Latent-Explorerを用いた 論理推論問題に対するLLM推論過程の分析

Analysis of LLM Reasoning Processes for Logical Tasks using Latent-Explorer

木村五郎[1] *　尾崎知伸[1]
Goro Kimura[1], Tomonobu Ozaki[1]

[1] 日本大学 文理学部 情報科学科
[1] Department of Information Science, Nihon University

Abstract: While Large Language Models (LLMs) excel in logical tasks, evaluations based solely on final outputs remain insufficient for understanding their internal reasoning processes. To address this black-box nature, this study investigates the mechanisms of knowledge formation and updates during logical inference. We employ Latent-Explorer, an activation-patching-based method, to extract layer-wise propositions from Llama-2-7b on the LogicQA and RuleTaker datasets. By constructing dynamic knowledge graphs, we analyze internal reasoning profiles through multi-faceted metrics, including conclusion emergence, stability, and convergence. Our findings quantitatively clarify how LLMs transform propositional knowledge across layers to reach final conclusions.

1 はじめに

Transformer アーキテクチャに基づく大規模言語モデル（Large Language Model, LLM）は，自然言語処理タスクにおいて飛躍的な性能向上を遂げ，機械翻訳や質問応答，文章生成など多様な分野に応用されている．また近年では，LLM の論理推論タスクへの適用も注目されており，正答率などの外部的な基準を用いた能力評価が行われている [1, 2]．しかし，論理的帰結の一致のみに着目した評価は，論理推論能力の評価としては必ずしも精密ではなく，推論過程を含めたより詳細な評価手法の確立が求められている．これに対し現状は，LLM 内部の推論過程，すなわち LLM がどの様なタイミングでどの様な知識を想起し，またそれらをどの様に組み合わせることで帰結を形成しているのかを，外部から把握することは容易ではなく，詳細評価の一つの障壁となっている．

これらのことを背景に，本研究では，ブラックボックスとされる LLM の内部推論メカニズムを段階的に評価するための初期的な試みとして，論理推論タスクにおける推論過程を可視化・分析し，その本質や原理の解明を試みる．具体的には，論理推論タスクに関するデータセット LogicQA[2] および RuleTaker[3] のそれぞれに対し，アクティベーション・パッチング（Activation Patching）[4] に基づく動的グラフ生成手法である Latent-Explorer[5] を適用することで，推論過程における各層の出現命題を抽出する．さらに得られる命題集合を分析することで，結論の形成時期や安定性，探索から収束への推移，相関の変化量など，多角的な指標により内部推論過程の分析を行う．これらを通じ，LLM が層ごとにどのような命題レベルの知識を構築・変容させ，最終的な結論へと収束させるのか，その内部推論過程を定量的に明らかにする．

本論文の構成は以下のとおりである．2 章で関連研究について述べる．3 章で，本研究における基礎技術である Latent-Explorer について概説する．次に 4 章で，論理タスクに対する LLM の推論過程を分析する枠組みを提案し，5 章でその評価実験の結果を示す．最後に 6 章でまとめを行い，今後の課題を述べる．

2 関連研究

LLM の内部推論過程はブラックボックスであり，モデルがどのように推論を行い，結論に至るかを明確に理解する手法は限られている．近年この課題を解決するために，LLM 内部の推論過程の可視化や知識の参照過程の追跡に関する研究が活発に行われている．

Heimersheim ら [4] は，アクティベーション・パッチング（AP 法）と呼ばれる，特定のプロンプトに対す

*連絡先：日本大学 文理学部 情報科学科
〒 156-8550 東京都 世田谷区 桜上水 3-25-40
E-mail: chgo22004@g.nihon-u.ac.jp

るモデル隠れ層の活性状態を別のプロンプトに転写することで推論の流れや参照知識を解析する技術を用い，モデル内部の知識参照の可視化手法を提案した．これにより，推論中にどの知識が呼び出されたかを部分的に明らかにすることが可能となった．

Bronzini ら [5] は，LLMs 内部の潜在表現を動的に追跡する手法として，AP 法を基礎とした動的知識グラフ構築手法 Latent-Explorer（LE 法）を提案した．詳細は後述するが，この手法は，モデルが推論過程において参照する知識の変化をグラフ構造として捉え，時系列的な推移を視覚化する．またその具体的な応用として，真偽判定タスクにおける複雑な命題推論の流れを可視化し，モデル内部における知識の変化を明らかにした．しかし現状，真偽判定以外のタスクに対する応用は確認されておらず，異なる系統の問題に対する推論過程の解明に関して検討の余地が残されている．

