Inductive-bias Learning：大規模言語モデルによる予測モデルの生成

田中 冬馬 [1]　江本 直史 [1]　弓林 司 [1]
[1] 株式会社ブレインパッド
{toma.tanaka,naofumi.emoto,tsukasa.yumibayashi}@brainpad.co.jp

概要

大規模言語モデル (LLM) はパラメータを更新することなくプロンプトに入力されたデータをもとに推論を行う In-context Learning(ICL) と呼ばれる能力が注目されている．また、コード生成も LLM の重要な応用先である．本論文では、ICL とコード生成を応用した”Inductive-bias Learning(IBL)”と呼ばれる新しい学習手法を提案する。IBL では ICL と同様に、学習データをプロンプトに入力し、推論を行うために必要な関係性を持つコード（Code Model と呼ぶ）を生成する。シンプルなアプローチであるが、IBL は学習データがどのような関係性を持つか捉えることができ，さらにコードを生成するため解釈性も兼ね備えている．また驚くべきことに、生成された Code Model は代表的な機械学習モデルに匹敵する予測精度を達成する．
IBL のコードはオープンソースです. 1)

1　はじめに

パラメータ数の多い言語モデルは、学習データ（入出力ペアなど）をプロンプトに直接入力することで、データの中の関係性を「文脈の中で」捉え，高精度な推論ができることが確認されており，この能力を生かした幅広いタスクへの応用が期待される。[1]．これは In-context Learning(ICL) と呼ばれる大規模言語モデルの能力である。(数式 1, 図 1):

$$\text{LLM.input}\left[D\left[(x_1, y_1), \cdots, (x_n, y_n)\right], x_{\text{input}}\right] \rightarrow y_{\text{pred}} \quad (1)$$

ここで、図 1 の左側は学習データと予測したデータの入力を表すプロンプト、図の右側は LLM の出力結果である。

ICL の発展を簡単に概観する．まず、その構造に関する研究として、([2][3]) は LLM が ICL 能力を獲得するために必要な訓練データの条件について研究されている．また、[4][5][6] では、ICL による LLM の推論過程と勾配降下との関係を研究している。ICL による推論の仕組みについてはまだ不明な点が多く，今後の研究が必要である．

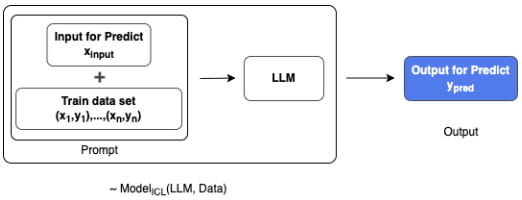

図 1　In-context Learning

さらに、その性能に関する研究として、[7] は、Transformer の事前学習により、スパース線形関数、2 層ニューラルネットワーク、決定木など、ICL による様々な関数の学習が可能であることを示している．したがって、適切な条件を満たす LLM は、ICL の能力に従って入力と出力の例をいくつか与えることで、未知のデータの関係を捉え、入力に対応する出力を予測することができる．

ここで、ICL についての一つの疑問が出てくる

LLM が入力例と出力例から学習した「モデル」を直接出力することはできないか？

上記の課題が解決できれば、LLM が入力例と出力例からどのようなデータ間の関係を見出したかを明示的に示すことができる．さらに、帰納バイアスを明示的に設定し学習を行う従来の機械学習 (Figure 2)」とは異なる、ICL のような汎用的な推論を行うことができる．

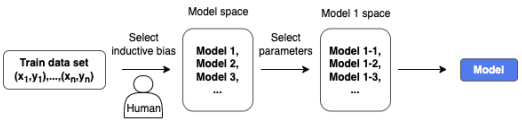

図 2　従来の機械学習．人による帰納バイアスの選択

これは、LLM がデータから学習方法を学習して

1）GitHub Link：https://github.com/fuyu-quant/IBLM

いると考えることができ、ICL よりも一段高い視点を扱うことができるということでもある。メタ学習にも関連する話題である．

この問題を解決するために、以下のように具体的な定式化を行う．

LLM に学習データ（ここでは説明変数と目的変数のペア）を入力とし、説明変数から目的変数を推論するモデルを出力することができるか？

この命題に対する我々の答えは「YES」である．この方法を Inductive-bias Learning(IBL) と名付けた (数式 2, 図 3)．Inductive-bias Learning という名前は、帰納バイアスを明示的に仮定することなく、データごとに様々な構造のモデルを出力することに由来する．

$$\begin{aligned}&\text{LLM.input}\left[D\left[(x_1,y_1),\cdots,(x_n,y_n)\right]\right]\\&\rightarrow \text{Code Model}_{\text{LLM},D}(x_{\text{input}})\rightarrow y_{\text{pred}}\end{aligned}\tag{2}$$

