ラベル付きデータの自己生成による大規模言語モデルのファインチューニング

高森 勇佑　　白井 清昭　　Natthawut Kertkeidkachorn
北陸先端科学技術大学院大学 先端科学技術研究科
{y-takamori,kshirai,natt}@jaist.ac.jp

概要

本論文ではラベル付きデータなしで大規模言語モデル (LLM) を下流タスクにファインチューニングするアプローチを提案する．まず，zero-shot プロンプトを LLM に与えて下流タスクのラベル付きデータを新たに生成する．次に，生成サンプルの信頼度を評価し，それが低いサンプルを除去するフィルタリングを行う．最後に，生成したラベル付きデータセットを用いて LLM をファインチューニングする．含意関係認識，感情分析，法律ドメインの自然言語推定，テキスト生成の 4 つの下流タスクを対象に評価実験を行い，提案手法の有効性を示す．

1　はじめに

近年，大規模言語モデル (Large Language Model; LLM) は様々な自然言語処理タスクで卓越した性能を示している．一般に，言語モデルの使用は，大量のコーパスを用いた事前学習と，事前学習したモデルを特定の下流タスクに適応させるファインチューニングの 2 段階に分けられる．LLM はパラメタ数が膨大であるため下流タスクへのファインチューニングが難しかったが，LoRA(Low-Rank Adaptation)[1] を始めとする Parameter Efficient Fine-Tuning (PEFT) と呼ばれる手法が提案され，LLM の効率的なファインチューニングが可能になった．ファインチューニングは下流タスクに対する LLM の性能を向上させる有効な手段であるが，大量のラベル付きデータが必要であり，その構築のコストが高いという問題点もある．特に医療や金融など専門性の高いドメインについては，既存のデータセットが少ないことに加え，ラベル付きデータの構築に専門家が携わる必要があり，ファインチューニングに十分な量のラベル付きデータの確保が難しい．

本研究では，LLM に特定の下流タスクのラベル付きデータを生成させ，それを用いて LLM をファインチューニングするアプローチを提案する．LLM は事前学習によって様々な知識を包含すると考えられるが，下流タスクに関する知識をラベル付きデータという明示的なフォーマットで引き出し，その知識を用いて LLM を下流タスクに適応させることによって，LLM の性能を向上させることを狙う．

2　関連研究

近年，モデル自身が生成したデータを利用してラベル付きデータ収集の負担を軽減する研究が注目されている．Wang らは，LLM 自身が生成した命令・入出力ペアを用いて LLM の Instruction Fine-Tuning(IFT) を行う Self-Instruct と呼ばれる方法を提案している [2]．公開データセットを用いた実験によって，同手法で Instruction Tuning された LLM は従来手法に匹敵する性能を達成したと報告している．Yeo らは自己生成データを用いた self-training の枠組みを提案し，自己生成データを選択的に学習に用いることで LLM の性能が向上することを示している [3]．

一方，IFT の内部メカニズムを分析する研究も行われている．Ren らは，高品質な人手作成データを用いた IFT を対象に，IFT による性能向上が単純な知識の追加によるものではなく，モデルが事前学習で獲得した知識や表現を指示に対してより整合的に用いるよう調整されることが主因である可能性を示している [4]．Alajrami らは，データセットにノイズや構造的変形を加えた場合の IFT への影響を分析し，データの品質や命令の構造がモデルの汎化性能に大きく関与すると述べている [5]．

本研究は，先行研究 [2, 5] と同じように LLM が自己生成したデータを用いるが，Instruction Tuning ではなく下流タスクの性能を直接的に向上させることを目的に LLM の自己学習を行う．また，LLM が自

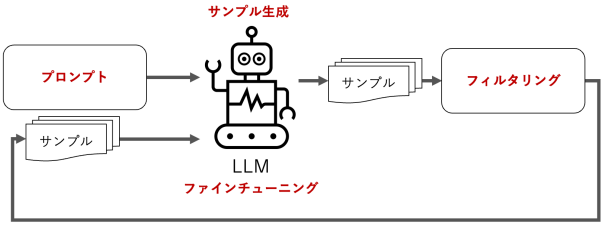

図 1: 提案手法の概要

己生成したデータのみを用いてファインチューニングを行う設定において，タスクの種類や生成データの性質が LLM のファインチューニングに与える影響を実験的に検証する．

