Discovering Transformer Circuits via a Hybrid Attribution and Pruning Framework

Hao Gu[*]
University of California, Los Angeles
haogu@ucla.edu

Vibhas Nair[*]
University of North Carolina, Chapel Hill
vibhasn@unc.edu

Amrithaa Ashok Kumar
Ohio State University
ashokkumar.15@osu.edu

Ryan Lagasse[†]
Algoverse AI Research
ryan@algoverseairesearch.org

Abstract

Interpreting language models often involves circuit analysis, which aims to identify sparse subnetworks, or *circuits*, that accomplish specific tasks. Existing circuit discovery algorithms face a fundamental trade-off: attribution patching is fast but unfaithful to the full model, while edge pruning is faithful but computationally expensive. This research proposes a hybrid attribution and pruning (HAP) framework that uses attribution patching to identify a high-potential subgraph, then applies edge pruning to extract a faithful circuit from it. We show that HAP is 46% faster than baseline algorithms without sacrificing circuit faithfulness. Furthermore, we present a case study on the Indirect Object Identification task, showing that our method preserves cooperative circuit components (e.g. S-inhibition heads) that attribution patching methods prune at high sparsity. Our results show that HAP could be an effective approach for improving the scalability of mechanistic interpretability research to larger models[3].

1 Introduction

Large language models (LLMs) are being increasingly deployed in high-stakes settings, motivating the need to uncover their "black-box" Alishahi et al. (2019) nature and understand how they "think." Hubinger (2020); Zhang et al. (2021) This is a key goal of mechanistic interpretability, a field focused on understanding transformer Vaswani et al. (2017) model behavior by analyzing the interactions between subnetworks of attention heads and multi-layer perceptrons (MLPs) Vig et al. (2020); Sharkey et al. (2025). The most common approach to mechanistic 8bis through circuit analysis, which identifies sparse subnetworks, or "circuits", responsible for specific behaviors Olah et al. (2020); Olah (2022); Erdogan (2025). Manual circuit discovery methods, such as Wang et al. (2022), have largely been replaced by automated approaches like Automated Circuit DisCovery (ACDC) Conmy et al. (2023), which uses a greedy search algorithm to ablate edges one by one.

To address the computational cost of ACDC, faster algorithms such as Edge Attribution Patching (EAP) Syed et al. (2023) and Edge Pruning (EP) Bhaskar et al. (2024) have been proposed. However, existing circuit discovery algorithms struggle to scale with larger models without sacrificing performance Hanna et al. (2024); Hsu et al. (2025); Zhang and Dong (2025). EAP uses a first-order Taylor series approximation to ablate all edges simultaneously. Although faster than ACDC, the

[*] Equal contribution.

[†] Corresponding author.

[3] Our code is available at: `https://github.com/nairvibhas18/HAP_Paper_NeurIPS_2025`

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

first-order approximations show low faithfulness to the full model. Conversely, EP efficiently applies a gradient-based pruning algorithm to discover circuits in parallel. Despite scaling well to larger models while maintaining exceptional circuit faithfulness, EP requires significant compute power.

This research proposes a novel Hybrid Attribution and Pruning (HAP) framework to enhance the scalability and maintain the faithfulness of discovered circuits. We leverage EAP to quickly filter out the majority of unimportant edges. This EAP-identified subgraph gives a narrowed search space for EP to find faithful circuits.

In summary, our main contributions are the following:

1. We propose a novel framework (HAP) that improves efficiency and preserves the faithfulness of discovered circuits.
2. We show that HAP matches or outperforms existing methods in efficiency and faithfulness.
3. We demonstrate in an IOI case study that HAP finds the often-missed S-Inhibition heads, preserving the quality of discovered circuits.

