大規模言語モデルを応用したエラー診断モデルによる学習支援に向けて

高橋　舞衣† 　小原百々雅†† 　相馬　菜生† 　倉光　君郎†

† 日本女子大学理学部数物科学科　〒 112–8681 東京都文京区目白台 2-8-1
†† 日本女子大学大学院理学研究科数理・物性構造科学専攻　〒 112–8681 東京都文京区目白台 2-8-1
E-mail: †{m1916046tm, m1916045sn}@ug.jwu.ac.jp, kuramitsuk@fc.jwu.ac.jp, ††m1816019om@ug.jwu.ac.jp

あらまし　近年，社会のデジタル化が進み，プログラミングを学び始める人が増えている．しかし，プログラミングの習得は難しく，特に初学者にとって頻繁に発生するエラーメッセージはつまずきの原因となる．本研究では，大規模言語モデルを用いて，エラーメッセージをより初学者に親切なメッセージに書き換えて提示する学習モデルの構築を進めている．具体的には，エラーが発生した行などの情報を追加し，単純な翻訳ではなく，エラーの原因と修正するためのヒントを詳しく提示することを目指している．本発表では，初学者から取得したエラー情報から作成したエラーコーパスを大規模言語モデルで学習し，構築したエラー診断モデルの精度を報告する．

キーワード　機械学習, エラー診断, プログラミング学習支援, 大規模言語モデル

1　はじめに

近年，社会のデジタル化にともない，プログラミング学習者が増えている．しかし，プログラミングの習得は容易ではない．特に，プログラミングのエラーメッセージは初学者にとって分かりづらく，エラーメッセージを正しく理解し，エラーを解決するのは困難である．そこで我々は，大規模言語モデルを用いて，エラー解決を手助けすることを目指す．

本論文では，エラーメッセージをより親切に提示するために，エラーの情報からエラー解決のヒントを出力するエラー診断モデルの構築について報告する．本論文の残りの構成は以下の通りである．2 節では，大規模言語モデルについて説明する．3 節では，エラー診断モデルの構築方法を提案する．4 節では，実験について述べる．第 5 節では，関連研究を外観し，6 節で本論文をまとめる．

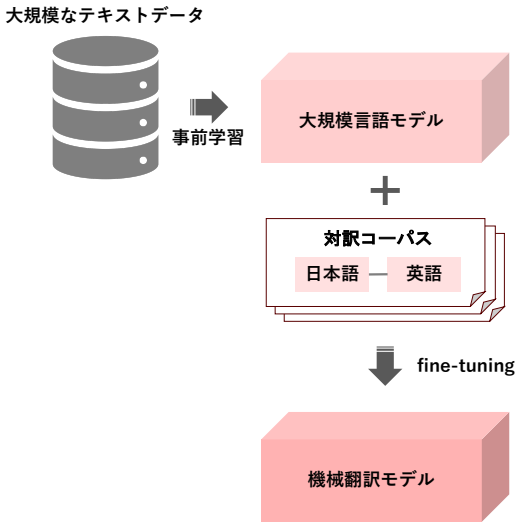

図 1　大規模言語モデル

2　大規模言語モデル

本節では，大規模言語モデルについて述べる．

大規模言語モデルとは，大規模なテキストデータを事前に学習しているモデルである．大規模言語モデルを，目的に合わせたデータで fine-tuning することで，よりタスクに特化したモデルを構築することができる．fine-tuning とは，事前学習した時のパラメータをそのまま利用し，下流タスクの訓練データでパラメータを微調整する学習方法である．下流タスクの例として，機械翻訳，質問応答，感情分析などが挙げられる．大規模言語モデルは大量のテキストデータを学習している為，fine-tuning に用いるデータが少なくても，精度の高いモデルを構築することが可能である．代表的な大規模言語モデルとして T5 [1] や BERT [2]，GPT-3 [3] が挙げられる．

