言語モデルの機械論的解釈可能性ツールの調査

Survey of tools for mechanistic interpretability of transformer language models

中内 遼吾 [1*]

Ryogo Nakauchi[1*]

[1] 東京都立産業技術大学院大学 Advanced Institute of Industrial Technology
*Corresponding author: Ryogo Nakauchi, nakauchi-ryogo@aiit.ac.jp

Abstract Tools for mechanistic interpretability (MI) of transformer language models provide systematic means to visualize and intervene in model activations. Analyses conducted with these tools help improve the explainability and interpretability of language model outputs. While analyzing internal model activations previously required advanced expertise, the emergence of increasingly mature MI tools has gradually lowered this barrier. This paper discusses the current trends in these MI tools.

Keywords mechanistic interpretability; large language models

1 はじめに

言語モデルの機械論的解釈可能性（MI: Mechanistic interpretability）ツールは，モデル内部の活性化状態の可視化と介入を行う MI 技術を簡易かつ系統的に用いるための機能を提供する．これを使って分析を施すことで，言語モデルの出力結果に対する説明性や解釈性の確保を一定程度目指すことができる．従来はモデル内部の活性化状態を分析するためには高度な専門知識を必要としたが，成熟しつつある MI ツールの登場によって徐々にその敷居が下がりつつある．

言語モデルが様々な分野の情報システムに組み込まれるようになったことで，その内部機序分析の必要性はますます増してきている．言語モデルを使うことに伴うハルシネーションの問題やモデル内部がブラックボックスであることの問題が意識され，組み込む情報システムの性質によっては説明性をどのように確保するかといった論点が重要になるケースがある．言語モデルは巨大なアーキテクチャと複雑な内部表現を持つため，その内部機序の全てが明らかになっているわけではない．しかしながら，MI ツールを使うことで探索的に解釈性を向上させることができる．

以下，MI 技術の適用レイヤごとに章を分け，各分野のツールについて述べる．様々に提案されている MI ツールのうち，特に Transformer ベースの言語モデルを明示的に対象とするものを扱う．

2 活性化状態の可視化・介入基盤

本章では，活性化状態の可視化・介入基盤レイヤのツールについて述べる． TransformerLens[1] と nnsight[2, 3] は，Transformer ベースの言語モデル内部の活性化状態を分析するための可視化・介入基盤の定番ライブラリとして定着している．TransformerLens は分析用にフック点を埋め込んで独自に再実装した HookedTransformer を使い，各点からキャッシュした層及びサブモジュール単位の活性化をもとに内部表現を分析する機能をパッケージ化している．主な機能には活性化キャッシュ，フック登録，パッチングのユーティリティ，可視化などがある．上流実装の個別性を抽象化することで系統的なアブレーションや活性化パッチング，ヘッド単位の分析を可能にしている．

TransformerLens と連携した可視化を簡易に行えるライブラリには CircuitsVis[4]がある．これは注意パターンやトークンごとの寄与を対話的に表示する機能を備える．Python と JavaScript の両方で利用することができる．

TransformerLens と並ぶもう一つの定番ライブラリである nnsight は，Hugging Face[5] の既存モデルをラップし，トレース文脈で入出力活性や勾配の追跡・介入を指定できる実験環境である．分析対象としてパラメータ規模の巨大なモデルを扱うケースに対応するため，リモート実行基盤である National Deep Inference Fabric[6]との連携機能が用意されている．これを使うことでリモート GPU 上に展開したモデルに対しても分析を行うことができる．新しいアーキテクチャを素早く扱いたい場合や，手持ちの計算機環境上に展開するのが難しい大規模パラメータのモデルを分析したい場合に利便性が高い．

nnterp[7]は NNsight を更に抽象化して可用性を高めるためのラッパである．NNsight をそのまま使う場合は分析対象とするコンポーネントを指定するときにモデルごとの命名規則の違いを考慮する必要がある．nnterp を使うことで，どのモデルを対象とした場合でも共通の指定方法で処理を実行できるようになる．それにより，複数のモデルを横断分析する際に簡素な実装で完結させることができる．