2 Related Works

Automated Circuit Discovery Algorithms such as ACDC construct computational graphs where nodes represent model components and edges represent information flow Conmy et al. (2023). ACDC recursively applies activation patching—replacing activations with those from "corrupted" examples—removing edges that do not degrade task metric performance Syed et al. (2023). This greedy search can rediscover known circuits, but it is computationally expensive for larger models or datasets due to the requirement for many forward passes, with scalability limited by the number of edges evaluated Conmy et al. (2023).

Edge Pruning and Optimization-based Methods frame circuit discovery as a gradient-based optimization problem, where edges between components of a model's computational graph are pruned using binary masks over edges Bhaskar et al. (2024). This method allows for finer-grained and more faithful recovery of causal pathways, but requires architectural modifications and additional memory for scalability. EP can parallelize training across multiple GPUs, which enables EP to scale to large models (e.g., CodeLlama-13B) and complex datasets, recovering circuits that are both smaller and more interpretable than those produced by prior methods Bhaskar et al. (2024).

Attribution and Gradient-based Approximations, like EAP, propose gradient-based, first-order approximations to activation patching Syed et al. (2023), enabling simultaneous computation of edge importance scores with one backward and two forward passes. EAP efficiently identifies circuits that align closely with those found by ACDC, as measured by ROC/AUC when compared to manually curated circuit ground-truths, but can miss critical component interactions due to its linear approximation and reduced faithfulness Bhaskar et al. (2024).

3 Methods

The HAP framework operates by leveraging EAP to perform a global search, quickly removing low-importance edges to isolate higher-importance edges. This EAP-identified subgraph gives a narrowed search space for the precise pruning algorithm, EP. The framework is shown in Figure 1.

3.1 Step 1: Computational Graph Construction

We start by representing our model as a computational graph following the convention of Bhaskar et al. (2024), where components of the Transformer architecture, namely attention layers and MLPs, are the nodes and the edges between any two nodes represent the connection between the output of one node to the input of the other node. The full model, in our case GPT-2 Small (from Radford et al. (2019)), can be represented at this granularity, and a circuit is a computational subgraph consisting of a set of edges that describe the full model's behavior on a particular task (see Section 4.1).

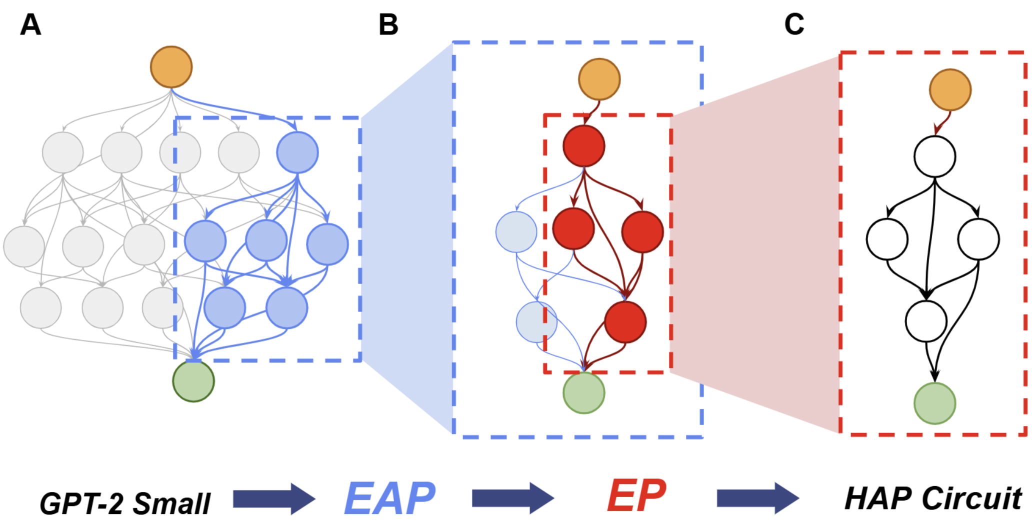


Figure 1: The HAP framework. A) EAP is applied on the full GPT2-small model to efficiently mask most low-importance edges, identifying a high potential subgraph based on the top-k edges. B) EP is applied to accurately discover a circuit within the EAP-derived subgraph, yielding a C) HAP circuit.