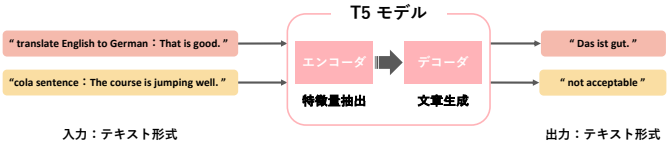

図 2　T5

2.1　T5

代表的な大規模言語モデルの 1 つである T5 について紹介する．

T5 とは Text-To-Text Transfer Transformer の略で，2020 年に発表された大規模言語モデルである．モデル構造は Transformer [4] をベースとしたエンコーダ・デコーダ型である．英語のデータセットである Colossal Clean Crawled Corpus(C4)

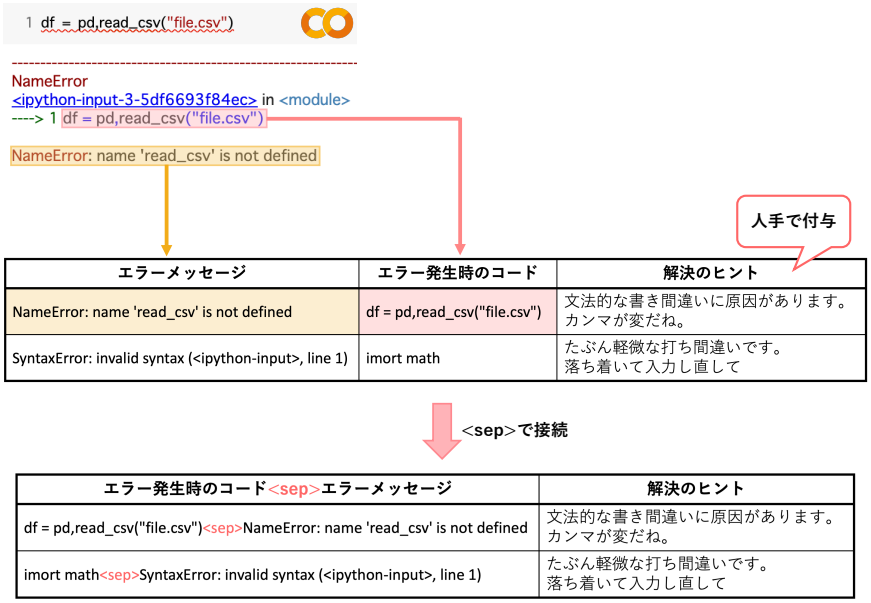

図 3　エラーコーパスの作成方法

表 1　エラーコーパスの例

入力	出力
sin(1)<sep>NameError: name 'sin' is not defined	たぶんインポートし忘れています.
prit(a,c)<sep>NameError: name 'prit' is not defined	たぶん，ミススペルです.
for i in range(2,N)<sep>SyntaxError: invalid syntax (, line 9)	構文エラー，つまり文法的に書き方が間違っています．コロンが必要だよ
X = df[' 通勤時間']<sep>KeyError: ' 通勤時間'	指定されたカラム名がデータフレームにありません.

を事前に学習している．入出力をテキスト形式で統一させることで，1 つのモデルで翻訳，分類，回帰，要約などのさまざまなタスクの対応を可能にした．

3　提　　案

本節では，エラーコーパスの作成方法と，エラー解決のヒントを分かりやすく示す「エラー診断モデル」の構築方法を提案する．

3.1　エラーコーパス

エラーコーパスとは，エラーの情報から原因や解決のヒントを提示するための対コーパスである．

3.1.1　データ収集

コーパス作成のために，2022 年度日本女子大学理学部数物情報科学科で開講されたプログラミング演習（2 年次向け：受講者 90 名），データサイエンス演習（3 年次向け: 履修者 70 名）の演習時間のプログラムとその実行結果について，発生した全てのエラーをログとして記録した．収集したデータ件数は 10879 件であった．

3.1.2　エラーコーパスの作成方法

まず，エラーコーパスに含ませる情報について考える．ここで，初学者に多い NameError を例に挙げる．NameError が発生する原因は，

- 変数が未定義
- 関数が未定義
- インポートし忘れ
- タイポ
- 文字列との勘違い
- 構文ミス，カンマの打ち間違い

と多岐にわたるが，メッセージは「`NameError: name '○○' is not defined`」とほぼ共通である．そのため，エラーメッセージだけでエラーの原因を突き止めるのは困難であると予想される．そこで，エラー発生時のコードを参照することで，エラーの原因や解決方法を推測することが可能になる．したがって，エラーコーパスには，エラーメッセージに加え，エラーの発生した行に相当するコードを加えることにした．