Jiang ら [6] は，数理推論に対する LLM の精度を高めるため，前向き推論と後ろ向き推論を統合した FORBAR 法を提案した．この手法は，推論の前方展開と逆方向の検証を組み合わせ，論理的な整合性を確認するものである．現時点では，モデル内部でどの知識がどのタイミングで参照されたかを段階的に可視化するまでには至っていないが，特に，数学的な証明に対して高い精度を発揮することが確認されている．

これらの研究ではいずれも，LLM の内部状態を示す情報を獲得することは可能であるが，多段推論における知識の時間的変化の詳細な追跡など，ステップごとの変化に対する形式的・定量的な解析までは行われていない．しかし実際には，複雑な数学的または論理的タスクを対象とした場合，モデルは複数の命題や事実を段階的に結び付けながら推論を進める必要があり，この過程を正確に観測・記録することは，推論メカニズムの解明のために不可欠であると言える．

3 Latent-Explorer

Latent-Explorer（LE 法）[5][1]とは，LLM モデルの各隠れ層に対するアクティベーション・パッチング（AP 法）の適用結果を集約することで，その推論過程を動的グラフとして抽出・可視化する手法である．

AP 法は，以下の手順に従い，L 層からなる LLM モデル $\mathcal{M}$ がタスク T を解く際における第 l 隠れ層での推論状況を自然言語表現 $o^{(l)}$ として抽出する（図 1 参照）．

1. LLM モデル $\mathcal{M}$ に，タスク T を伴うソースプロンプト $P_{\mathcal{S}}$ を表すトークン列 $X^{\mathcal{S}}(T) = x_1^{\mathcal{S}}, \cdots, x_m^{\mathcal{S}}$ を与え，タスク（推論）を実行する．

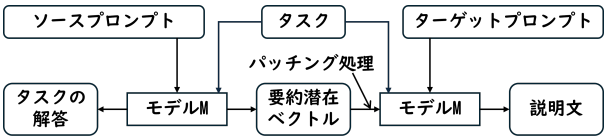

図 1: アクティベーション・パッチングの処理フロー

2. $P_{\mathcal{S}}$ によって事前に既定されるトークン群を対象に，推論の過程で得られる第 l 層における第 t トークン $x_t^{\mathcal{S}}$ の潜在ベクトル表現 $h_t^{(l)}$ を

$$\overline{h^{(l)}} = \sum_t w_t h_t^{(l)} \,/\, \sum_t w_t$$

のように集約し，第 l 層の要約潜在ベクトル $\overline{h^{(l)}}$ を算出する．ここで w_t は $x_t^{\mathcal{S}}$ に対する重みであり，名詞や動詞などのエンティティや関係を表すトークンには大きな重みを，それ以外のトークンには小さい重みを与えることで，重要性を反映した要約表現を獲得する．

3. 再び $\mathcal{M}$ に，T を伴うターゲットプロンプト $P_{\mathcal{T}}$ を表すトークン列 $X^{\mathcal{T}}(T) = x_1^{\mathcal{T}}, \cdots, x_n^{\mathcal{T}}$ を与え，タスクを実行し，結果である自然言語表現 $o^{(l)}$ を得る．なおこのとき，第 l 隠れ層における特定位置の潜在ベクトル表現を，要約潜在ベクトル $\overline{h^{(l)}}$ に置き換える処理（パッチング処理）を行った上で，推論処理を継続・完遂する．このパッチング処理により，結果である $o^{(l)}$ に，$\overline{h^{(l)}}$ で表現される推論の内部状態が反映される．

LE 法は，AP 法の後処理として，テンプレートに基づく命題表現抽出処理を $o^{(l)}$ に適用し，命題集合 $P^{(l)} = \{p_1^{(l)}, \cdots, p_{N_l}^{(l)}\}$ の抽出と，$P^{(l)}$ に基づく知識グラフ $G^{(l)}$ の構築を行う．さらにこの処理を全隠れ層に適用することで，層位置 $l \in \{1, \cdots, L\}$ をタイムスタンプとする動的知識グラフ $G = G^{(1)}, \cdots, G^{(L)}$ を導出する．

以上概観した通り，LE 法は，AP 法の適用も含めたこれら一連の処理を通じ，LLM モデル $\mathcal{M}$ におけるタスク T の推論過程，すなわち時間軸（層方向）に沿って知識がどのように追加・変形されていくかを動的知識グラフ G として抽出・可視化する．