3 特徴抽出

本章では，特徴抽出のツールについて述べる．Sparse Auto-Encoder (SAE) [8]は，モデル内部の密な表現を解釈しやすい疎な表現に変換することで分析を図る特徴抽出の代表的手法である．SAE で得られる表現は人間が理解可能な特定の語彙属性や構文の役割などの特徴に対応することが期待される．

SAE を簡易に扱うための代表的なライブラリに，前述の TransformerLens から分かれて発展した SAELens[9]がある．このライブラリでは，訓練ループや学習済み SAE の Hugging Face からの読み込み，評価・分析ユーティリティなどがパッケージされている．これを用いることでモデル内部の活性化状態に対する SAE の訓練・評価・介入を系統的に実施できる．訓練ループでは，対象層の活性化キャプチャやバッチ構成，正則化，スパース率やコード次元を管理する機能を備える．評価ユーティリティでは特徴量の分離性と安定性を Weights & Biases[10]でロギングする機能を統合する．学習済み SAE の読み込みができるため，外部で訓練した特徴辞書の再評価や介入実

Received: 2025-10-31; Published online: 2026-02-28

験を行うことも可能である．

モデルの重みがオープンソースになっていない API 専用モデルのためのツールも近年公開されてきた．OpenAI は GPT-4 を使って学習した大規模な潜在変数を持つ SAE[11, 12]を公開している．その Web 版ビューア[13]では GPT-2 small に加えて GPT-4 について，層・位置・特徴を選択して活性化例と寄与度を可視化することができる．Google DeepMind の Gemma Scope[14, 15]では Gemma 2 に対する JumpReLU SAE が全ての層で公開されている．ランディングページやモデルの挙動評価，Hugging Face 上での重みも示されている．Anthropic も Claude 3 Sonnet の内部表現から学習した SAE を使ったビューアである Feature Browser[16, 17]を公開する．

SAE の有効性を確保するためには適宜必要なバリデーションを実施する必要がある．そうしたプロセスに必要な機能を統合した評価スイートに SAEBench[18, 19]がある．これは SAE の解釈可能性・分離性・応用有用性などの評価に関わる多数の指標を集約し，Neuronpedia[20]上に展開する Web UI[21]から対話的に利用することができる．異なる設定で学習された多数の SAE がまとめられてあり，横断比較が可能である．

上述した Neuronpedia は，SAE 特徴量の可視化・探索・共有を中心に据えたオープンプラットフォームであり，多数のオープン SAE の特徴量ダッシュボードのホストとして利用される．また，API とツール群により活性化状態の即時評価も可能である．SAE にとどまらず，知識編集やステアリングの機能も備える共有インフラとしても利用できるため，再現性と仮説検証の共有にも有用である．

4 回路追跡

本章では，回路追跡を効率化するツールについて述べる．回路追跡とは，モデル内部を計算グラフで表現し，分析対象とする出力傾向に対する寄与度の大きいサブグラフを見つけ出す手法である．モデル内部での情報伝達回路を明らかにすることで，モデルの振る舞いやコンポーネントの役割を理解する意義がある．

回路追跡の自動化を提案した初期の手法に Automated Circuit Discovery (ACDC) [22, 23]が挙げられる．この手法では分析対象とする一つの出力を起点として，出力側から計算グラフ上の各エッジに対して活性化パッチングを自動実行する．当該出力への影響が小さいエッジを反復的に除去していくことで寄与度が大きい回路を抽出するアプローチをとっている．

Edge Attribution Patching (EAP) [24, 25]は，活性化パッチングを線形近似することでこの過程の高速化を図った．網羅的なパッチングの代わりに順伝播と逆伝播の双方でエッジの重要度を計算し，寄与度が小さいエッジを削除して回復度を評価する．論文では既存の自動化手法に比べて ROC-AUC 等の識別指標で性能が上回ることが示されている．

Information Flow Routes (IFR) [26, 27]は，与えられた予測に対して重要度の高いノードとエッジを選択するために帰属を用い，トップダウンの回路グラフを構成する手法を提案する．Llama-2 を使った実験では，繰り返し現れるヘッドの役割やドメイン特化が示されている．手作業でテンプレートを作成せずにデータセットを集約できる予測ごとの回路グラフに重点を置く．EAP と同じく，パッチングよりも効率的であるため，より大規模なモデルやより広範なカバレッジへの拡張がしやすい．

