number of edges in the circuit:

1. Select $\lceil \text{PNR} \cdot k \rceil$ edges from the top positively scored edges, sorted by raw signed-score.
2. Select the remaining $k - \lceil \text{PNR} \cdot k \rceil$ edges from the top remaining edges, sorted by absolute score (positive or negative).

Hence, PNR sets the minimum fraction of positively contributing edges in the circuit.

3.3 Integer Linear Programming (ILP)

Lastly, we replace the greedy algorithm by formulating graph construction as an ILP problem (Wolsey, 1998).

Formally, we define the computation graph as a multi-edge directional graph, $G = (V, E)$, with a scoring function, $a : E \to \mathbb{R}$. G has a unique source node y_s with no incoming edges ($d_{in}(y_s) = 0$), and a unique target node y_t with no outgoing edges ($d_{out}(y_t) = 0$). Let $x_e \in \{0, 1\}$ indicate whether edge $e = (u, v, w) \in E$ is selected, and $y_v \in \{0, 1\}$ indicate whether node $v \in V$ is used.

Given a budget k on the maximum number of edges in the circuit, the ILP maximizes the total score of the selected edges (Equation 2a) under the following constraints:

$$\max \quad \sum_{e \in E} a(e) \cdot x_e \tag{2a}$$

$$\text{s.t.} \quad \sum_{e \in E} x_e \leq k \tag{2b}$$

$$y_s = y_t = 1 \tag{2c}$$

$$x_{(u,v,w)} \leq \min\{y_u, y_v\}, \forall (u, v, w) \in E \tag{2d}$$

$$\sum_{(u,v,w) \in E} x_{(u,v,w)} \geq y_u, \forall u \in V \tag{2e}$$

$$\sum_{(u,v,w) \in E} x_{(u,v,w)} \geq y_v, \forall v \in V \tag{2f}$$

$$\sum_{e \in E, a(e) > 0} x_e \geq \text{PNR} \cdot k \tag{2g}$$

where the constraints correspond to:

- **Edge budget (2b):** select at most k edges.
- **Source and target (2c):** source node y_s and target node y_t must be selected.
- **Node-edge consistency (2d):** if (u, v, w) is selected, both u and v must be selected.
- **Connectivity (2e, 2f):** every used non-source node has at least one incoming selected edge; every used non-target node has at least one outgoing selected edge.
- **PNR (2g):** (when using ILP with PNR) the number of positive-scoring edges should exceed the PNR value.

4 Experimental Setup

MIB provides a standardized framework for evaluating circuit discovery methods across four models and four tasks. In this submission, we focus on a subset of these models and tasks due to computational limitations of the ILP problems, and include results on Gemma-2 2B (Riviere et al., 2024), Qwen-2.5 0.5B (Yang et al., 2024), and GPT-2 Small (Radford et al., 2019) on indirect object identification (IOI) (Wang et al., 2022), multiple-choice question answering (MCQA) (Wiegreffe et al., 2024), and the easy partition of the AI2 Reasoning Challenge (ARC-E) (Clark et al., 2018).

We used the validation sets to select the best combination of methods and hyper-parameters for each metric, and additionally report results of the leading method on the test sets. Additional implementation details are described in Appendix B.

5 Results

Our main results are displayed in Table 1. We evaluate using both CMD (lower is better) and CPR (higher is better), reporting our best-performing combination of proposed methods against the greedy graph building baseline (Mueller et al., 2025). For CMD evaluation, we employ ILP to construct optimal graphs, combined with PNR selection, which prioritizes positive edges. The PNR value varies per model-task combination and was chosen for each combination individually, to account for task-specific distributions. This approach enhances faithfulness towards the optimal threshold of 1.0, particularly for smaller subgraphs.

