算術推論問題における自己回帰型言語モデルの内部機序

工藤慧音 [1,2] 青木洋一 [1,2] 栗林樹生 [3] 谷口雅弥 [2] 曾根周作 [1] 坂口慶祐 [1,2] 乾健太郎 [3,1,2]
[1] 東北大学 [2] 理化学研究所 [3]MBZUAI
{keito.kudo.q4,youichi.aoki.p2,sone.shusaku.r8}@dc.tohoku.ac.jp
{tatsuki.kuribayashi,kentaro.inui@}@mbzuai.ac.ae masaya.taniguchi@riken.jp
keisuke.sakaguchi@tohoku.ac.jp

概要

自己回帰型言語モデルのような逐次的に単語を生成するモデルが「複合的な問題について，いつどのような部分問題を問いているか」という内部機序についての時間方向の解析は現状不十分である．そこで本研究では，人工的な算術推論タスクを解く際の言語モデル内部機序を分析する．本論文の前半では，モデルの内部機序の予測が立つ人工的な算術タスクのみで学習されたモデルに対し分析を行い，提案法の妥当性を示した．後半では，思考の連鎖 [1] を用いて推論を行う事前学習済みの言語モデルに対して提案法を適用し分析を行ない，モデルは推論過程を出力すると同時に途中の計算も行っていることが明らかとなった．

1 はじめに

大量のテキストによって学習された自己回帰型言語モデルが自然言語処理のタスクにおいて高い性能を達成している [2, 3]．しかし，タスクを解く際のモデル内部における処理の種類や導出過程の詳細は未解明である．モデルの内部分析については，既存研究においても様々行なわれてきたものの [4, 5, 6, 7]，とりわけ，自己回帰型言語モデルのような逐次的に単語を生成するモデルが「複合的な問題について，いつどのような部分問題を問いているか」という時間方向の解析については研究が途上である．

本研究では，算術推論タスクを解いた時の自己回帰型の言語モデルの隠れ状態を入力として計算の途中結果を予測するように学習した回帰モデルの性能の違いから，計算の途中結果がモデル内部のどこに表現されているかを観察 (プロービング) することで，言語モデルが推論を行う際の内部機序を明らかにする．まず，モデルの振る舞いがある程度予測可能な人工的設定かつ，事前訓練をしていないモデルを用いた実験を通して，今回用いる解釈方法により妥当な観察が得られることを確認した．次に，事前学習済みの言語モデルに対して提案法を適用し推論時のモデルの内部機序の分析を行った．結果として，事前学習済み言語モデルは思考の連鎖 (Chain of Thoughts) [1] による推論時，モデル内部で結論を得て後付け的に中間ステップを生成しているのではなく，推論過程を出力すると同時に途中の計算も行っていることなどが明らかになった．

2 分析手法

2.1 算術推論タスク

言語モデルが推論を行う際の内部機序を明らかにするため，数量推論タスクを抽象化したような人工的な算術タスクを用いる．本タスクは，既存研究 [8] で提案された算術タスクをベースとしものであり，タスクを構成する各問題は (i) A=1，(ii) A=B，(iii) A=1+2， (iv) A=B+2，という 4 つの**基本式**の並びと，モデルに出力を求める変数名を示す質問 (A=) によって構成されている．ただし，数式中に現れる数字や変数名，演算子は無作為に選ばれる．本論文の評価データとして用いる問題例は以下の通りである．

入力： A=1+B,B=2+3,C=4+5;A= 出力： 6 (1)

モデルは入力として定義部 (A=B+1,B=2+3,C=4+5;) と質問部 (A=) を受け取り，解答部 (6) の出力を行うように学習を行う．式 1 は 3 つの基本式からなる問題である．質問部 (A=) で問われている変数の値を求めるためには，最初に B の値を求める必要があるという依存関係が存在する．また，モデルには B の定義よりも先に A の定義が入力されるため，単純に入力順に変数の値を求めることは不可能であるという特徴を持つ．加えて，この問題には質問部の変数の値を求めるためには不必要な式 (C=4+5) が含まれて