最終的に，エラーコーパスは，以下の手順で作成した．

1. 収集したエラーログからエラーメッセージとエラー発生時のコードを取り出し，前処理を行う．
   (前処理：改行→`<nl>`，tab →空白 2 つ)
2. ヒントを付与できるデータのみを取り出し，解決のヒン

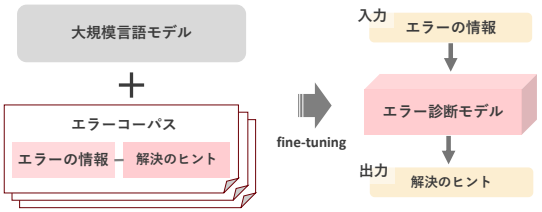

図 4　エラー診断モデルの構築方法

トを人手で付与する．

3. エラー発生時のコードとエラーメッセージを<sep> で接続する．

エラーコーパスのデータ件数は 5255 件であった．表 1 に示すように，エラーコーパスの入出力は以下の通りである．

入力：エラーメッセージ，エラー発生時のコード

出力：解決のヒント

また，エラー発生時のコードは Python，エラーメッセージは英語，解決のヒントは日本語であり，3 つの言語から成るのもエラーコーパスの特徴である．

3.2　モデル構築

エラー診断モデルは，図 4 のように，大規模言語モデルをエラーコーパスで fine-tuning することで構築する．構築したエラー診断モデルにエラーの情報を入力することで解決のヒントが出力される．

4　実　　験

本節では，実験について述べる．我々は，実際にエラーコーパスからエラー診断モデルを構築することで，精度の確認を行なった．

4.1　大規模言語モデル

エラー診断モデルの特徴は，Python/英語/日本語と複数の言語が含まれている点である．我々は，以下の 4 つの T5 モデルを用意した．

- **CodeT5** [5]: 英語と，Python を含む 8 つのプログラミング言語からなるデータセットを学習
- **日本語 T5**: 日本語のみのデータセットを学習
- **mT5** [6]: 日本語と英語を含む 101 言語からなるデータセットを学習
- **mPyT5** [7]: mT5 に python を追加学習

4.2　実 験 概 要

3.1.2 節で作成したエラーコーパスのデータ 5255 件を，訓練用データ 3678 件，検証用データ 789 件，テスト用データ 788 件に分割した．エラーコーパスに含まれるエラーの種類を表 3 に示す．

訓練用データで 4.1 節の 4 種類の大規模言語モデルを fine-tuning し，4 種類のエラー診断モデルを構築した．テスト用データを用いて構築した 4 種類のエラー診断モデルの精度を比較する．

表 2　各モデルが学習している言語

	Python	英語	日本語 T5
CodeT5	○	○	
日本語 T5			○
mT5		○	○
mPyT5	○	○	○

表 3　エラーコーパスのデータの件数

エラーの種類	全データ	訓練用	検証用	テスト用
NameError	1636	1156	240	240
SyntaxError	1118	780	168	170
TypeError	806	560	124	122
KeyError	507	352	87	68
AttributeError	475	338	70	67
IndexError	439	294	66	79
IndentationError	163	122	21	20
FileNotFoundError	111	76	13	22
合計	**5255**	**3678**	**789**	**788**

4.3　評 価 指 標

モデルの評価には以下の 4 つを用いる．

- **正答率** [8]: 正解文とモデルの予測文の完全一致率
- **BLEU** [9]: 正解文とモデルの予測文の N-gram に基づく類似度 (主に機械翻訳の評価に使用)
- **ROUGE-L** [10]: 最長共通部分列に基づいた正解文とモデルの予測文の構造的な類似度 (主にテキストの要約の評価に使用)
- **EditSim**: Levenshtein 距離 [11] に基づく正解文とモデルの予測文の文字列間の類似度