3.2 Step 2: Edge Attribution Patching

We then use Edge Attribution Patching to quickly get absolute attribution scores that measure the importance of all edges in the computational graph using:

$$L(\mathbf{x} \mid e_{\text{ablated}}) - L(\mathbf{x}) \;\approx\; (e_{\text{clean}} - e_{\text{ablated}})^{\top} \frac{\partial L(\mathbf{x} \mid e_{\text{clean}})}{\partial e_{\text{clean}}} \tag{1}$$

where $L(\mathbf{x})$ is the logit loss, e_{ablated} denotes predictions after ablation of the target edge, and the right side of Equation (1) represents the computed absolute attribution score Syed et al. (2023). After ranking the scores, we keep the top-k edges for further processing.

3.3 Step 3: Subgraph Selection and Edge Pruning

From here, edges with low attribution scores are masked to produce a high-potential subgraph. The masking threshold balances sparsity against the retention of potentially cooperative but weakly attributed components (e.g. S-inhibition heads). We use the edges found by EAP to "jumpstart" the EP training process. EP proceeds by optimizing a binary mask $z \in [0, 1]^{N_{\text{edge}}}$ to minimize output divergence between the original and pruned graphs, under a targeted sparsity constraint:

$$1 - \frac{|H|}{|G|} \geq c \tag{2}$$

This step is performed via gradient-based optimization using clean and corrupted examples Bhaskar et al. (2024).

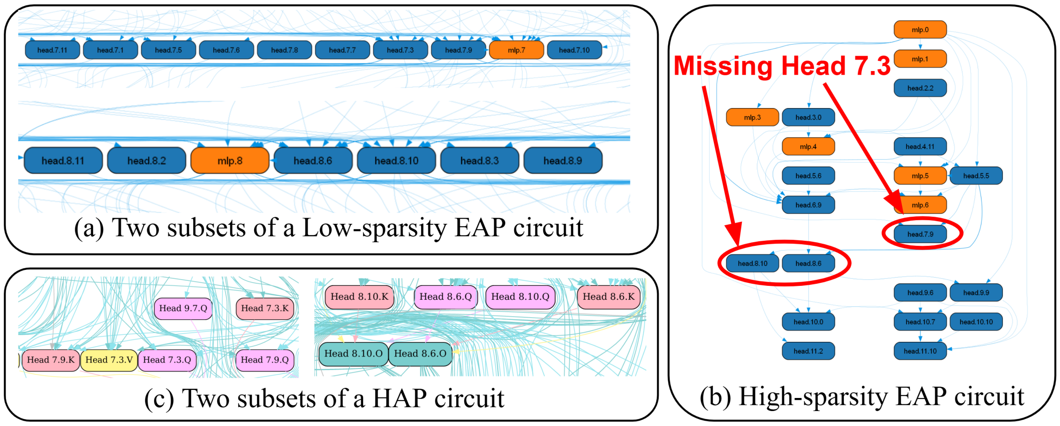
4 Experiment

4.1 Task Description

The task being studied is defined by a set of prompts that elicit a clearly defined response from the model predictions. We study the Indirect Object Identification (IOI) task, which is in the general format of "*When Dylan and Ryan went to the store, Dylan gave a popsicle to* → *Ryan*". We use Wang et al. (2022)'s prompt templates to generate an IOI dataset of 200 randomly selected examples with lexical and syntactic diversity, each for training and validation. Our test split involved 36,084 examples as per Bhaskar et al. (2024).

Algorithm	Sparsity	GPT-2 Small			
		Accuracy ↑	**Logit Diff** ↑	**KL** ↓	**Runtime** (s) ↓
EAP	94±0.5%	0.698	3.13	–	4
EP	94±0.5%	0.772	3.48	0.190	2921
HAP	94±0.5%	0.759	3.42	0.188	1579

Table 1: Efficiency of HAP compared to existing works.



