全域木による言語モデルの分析

坂上 温紀 *　Frederikus Hudi*　坂井 優介　上垣外 英剛　渡辺 太郎
奈良先端科学技術大学院大学
sakajo.haruki.sd9@naist.ac.jp
{frederikus.hudi.fe7,sakai.yusuke.sr9,kamigaito.h,taro}@is.naist.jp

概要

言語には何らかの構造が見られ，言語獲得や言語変化はその構造によって説明される．この特徴を踏まえると，言語モデルもまた，内部に何らかの構造を有していると考えられる．本研究ではモデルの層内におけるトークン表現間の意味的関係性に基づいて層ごとに全域木を導出し，層間で内部構造がどのように関係するかを木構造の類似度や特性の変化から分析する手法を提案する．実験の結果，この手法によって得られた木構造のパターンはモデルのよって異なることが明らかになった．さらに，この構造を考慮した層間の類似度は層の枝刈りにおいても有効であり，言語モデルの解釈や最適化における有用性を示している．

1　序論

言語は構造を有している．言語獲得や言語変化といった言語現象は，生成文法の立場でも認知言語学の立場でも，背後にある何らかの構造よって説明されてきた [1, 2, 3]．このような性質を踏まえると，言語を計算的にモデル化する言語モデルもまた，何らかの構造的性質を示すと考えられる [4]．

このような言語の構造的性質にもかかわらず，解釈可能性研究や層枝刈りといった大規模言語モデル（LLM）に関する研究では，層間あるいはモジュール間の分析を行う際に，これらの構造がしばしば見過ごされてきた．これまでの研究の中には，構文解析手法を用い，トークン間関係に基づいて層間分析を行うものも存在する．生成文法的観点からの言語学的分析に基づく研究では，注意重みが統語構造を反映していること [5, 6, 7, 8]，表現が統語情報を符号化していること [9, 10, 11, 12, 13]，そして統語構造がボトムアップ的に出現すること [14] が示されている．これらの研究は，LLM がトークン間の関係から得られる構造を保持し利用していることを明らかにしている一方で，特定の正解構造を前提とした静的な構造に焦点を当てている．しかし，言語に動的な構造を見出すことができる [2, 3] ことを考えると，言語モデルの内部構造の分析としては，ボトムアップな方法の方がより適切であると考えられる．

そこで本研究では，依存構造解析 [15, 16] に着想を得た層分析の手法を提案する．この手法では，LLM の内部表現を用いて最大全域木を構築する．各層の出力における残差ストリームを用いて，トークン表現間の L2 距離に基づく類似度を計算することで，各層ごとに最大全域木を構築する．これらの木は，言語モデル内部における意味的な表現構造を大域的な観点から捉えることを可能にする．

実験の結果，提案手法によって，層間類似度を木構造の類似度で比較すると，モデルによって異なる変化や構築のパターンを示すことが明らかになった．さらに，木編集距離 [17] は層の枝刈りの指標として有効であることが確認された．これらの結果は，構造を考慮した大域的な視点が言語モデルの分析や最適化に役立つことを示している．

2　提案手法

2.1　全域木の構築

この研究では，トークン表現間の意味的な関係に基づき，トークン列から，トークン表現の類似度を用いて，総エッジ重みが最大となる単一根の木構造，すなわち最大全域木を構築する．形式的には，長さ n の入力トークン列 $\boldsymbol{x}$ が与えられたとき，$\mathcal{G}$ を n 個のノードからなる完全有向グラフと定義する．ただし自己ループは含まず，エッジの総数は $n(n-1)$ 本である．各ノードは $\boldsymbol{x}$ 中の各トークンに対応し，各有向エッジは 2 つのトークン間の関係を表す．ここで，モデルの隠れ表現の次元数を d，層 ℓ の直後の i 番目のトークンの残差ストリームを

* 共同筆頭著者

$\boldsymbol{h}_i^{(\ell)} \in \mathbb{R}^{n\times d}$，$i$ 番目のノードから j 番目のノードへのエッジの重みを $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ として，$\boldsymbol{A}$ を以下のように定義する：

$$\boldsymbol{A}_{i,j} = \begin{cases} \exp\left(-\|\boldsymbol{h}_i^{(\ell)} - \boldsymbol{h}_j^{(\ell)}\|\right) & (i < j \text{ の場合}) \\ 0 & (\text{それ以外}). \end{cases} \tag{1}$$