IFR の考え方を組み込んだ対話的ツールに LM Transparency Tool[28, 29]がある．上位層の表現からヘッドやニューロンに至るまでのモデルの予測プロセスを追跡して変化量を特定のコンポーネントに帰属させ，計算の寄与度の大きい部分を検査用に可視化することができる．IFR などの帰属手法を基盤とすることで処理の高速化が図られており，探索的に可視化を行いたい場合にも適する．

回路追跡で得た仮説回路や特徴集合が特定の挙動に対して決定的であるかを確認する段階では因果検証のための手法が必要となる．Causal Scrubbing[30]は，分布操作下における挙動に対して仮説上のメカニズムが持つ寄与度を検証する手法として提案された．仮説が真のメカニズムを捉えている場合にのみ行動を維持するリサンプリング介入を定義することで，回路パッチングを一般化する考え方をとっている．

Tracr[31, 32]は，Restricted Access Sequence Processing プログラムを内部構造が既知のデコーダモデルにコンパイルすることで，発見されたパターンが正しいか否かを考えることなく手法の評価を行えるように図られている．Tracr モデルは，スパース特徴を圧縮する際の重ね合わせの検証やトークン頻度などのタスクにおける回路追跡や因果関係の検証に適する．

5 モデル編集と潜在ステアリング

本章では，モデル編集と潜在ステアリングについて述べる．モデル編集は，モデルの重みを書き換えることで特定の知識や挙動を更新する操作を指す．潜在ステアリングは推論時に隠れ状態にベクトルを加えるなどの線形操作を行い，出力を特定の方向に誘導するものである．これらは生成結果を制御するための手法であると同時に，モデル内部で事実知識や属性情報が集中している位置や集中の程度を検証するための手段としても用いられる．

Rank-One Model Editing (ROME) [33, 34]は，中間層 MLP の一部を事実知識が key-value ペアで保存された線形連想メモリとして扱う．知識編集するにあたって解析的に求められるランク 1 の重み更新を行うだけで済ませる軽量化された手法を提案する．MEMIT [35]はこれを数千の関連のバッチ編集に拡張し，より大きなモデルにスケーリングする．ここで直接的な重み介入を体系的に組織化できることが示されている．

Model Editor Networks using Gradient Decomposition (MEND) [36, 37]は，勾配の低ランク分解を利用して入力と誤差のベクトルを単一隠れ層の小規模 MLP で変換し，その出力から重み更新を作成して効率的に知識編集を図る手法である．検証結果では 10 億パラメータを超えるモデルでも単一 GPU で実行できる例が示されており，大規模モデルの局所編集を効率化することができる．

知識編集がモデルの重みを永続的に変更するアプローチを

とるのに対し，推論時の動的介入によってモデル制御に変更を加えるアプローチをとる手法に潜在ステアリング[38]がある．これは，隠れ状態を直接操作することで，ファインチューニングなしで生成を目標の文や属性へと導くことを目指す手法である．

これら各手法を統合して系統的に扱えるようにしたツールに EasyEdit[39, 40]と EasyEdit2[41, 42]がある．前者が知識編集，後者が潜在ステアリングに対応しており，様々な手法を単一 API で扱えるフレームワークとして提案された．知識編集では基本的な知識更新・挿入・消去に加えて，マルチモーダルにも対応する．また，複数のベンチマークを統合した評価機能も備えている．

6 軽量検証ツール

本格的なツールを利用する手前で，Jupyter Notebook などで短時間に可視化・初期検証を行うユースケースではより軽量なツールが適する場合もある．LLM 登場以前から使われている軽量可視化ツールである BertViz[43]や Ecco[44]は依然として LLM の MI にとっても有用である．これらは軽量な実行環境内であっても注意重みやトークンレベルの寄与を素早く可視化できる．より重い実験を行う前段階で，簡易的な確認を実施するのに適している．