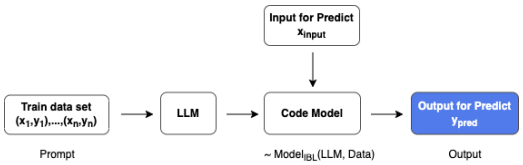

図 3 Inductive-bias Learning

本論文では、次のような検証を行った．複数のデータセットとシード値を用いてデータをサンプリングし、代表的な機械学習アルゴリズムと精度の比較を行った．タスクとして二値分類問題を，LLM としては GPT-4 を使用した．命題にあるように、IBL は学習データから推論構造そのものを出力する．さらに、これは様々なデータに対して行うことができる。このことは，LLM がデータが与えられたときに入力と出力の関係をどのように学習するかを学習している可能性を示唆しており，LLM は一種のメタ学習を行うことができていると考えられる．

2 Inductive-bias Learning

Inductive-bias Learning（IBL）は、通常の教師あり機械学習アルゴリズムと同様に、学習と推論の 2 つのフェーズに分けることができる．学習フェーズでは、説明変数と目的変数のペアを入力とし、説明変数から目的変数を予測するためのコードを生成する．推論では、生成したコードを用いて未知のデータを予測することができる．本研究では二値分類タスクで検証を行っている．詳細は後述する (図 4)。

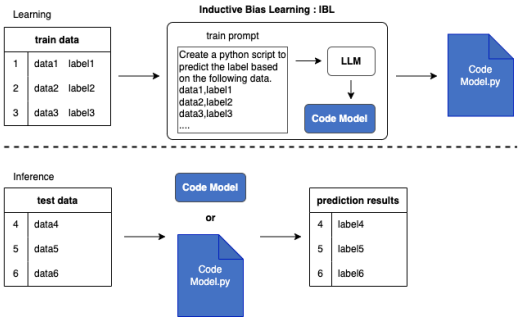

図 4
上図：IBL によるモデル生成... 通常の教師あり学習と同様に、説明変数と目的変数の複数の組からなる訓練データを入力しコードを作成する．コードは Python コードとして生成した．
下図：IBL で生成したモデルの予測... 通常の機械学習モデルと同様に予測したいデータの説明変数を入力する．生成されたコードを用いて予測を行う．

2.1 学習

IBL の学習フェーズでは、与えられた学習データを用いて、説明変数から目的変数を出力するための Python で書かれたコードを生成する．この生成された Python コードを “Code Model” と呼ぶ。

また，LLM に入力するプロンプトには入出力ペアである学習データに対して，二値分類タスクとしての精度評価のためにラベル 1 に分類される確率値を出力するようなコードを生成する．今回の検証では、GPT-4 がデータから予測を行う “論理構造” を出力できるかに関心があったため機械学習モデルを使うコードは扱わない．

出力は，説明変数を入力とし目的変数のラベルが 1 である確率値を出力値とする Pyhton のコードを生成する．生成された Python コードは，特定の特徴量による条件分岐や，複数の特徴量を含む線形の関係式などが生成される．また，学習に用いるデータセットやデータ数に応じて様々な出力結果が得られた．

2.2 推論

IBL の推論フェーズは，学習によって生成された Python コードを用いて実行される．

IBL で生成される Python コードは，特定の特徴量による条件分岐や、複数の特徴量を含む線形な関係式などの単純な処理が生成されるため非常に高速である．一方で、そのような単純なロジックにもかかわらず生成された Code Model は非常に高い予測精度を持つことが確認されている．

さらに，IBL が生成した Code Model は Python のコードとして出力されるためモデルの解釈も非常に容易である．

3　実験と結果

本節では、IBL の性能を検証するための実験結果を記載している．はじめにいくつかのデータセットについて IBL と典型的な機械学習モデルの予測能力を比較した．

3.1　実験概要

データセットとタスク

IBL によって生成された Code Model の精度を機械学習アルゴリズムや ICL と、二値分類タスクを使い比較した．機械学習モデルとの比較では scikit-learn で利用可能な擬似データセットと moon データセットを扱った．各シード値で学習に使用されなかったデータはテストデータとした。scikit-learn の擬似データと moon データセットについては，3 つの異なるシード値を用いて合成データを生成し，訓練データとテストデータに分割した．小数点以下が多すぎると IBL や ICL に入力できるデータ点数が少なくなるため生成した値はすべて小数点 3 桁とした．