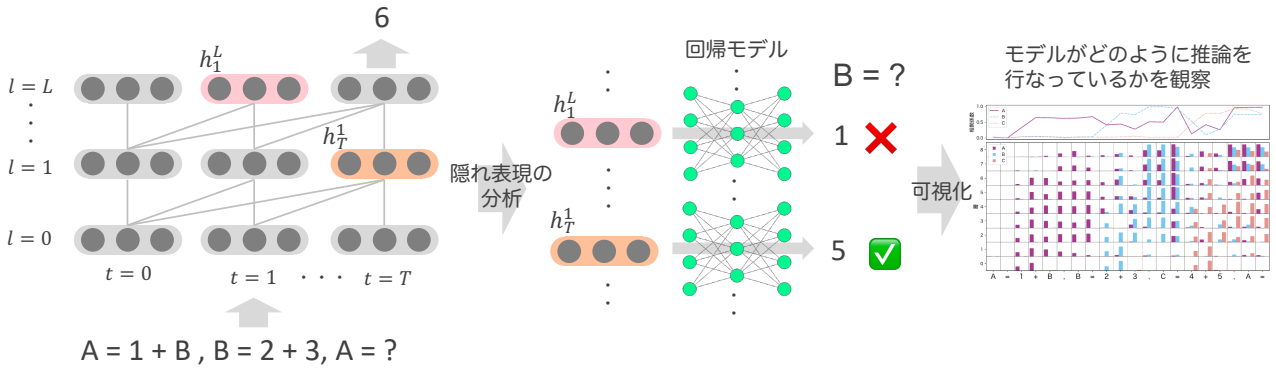

図 1　本研究の概観. 言語モデルの隠れ状態を入力として計算の途中結果 (図中では B の値) を予測するように学習した回帰モデルの性能の違いから，モデル内部のどこに表現されているかを観察することを通して，言語モデルが推論を行う際の内部機序を明らかにする．

いる．このような性質を持つ問題を解いた際の言語モデルに対する分析結果を観察することで，言語モデルが推論を行う際の内部機序を議論する．

分析のための回帰モデルの学習データ，評価データを構築時には，この数式と同様の基本式の並びの問題を，数式中に現れる数字や変数名，演算子を無作為に変更することで大量に生成する．式 1 の場合の各変数の値はそれぞれ，6 (=A)，5 (=B)，9 (=C) であり，これらの値を予測する回帰モデルを学習する．本論文では，説明のために式 1 で用いられている変数名や数字を参照して議論を行う．実際の評価データでの各事例で使われる変数名や数字は固定ではなく，無作為に変数名や数字が選ばれた 2,000 事例を評価データとして利用していることを留意されたい．その他の詳細は B 節を参照されたい．

2.2　回帰モデルの学習

モデルが推論を行う際の内部機序を明らかにするために，本研究では回帰モデルを用いた分析を行う．具体的には 2.1 節で述べた算術タスクを言語モデルに入力した際の各時刻の隠れ状態を入力として，問題中の特定の変数の値を出力する回帰モデルを作成し，このモデルの予測を通して，どの変数に対する計算がモデルのどこ (層の深さ) でいつ (時刻) 行われたかを分析する．

具体的に，自己回帰型言語モデルに対して，t トークン目の入力が与えられた時の l 層目のモデルの出力 (隠れ状態) を $\mathbf{h}_t^l \in \mathbb{R}^d$ とする．ここで，d はモデルの隠れ状態の次元数である．この隠れ状態 $\mathbf{h}_t^l$ を入力として，特定の変数の値を予測する回帰モデルを学習する．

$$\hat{y} = f_t^l(\mathbf{h}_t^l; W_t^l) \tag{2}$$

回帰モデルのパラメータを W とし，$\hat{y} \in \mathbb{R}^1$ は回帰モデルの予測値である．今回用いる数式はフォーマットが固定されており，変数が 3 つ導入される．それぞれの変数の予測に特化した回帰モデルを個別に学習する．本研究では回帰モデルとして 2 層の多層パーセプトロンを用いる．この回帰モデルは以下のように，正解の変数の値と予測値の平均二乗誤差を最小化するように学習する．また，この回帰モデルの学習には，対象となる言語モデルが問題に正解した際の隠れ状態のみを用いる．学習データ数等の詳細は A.1 節を参照されたい．