3 提案手法

3.1 概要

ある下流タスクに LLM を適応させるとき，その下流タスクのラベル付きデータが全く存在しない，あるいはごく少数の事例のみ存在するという条件下で LLM をファインチューニングする．提案手法の概要を図 1 に示す．本手法は，(1)LLM によるラベル付きサンプルの生成，(2) 自己生成サンプルのフィルタリング，(3) 自己生成サンプルを用いた LLM のファインチューニング，の 3 段階で構成される．

3.2 下流タスク

本論文では，ケーススタディとして，以下の 4 つの下流タスクに対して提案手法を適用する．

RTE (Recognizing Textual Entailment; 含意関係認識) は，前提 (premise) と仮説 (hypothesis) の 2 つの文が与えられたとき，仮説が前提を含意するか否かを判定する分類タスクである．出力は entailment または non-entailment のいずれかである．

SA (Sentiment Analysis; 感情分析) は，ここでは与えられた文が肯定的か否定的かを判定する分類タスクと定義する．出力は positive または negative のいずれかである．

ContractNLI は，契約書を対象にした法律ドメインの自然言語推論 (Natural Language Inference; NLI) タスクである．前提として契約書，仮説として別の文が与えられたとき，両者の関係を entailment, contradiction, neutral のいずれかに分類する．ドメインに特化した分類タスクに対する提案手法の有効性を評価するために対象タスクとして選んだ．

E2E NLG (End-to-End Natural Language Generation) [6] は，意味表現 (Meaning Representation; MR) を入力とし，それを説明する自然言語文を生成するタスクである．MR は属性–値の集合として与えられ，ここではレストランに関する MR を与える．テキスト生成タスクに対する提案手法の有効性を評価するために対象タスクとして選んだ．

3.3 ラベル付きサンプルの生成

下流タスク毎に LLM を用いてラベル付きサンプルを生成する手法を述べる．サンプル生成の際に用いたプロンプトおよび生成したサンプルの例を付録 A に示す．

RTE タスクの定義とキーワードを与え，キーワードに関連し含意関係が成立する前提と仮説のペアを生成させる zero-shot のプロンプトを LLM に与える．プロンプトにキーワードを与えるのは多様なサンプルを生成するためである．Wikipedia のカテゴリ名[1)]をキーワードとし，Research, Library science など 727 個のカテゴリを抽出してキーワードリストを作成した．生成したサンプルには entailment のラベルを付与する．一方，non-entailment のラベルが付与されたペアは，LLM で生成したサンプルの前提と仮説をランダムに組み合わせて作成する．

SA 感情分析タスクの定義とキーワードを与え，そのキーワードに関連する文と極性ラベルの生成を指示するプロンプトを与える．感情分析タスクについてもプロンプトは zero-shot とする．キーワードとして映画のジャンル (Action, Historical など) とアスペクト (camera angle, function など) の組み合わせを与える．ジャンルとアスペクト数はそれぞれ 30 と 58 であり，$30 \times 58 = 1740$ 個のキーワードを与えてサンプルを生成する．

ContractNLI タスクの定義とキーワードを与え，契約書 (前提)，仮説，ラベルを生成するプロンプトを LLM に与える．プロンプトは one-shot とし，生成対象のラベルの例を 1 つ与える．キーワードとして RTE タスクと同じく Wikipedia のカテゴリ名を用いる．

E2E NLG MR が持つべきレストランの属性の定義を与え，MR と，それに含まれる全ての属性および値を反映したテキストを生成するよう LLM に指示する．また，キーワードとして都市名 (Tokyo, London など) を与え，その都市のレストランのサン