この隣接行列 $\boldsymbol{A}$ を用いて，Chu-Liu/Edmonds アルゴリズム [18, 19] に基づく Tarjan [20] のアルゴリズムにより最大全域木を構築する．

2.2 全域木に基づく層間類似度

LLM における層の冗長性を分析するために，コサイン類似度などの既存手法が層間の類似度を定量化する目的で広く用いられてきた [21, 22]．これらの従来手法は，対応する位置における表現同士の類似度を測定することで，局所的なペアワイズ関係を捉えるものである．しかし，層内におけるトークン間関係を含む大域的な視点を欠いており，層レベルの相互作用を包括的に捉えることはできない．本研究では，提案手法を用いて，包括的かつ大域的な関係を測定するための木構造に着目した 2 つの類似度指標を用いて層間類似度を算出する．

Centered Kernel Alignment（CKA） ベースラインとして，層間の大域的類似度を比較するための標準的な指標である Centered Kernel Alignment（CKA）[23] について概説する．本研究では，Hilbert-Schmidt Independence Criterion（HSIC）による不偏推定量 [24] を用いる．$\boldsymbol{H}^{\ell} \in \mathbb{R}^{n\times d}$ を層 ℓ 直後の残差ストリームとし，$\boldsymbol{K} = \boldsymbol{H}^{(\ell_a)}\boldsymbol{H}^{(\ell_a)\top}$ および $\boldsymbol{L} = \boldsymbol{H}^{(\ell_b)}\boldsymbol{H}^{(\ell_b)\top}$ とすると，CKA による層間類似度は以下のように定義される：

$$\text{score}_{\text{CKA}}(\ell_a, \ell_b) = \frac{\text{HSIC}(\boldsymbol{K}, \boldsymbol{L})}{\sqrt{\text{HSIC}(\boldsymbol{K}, \boldsymbol{K})\text{HSIC}(\boldsymbol{L}, \boldsymbol{L})}}. \tag{2}$$

コサイン類似度（Cos-Base） 別のベースラインとして，コサイン類似度はベクトル表現間の比較において一般的に用いられる指標である．本研究では，隣接層間の類似度計算を全層ペアに拡張する．長さ n の入力トークン列 $\boldsymbol{x}$ が与えられたとき，a 層目および b 層目をそれぞれ ℓ_a，ℓ_b とする．各層の残差ストリームから得られるトークン表現を用いて，層 ℓ_a と ℓ_b の類似度を次のように計算する：

$$\text{score}_{\text{Cos-Base}}(\ell_a, \ell_b) = \sum_{i}^{n} \cos\left(\boldsymbol{h}_i^{(\ell_a)}, \boldsymbol{h}_i^{(\ell_b)}\right). \tag{3}$$

木編集距離（Tree-Edit） 木編集距離 [17] は，順序付きラベル付き木間の非類似度を定量化する手法として広く利用・研究されてきた [25]．本研究では，この指標を負の類似度スコアとして用いる．

グラフ $\mathcal{G}$ が与えられたとき，層 ℓ_a に対応する最大全域木を $\mathcal{T}_a$ とする．$\mathcal{P}(\mathcal{T}_a, \mathcal{T}_b)$ を $\mathcal{T}_a$ を $\mathcal{T}_b$ に変換する編集スクリプトの集合とし，π に含まれる編集操作 o のコストを $c(o)$ とする．Tree-Edit スコアは次のように定義される：

$$\text{score}_{\text{Tree-Edit}}(\ell_a, \ell_b) = -\left(\min \pi \in \mathcal{P}(\mathcal{T}_a, \mathcal{T}_b) \sum_{o\in\pi} c(o)\right). \tag{4}$$

文字列編集距離 [26] と同様に，Tree-Edit では以下の 3 種類の編集操作を許容する：

- **挿入**：既存ノードの子として新しいノードを挿入する．
- **削除**：ノードを削除し，その子ノードを親ノードに再接続する．
- **再ラベル付け**：ノードのラベルを $\mathcal{G}$ 内の別のラベルに変更する（変更がない場合はコスト 0）．

ただし Tree-Edit は，部分木全体の移動に大きなコストを与える．これは，そのような変更が部分木内のすべてのノードおよびエッジに対する再帰的な削除・挿入操作を必要とするためである．