(a) Two subsets of a Low-sparsity EAP circuit

(b) High-sparsity EAP circuit

(c) Two subsets of a HAP circuit

Figure 2: Recovered IOI circuits. While EAP on its own is unable to recover all S-Inhibition Heads at high sparsity, HAP preserves S-Inhibition Heads because it only uses EAP at low sparsity.

4.2 Experimental Setup

We evaluate our Hybrid Attribution and Pruning (HAP) framework on the Indirect Object Identification (IOI) task using GPT-2 Small (117M). The attribution score threshold in EAP is set very low to preserve possibly cooperative edges that might score low individually. For EP, we use the hyperparameters as detailed in Bhaskar et al. (2024). All training runs were performed on one NVIDIA H100 GPU. We quantify circuit quality with faithfulness via KL divergence and logit difference between model predictions and circuit predictions, and report standard metrics such as accuracy and runtime.

5 Results

5.1 HAP vs Existing Methods

5.2 Case Study: S-inhibition Heads in IOI

To compare the performance between different models, we leverage manually discovered circuits in Wang et al. (2022) as a reference to calculate the accuracy of circuits recovered by automatic methods. As shown in Table 1, HAP outperforms EAP in accuracy while having only slightly lower accuracy compared to EP. Similarly, circuits recovered by HAP are much more faithful to the full model compared to EAP (when comparing logit difference), while also maintaining similar faithfulness to EP circuits. It is shown in both KL divergence and logit difference metrics that HAP circuits are only slightly less faithful than EP circuits.

When GPU and target sparsity are controlled, HAP is at least 46% faster than EP while maintaining high accuracy and faithfulness to the full model. This shows that HAP can be a valuable framework for reducing the computational cost of circuit discovery, possibly enabling scalability to larger models.

To present the qualitative advantages of our hybrid framework, we present a case study on the IOI task in GPT-2 Small (see Section 4.1). In IOI, the role of S-inhibition heads (or Subject-Inhibition Heads) is cooperative: they suppress the Name Mover Heads from incorrectly flagging the subject of a sentence due to their proximity to the verb. Thus S-Inhibition Heads, although critical for accurate task performance, are difficult to detect due to the low individual importance assigned by methods like Syed et al. (2023) at high sparsity.

We found that existing methods do not recover the complete circuit. For example, EAP falls short since S-inhibition heads do not receive high attribution scores, causing them to be undervalued as shown in Figure 2B. In contrast, HAP successfully captures the complete, functional circuit. By first using EAP to define a constrained search space with a generous threshold, we created a "safe zone" that retains these S-inhibition heads despite their low individual scores. The subsequent EP algorithm, operating on this focused and less noisy subgraph, correctly identifies their cooperative importance. As shown in Figure 2C, the S-inhibition heads 7.3, 7.9, 8.6, and 8.10 are all preserved by HAP. This serves as qualitative evidence that our method is not only efficient but also preserves cooperative components that are overlooked by prior approaches.

6 Limitations

Our experiments are conducted exclusively on the IOI task with the GPT-2 Small model. Although this task is a well-established benchmark for mechanistic interpretability, further evaluation on a broader set of models and tasks is necessary to assess the generality, robustness, and scalability of HAP. Furthermore, the current implementation has not optimized the threshold to select edges during the EAP stage, which will require future hyperparameter tuning. We also acknowledge that variations in the generated training dataset may result in minor performance differences across different runs.

7 Conclusion

We introduce HAP, a hybrid framework that resolves the longstanding speed-faithfulness tradeoff in circuit discovery by strategically sequencing EAP, a fast and approximate algorithm, with EP, a fine-grained and precise one. Our experiments show this approach is not only 46% faster than EP while maintaining comparable faithfulness, but is also qualitatively superior. As demonstrated in our IOI case study, HAP successfully preserves the S-inhibition heads that attribution methods fail to recover in isolation. The results challenge the notion that the speed-faithfulness trade-off is fundamental and provide a simple framework to scale up future mechanistic interpretability research to interpret larger models.