2.3　評価指標

学習した回帰モデルの評価には，N 個の評価データに対する回帰モデルの予測値 $\hat{\mathbf{y}}_t^l$ と正解の変数の値 $\mathbf{y}$ のピアソンの積率相関係数 ρ_t^l を用いる．

3　人工データのみで学習した言語モデルに対する分析

初めに，統制された人工データのみで学習した言語モデルに対して 2 節で述べた提案法を適用し，提案法の妥当性を確認する．具体的には，2.1 節で述べた算術タスクを解くためのモデル 2 つを異なる学習データを用いてそれぞれ学習する．これら 2 つのモデルに対して 2.2 節で述べた回帰モデルの学習を行い，それぞれのモデルの推論時の内部機序の狙った違いが可視化できることを示す．

3.1　実験設定

評価データ　本実験では，評価タスクとし式 1 に例として示した基本式の並びの数式を用いる．

学習データとモデル　異なる学習データを用いた二つの自己回帰型言語モデルを分析する：

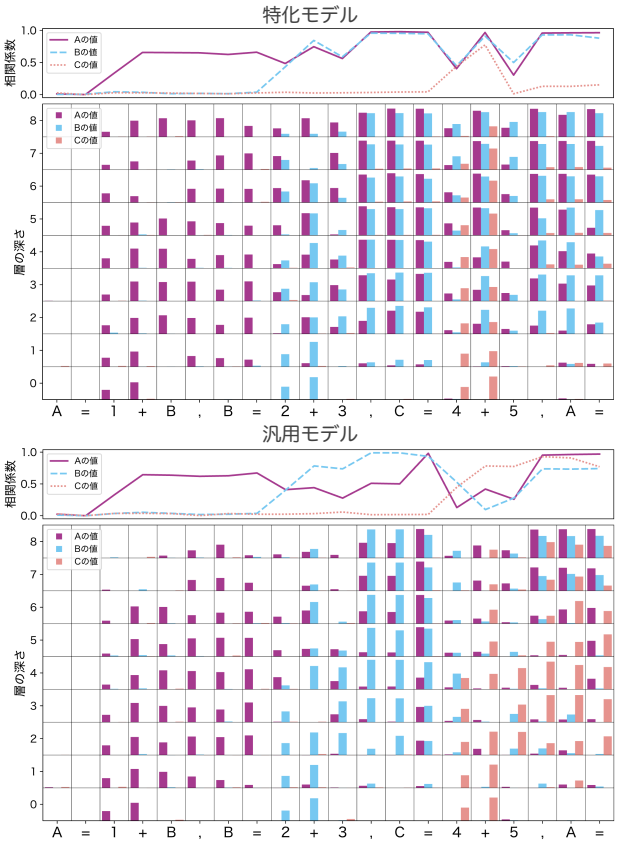

図 2　人工データで学習したモデルに対する分析結果．横軸は時間方向にモデルに入力されたトークンを表している．各グラフ上部，折れ線グラフの縦軸は各トークンの位置 t ごとの最大の相関係数をあらわす，またグラフ下部の縦軸は入力から数えた層の深さを表し，各ブロックは問題文中の変数との相関を示している．解答に不必要な式の値 C との相関の強さがモデル間で異なっていた．

- **特化モデル**: 評価データと基本式の並びが同一の問題のみを学習データとして用いており，式 1 において，C の値を計算しなくてよいことを知っている．
- **汎用モデル**: 3 つ以下の基本式のランダムな組み合わせからなる問題を学習データとして用いており，式 1 において，C の値を計算しなくてよいことを知らない．