エッジ編集距離（Edge-Edit） 本研究では，層間における部分木移動によって生じるスコア変動を緩和し，より直接的かつ安定した構造比較を行うために，より単純なエッジベースの編集距離指標を用いる．この指標を負の類似度スコアとして定義する．

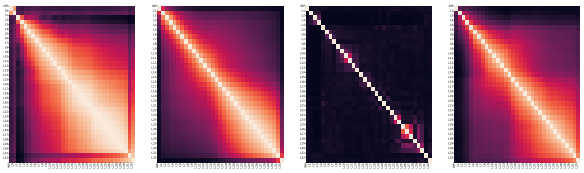
木 $\mathcal{T}$ のエッジ集合 $\mathbb{S}_{\mathcal{T}}$ において，r を根のノードとし，$Pa_T(i)$ を r 以外の任意のノード i の親ノードとする．このとき，エッジ集合は以下のように与えられる：

$$\mathbb{S}_{\mathcal{T}} = \{\,(i, Pa_T(i)) \mid i \in \{1, \ldots, n\},\ i \neq r\,\}. \tag{5}$$

層 ℓ_a および ℓ_b に対応する全域木をそれぞれ $\mathcal{T}_a$，$\mathcal{T}_b$ とし，式 5 で定義されるエッジ集合をそれぞれ $\mathbb{S}_{\mathcal{T}_a}$，$\mathbb{S}_{\mathcal{T}_b}$ とする．2 つの木は同一のノード集合と同数のエッジを持つため，Edge-Edit スコアはエッジ集合の差分として次のように定義される：

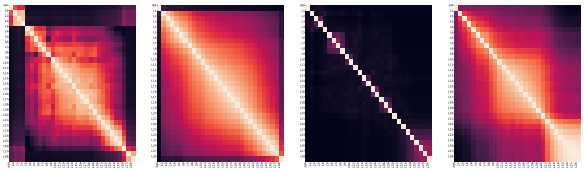
$$\text{score}_{\text{Edge-Edit}}(\ell_a, \ell_b) = -\left(|\mathbb{S}_{\mathcal{T}_a} \backslash \mathbb{S}_{\mathcal{T}_b}| + |\mathbb{S}_{\mathcal{T}_b} \backslash \mathbb{S}_{\mathcal{T}_a}|\right). \tag{6}$$

この指標は，エッジの挿入および削除を直接数えることで，部分木移動によるコストの過大評価を回避し，層間の構造的類似度をより安定して測定することができる．



(a) CKA (b) Cos-Base (c) Tree-Edit (d) Edge-Edit

図 1: MMLU における Llama3.1 8B の層間類似度．色の明るさは類似度の高さを示す．



(a) CKA (b) Cos-Base (c) Tree-Edit (d) Edge-Edit

図 2: MMLU における Qwen2.5 7B の層間類似度

3 全域木に基づく層の分析

3.1 層間類似度

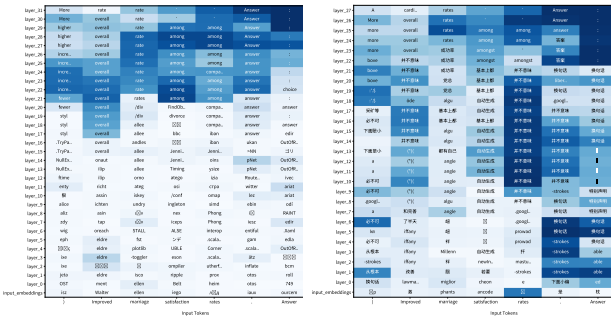
各データセットからサンプリングしたデータに対して各層の全域木を構築し，その後，式 2，3，4，および 6 を用いて層間類似度を計算する．

実験設定 実験には，Llama3.1 8B [27] および Qwen2.5 7B [28] を用いる．評価データセットとしては，4 択（A，B，C，D）の多肢選択型質問応答データセットである MMLU [29] を使用する．各データセットからランダムに 100 件のデータを抽出し，各データセットの dev セットから取得した 5 つの例を含むプロンプトを用いる．詳細は付録 A に示す．

結果 MMLU における各モデルの層間類似度を，図 1 および図 2 に示す．これらの図では，横軸および縦軸はいずれも層インデックスを表している．結果から，Edge-Edit は対角方向にクラスタリングしたパターンを示し，高い層間類似度によって特徴付けられる離散的なグループを形成していることが分かる．本研究では，これらのグループを「島」と呼ぶ．