For CPR evaluation, where negative edges are penalized due to the objective of maximizing faithfulness, we replace PNR with bootstrapping to consistently retain positive edges. We use $\tau = 10$ bootstrap iterations. We choose these combinations of methods as they achieve better results than the baseline across the widest range of analyzed models and tasks (see Section 5.1). Our proposed

	IOI			MCQA		ARC (E)
Method	GPT-2	Qwen-2.5	Gemma-2	Qwen-2.5	Gemma-2	Gemma-2
Greedy	0.0411	0.0254	0.0564	0.1403	0.1234	**0.0417**
ILP	0.0370	0.0242	0.0646	**0.1348**	0.1253	0.0477
Bootstrapping	0.0965	0.2849	0.0350	0.3260	0.0906	0.0489
PNR	**0.0221**	**0.0217**	0.0606	0.1484	0.1058	0.0548
ILP + Bootstrapping	0.0961	0.2844	**0.0295**	0.3290	0.0902	0.0438
ILP + PNR	0.0370	0.0242	0.0590	**0.1348**	0.1047	0.0477
Bootstrapping + PNR	0.0495	0.1572	0.0336	0.1404	**0.0586**	0.0489
ILP + Bootstrapping + PNR	0.0886	0.2734	**0.0295**	0.2582	0.0879	0.0427

Table 2: CMD scores across all combinations of our methods on the public validation sets (lower is better). We **bold** the best method per column and underline any result better than the greedy baseline.

	IOI			MCQA		ARC (E)
Method	GPT-2	Qwen-2.5	Gemma-2	Qwen-2.5	Gemma-2	Gemma-2
Greedy	2.9302	2.2553	5.4410	1.2421	1.8217	1.9217
ILP	2.9153	2.1928	5.3489	**1.3844**	1.8240	1.9197
Bootstrapping	**3.1865**	**2.6196**	**5.6091**	1.2548	1.9142	1.9846
ILP + Bootstrapping	3.1772	2.5894	5.4410	1.3510	**1.9261**	**2.0099**

Table 3: CPR scores across all combinations of our methods on the public validation sets (higher is better). We **bold** the best method per column and underline any result better than the greedy baseline.

methods demonstrate improvements over the baseline across almost all models and tasks evaluated.

5.1 Ablations

We conduct an ablation study to evaluate our method design choices. We ablate both the combination of methods and method-specific hyperparameters. Tables 2 and 3 present the CMD and CPR results, respectively, across different model and task combinations. All ablations are performed on the MIB validation sets. Additional ablation studies on the number of bootstraps and PNR ratio values are provided in Appendix A.

6 Discussion and Limitations

Our work focused on the second stage of circuit building: using edge scores to select a fully connected sub-graph. This aspect has received limited attention in prior work, with most approaches relying on naive top-n selection or greedy algorithms (Hanna et al., 2024; Conmy et al., 2023). We demonstrated that the greedy algorithm can be improved through techniques such as ILP, bootstrapping, and PNR. However, our methods come with important limitations. First, ILP optimization scales poorly with the number of edges, limiting applicability to larger models while providing only modest faithfulness gains over the greedy approach. Second, the optimal ratio of positive edges varies across models and tasks, requiring task-specific tuning. These limitations increase the computational overhead required to achieve higher-faithfulness graphs from existing edge scores.

Despite these limitations, our results show that principled edge selection improves faithfulness and enables more robust circuit discovery.

While our early experiments showed that ILP significantly outperformed greedy methods at maximizing total edge scores, this improvement did not translate to significantly higher faithfulness scores. This suggests a gap between the scores generated by state-of-the-art attribution methods (Hanna et al., 2024) and the "ground truth" edge importance scores. Better score attribution methods could thus potentially unlock the full benefits of using ILP as an optimal solution to graph building.

References

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.

John Forrest, Ted Ralphs, Stefan Vigerske, Haroldo Gambini Santos, Lou Hafer, Bjarni Kristjansson, jpfasano, Edwin Straver, Jan-Willem, Miles Lubin, rlougee, a andre, jpgoncal1, Samuel Brito, h-i gassmann, Cristina, Matthew Saltzman, tosttost, and to st. 2024. coin-or/cbc.

Michael Hanna, Sandro Pezzelle, and Yonatan Belinkov. 2024. Have faith in faithfulness: Going beyond circuit overlap when finding model mechanisms. In *First Conference on Language Modeling*.

Stuart Mitchell, Michael OSullivan, and Iain Dunning. 2011. Pulp: a linear programming toolkit for python. *The University of Auckland, Auckland, New Zealand*, 65:25.