また学習用データの不均衡を解消するため、正例と負例の数を同じにした．

IBL 実行回数

IBL の問題点として，生成されたコードモデルの出力が不安定であることが挙げられる．IBL が生成したコードモデルは、必ずしも安定して高い予測精度を出すとは限らず精度にばらつきがある．そこで，データセット，学習データサイズ，シード値ごとに 30 個のコードモデルを生成し AUC が最も高いものを比較した。本稿では IBL の実行に OpenAI の API で gpt-4-0613 を用いて検証を行った．ファインチューニングなどの追加の学習は行わずプロンプト入力のみを使用した．

3.2　IBL と機械学習モデルの比較

最初の検証では、擬似データセット、Moon データセットを用いて，IBL が生成したコードモデルの AUC の値を代表的な機械学習モデルと比較した．比較のための機械学習モデルとしては、ロジスティック回帰、K-NN（K-Nearest Neighbors）、リニアカーネルの SVM を用いた。IBL では、各シード値に対してコードモデルを 30 回生成し、AUC が最も高いモデルを選択した．なお，生成に成功した Code Model がない場合はグラフにプロットを行っていない．

擬似データセット

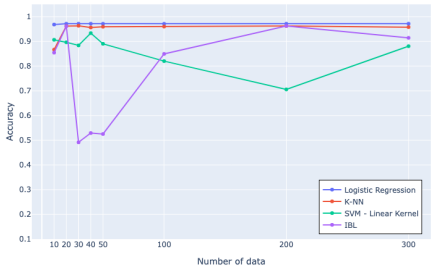

図 5　seed=3655

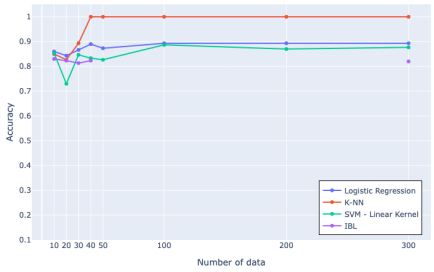

図 6　seed=3656

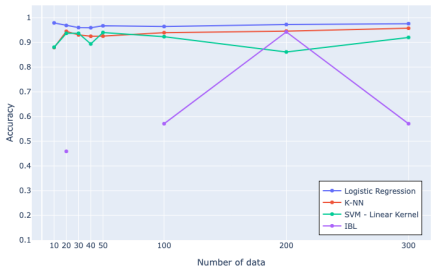

図 7　seed=3657

図 5，6，7 より，他の機械学習モデルと同等の AUC を出力する場合もある．

Moon データセット

図 8，9，10 より，Moon データセットにおける AUC は，K-NN やロジスティック回帰の AUC には劣るものの SVM の AUC を上回ることも多々ある．

疑似データセットや Moon データセットでは説明変数の名前から目的変数の値を予測することは不可能であり、また適当な seed 値により生成したデータから scikit-learn のライブラリを用いて生成されたデータであることを識別することは非常に困難であ

図 8　seed=3655

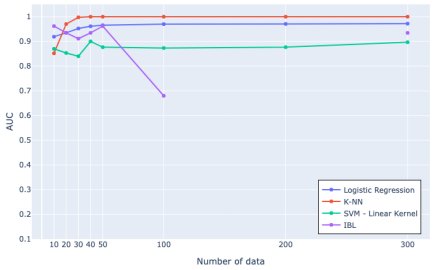

図 9　seed=3656

ると考えられる．そのため，LLM はデータ間の論理構造に基づいて推論のためのロジックを出力していると考えられる．連続的な説明変数と離散的な説明変数のどちらが IBL で扱いやすいかを判断するためには、異なるデータセットでさらなる検証を行う必要がある。生成された Code Model の一部を付録として添付している．

4　今後の展望

IBL は，新しい学習手法のパラダイムシフトとなる可能性を秘めている．以下に今後重要になると思われる項目をまとめる．

IBL の精度向上と応用性

現在の IBL ではファインチューニングなどの追加の学習を行わずに実行しているため出力の不安定性や精度の課題がある．IBL 専用のデータセット (例えば予測したい論理関係の Python コードとその論理関係からサンプリングしたデータ) を使い学習する方法が考えられる．GPT-4 などの LLM では大量の自然言語データによる学習を行っているため自然言語内にあるような論理構造を把握することはできるかもしれないがそうでないような論理関係などは捉えることができないと考えている．