3.2 モデルの振る舞いと木構造の変換

本研究では，Edge-Edit によって示される「島」でのモデルの振る舞いを，logit lens を用いて分析する．図 3 に示すように，各層における logit lens の最終トークン出力に注目すると，Llama3.1 8B では層 18 から指示追従的な挙動（A/B/C/D の選択）が現れ始めるのに対し，Qwen2.5 7B では層 22 からこの



(a) Llama3.1 8B (b) Qwen2.5 7B

図 3: MMLU での Logit lens．入力の最後の 8 トークンの予測を示す．色の濃さは予測確率の強さを表す．

表 1: Edge-Edit による層間類似度を用いた層のスペクトラルクラスタリング [30, 31] の結果 ($k = 3$)．Layer 0 は入力埋め込みを示す．

	Llama3.1 8B	Qwen2.5 7B
	Layers	Layers
Cluster 1	0, 1, 2, 3.	0, 1, 2, 3, 4, 5, 6, 7.
Cluster 2	4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17.	8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20.
Cluster 3	18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32.	21, 22, 23, 24, 25, 26, 27, 28.

挙動が始まる．島の境界は Llama3.1 8B では層 18，Qwen2.5 7B では層 21 である（表 1）ことから，提案手法によって明らかにされる構造的変換が，出力形式の形成につながっていることを示唆している．

3.3 頻出部分木

どのような構造が構築されているのかを明らかにするため，ケーススタディとして MMLU の 1 インスタンスに対して頻出部分木マイニング [32, 33] を実施する．頻出部分木マイニングには FREQT[1] を用い，8 ノードからなる部分木を抽出する．本研究では，各層ごとに構築された木の集合全体にわたって少なくとも 2 回出現した，すなわち最低 2 層で観測された部分木を抽出する．

表 2 および 表 3 に示すように，いずれも，中間層および深層において連続するトークンから構成される深さ 8 の部分木を構築しており，さらに深い層では，後方位置にある連続トークンが木構造を形成していることが確認される．このような部分木の出現パターンは，モデルが左から右へと逐次的に部分木を構築している．さらに，Qwen2.5 7B は最初の数層で構築する，選択肢のトークンから構成される部分木がそれ以降の層では再利用していない．

1 http://chasen.org/~taku/software/freqt/

表 2: MMLU における Llama3.1 8B の頻出部分木の例（S 式）．“_” の前の数字は入力系列内の位置を表す．

Subtree
(14_A(36_[A(41_[B(48_[C(54_[D(143_[D(1352_[D]))(131_[C]))))   Layers: 1, 2, 3, 4, 5, 6, 7, 8, 29
(1_The(2_following(3_are(4_multiple(5_choice(6_questions(7_about(8_college))))))))   Layers: 4, 5, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16
(520_io(521_Is(522_chem(523_ic(524_Heart(525_Disease(530_HD)(531_]))))))))   Layers: 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32

表 3: MMLU における Qwen2.5 7B の頻出部分木の例（S 式）．“_” の前の数字は入力系列内の位置を表す．

Subtree
(35_[A(40_[B(47_[C(53_[D(142_[D(246_[D(324_[D])))))(101_[A))   Layers: 0, 1, 2, 3, 4
(27_side(28_effect(29_of(33_is(36_](37_muscle(49_muscle))(41_]))))))   Layers: 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20
(1013_while(1014_the(1015_heart(1016_rate(1017_[(1018_the(1019_number(1020_of)))))))))   Layers: 21, 22, 23, 24, 25, 26, 27, 28

4　全域木に基づく層枝刈り

モデルの各層から得られた全域木を用いた応用として，層枝刈りを行う．

層枝刈りアルゴリズム　LLM における層枝刈り手法 [34, 22] は，層間の表現類似度に基づいて，表現に与える変化が比較的小さい層を特定し，それらを削除する．削除対象となる層を決定するために，まず各層の重要度を定量化する．

層の影響度　ShortGPT [22] では，層間のコサイン類似度を用いて層の影響度（重要度）を算出し，その重要度が低い順に層をモデルから削除する．本研究では同様の方法で木構造に基づく類似度を用いて層の影響度を計算する．