ハイパーパラメータなどは A.2 に示す．

3.2　実験結果

汎用モデル，特化モデルに対して提案法による分析を行った結果を図 2 に示した．図 2 において，各グラフの下部のそれぞれのブロックには，言語モデルの入力部分から数えた層の深さ l，横軸はモデルへの入力のトークンの位置 t における隠れ状態を入力として学習した回帰モデルの予測値と正しい変数の値との相関を表している．各ブロックに表された棒グラフは，3 種類の変数 (A, B, C) の値との相関を表している．各グラフ上部の折れ線グラフでは，モデルへの入力のトークンの位置 t ごとの最大の相関値を表している．また，特化モデル，汎用モデルいずれの場合も正解率は評価データに対する正解率は 98.8 %，99.5 %であった．

計算の実行タイミング　2.1 節で述べた，式 1 における変数 A，B の依存関係着目すると，いずれのモデル場合でも変数 B を求めるための数式 B=2+3 がモデルに入力された直後に B の値との強い相関が見られた．その直後に，B に依存している A との高い相関観察された．このことから，各変数について変数の値が計算可能になった時点で計算を行うという貪欲な戦略，特にプログラミング言語の評価戦略におけるユニフィケーションに相当する戦略をとっていると考えられる．

解答に不必要な数式の扱い　2.1 節で述べたように，式 1 のような評価タスクに用いる数式には解答に不必要な数式 (C=4+5) が含まれている．このような数式の計算結果 (C の値) と回帰モデルの予測値の相関は，汎用モデルにおいては 0.9 程度の高い一方で，特化モデルにおいては最大 0.7 程度と比較的低かった．このことから，汎用モデルは解答に不必要な数式に対しても計算を行っており，全ての計算式についてその値を最終的に生成するかどうかにかかわらず計算を行う戦略を取るモデルであると考えられる．一方で，特化モデルは解答に不必要な数式に対しては厳密に計算を行はず，必要最低限の計算のみを行う効率的な戦略を取っているモデルになっていると考えられ，狙った差異が得られている．以上から，提案法を用いることで言語モデルが推論を行う際の内部機序を議論できる可能性があることが示唆された．

4　事前学習済み言語モデルの分析

3 節では，人工データのみで学習した言語モデルに対して提案手法を適用し，言語モデルが推論を行う際の内部機序を可視化する手段として有望であることを示した．本節では，提案法を事前学習済みの言語モデルに対して適用し，言語モデルが推論を行う際の内部機序の分析を行う．特に，事前学習済みモデルが言語モデルが解答を生成する際に，問題に対する答えだけでなく，その解答を導くまでの推論過程を出力させる手法である思考の連鎖 [1] による