4 LE 法の論理推論タスクへの適用

本研究では，LE 法を用いた論理推論タスクに対する LLM 推論過程の分析に関し，以下の 2 点について検討を行う．

1. タスク構造に即したソースプロンプトの設計
2. 動的グラフを対象とした多角的な評価指標の策定

[1] https://github.com/Ipazia-AI/latent-explorer

4.1 ソースプロンプトの設計

LLM へ与えるソースプロンプト P_S は，モデル内部で形成される潜在表現の構造および抽出される推論過程に直接的な影響を及ぼす．本研究の目的は，外的性能の最大化ではなく，内部推論過程の比較・可視化である．従って，ソースプロンプトには再現性と比較可能性を担保するための設計上の妥当性が求められる．これに基づき，以下の指針に従って設計を行うこととする．

- タスク構造の忠実な反映：元の問題が持つ構造を保ったまま提示する．
- 不要ヒントの排除：正解ラベルや解き方を直接示すような追加文は含めず，同タスクを人間が解く際に与えられる以上の情報を与えない．
- 形式的一貫性：同一タスク内では，すべての問題に対して同一の指示文を用いる．これにより，プロンプト構造の変化による影響を抑え，問題ごとの差は内容そのものに起因するようにする．
- 役割指示：モデルには “論理推論の専門家” として回答する役割のみを与え，中間的な思考ステップの形式や出力スタイルを細かく制約しない．

4.2 評価指標の策定

動的グラフ・命題集合として抽出される推論プロセスを定量的に評価するため，結論導出の速さや安定性，考慮知識の収束率，および層間での知識の更新量に着目し，以下の多角的な評価指標を導入する．

最終帰結初出層： LLM の思考速度を評価することを目的に,「タスクに関するすべての帰結が出現するまでに必要とされた層の数」を評価基準とする．形式的には，タスク T の正答に対する命題表現集合を P_T，L 層モデル $\mathcal{M}$ の T に対する命題集合系列を $P = P^{(1)}, \cdots, P^{(L)}$ とし，最終帰結初出層を次のように定義する．

$$\max_{p \in P_T} \min_{l \in \{1,\cdots,L\}} \{l \mid p \in P^{(l)}\}$$

帰結安定層： 最終帰結初出層が「いつ答えに辿り着いたか」を示すのに対し，帰結の安定性，すなわち LLM が「いつ答えを確信し，維持し始めたのか」を評価することを目的に,「全帰結が最終層まで一貫して出現し始めた最小の層」を評価基準とする．形式的には，次のように定義する．

$$\min_{s \in \{1,\cdots,L\}} \forall_{l \in \{s,\cdots,L\}} P_T \subseteq P^{(l)}$$

推論圧縮率： 推論の洗練度，すなわち推論の途上で生成された膨大な中間仮説や候補（探索の広がり）から，最終的な結論に向けていかに不要な情報を削ぎ落とし，論理を収束させたかという情報の整理能力を定量化することを目的に,「（正規化した）最大層内命題数と最終層内命題数の差」を評価基準とする．形式的には，以下のように定義する．

$$\min_{l \in \{1,\cdots,L\}} \left(1 - \frac{|P^{(L)}|}{|P^{(l)}|}\right)$$

層間知識更新率： 各層における推論の活動量を測定することを目的に,「隣接する層間で行われた知識グラフの正規化更新量」を評価基準とする．形式的には，隣接する層間における導出命題集合の Jaccard 距離と定義する．

$$1 - \left(\frac{\left|P^{(l)} \cap P^{(l-1)}\right|}{\left|P^{(l)} \cup P^{(l-1)}\right|}\right)$$

5 評価実験

5.1 実験設定とソースプロンプト

対象モデルとして Meta 社によるチャット向け事前学習モデル `meta-llama/Llama-2-7b-chat-hf` を採用し，評価実験を行う．またデータセットとして，読解型多肢選択問題を扱う LogicQA[2][2]および伴意判定問題を扱う RuleTaker[3][3] を利用する．以下，各データセットの詳細と，LE 法で利用するソースプロンプトについて説明する．

LogicQA： このデータセットは，読解・論理推論問題から構成されるデータセットであり，短い文章（context）と質問文（question），4 つの解答選択肢（options）から構成される．文章中の事実や因果関係を基に正しい選択肢を推定することが求められ，多段の推論や条件の組み合わせが必要となる問題が含まれる．本タスクに対するソースプロンプトを図 2 に示す．