ShortGPT においては，i 番目の層の影響度（Block Influence; BI）は式 3 に基づいて次のように定義される：

$$\mathrm{CosBaseBI}_i = 1 - \mathrm{score}_{\text{Cos-Base}}(\ell_i, \ell_{i-1}) \tag{7}$$

さらに，本研究では木構造に基づく類似度指標，すなわち Tree-Edit（式 4），および Edge-Edit（式 6）を用いて，以下のように層の影響度を算出する：

$$\mathrm{TreeBI}_i = 1 - \mathrm{score}_{\text{Tree-Edit}}(\ell_i, \ell_{i-1}) \tag{8}$$

$$\mathrm{EdgeBI}_i = 1 - \mathrm{score}_{\text{Edge-Edit}}(\ell_i, \ell_{i-1}) \tag{9}$$

TreeBI および EdgeBI のスコア範囲は入力に依存するため，それぞれの指標の理論的な上限・下限を考慮し，サンプルごとに正規化を行った．

実験設定　実験には Llama3.1 8B および Qwen2.5 7B を用いる．評価データセットとしては，質問応答タスクである MMLU を用いる．MMLU については，第 3.1 節の実験と同様の設定で実験を行う．各モデルに対して，各指標に基づく BI スコアの低い順に，層の約 10%を削除し，そのときの正解率を計測する．正解率の差に対する統計的有意性の検定には McNemar 検定 [35] を，パープレキシティに対してはブートストラップ検定を用いる．層削除のキャリブレーションには，英語版 Wikipedia データセット [36] から抽出した 10 サンプルを用いる．

表 4: 枝刈りの結果．Acc. は正解率を表し，PPL はパープレキシティを表す．TreeBI，および EdgeBI において * が付された値は，CosBaseBI と比較して統計的に有意である（$p < 0.05$）．

削除割合	指標	削除した層	Acc. (↑)	PPL (↓)
Llama3.1 8B				
0.0%	Dense	—	66.6	1,038.5
12.5%	CosBaseBI	24 25 26 27	63.0	221.8
	TreeBI	23 24 26 27	**66.2***	**57.5***
	EdgeBI	23 24 25 26	65.8*	358.5*
Qwen2.5 7B				
0.0%	Dense	—	75.5	11.4
10.7%	CosBaseBI	15 16 17	55.8	14.1
	TreeBI	15 16 26	**65.3***	**11.9***
	EdgeBI	24 25 26	55.6*	23.4*

結果　表 4 は，木構造に基づく TreeBI は，コサイン類似度と比較して，層枝刈り時の性能劣化をより抑えられることを示している．つまり，効果的な層影響度の評価には，トークンごとの対応関係のみを調べる局所的視点ではなく，提案手法が提供するような，各層内におけるトークン間関係を反映した大域的視点が必要であることを示唆している．

5　結論

本研究では，言語モデルの各層で全域木を構築し，その全域木を通じて言語モデルを分析するための枠組みを提案した．実験結果から，提案手法は，コサイン類似度などの従来指標とは異なる層間類似度パターンを示すことが明らかになり，モデルによって異なる構築・変換のパターンが示された．さらに，提案手法を取り入れた指標は，層枝刈りにおいても有用な知見を与えることが示された．これらの結果は，提案手法が有益な分析視点を提供し，本分野の研究を拡張する可能性を有していることを示している．

謝辞

本研究は JST 次世代研究者挑戦的研究プログラム JPMJSP2140 の支援を受けたものである.

参考文献

[1] N. Chomsky. **Syntactic Structures**. Janua Linguarum : Studia Memoriae Nicolai van Wijk dedicata. Mouton, 1962.

[2] Michael Tomasello. **Constructing a Language: A Usage-Based Theory of Language Acquisition**. Harvard University Press, March 2005.

[3] Joan L Bybee. From usage to grammar: The mind's response to repetition. **Language**, Vol. 82, No. 4, p. 711–733, December 2006.

[4] Jin Hwa Lee, Thomas Jiralerspong, Lei Yu, Yoshua Bengio, and Emily Cheng. Geometric signatures of compositionality across a language model's lifetime. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar, editors, **Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)**, pp. 5292–5320, Vienna, Austria, July 2025. Association for Computational Linguistics.

[5] Alessandro Raganato and Jörg Tiedemann. An analysis of encoder representations in transformer-based machine translation. In Tal Linzen, Grzegorz Chrupała, and Afra Alishahi, editors, **Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP**, pp. 287–297, Brussels, Belgium, November 2018. Association for Computational Linguistics.