References

Afra Alishahi, Grzegorz Chrupała, and Tal Linzen. Analyzing and interpreting neural networks for nlp: A report on the first blackboxnlp workshop. *Natural Language Engineering*, 25(4):543–557, 2019. doi: 10.1017/S135132491900024X.

Adithya Bhaskar, Alexander Wettig, Dan Friedman, and Danqi Chen. Finding Transformer Circuits with Edge Pruning. *arXiv preprint arXiv:2406.16778*, 2024. URL `http://arxiv.org/abs/2406.16778`.

Steven Cao, Victor Sanh, and Alexander Rush. Low-Complexity Probing via Finding Subnetworks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 960–966, Online, June 2021. Association for Computational Linguistics. URL `https://aclanthology.org/2021.naacl-main.74`.

Hang Chen, Jiaying Zhu, Xinyu Yang, and Wenya Wang. Rethinking Circuit Completeness in Language Models: AND, OR, and ADDER Gates. *arXiv preprint arXiv:2505.10039*, 2025. URL `http://arxiv.org/abs/2505.10039`.

Arthur Conmy, Augustine N. Mavor-Parker, Aengus Lynch, Stefan Heimersheim, and Adrià Garriga-Alonso. Towards Automated Circuit Discovery for Mechanistic Interpretability. *arXiv preprint arXiv:2304.14997*, 2023. URL `http://arxiv.org/abs/2304.14997`.

Ege Erdogan. Automated Circuit Discovery for Mechanistic Interpretability. 2025.

Michael Hanna, Sandro Pezzelle, and Yonatan Belinkov. Have Faith in Faithfulness: Going Beyond Circuit Overlap When Finding Model Mechanisms. *arXiv preprint arXiv:2403.17806*, 2024. URL http://arxiv.org/abs/2403.17806.

Stefan Heimersheim and Jett Janiak. A circuit for Python docstrings in a 4-layer attention-only transformer. *Alignment Forum*, February 2023. URL https://www.alignmentforum.org/posts/u6KXXmKFbXfWzoAXn/a-circuit-for-python-docstrings-in-a-4-layer-attention-only.

HF Canonical Model Maintainers. gpt2 (revision 909a290). *Hugging Face*, 2022. doi: 10.57967/hf/0039. URL https://huggingface.co/gpt2.

Aliyah R. Hsu, Georgia Zhou, Yeshwanth Cherapanamjeri, Yaxuan Huang, Anobel Y. Odisho, Peter R. Carroll, and Bin Yu. Efficient Automated Circuit Discovery in Transformers using Contextual Decomposition. *arXiv preprint arXiv:2407.00886*, 2025. URL http://arxiv.org/abs/2407.00886.

Evan Hubinger. An overview of 11 proposals for building safe advanced ai. *arXiv preprint arXiv:2012.07532*, 2020. URL https://arxiv.org/abs/2012.07532.

Jianwei Li, Yijun Dong, and Qi Lei. Greedy Output Approximation: Towards Efficient Structured Pruning for LLMs Without Retraining. *arXiv preprint arXiv:2407.19126*, 2024. URL http://arxiv.org/abs/2407.19126.

Gui Ling, Ziyang Wang, Yuliang Yan, and Qingwen Liu. SlimGPT: Layer-wise Structured Pruning for Large Language Models.

Samuel Marks, Can Rager, Eric J. Michaud, Yonatan Belinkov, David Bau, and Aaron Mueller. Sparse Feature Circuits: Discovering and Editing Interpretable Causal Graphs in Language Models. *arXiv preprint arXiv:2403.19647*, 2025. URL http://arxiv.org/abs/2403.19647.