RuleTaker： このデータセットは，問題文（context）と質問文（query）から構成されるデータセットであり，質問文が伴意される場合は ENTAILMENT を，そうでない場合は NOT ENTAILMENT を出力することが求められる．本タスクに対するソースプロンプトを図 3 に示す．

[2] https://github.com/lgw863/LogiQA-dataset
[3] https://github.com/allenai/ruletaker

```
You are an expert in logical reasoning. Read the passage carefully and answer the question
by choosing one option from (A), (B), (C), (D). Answer only with the letter of the correct option.

Example 1: Passage: Tom had three apples. He gave one apple to Mary and one apple to John.
Question: How many apples does Tom have now?
Options: (A) 1 (B) 2 (C) 3 (D) 4
Answer: A

Example 2: Passage: Alice is older than Betty. Betty is older than Chris.
Question: Who is the youngest person?
Options: (A) Alice (B) Betty (C) Chris (D) None of the above
Answer: C

Now solve the following problem.
Passage: [context of the target problem]
Question: [question of the target problem]
Options: (A) [option A] (B) [option B] (C) [option C] (D) [option D]
Answer:
```

図 2: LogicQA に対するプロンプト

```
You are an expert in logical reasoning. Read the context carefully and determine whether the question
is entailed by the context. Answer only with one of the following labels: ENTAILMENT or NOT_ENTAILMENT.

Example 1:
Context: Tom is tall. Tall things are strong.
Question: Tom is strong.
Answer: ENTAILMENT

Example 2:
Context: Alice is older than Betty. Betty is older than Chris.
Question: Alice is the youngest person.
Answer: NOT_ENTAILMENT

Now solve the following problem.
Context: [context of the target problem]
Question: [question of the target problem]
Answer:
```

図 3: RuleTaker に対するプロンプト

5.2 定量評価

各データセットから 30 問ずつ，計 60 問を選定して定量評価を行った．問題の選定にあたっては，入力長が極端に長い問題は除外し，モデルが安定して推論可能な範囲の問題を選択した．具体的には，LogicQA については，選択肢数や問題形式が標準的な問題を選び，特殊なフォーマットを含むものは除外した．一方 RuleTaker については，多数の推論ステップが必要とされる難易度の高い問題は対象外とした．

実験結果を表 1 にまとめる．なお，表中の各値は全 30 問の平均値である．また層間知識更新率に関しては，全層を前段 (02–08 層)・中段 (09–16 層)・後段 (17–24 層) の 3 つのフェーズに分割して集計した．加えて，後段層に関しては，詳細分析のために，タスクの正誤別での結果も算出した．以下，各評価基準の観点から考察を行う．

正答率： 正答率に関しては，LogicQA が RuleTaker を下回る結果となった．これは，ルールに基づく推論が核となる RuleTaker に対し，LogicQA は複数の選択肢から妥当な結論を導出するために広範な探索を要するという，タスク難易度の差を反映しているものと考えられる．

帰結の導出： 帰結の導出プロセスにおいて，RuleTaker は LogicQA よりも最終帰結初出層・帰結安定層ともに小さな値を示し，両指標の乖離（安定までの遅延）も小さい．これは，RuleTaker が問題文から中間命題を導出すれば結論が定まりやすいという特性を持ち，必要な命題集合が比較的浅い層で揃うためと推察される．対照的に，LogicQA は選択肢を検証するための条件探索を伴うため，結論の到達および安定までにより多くの層を要する傾向が伺える．

表 1: 評価結果

指標	LogicQA	RuleTaker
正答率	0.73	0.90
最終帰結初出層	16.27	9.10
帰結安定層	20.07	11.70
推論圧縮率	0.676	0.479
最大層内命題数	38.47	23.90
層間知識更新率（02–08）	0.447	0.383
層間知識更新率（09–16）	0.206	0.115
層間知識更新率（17–24）	0.236	0.103
層間知識更新率（17–24 正）	0.214	0.091
層間知識更新率（17–24 誤）	0.298	0.215

推論圧縮率： LogicQA は最大層内命題数が大きく，推論圧縮率も高い値を示した．これは，推論の過程で膨大な中間命題を一時的に生成（拡散）し，その後，結論に不要な情報を大幅に削除（収束）するという動的な情報整理が行われていることを示唆している．一方，RuleTaker は最大層内命題数が小さくまた圧縮率も高くなく，早期から必要な情報のみに絞られた効率的な推論が行われていることが伺える．