[6] Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. What does BERT look at? an analysis of BERT's attention. In Tal Linzen, Grzegorz Chrupała, Yonatan Belinkov, and Dieuwke Hupkes, editors, **Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP**, pp. 276–286, Florence, Italy, August 2019. Association for Computational Linguistics.

[7] Vinit Ravishankar, Artur Kulmizev, Mostafa Abdou, Anders Søgaard, and Joakim Nivre. Attention can reflect syntactic structure (if you let it). In Paola Merlo, Jorg Tiedemann, and Reut Tsarfaty, editors, **Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume**, pp. 3031–3045, Online, April 2021. Association for Computational Linguistics.

[8] Ruochen Zhang, Qinan Yu, Matianyu Zang, Carsten Eickhoff, and Ellie Pavlick. The same but different: Structural similarities and differences in multilingual language modeling. In **The Thirteenth International Conference on Learning Representations**, 2025.

[9] John Hewitt and Christopher D. Manning. A structural probe for finding syntax in word representations. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, **Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)**, pp. 4129–4138, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[10] Jacob Andreas. Measuring compositionality in representation learning. In **International Conference on Learning Representations**, 2019.

[11] Xiang Lisa Li and Jason Eisner. Specializing word embeddings (for parsing) by information bottleneck. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, **Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)**, pp. 2744–2754, Hong Kong, China, November 2019. Association for Computational Linguistics.

[12] Shikhar Murty, Pratyusha Sharma, Jacob Andreas, and Christopher D Manning. Characterizing intrinsic compositionality in transformers with tree projections. In **The Eleventh International Conference on Learning Representations**, 2023.

[13] Frederikus Hudi, Zhi Qu, Hidetaka Kamigaito, and Taro Watanabe. Disentangling pretrained representation to leverage low-resource languages in multilingual machine translation. In Nicoletta Calzolari, Min-Yen Kan, Veronique Hoste, Alessandro Lenci, Sakriani Sakti, and Nianwen Xue, editors, **Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)**, pp. 4978–4989, Torino, Italia, May 2024. ELRA and ICCL.

[14] Taiga Someya, Ryo Yoshida, Hitomi Yanaka, and Yohei Oseki. Derivational probing: Unveiling the layer-wise derivation of syntactic structures in neural language models. In Gemma Boleda and Michael Roth, editors, **Proceedings of the 29th Conference on Computational Natural Language Learning**, pp. 93–104, Vienna, Austria, July 2025. Association for Computational Linguistics.

[15] Jason M. Eisner. Three new probabilistic models for dependency parsing: An exploration. In **COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics**, 1996.

[16] Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajič. Non-projective dependency parsing using spanning tree algorithms. In Raymond Mooney, Chris Brew, Lee-Feng Chien, and Katrin Kirchhoff, editors, **Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing**, pp. 523–530, Vancouver, British Columbia, Canada, October 2005. Association for Computational Linguistics.

[17] Kaizhong Zhang and Dennis Shasha. Simple fast algorithms for the editing distance between trees and related problems. **SIAM J. Comput.**, Vol. 18, pp. 1245–1262, 12 1989.

[18] Yoeng-Jin Chu and Tseng-Hong Liu. On the shortest arborescence of a directed graph. **Scientia Sinica**, Vol. 14, pp. 1396–1400, 1965.

[19] Jack Edmonds, et al. Optimum branchings. **Journal of Research of the national Bureau of Standards B**, Vol. 71, No. 4, pp. 233–240, 1967.

[20] R. E. Tarjan. Finding optimum branchings. **Networks**, Vol. 7, No. 1, p. 25–35, March 1977.

[21] Jiachen Jiang, Jinxin Zhou, and Zhihui Zhu. Tracing representation progression: Analyzing and enhancing layer-wise similarity. In **The Thirteenth International Conference on Learning Representations**, 2025.

[22] Xin Men, Mingyu Xu, Qingyu Zhang, Qianhao Yuan, Bingning Wang, Hongyu Lin, Yaojie Lu, Xianpei Han, and Weipeng Chen. ShortGPT: Layers in large language models are more redundant than you expect. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar, editors, **Findings of the Association for Computational Linguistics: ACL 2025**, pp. 20192–20204, Vienna, Austria, July 2025. Association for Computational Linguistics.