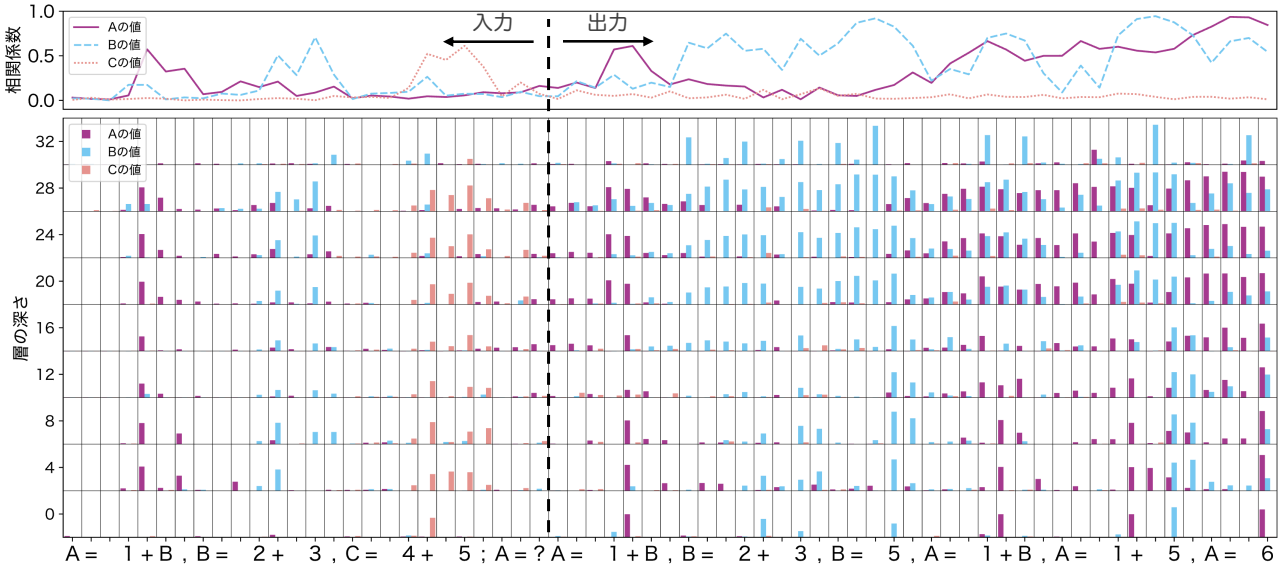

図 3　事前学習済み言語モデルに対する分析結果．横軸は時間方向にモデルに入力されたトークンを表している．各グラフ上部，折れ線グラフの縦軸は各トークンの位置 t ごとの最大の相関係数をあらわす，またグラフ下部の縦軸は入力から数えた層の深さを表し，各ブロックは問題文中の変数との相関を示している．事前学習済みモデルの場合，出力部で途中の計算結果との比較的強い相関が観察された．

推論時の内部機序に焦点を当てる．

4.1　実験設定

評価データ　本実験では，2.1 節で述べたタスクと同様の数式を用いる．ただし，解答部は以下のように途中の推論過程も出力するように変更する．

$$\begin{aligned} &\text{入力 :A=1+B,B=2+3,C=4+5,A=?} \\ &\text{出力 :A=1+B,B=2+3,B=5,A=5+1,A=6} \end{aligned} \tag{3}$$

その他の詳細については B 節を参照されたい．

モデルと文脈内学習　事前学習済みの言語モデルに評価タスクを解かせるために，文脈内学習 [9] を行う．本研究では評価タスクと同形式の 3 つの入出力の組を文脈としてモデルに入力し，その後に評価データの入力を与える．文脈として与える事例は，評価データと同様の形式の数式ではあるが，評価データに含まれる全ての事例とは異なるものである．分析対象のモデルには Mistral [10] を用いる．

4.2　実験結果

事前学習済み言語モデルに対して提案法による分析を行った結果を図 3 に示した．本評価タスクに対するモデルの正解率は 98.3 %であった．

計算の実行タイミング　図 3 の結果から，各変数の値との相関は入力部分では小さく，モデルが途中の推論過程を生成する過程 (出力部分) で高くなる様子が観察された．したがって，モデルは，問題文が入力された時点では解答を導けているのではなく，推論過程を生成する過程で同時に推論も行い解答に至っていることが示された．

解答に不必要な数式の扱い　図 3 の結果から，解答に不必要な数式の計算結果 (変数 C の値) との相関は最大 0.6 程度と比較的低い結果となった．したがって，本研究で対象とした事前学習済み言語モデルは，解答に不必要な数式の計算は行わない効率的な戦略を取っていることが示唆された．

5　おわりに

本研究では，算術推論タスクを解く自己回帰型の言語モデルの内部表現を対象に，計算の途中結果がモデル内部のどこに表現されているかを観察することを通して，言語モデルが推論を行う際の内部機序を明らかにした．実験では，初めにモデルの振る舞いがある程度予測可能な人工的設定で，今回用いる解釈方法により妥当な観察が得られることを確認した．次に，事前学習済み言語モデルに対して提案法を適用し，モデルは推論過程を出力すると同時に途中の計算も行っていることが明らかとなった．

本論文の実験では単一の事前学習済み言語モデル，文脈の設定での分析に留まっている．そのため，今後は提案法を利用して異なるモデルや文脈に対する包括的な分析を行うことが必要である．また，将来的には算術推論に限らない内部機序の分析手法の検討していきたい．

謝辞

本研究は，JST CREST JPMJCR20D2 及び JSPS 科研費 JP21K21343 及び理化学研究所の基礎科学特別研究員制度の支援を受けたものですまた，本研究を進めるにあたり多くの協力を賜りました Tohoku NLP グループの皆様に感謝申し上げます.