Patchscopes[45]は，内部表現を分析するためにモデル自体を用いて自然言語の活性化状態を説明するフレームワークである．既往の語彙への投影手法と介入手法を統合し，初期層の分析や表現力の問題に対処する．よりパラメータ規模の大きなモデルを用いて小規模モデル内の表現を説明することが提案され，多段推論のエラー解析に似た応用例が示されている．

7 Vision Language モデルへの適用

本章では Vision Language モデル（VLM）応用のための MI ツールについて述べる．VLM は画像埋め込みとテキスト表現を同時に処理し，マルチモーダルな意味空間を生成する点で特有の内部構造を持つ．そのため，言語モデル用ツールをそのまま適用することは難しく，モダリティ横断に対応した新しい MI 基盤が模索されている．

Prisma[46, 47]は，視覚および動画変換モデルにおける機械論的解釈を目的としたツールであり，活性化キャッシュや回路追跡，SAE やトランスコーダの訓練機能などを統合する．初期研究では，言語より低いスパース率でも有効となる傾向や，SAE 再構成の注入でモデル損失が下がる場合のあることが示されている．

大規模 VLM の解析を目指したツールに LVLM-Interpret[48, 49]がある．これは大規模 VLM における回答生成時の寄与量が大きい画像パッチとテキストトークンを対話的に解析するフレームワークである．入力画像とテキストプロンプトをもとに，モデル出力に対して視覚的注意の寄与度マップを可視化する．

VLM-Lens[50, 51]は複数の VLM を横断的に扱い，指定した層の内部表現を抽出して特徴分布や層間差異を比較することを目的としたツールである．実験パラメータ設定を抽象化しており，モデルの固有差を吸収して横断的な解析を容易にしている．得られた内部表現から，特定の概念方向やモダリティ間対応を検出する試行方法も示されている．

8 おわりに

本稿では Python 等で利用可能なライブラリや Web UI で対話的に利用可能なものを中心に言語モデルの内部機序の理解に寄与する MI ツールについて述べた．多様な問題関心から提案されてきたツール群を整理するため，やや乱雑ではあるが概ねの傾向をもとに分類して述べた．しかしながら，主要な MI ツールは現在進行形で機能追加が行われている状況にある．また，複数の手法を統合する方向性もみられる．このため，特定の分類に収まっていないツールも多々あるのが実態である．

言語モデルの内部機序には未解明の点も多くあるが，MI 手法の発展は急速に進んでいる．今後も基礎的な研究が充実していくことによって，その知見を取り入れた MI ツール開発も継続的に進展していくはずである．ツールの成熟に伴う可用性の改善により，幅広いユースケースで言語モデル利用時の説明性が担保しやすくなることが期待できる．

参考文献

1. TransformerLens: A library for mechanistic interpretability of GPT-style language models. Github; Available: https://github.com/TransformerLensOrg/TransformerLens
2. nnsight: The nnsight package enables interpreting and manipulating the internals of deep learned models. Github; Available: https://github.com/ndif-team/nnsight
3. Fiotto-Kaufman J, Loftus AR, Todd E, Brinkmann J, Pal K, Troitskii D, et al. NNsight and NDIF: Democratizing access to open-weight foundation model internals. arXiv [cs.LG]. 2024. doi:10.48550/arXiv.2407.14561
4. CircuitsVis: Mechanistic Interpretability Visualizations using React. Github; Available: https://github.com/TransformerLensOrg/CircuitsVis
5. Hugging Face – The AI community building the future. Available: https://huggingface.co/
6. NSF National Deep Inference Fabric. Available: https://ndif.us/
7. Dumas C. nnterp: Unified access to Large Language Model modules using NNsight. Github; Available: https://github.com/Butanium/nnterp
8. Towards Monosemanticity: Decomposing Language Models With Dictionary Learning. Available: https://transformer-circuits.pub/2023/monosemantic-features/index.html
9. Bloom J. SAELens: Training Sparse Autoencoders on Language Models. Github; Available: https://github.com/jbloomAus/SAELens
10. Weights & Biases. In: Weights & Biases [Internet]. 12 Apr 2024 Available: https://wandb.ai/site/ja/
11. Gao L, la Tour TD, Tillman H, Goh G, Troll R, Radford A, et al. Scaling and evaluating sparse autoencoders. arXiv [cs.LG]. 2024. Available: http://arxiv.org/abs/2406.04093
12. sparse_autoencoder. Github; Available: https://github.com/openai/sparse_autoencoder
13. SAE viewer. Available: https://openaipublic.blob.core.windows.net/sparse-autoencoder/sae-viewer/index.html