Paul Michel, Omer Levy, and Graham Neubig. Are Sixteen Heads Really Better than One? In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/hash/2c601ad9d2ff9bc8b282670cdd54f69f-Abstract.html.

Philipp Mondorf, Sondre Wold, and Barbara Plank. Circuit Compositions: Exploring Modular Structures in Transformer-Based Language Models. *arXiv preprint arXiv:2410.01434*, 2025. URL http://arxiv.org/abs/2410.01434.

Neel Nanda. Attribution Patching: Activation Patching At Industrial Scale. URL https://www.neelnanda.io/mechanistic-interpretability/attribution-patching.

Chris Olah. Mechanistic Interpretability, Variables, and the Importance of Interpretable Bases. 2022. URL https://www.transformer-circuits.pub/2022/mech-interp-essay.

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. doi: 10.23915/distill.00024.001. URL https://distill.pub/2020/circuits/zoom-in.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019. URL https://api.semanticscholar.org/CorpusID:160025533.

Lee Sharkey, Bilal Chughtai, Joshua Batson, Jack Lindsey, Jeff Wu, Lucius Bushnaq, Nicholas Goldowsky-Dill, Stefan Heimersheim, Alejandro Ortega, Joseph Bloom, Stella Biderman, Adria Garriga-Alonso, Arthur Conmy, Neel Nanda, Jessica Rumbelow, Martin Wattenberg, Nandi Schoots, Joseph Miller, Eric J. Michaud, Stephen Casper, Max Tegmark, William Saunders, David Bau, Eric Todd, Atticus Geiger, Mor Geva, Jesse Hoogland, Daniel Murfet, and Tom McGrath. Open problems in mechanistic interpretability. *arXiv preprint arXiv:2501.16496*, 2025. URL https://arxiv.org/abs/2501.16496.

Aaquib Syed, Can Rager, and Arthur Conmy. Attribution Patching Outperforms Automated Circuit Discovery. *arXiv preprint arXiv:2310.10348*, 2023. URL `http://arxiv.org/abs/2310.10348`.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017. URL `https://arxiv.org/abs/1706.03762`.

Jesse Vig, Sebastian Gehrmann, Yonatan Belinkov, Sharon Qian, Daniel Nevo, Simas Sakenis, Jason Huang, Yaron Singer, and Stuart Shieber. Causal mediation analysis for interpreting neural nlp: The case of gender bias. *arXiv preprint arXiv:2004.12265*, 2020. URL `https://arxiv.org/abs/2004.12265`.

Kevin Ro Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. Interpretability in the Wild: a Circuit for Indirect Object Identification in GPT-2 Small. In *The Eleventh International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=NpsVSN6o4ul`.

Ziheng Wang, Jeremy Wohlwend, and Tao Lei. Structured Pruning of Large Language Models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6151–6162, Online, November 2020. Association for Computational Linguistics. URL `https://aclanthology.org/2020.emnlp-main.496`.

Mengzhou Xia, Zexuan Zhong, and Danqi Chen. Structured Pruning Learns Compact and Accurate Models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1513–1528, Dublin, Ireland, May 2022. Association for Computational Linguistics. URL `https://aclanthology.org/2022.acl-long.107`.

Lin Zhang, Wenshuo Dong, and et al. EAP-GP: Mitigating Saturation Effect in Gradient-based Automated Circuit Identification. *arXiv preprint arXiv:2502.06852*, 2025. URL `http://arxiv.org/abs/2502.06852`.

Yu Zhang, Peter Tiňo, Aleš Leonardis, and Ke Tang. A survey on neural network interpretability. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 5(5):726–742, 2021. doi: 10.1109/TETCI.2021.3100641.

A IOI Dataset Generation

In Table 2, we provide the full set of IOI templates from Wang et al. (2022) used to generate our dataset described in Section 4.1. Names were sampled from a list of 100 common English first names, while places and objects were selected from a curated set of 20 frequent options.