参考文献

[1] Dingjun Wu, Jing Zhang, and Xinmei Huang. Chain of Thought Prompting Elicits Knowledge Augmentation. In **Findings of the Association for Computational Linguistics (ACL)**, pp. 6519–6534, 2023.

[2] OpenAI. GPT-4 Technical Report. **CoRR**, Vol. arXiv preprint, cs.CL/arXiv:2303.08774, , 2023.

[3] Gemini Team, Rohan Anil, and Sebastian Borgeaud et al. Gemini: A family of highly capable multimodal models. **CoRR**, Vol. arXiv preprint, cs.CL/arXiv:2312.11805, , 2023.

[4] Yuta Matsumoto, Benjamin Heinzerling, Masashi Yoshikawa, and Kentaro Inui. Tracing and Manipulating intermediate values in Neural Math Problem Solvers. In **Proceedings of the 1st Workshop on Mathematical Natural Language Processing (MathNLP)**, pp. 1–6, 2022.

[5] Alessandro Stolfo, Yonatan Belinkov, and Mrinmaya Sachan. A Mechanistic Interpretation of Arithmetic Reasoning in Language Models using Causal Mediation Analysis. In **Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)**, pp. 7035–7052, December 2023.

[6] Kenneth Li, Aspen K Hopkins, David Bau, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Emergent World Representations: Exploring a Sequence Model Trained on a Synthetic Task. In **The Eleventh International Conference on Learning Representations (ICLR)**, 2023.

[7] Koyena Pal, Jiuding Sun, Andrew Yuan, Byron Wallace, and David Bau. Future Lens: Anticipating Subsequent Tokens from a Single Hidden State. In **Proceedings of the 27th Conference on Computational Natural Language Learning (CoNLL)**, pp. 548–560, 2023.

[8] Keito Kudo, Yoichi Aoki, Tatsuki Kuribayashi, Ana Brassard, Masashi Yoshikawa, Keisuke Sakaguchi, and Kentaro Inui. Do Deep Neural Networks Capture Compositionality in Arithmetic Reasoning? In **Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics (EACL)**, pp. 1351–1362, 2023.

[9] Tom Brown, Benjamin Mann, and Nick et al. Ryder. Language Models are Few-shot Learners. In **Advances in Neural Information Processing Systems (NeurIPS)**, Vol. 33, pp. 1877–1901, 2020.

[10] Albert Q. Jiang, Alexandre Sablayrolles, and Arthur Mensch et al. Mistral 7B. **CoRR**, Vol. arXiv preprint, cs.CL/arXiv:2310.06825, , 2023.

[11] Herbert E. Robbins. A stochastic approximation method. **Annals of Mathematical Statistics**, Vol. 22, pp. 400–407, 1951.

[12] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep Sparse Rectifier Neural Networks. In **Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics (PMLR)**, Vol. 15, pp. 315–323, 2011.

[13] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. In Yoshua Bengio and Yann LeCun, editors, **3rd International Conference on Learning Representations, ICLR, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings**, 2015.

[14] Ilya Loshchilov and Frank Hutter. SGDR: stochastic gradient descent with warm restarts. In **5th International Conference on Learning Representations, ICLR, Toulon, France, April 24-26, 2017, Conference Track Proceedings**, 2017.

[15] Susan Zhang, Stephen Roller, and Naman Goyal et al. OPT: Open Pre-trained Transformer Language Models. **CoRR**, Vol. arXiv preprint, cs.CL/arXiv:2205.01068, , 2022.

[16] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention Is All You Need. In **Advances in Neural Information Processing Systems 31 (NIPS)**, pp. 5998–6008, 2017.