14. Lieberum T, Rajamanoharan S, Conmy A, Smith L, Sonnerat N, Varma V, et al. Gemma Scope: Open sparse autoencoders everywhere all at once on Gemma 2. arXiv [cs.LG]. 2024. doi:10.48550/arXiv.2408.05147

15. google/gemma-scope · Hugging Face. Available: https://huggingface.co/google/gemma-scope

16. Scaling Monosemanticity: Extracting Interpretable Features from Claude 3 Sonnet. Available: https://transformer-circuits.pub/2024/scaling-monosemanticity/

17. Feature Browser. Available: https://transformer-circuits.pub/2024/scaling-monosemanticity/features/index.html

18. Karvonen A, Rager C, Lin J, Tigges C, Bloom J, Chanin D, et al. SAEBench: A comprehensive benchmark for sparse autoencoders in language model interpretability. arXiv [cs.LG]. 2025. doi:10.48550/arXiv.2503.09532

19. Karvonen A. SAEBench. Github; Available: https://github.com/adamkarvonen/SAEBench

20. Neuronpedia Docs. Available: https://docs.neuronpedia.org/

21. SAE Bench - Evals. Available: https://www.neuronpedia.org/sae-bench

22. Conmy A, Mavor-Parker AN, Lynch A, Heimersheim S, Garriga-Alonso A. Towards automated circuit Discovery for mechanistic interpretability. arXiv [cs.LG]. 2023. doi:10.48550/arXiv.2304.14997

23. Conmy A. Automatic-Circuit-Discovery. Github; Available: https://github.com/ArthurConmy/Automatic-Circuit-Discovery

24. Syed A, Rager C, Conmy A. Attribution patching outperforms automated circuit discovery. arXiv [cs.LG]. 2023. doi:10.48550/arXiv.2310.10348

25. Syed A. edge-attribution-patching: Code for my NeurIPS 2024 ATTRIB paper titled “Attribution Patching Outperforms Automated Circuit Discovery.” Github; Available: https://github.com/Aaquib111/edge-attribution-patching

26. Ferrando J, Voita E. Information flow routes: Automatically interpreting language models at scale. arXiv [cs.CL]. 2024. doi:10.48550/arXiv.2403.00824

27. Ferrando J, Voita E. Information flow routes: Automatically interpreting language models at scale. Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing. Stroudsburg, PA, USA: Association for Computational Linguistics; 2024. pp. 17432–17445. doi:10.18653/v1/2024.emnlp-main.965

28. llm-transparency-tool: LLM Transparency Tool (LLM-TT), an open-source interactive toolkit for analyzing internal workings of Transformer-based language models. *Check out demo at* https://huggingface.co/spaces/facebook/llm-transparency-tool-demo. Github; Available: https://github.com/facebookresearch/llm-transparency-tool

29. Tufanov I, Hambardzumyan K, Ferrando J, Voita E. LM Transparency Tool: Interactive tool for analyzing Transformer language models. arXiv [cs.CL]. 2024. doi:10.48550/arXiv.2404.07004

30. LawrenceC, Garriga-alonso A, Goldowsky-Dill N, ryan_greenblatt, Radhakrishnan A, Buck, et al. Causal Scrubbing: a method for rigorously testing interpretability hypotheses [Redwood Research]. 3 Dec 2022
Available: https://www.lesswrong.com/posts/JvZhhzycHu2Yd57RN/causal-scrubbing-a-method-for-rigorously-testing

31. Lindner D, Kramár J, Farquhar S, Rahtz M, McGrath T, Mikulik V. Tracr: Compiled transformers as a laboratory for interpretability. arXiv [cs.LG]. 2023. doi:10.48550/arXiv.2301.05062

32. tracr. Github; Available: https://github.com/google-deepmind/tracr

33. Meng K. rome: Locating and editing factual associations in GPT (NeurIPS 2022). Github; Available: https://github.com/kmeng01/rome