層間知識更新率： RuleTaker は前段で命題集合が大きく更新されるものの，中段以降は書き換え割合が減少し，推論過程が早期に定常状態へと移行していることが示唆される．これに対し LogicQA は，前段から大きな更新が続くほか，後段においても相対的に高い値が維持される．これは，選択肢の妥当性を保証するための条件探索や候補の切り替えが終盤まで継続し，命題集合が動的に組み替わりやすいタスクの特性を示していると考えられる．

また正誤別の考察では，両タスクに共通して，誤答時に後段層での層間知識更新率が上昇する傾向が確認された．これは，最終段階においても命題集合が大きく入れ替わり続け，推論が安定状態へ十分に収束していないことを示している．特に RuleTaker では，正答時後段の値が 0.091 と極めて低いのに対し，誤答時は 0.215 まで値が増大している．LogicQA においても誤答時の更新率は高まるが，正答時でも一定の値を示すことから，タスク自体が本質的に後段までの試行錯誤を伴いやすい性質を持つと推察される．

5.3 ケーススタディ

前節までの定量的な分析結果を補完するため，本節では LogicQA および RuleTaker の代表的な問題に対する LLM の推論過程を可視化し，定性的な考察を行う．図 4 に LogicQA の問題と得られた動的知識グラフの例を，図 5 に Ruletaker の問題と得られた動的知識グラフをそれぞれ示す．

図 4 より，動的グラフの初期層では，まず “Cantonese some not like chili” といった，文脈から抽出された主要な事実構造が形成されていることが分かる．また中段層に至ると，前提命題と結合可能な論理的橋渡しの候補として，各選択肢に対応する命題がグラフ内に組み込まれ，それらの妥当性が探索される．後段層では，結論導出に寄与しない不要な候補命題が順次削除され，最終的に前提命題，Option C の規則，および結論に関連する最小限の命題集合へと収束する．その結果，最終層において Option C が選択・出力されている．以上のプロセスから，選択肢型の論理タスクにおいて LLM は，前提と結論の間を埋める中間命題を適応的に探索し，後段にかけて必要な論理パスへと情報を絞り込むことで回答を導出していることが見て取れる．

一方，図 5 においては，動的グラフの初期層で “Bob is kind” や “Bob is young” といった入力文に基づく事実命題が形成されていることが分かる．中段層では，関連するルールが活性化されることで，“Bob is kind” と “Bob is young” の同時成立から “Bob is not smart” が導出されるなど，結論の判定に直接関与しない命題も一時的に増加する．これは，与えられた複数のルールを逐次的に適用し，推論候補を拡張している過程を反映している．後段層に入ると，最終的な判断に不要な命題は順次削除され，結論に直結する命題のみが保持されるようになる．最終的には “Bob is kind” を含む最小限の命題集合へと整理され，最終層において “ENTAILMENT” が出力される．以上のプロセスから，結論の根拠が入力に含まれる比較的単純なタスクにおいても，モデルは早期に必要命題を特定するだけでなく，層を通じた探索過程で一時的に派生命題を生成（拡散）し，その後に必要な論理パスへと収束させるという動的な推論プロセスを経ていることが見て取れる．

6 結論

本研究では，Latent-Explorer を用いた論理推論タスクに対する LLM 推論過程の分析を目的とし，タスク構造を反映したソースプロンプトの設計および動的グラフを対象とした多角的な評価指標の提案を行った．また論理推論に関する 2 つのデータセット（LogicQA, RuleTaker）に提案手法を適用し，内部推論の過程を定量的・定性的に評価した．

実験の結果，本手法は「帰結がいつ形成・安定し，推論過程でどの程度探索的に情報が拡散し，層間でいかに書き換わりながら収束するか」という，推論過程の時間的・空間的構造をタスク間や正誤間で比較可能にすることを示した．具体的な知見としては，RuleTaker

問題文: Some Cantonese don't like chili, so some southerners don't like chili.

質問文: Which of the following can guarantee the above argument?

選択肢: (a)Some Cantonese love chili. (b)Some people who like peppers are southerners. (c)All Cantonese are southerners. (d)Some Cantonese like neither peppers nor sweets.