学習データ数	10,000 事例
最適化器	SGD [11]
学習率	1.0×10^{-3} (一定)
バッチサイズ	10,000 事例
エポック数	30

表 1　回帰モデルの学習時のハイパーパラメータ

A　学習設定の詳細

A.1　回帰モデルの学習設定

2.2 節で述べたように，回帰モデルとして 2 層の多層パーセプトロンを用いる．具体的には，以下のような式で表される関数 f_t^l を用いる．

$$f_t^l(\mathbf{h}_t^l, {}^1\mathbf{W}_t^l, {}^2\mathbf{W}_t^l, {}^1\mathbf{b}_t^l, {}^2\mathbf{b}_t^l) = {}^2\mathbf{W}_t^l \sigma({}^1\mathbf{W}_t^l \mathbf{h}_t^l + {}^1\mathbf{b}_t^l) + {}^2\mathbf{b}_t^l \tag{4}$$

ここで，${}^1\mathbf{W}_t^l \in \mathbb{R}^d$，${}^2\mathbf{W}_t^l \in \mathbb{R}^{1\times d}$，${}^1\mathbf{b}_t^l \in \mathbb{R}^d$，${}^2\mathbf{b}_t^l \in \mathbb{R}^1$ は回帰モデルのパラメータであり，σ は活性化関数である．本研究では σ として ReLU [12] を用いる．

また，回帰モデルの学習は正解の変数の値と予測値の平均二乗誤差を最小化するように行う．

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2 \tag{5}$$

ここで N は学習データのサンプル数である．その他のハイパーパラメータについては，表 1 に示す．

A.2　人工データのみで学習するモデルアーキテクチャの詳細

Transformer のデコーダをベースとした自己回帰型の言語モデルを用いる．モデルは 8 層の Transformer 層からなり，アテンションヘッドの数は 16，隠れ層の次元数 1024 である．アーキテクチャの実装としては Hugging Face Transformers の OPT [15] (https://huggingface.co/docs/transformers/model_doc/opt) を用いた．また，入力文のトークナイズは数字単位で行い，各数字の単語埋め込みには Transformer [16] の絶対位置埋め込みとして用いられている埋め込みを利用している．これは，数字の埋め込みが規則的に並ぶことによって．学習時にモデルが数字の大小関係等を容易に学習できることを期待したものである．また，変数名や演算子等の数字以外の埋め込みについてはランダム初期化した埋め込みを用いる．いずれの埋め込みについても学習時にはパラメータを固定し更新しない．その他のハイパーパラメータについては表 2 に示す．

特化モデル	
学習データ数	200,000 事例
最適化器	Adam [13] ($\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 1 \times 10^{-8}$)
学習率のスケジューラ	コサインスケジューラ [14]
学習率のウォームアップ	500
学習率 (最大)	3.0×10^{-5}
勾配のクリッピング	1.0
ドロップアウト	0.1
バッチサイズ	16,384 トークン
エポック数	500
汎用モデル	
学習データ数	1,000,000 事例
最適化器	Adam [13] ($\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 1 \times 10^{-8}$)
学習率のスケジューラ	コサインスケジューラ [14]
学習率のウォームアップ	500
学習率 (最大)	1.0×10^{-4}
勾配のクリッピング	1.0
ドロップアウト	0.1
バッチサイズ	16,384 トークン
エポック数	50

表 2　特化モデル，汎用モデルの学習時のハイパーパラメータ

B　評価データの詳細

人工データのみで学習した言語モデルに対する分析で用いる評価データ　各事例に含まれる数字は 0 から 99 までの自然数，変数名は 25 種類としている．また，使用する演算子は足し算または引き算としている．

事前学習済みモデルの分析で用いる評価データ　事前学習済みモデルの分析の場合，評価データに含まれる数字は，出力部も含めて全て 1 桁の自然数となる事例のみを用いている．これは，本実験で用いる事前学習済みモデルである Mistral は数字が桁ごとにトークン化されているため，入出力のトークン数を全ての評価事例で揃えることを目的としている．