さらに，バギングやブースティングのような統計的学習理論のテクニックの活用や作成された Code Model の編集，学習データと一緒に入力されるプロンプトの最適化 [8] などでも，より精度の高いモデルが生成されることが期待される．

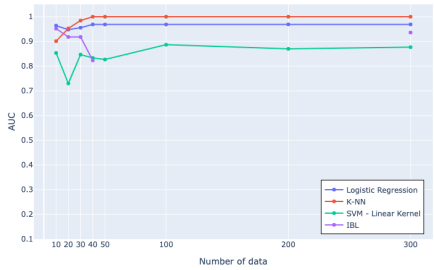

図 10　seed=3657

また，LightGBM[9] のような社会実装されている機械学習アルゴリズムを IBL に置き換える可能性も検討している．予測の精度や出力の安定性などがクリアできればモデルの推論速度や解釈可能性という点で，Code Model を使う意義は多い．

LLM の論理構造の把握能力

IBL が学習データからどのように予測ロジックを推論して Code Model を生成するのか，また生成された Code Model の予測ロジックがどの程度優れているのかについては，さらに詳しい分析が必要である．後者については、[7] による研究では ICL は最小二乗推定に匹敵する精度で線形関数を近似できることが示唆されている．この方法を使い IBL を用いて LLM がどれだけ正確にデータから回帰係数を求めることができるかを検証することは可能である．

また，IBL がどのようなデータに対しても最適な予測ロジックを出力できるかは興味深い話題である．sequence-to-sequence の万能近似定理 [10] が適用できる場合入力されたデータに対して理想的な予測ロジックを出力するように学習することもできる可能性がある．

5　結論

本論文では、LLM を用いて予測モデルを生成する Inductive-bias Learning(IBL) と呼ばれる新しい学習法を提案した．この学習手法によって生成された予測モデルである Code Model は，従来の機械学習モデルに匹敵する予測精度を達成する場合もあった．IBL によって生成された予測モデル（Code Model）は、今後精度が向上すれば，解釈可能性や推論速度の点で既存の機械学習モデルを置き換える可能性があると考えている．さらに，LLM の論理構造の把握能力について理解するのに寄与するとも考えている．

謝辞

この研究を進める中で、絶えず支援と励ましをくださった先輩方に感謝します。弓林さんには理論的な議論において貴重な助言をくださりました．また．江本さんには実装について多くのアドバイスをいただいたり実際に実装を手伝っていただきました．感謝いたします．

参考文献

[1] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. **Advances in neural information processing systems**, Vol. 33, pp. 1877–1901, 2020.

[2] Stephanie Chan, Adam Santoro, Andrew Lampinen, Jane Wang, Aaditya Singh, Pierre Richemond, James McClelland, and Felix Hill. Data distributional properties drive emergent in-context learning in transformers. **Advances in Neural Information Processing Systems**, Vol. 35, pp. 18878–18891, 2022.

[3] Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. An explanation of in-context learning as implicit bayesian inference. **arXiv preprint arXiv:2111.02080**, 2021.

[4] Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. What learning algorithm is in-context learning? investigations with linear models. **arXiv preprint arXiv:2211.15661**, 2022.

[5] Arvind Mahankali, Tatsunori B Hashimoto, and Tengyu Ma. One step of gradient descent is provably the optimal in-context learner with one layer of linear self-attention. **arXiv preprint arXiv:2307.03576**, 2023.

[6] Johannes Von Oswald, Eyvind Niklasson, Ettore Randazzo, João Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. Transformers learn in-context by gradient descent. In **International Conference on Machine Learning**, pp. 35151–35174. PMLR, 2023.

[7] Shivam Garg, Dimitris Tsipras, Percy S Liang, and Gregory Valiant. What can transformers learn in-context? a case study of simple function classes. **Advances in Neural Information Processing Systems**, Vol. 35, pp. 30583–30598, 2022.

[8] Alessandro Sordoni, Xingdi Yuan, Marc-Alexandre Côté, Matheus Pereira, Adam Trischler, Ziang Xiao, Arian Hosseini, Friederike Niedtner, and Nicolas Le Roux. Deep language networks: Joint prompt training of stacked llms using variational inference. **arXiv preprint arXiv:2306.12509**, 2023.

[9] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. **Advances in neural information processing systems**, Vol. 30, , 2017.

[10] Chulhee Yun, Srinadh Bhojanapalli, Ankit Singh Rawat, Sashank J Reddi, and Sanjiv Kumar. Are transformers universal approximators of sequence-to-sequence functions? **arXiv preprint arXiv:1912.10077**, 2019.