[23] Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, **Proceedings of the 36th International Conference on Machine Learning**, Vol. 97 of **Proceedings of Machine Learning Research**, pp. 3519–3529. PMLR, 09–15 Jun 2019.

[24] Le Song, Alex Smola, Arthur Gretton, Karsten M. Borgwardt, and Justin Bedo. Supervised feature selection via dependence estimation. In **Proceedings of the 24th International Conference on Machine Learning**, ICML '07, p. 823–830, New York, NY, USA, 2007. Association for Computing Machinery.

[25] Benjamin Paaßen. Revisiting the tree edit distance and its backtracing: A tutorial, 2022.

[26] V. I. Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. **Soviet Physics – Doklady**, Vol. 10, No. 8, pp. 707–710, 1966. Translated from Doklady Akademii Nauk SSSR, 163 No. 4 (845–848), 1965.

[27] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models, 2024.

[28] Qwen. Qwen2.5 technical report, 2025.

[29] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In **International Conference on Learning Representations**, 2021.

[30] Jianbo Shi and J. Malik. Normalized cuts and image segmentation. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, Vol. 22, No. 8, pp. 888–905, 2000.

[31] Ulrike von Luxburg. A tutorial on spectral clustering. **Statistics and Computing**, Vol. 17, No. 4, p. 395–416, August 2007.

[32] Kenji Abe, Shinji Kawasoe, Tatsuya Asai, Hiroki Arimura, and Setsuo Arikawa. Optimized substructure discovery for semi-structured data. In **Proceedings of the 6th European Conference on Principles of Data Mining and Knowledge Discovery**, PKDD '02, p. 1–14, Berlin, Heidelberg, 2002. Springer-Verlag.

[33] M.J. Zaki. Efficiently mining frequent trees in a forest: algorithms and applications. **IEEE Transactions on Knowledge and Data Engineering**, Vol. 17, No. 8, pp. 1021–1035, 2005.

[34] Yifei Yang, Zouying Cao, and Hai Zhao. LaCo: Large language model pruning via layer collapse. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, **Findings of the Association for Computational Linguistics: EMNLP 2024**, pp. 6401–6417, Miami, Florida, USA, November 2024. Association for Computational Linguistics.

[35] Quinn McNemar. Note on the sampling error of the difference between correlated proportions or percentages. **Psychometrika**, Vol. 12, No. 2, p. 153–157, June 1947.

[36] Wikimedia Foundation. Wikimedia downloads.

[37] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. Transformers: State-of-the-art natural language processing. In Qun Liu and David Schlangen, editors, **Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations**, pp. 38–45, Online, October 2020. Association for Computational Linguistics.

[38] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. **PyTorch: an imperative style, high-performance deep learning library**. Curran Associates Inc., Red Hook, NY, USA, 2019.

[39] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[40] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. **Journal of Machine Learning Research**, Vol. 12, pp. 2825–2830, 2011.

A　実験設定（詳細版）

プロンプト　本論文で MMLU を用いた実験では以下のプロンプトを用いる．

MMLU

The following are multiple choice questions about {subject}. Respond with either A, B, C, or D as your answer.
{Question of Example1}
(A) {Choice A of Example1}
(B) {Choice B of Example1}
(C) {Choice C of Example1}
(D) {Choice D of Example1}
Answer: {Answer of Example1}
...
{Question}
(A) {Choice A}
(B) {Choice B}
(C) {Choice C}
(D) {Choice D}
Answer:

実装　本研究では，HuggingFace を通じてモデルとデータセットを利用した．表 5 に HuggingFace ID を示す．また，HuggingFace Transformers [37] と PyTorch [38] を用いてモデルによる推論を行い，最大全域木の構築には TensorFlow Text [39]，スペクトラルクラスタリングには scikit-learn [40] を用いた．実験は それぞれ 1 枚の NVIDIA GeForce RTX 3090 上で行った．パイパーパラメータは表 6 に示す．

表 5: モデルとデータセットの ID

	HuggingFace ID
Llama3.1 8B	meta-llama/Llama-3.1-8B
Qwen2.5 7B	Qwen/Qwen2.5-7B
MMLU	cais/mmlu
Wikipedia	wikimedia/wikipedia

表 6: ハイパーパラメータ

パラメータ	値
復号化手法	貪欲法
精度	BF16
シード値	42