4.4　実 験 結 果

各モデルの予測例を表 4 に，予測精度を表 5 に示す．

どのモデルも正答率が 80%以上と高精度だが，mPyT5 を用いて構築したエラー診断モデルの精度が最も高くなった．エラーコーパスは，Python，英語，日本語の 3 つの言語で構成されている為，この 3 つの言語を事前に学習している mPyT5 を用いたモデルの精度が高くなったと考えられる．

5　関 連 研 究

本節では，関連研究について述べる．

表 4　各モデルの予測例

入力	エラー発生時のコード	ans_1 = (1+sqrt(1-4*1*(-2)*c))/2
	エラーメッセージ	NameError: name 'sqrt' is not defined
正解		たぶんインポートし忘れています.
CodeT5 の予測		文法的な書き間違いに原因があります．カンマが変だね.
日本語 T5 の予測		文法的な書き間違いに原因があります．カンマが変だね.
mT5 の予測		たぶんインポートし忘れています.
mPyT5 の予測		たぶんインポートし忘れています.

表 5　エラー診断モデルの予測精度

	正答率	BLEU	ROUGE-L	EditSim
CodeT5	81.73	88.59	85.96	90.97
日本語 T5	80.71	87.68	84.59	88.94
mT5	82.49	89.06	85.75	90.26
mPyT5	**84.65**	**89.51**	**87.11**	**91.58**

大規模言語モデルを用いた研究では，Codex [5] を使用してエラーメッセージの解説やコードの修正案の提案を行う研究が行われている [12]．具体的には，エラーを発生させるコードと処理の内容を入力し，それに対する出力を評価する研究である．エラーメッセージの解説については，入力の 84%に対して出力が得られたが，このうち正しいと判断されたのは 57%であった．これは全入力の 48%にあたる．また，コードの修正案の提案については，入力の 70%に対して出力が得られたが，このうち正しいと判断されたのは 47%であった．これは全入力の 33%にあたる．本研究の相違としては，コーパスの作成および学習を行っていない点が挙げられる．

6 むすびに

本論文では，プログラミング学習者のエラー解決を支援するために，エラー解決のヒントを提示するエラー診断モデルの構築方法を提案した．具体的には，まず，初学者のエラー情報に解決のヒントを付与することで，エラーコーパスを作成した．そして，作成したエラーコーパスで大規模言語モデルを fine-tuning することで，エラー診断モデルを構築した．

さらに，使用する大規模言語モデルを変え，4 種類のエラー診断モデルを構築し，精度の比較を行った．その結果，Python，英語，日本語の 3 つの言語を事前に学習している mPyT5 を用いて構築したエラー診断モデルの精度が一番高くなることが確認できた．

今後は，構築したエラー対応モデルを実際にプログラミング学習者に使ってもらい，評価を受けたい．

文　献

[1] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67, 2020.

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.

[3] Mann Benjamin Ryder Nick Subbiah Melanie Kaplan Jared Dhariwal Prafulla Neelakantan Arvind et al Brown, Tom B. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

[4] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA, 2017. Curran Associates Inc.

[5] Yue Wang, Weishi Wang, Shafiq Joty, and Steven C. H. Hoi. Codet5: Identifier-aware unified pre-trained encoder-decoder models for code understanding and generation, 2021.

[6] Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. mT5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498, Online, June 2021. Association for Computational Linguistics.

[7] Miyu Sato Mai Takahashi Kimio Kuramitsu Teruno Kajiura, Nao Souma. An additional approach to pre-trained code model with multilingual natural languages. In *29th Asia-Pacific Software Engineering Conference (APSEC 2022)*, 2022.

[8] Xinyun Chen, Chang Liu, and Dawn Song. Tree-to-tree neural networks for program translation. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 2552–2562, Red Hook, NY, USA, 2018. Curran Associates Inc.

[9] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics.

[10] Chin-Yew Lin and Franz Josef Och. Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 605–612, Barcelona, Spain, July 2004.

[11] Li Yujian and Liu Bo. A normalized levenshtein distance metric. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29:1091–1095, 2007.

[12] Sami Sarsa Brent Reeves Paul Denny James Prather Juho Leinonen, Arto Hellas and Brett A. Becker. Using large language models to enhance programming error messages. *arXiv:2210.11630*, 2022.