1) https://en.wikipedia.org/wiki/Wikipedia:Contents/Categories

プルを生成する．都市名はウェブ上の首都のリストなどから取得し，1,972 個の都市名リストを用意した．

3.4 生成サンプルのフィルタリング

LLM によって生成したサンプルの信頼度を評価し，それが低いものを除外するフィルタリングを行う．本研究では以下に述べる 3 種類のフィルタリング手法を用いる．

類似度フィルタリング RTE タスクにおいて，正解ラベルが entailment のとき，生成した前提と仮説の類似度が低いときは正しいサンプルではない可能性が高いとみなし，類似度が閾値より高いサンプルのみを残す．また，正解ラベルが non-entailment のときは類似度が閾値より低いサンプルを残す．同様に，ContractNLI タスクでは，ラベルが entailment, contradcition, neutral のとき，前提と仮説の類似が高いサンプル，低いサンプル，0.5 に近いサンプルを残す．閾値はラベル毎に 20%のサンプルを除外するように設定する．前提と仮説の類似度は，RTE タスクでは文を TF-IDF を重みとする単語ベクトルに，ContractNLI タスクでは文を Sentence Transformer [7] を用いて埋め込みに変換し，両者のコサイン類似度によって求める．なお，このフィルタリングは SA タスクと E2E NLG タスクには適用しない．

生成確率フィルタリング 自己生成したサンプルのテキストの生成確率が低いとき，そのテキストは不自然であり，適切なサンプルではない可能性が高い．そこで，式 (1) に示すように，テキスト s に含まれる次単語の予測確率の平均を信頼度のスコアと定義する．

$$Score(s) = \frac{1}{N}\sum_{i=1}^{N} P_{\mathrm{LLM}}(w_i|\mathrm{context}) \qquad (1)$$

ここで単語の予測確率 P_{LLM} はサンプル生成に用いた LLM を用いて推定する．このスコアが閾値 T_p 以上のサンプルを残し，残りのサンプルは除外する．T_p は，およそ 1 割から 3 割のサンプルを除外するように，RTE では 0.85，SA では 0.7，ContractNLI では 0.7，E2E NLG では 0.85 と設定した．

LLM 評価フィルタリング サンプルを生成した LLM 自身にそのサンプルの信頼性を評価させる．生成テキストの流暢さや自然さではなく，入力と出力の関係がタスクの要件を満たしているかどうかに着目して評価する．例えば，E2E NLG タスクについては，生成文が入力 MR に含まれる属性–値を過不足なく含むかを LLM に評価させる．評価値は 1〜5 の 5 段階とし，評価値が 2 以下のサンプルを除外する．

3.5 ファインチューニング

LoRA[1] を用いて，フィルタリング後のラベル付きデータを用いて LLM をファインチューニングする．

4 評価実験

4.1 実験条件

提案手法の評価実験のため，対象下流タスクの公開ベンチマークデータセットを使用する．RTE と感情分析タスクについては GLUE ベンチマークの RTE[8]，SST-2[9] データセット，ContractNLI タスクについては ContractNLI データセット [10]，E2E NLG タスクについては E2E Challenge Dataset[11] をそれぞれ用いる．各データセットのテストデータのみ使用し，訓練・開発データは使用しない．データセットの統計を表 1 に示す．

表 1: 評価データセットの統計

RTE[8]	SST-2[9]	ContractNLI[10]	E2E NLG[12]
3,000	1,821	1,991	1,847

LLM として Llama-3.2-3B-Instruct [13] を使用する．3.5 項で述べたように，ファインチューニングには LoRA[1] を用いる．学習パラメタなどの詳細は付録 B で述べる．

評価指標として，RTE および SA タスクでは正解率 (Acc.) を，ContractNLI タスクでは正解率とマクロ F1 スコア (macro F1) を用いる．生成タスクである E2E NLG タスクでは，Corpus BLEU，ROUGE-L F1，METEOR，および BERTScore F1 を評価指標とする．

ファインチューニングをせず事前学習済み LLM をそのまま用いる手法をベースラインとし，自己生成したサンプルを用いてファインチューニングする提案手法と比較する．また，フィルタリングを行わない場合，ならびに 3 種類のフィルタリング手法を適用した場合を比較する．

4.2 結果と考察

自己生成したサンプル数，およびフィルタリング後のサンプル数を表 2 に示す．LLM 評価フィルタリングでは削除されるサンプルの数がタスクによっ

て大きく異なる．SA では多くのサンプルが削除される一方，E2E NLG ではほとんどのサンプルが残されている．SA では入力と出力の対応関係が比較的明確でサンプルの妥当性が判断しやすいが，E2E NLG のような生成タスクでは入力に対する妥当な出力は多様であり，LLM による妥当性判定が難しかった可能性がある．