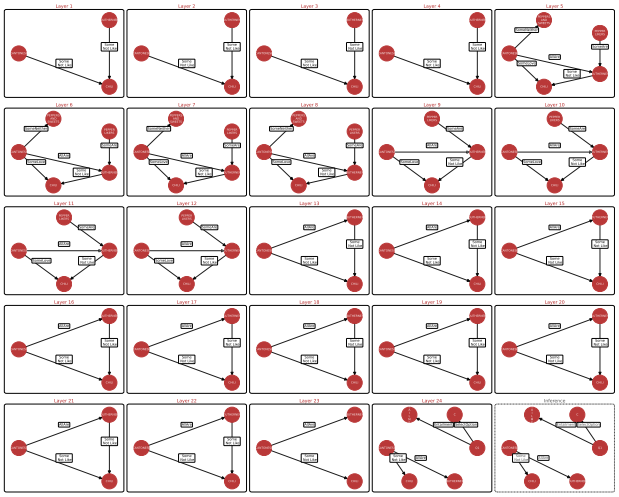

図 4: LogicQA の問題と得られた動的知識グラフの例

問題文: Anne is quiet. Anne is not young. Bob is kind. Bob is young. Dave is rough. Dave is round. Dave is smart. Dave is not young. Fiona is quiet. Fiona is not round. Kind, young things are not smart.

質問文: Bob is kind.

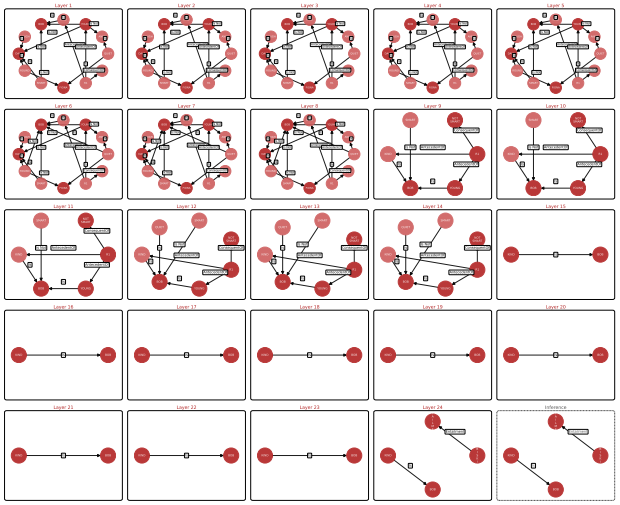

図 5: RuleTaker の問題と得られた動的知識グラフの例

では比較的早期に必要命題が揃うのに対し，LogicQA では未既知の橋渡し探索を反映して，後段まで命題の動的な組み替えが継続するといった，問題の系統性に起因する推論過程の差異を定量的に明らかにした．また，誤答時には最終層付近でも命題集合が不安定なまま推移する現象を確認しており，これはモデルの「迷い」を内部指標から検知できる可能性を示唆している．

今後の課題として，本実験は各 30 問という小規模なサンプルに基づいているため，より大規模かつ多様なデータセットおよび異なるモデル規模への拡張が必要である．また，命題抽出プロセスがモデルの表現揺れや変換規則の影響を少なからず受けることから，観測された推論過程が純粋な推論の変化なのか，あるいは LLM 由来のノイズであるかを厳密に峻別する手法の検討が求められる．今後は，誤答タイプの体系化や詳細な分類を進めることで，知識グラフに基づく内部推論プロセス分析を，LLM の信頼性を担保するための比較・検証基盤へと発展させたいと考えている．

参考文献

[1] X. Wu, Y. Cai, and H.-F. Leung : Abstract-level Deductive Reasoning for Pre-trained Language Models, The 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation, pp.70–76, 2024.

[2] J. Liu, L. Cui, H. Liu, D. Huang, Y. Wang, and Y. Zhang : LogiQA: A Challenge Dataset for Machine Reading Comprehension with Logical Reasoning, The 29th International Joint Conference on Artificial Intelligence, pp.3622–3628, 2020.

[3] P. Clark, O. Tafjord, and K. Richardson : Transformers as Soft Reasoners over Language, The 29th International Conference on International Joint Conferences on Artificial Intelligence, pp.3882–3890, 2020.

[4] S. Heimersheim and N. Nanda : How to use and interpret activation patching, arXiv:2404.15255, 2024.

[5] M. Bronzini, C. Nicolini, B. Lepri, J. Staiano, and A. Passerini : Unveiling LLMs: The Evolution of Latent Representations in a Dynamic Knowledge Graph, arXiv:2404.03623, 2024.

[6] W. Jiang, H. Shi, L. Yu, Z. Liu, Y. Zhang, Z. Li, and J. Kwok : Forward-Backward Reasoning in Large Language Models for Mathematical Verification, *Findings of Annual Meeting of the Association for Computational Linguistics*, pp.6647–6661, 2024.