34. Meng K, Bau D, Andonian A, Belinkov Y. Locating and editing factual associations in GPT. Koyejo S, Mohamed S, Agarwal A, Belgrave D, Cho K, Oh A, editors. arXiv [cs.CL]. 2022. pp. 17359–17372.
Available: https://proceedings.neurips.cc/paper_files/paper/2022/hash/6f1d43d5a82a37e89b0665b33bf3a182-Abstract-Conference.html

35. Meng K. memit: Mass-editing thousands of facts into a transformer memory (ICLR 2023). Github; Available: https://github.com/kmeng01/memit

36. Mitchell E. mend: MEND: Fast Model Editing at Scale. Github; Available: https://github.com/eric-mitchell/mend

37. Mitchell E, Lin C, Bosselut A, Finn C, Manning CD. Fast model editing at scale. arXiv [cs.LG]. 2021. doi:10.48550/arXiv.2110.11309

38. Subramani N, Suresh N, Peters M. Extracting latent steering vectors from pretrained language models. Findings of the Association for Computational Linguistics: ACL 2022. Stroudsburg, PA, USA: Association for Computational Linguistics; 2022. pp. 566–581. doi:10.18653/v1/2022.findings-acl.48

39. EasyEdit: [ACL 2024] An Easy-to-use Knowledge Editing Framework for LLMs. Github; Available: https://github.com/zjunlp/EasyEdit

40. Wang P, Zhang N, Tian B, Xi Z, Yao Y, Xu Z, et al. EasyEdit: An easy-to-use knowledge editing framework for large language models. Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations). Stroudsburg, PA, USA: Association for Computational Linguistics; 2024. pp. 82–93. doi:10.18653/v1/2024.acl-demos.9

41. EasyEdit2.
Available: https://zjunlp.github.io/project/EasyEdit2/?utm_source=chatgpt.com

42. Xu Z, Wang S, Xu K, Xu H, Wang M, Deng X, et al. EasyEdit2: An easy-to-use steering framework for editing large language models. arXiv [cs.CL]. 2025. doi:10.48550/arXiv.2504.15133

43. Vig J. bertviz: BertViz: Visualize Attention in NLP Models (BERT, GPT2, BART, etc.). Github; Available: https://github.com/jessevig/bertviz

44. Alammar J. ecco: Explain, analyze, and visualize NLP language models. Ecco creates interactive visualizations directly in Jupyter notebooks explaining the behavior of Transformer-based language models (like GPT2, BERT, RoBERTA, T5, and T0). Github; Available: https://github.com/jalammar/ecco

45. Ghandeharioun A, Caciularu A, Pearce A, Dixon L, Geva M. Patchscopes: A Unifying Framework for Inspecting Hidden Representations of Language Models. Forty-first International Conference on Machine Learning. 2024.
Available: https://openreview.net/forum?id=5uwBzcn885

46. Joseph S, Suresh P, Hufe L, Stevinson E, Graham R, Vadi Y, et al. Prisma: An open source toolkit for mechanistic interpretability in vision and video. arXiv [cs.CV]. 2025. doi:10.48550/arXiv.2504.19475

47. ViT-Prisma: ViT Prisma is a mechanistic interpretability library for Vision and Video Transformers (ViTs). Github; Available: https://github.com/Prisma-Multimodal/ViT-Prisma

48. Stan GBM, Aflalo E, Rohekar RY, Bhiwandiwalla A, Tseng S-Y, Olson ML, et al. LVLM-interpret: An interpretability tool for large vision-language models. arXiv [cs.CV]. 2024. Available: http://arxiv.org/abs/2404.03118

49. Aflalo E. LVLM Interpret.
Available: https://intellabs.github.io/multimodal_cognitive_ai/lvlm_interpret/

50. Sheta H, Huang E, Wu S, Alenabi I, Hong J, Lin R, et al. From behavioral performance to internal competence: Interpreting vision-language models with VLM-Lens. arXiv [cs.CL]. 2025. doi:10.48550/arXiv.2510.02292

51. vlm-lens: [EMNLP 2025 Demo] Extracting internal representations from vision-language models. Beta version. Github; Available: https://github.com/compling-wat/vlm-lens