Aaron Mueller, Atticus Geiger, Sarah Wiegreffe, Dana Arad, Iván Arcuschin, Adam Belfki, Yik Siu Chan, Jaden Fried Fiotto-Kaufman, Tal Haklay, Michael Hanna, Jing Huang, Rohan Gupta, Yaniv Nikankin, Hadas Orgad, Nikhil Prakash, Anja Reusch, Aruna Sankaranarayanan, Shun Shao, Alessandro Stolfo, and 4 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.

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.

Daking Rai, Yilun Zhou, Shi Feng, Abulhair Saparov, and Ziyu Yao. 2024. A practical review of mechanistic interpretability for transformer-based language models. *ArXiv*, abs/2407.02646.

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.

Naomi Saphra and Sarah Wiegreffe. 2024. Mechanistic? In *The 7th BlackboxNLP Workshop*.

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*.

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*.

Sarah Wiegreffe, Oyvind Tafjord, Yonatan Belinkov, Hannaneh Hajishirzi, and Ashish Sabharwal. 2024. Answer, assemble, ace: Understanding how lms answer multiple choice questions. *arXiv preprint arXiv:2407.15018*.

L.A. Wolsey. 1998. *Integer Programming*. Wiley Series in Discrete Mathematics and Optimization. Wiley.

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.

A Hyper-parameter Ablations

We report additional ablations on method-specific hyperparameters. For bootstrapping, we tested different numbers of iterations with $\tau \in \{5, 10, 15\}$. For the PNR method, we explored values of PNR $\in \{0.3, 0.4, 0.45, 0.5, 0.55, 0.6, 0.7, 0.8, 0.9\}$. Tables 4 and 5 present the results for bootstrapping and PNR ablations, respectively. These results motivated our selection of $n = 10$ bootstrapping iterations and model-task-specific PNR values that yielded the best validation performance.

B Implementation Details

Our implementation is based on the code provided in MIB.[2] The EAP-IG attribution scores were computed using the implementation by Hanna et al.. Our ILP approach employs Pulp (Mitchell et al., 2011) as a linear integer programming modeler, using Cbc (Forrest et al., 2024) as the solver.

[2] https://github.com/hannamw/MIB-circuit-track

		IOI			MCQA		ARC (E)
Method	**n**	GPT-2	Qwen-2.5	Gemma-2	Qwen-2.5	Gemma-2	Gemma-2
Bootstrapping	5	3.1617	2.5794	**5.6895**	**1.3918**	1.8130	1.9615
Bootstrapping	10	3.1865	**2.6196**	5.6091	1.2548	**1.9142**	**1.9846**
Bootstrapping	15	**3.1988**	2.5527	5.4732	1.0808	1.9136	1.9616

Table 4: CPR scores (higher is better) across different amounts of bootstrap iterations n. **Bold** indicates best result per task-model combination.

	IOI			MCQA		ARC (E)
Method	GPT-2	Qwen-2.5	Gemma-2	Qwen-2.5	Gemma-2	Gemma-2
ILP + PNR 0.3	0.0416	0.0322	**0.0590**	0.1852	**0.1047**	**0.0477**
ILP + PNR 0.4	0.0416	0.0322	**0.0590**	0.1852	**0.1047**	**0.0477**
ILP + PNR 0.45	0.0416	0.0322	**0.0590**	0.1852	**0.1047**	**0.0477**
ILP + PNR 0.5	0.0416	0.0322	**0.0590**	0.1852	0.1068	**0.0477**
ILP + PNR 0.55	0.0416	0.0322	**0.0590**	0.1852	0.1068	**0.0477**
ILP + PNR 0.6	**0.0370**	**0.0242**	0.0646	**0.1348**	0.1253	**0.0477**
ILP + PNR 0.7	**0.0370**	**0.0242**	0.0646	**0.1348**	0.1253	**0.0477**
ILP + PNR 0.8	**0.0370**	**0.0242**	0.0646	**0.1348**	0.1253	**0.0477**
ILP + PNR 0.9	**0.0370**	**0.0242**	0.0646	**0.1348**	0.1253	**0.0477**

Table 5: CMD scores (lower is better) across different PNR values. **Bold** indicates best result per task-model combination.