IOI prompt templates
Then, [B] and [A] went to the [PLACE]. [B] gave a [OBJECT] to [A]
bThen, [B] and [A] had a lot of fun at the [PLACE]. [B] gave a [OBJECT] to [A]
Then, [B] and [A] were working at the [PLACE]. [B] decided to give a [OBJECT] to [A]
Then, [B] and [A] were thinking about going to the [PLACE]. [B] wanted to give a [OBJECT] to [A]
Then, [B] and [A] had a long argument, and afterwards [B] said to [A]
After [B] and [A] went to the [PLACE], [B] gave a [OBJECT] to [A]
When [B] and [A] got a [OBJECT] at the [PLACE], [B] decided to give it to [A]
When [B] and [A] got a [OBJECT] at the [PLACE], [B] decided to give the [OBJECT] to [A]
While [B] and [A] were working at the [PLACE], [B] gave a [OBJECT] to [A]
While [B] and [A] were commuting to the [PLACE], [B] gave a [OBJECT] to [A]
After the lunch, [B] and [A] went to the [PLACE]. [B] gave a [OBJECT] to [A]
Afterwards, [B] and [A] went to the [PLACE]. [B] gave a [OBJECT] to [A]
Then, [B] and [A] had a long argument. Afterwards [B] said to [A]
The [PLACE] [B] and [A] went to had a [OBJECT]. [B] gave it to [A]
Friends [B] and [A] found a [OBJECT] at the [PLACE]. [B] gave it to [A]

Table 2: Templates used in the IOI dataset. The table displays templates following the "BABA" pattern; templates with the "ABBA" pattern were also employed but are omitted here for conciseness.

B Connecting EAP to EP

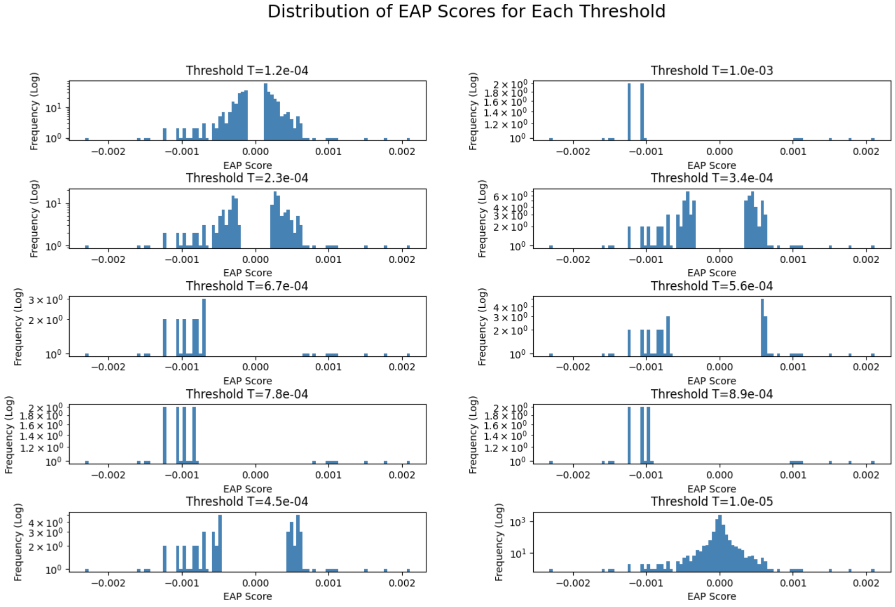


Figure 3: Attribution score distribution over different EAP thresholds.

To map the high-potential edges identified by EAP to the binary masks z of EP, we first show that EAP attribution scores are generally normally distributed (Figure 3). Then, we normalize the output attribution scores to a range $\in [-1, 1]$. To integrate the normalized attribution scores into the binary masks z of EP, we create the initial $log\alpha$ tensor. We then modify the EP initialization by changing the relevant mask parameters using the computed $log\alpha$ tensor. EP then undergoes training.