表 2: 生成したサンプル数

	サンプル数	フィルタリング後		
		類似度	生成確率	LLM 評価
RTE	1,454	1,162	1,242	1,277
SA	3,480	—	3,108	2,243
ContractNLI	1,434	1,145	1,032	1,422
E2E NLG	1,972	—	1,965	1,958

RTE，SA，ContractNLI タスクの実験結果を表 3 に，E2E NLG タスクの実験結果を表 4 に示す．ContractNLI では正解率とマクロ F1 とで手法の優劣に大きな乖離が見られる．これはテストデータにおけるラベル分布の偏りが大きいことが原因であり，評価指標としてはマクロ F1 の方が適切である．E2E NLG タスクについては，BLEU や ROUGE-L は全体的にスコアが低いが，BERTScore は高い値を示している．これは，LLM が生成した文が参照文と表層的には一致しない場合でも，意味内容は類似していることを示唆する．

いずれのタスクにおいても，自己生成サンプルを用いたファインチューニングによって評価指標が向上したことが確認できる．特に SA タスクではフィルタリングなしのファインチューニングでもベースラインと比べて正解率が 0.28 ポイント程度向上し，モデルの性能を大幅に向上させることができた．このことは，下流タスクに関する情報を LLM から引き出し，これを用いて LLM を下流タスクに適応させるアプローチの妥当性を示している．

生成サンプルのフィルタリングなし (FT wo. FIL) とフィルタリングありの手法を比較すると，全体的にはフィルタリングによってモデルの性能が改善する．LLM 評価によるフィルタリングは，SA タスクと E2E NLG タスクで最高の性能を示しており，特に SA タスクではフィルタリングなしの手法との差が大きい．LLM による信頼度評価によるフィルタリングは分類タスクでも生成タスクでも効果的であると言える．それに比べて，生成確率によるフィルタリングは，SA タスクではフィルタリングなしの手法よりも正解率が低下するなど，その効果は限定的である．同フィルタリングは主にサンプルテキストの流暢性を評価するが，タスクの定義に沿ったサンプルであるかといった観点の評価の方がファインチューニング用のデータのフィリタリングとして適切である可能性がある．類似度フィルタリングは全てのタスクに適用できるわけではないが，RTE タスクの正解率や ContractNLI のマクロ F1 では最高の数値が得られている．下流タスクに応じた適切なフィルタリング手法を発見し適用することが重要であると言える．

以上の結果から，自己生成データによるファインチューニングは分類タスク・生成タスクを問わず多くのタスクに有効に働くことが確認された．ただし，その効果はタスクに大きく依存する．入力-出力関係が比較的明確なタスク (SA，E2E NLG) では安定した性能向上が得られた一方，文間推論や専門知識を要するタスク (RTE, ContractNLI) ではモデルの性能改善は限定的であった．

表 3: 分類タスクの評価結果

	RTE	SA	ContractNLI	
	Acc.	Acc.	Acc.	macro-F1
Baseline	0.4693	0.5356	0.4385	0.2125
FT wo. FIL	0.5235	0.8234	0.3651	0.2422
FT w. FIL(sim)	**0.5632**	—	0.3134	**0.2734**
FT w. FIL(prob)	0.5307	0.6881	0.2983	0.2577
FT w. FIL(llm)	0.4908	**0.9071**	**0.4134**	0.2376

FIL(sim)，FIL(prob)，FIL(llm) はそれぞれ類似度フィルタリング，生成確率フィルタリング，LLM 評価によるフィルタリングを表す．「wo. FIL」はフィルタリングをしない手法を表す．

表 4: E2E NLG タスクの実験結果

	BLEU	ROU.	MET.	BERT.
Baseline	0.0458	0.2138	0.4045	0.8768
FT wo. FIL	0.0874	0.2949	0.4693	0.9116
FT w. FIL(prob)	0.0881	0.2914	0.4712	0.9116
FT w. FIL(llm)	**0.0910**	**0.2990**	**0.4743**	**0.9126**

BLUE = Corpus BLUE, ROU. = ROUGE-L, MET. = METOR, BERT. = BERTScore F1

5 おわりに

本論文の貢献を以下にまとめる．(1)LLM からラベル付きデータという形式で下流タスクの知識を引き出し，それを元に LLM をファインチューニングすることで，下流タスクに対する LLM の性能を向上させる手法を提案した．(2) 分類タスク・生成タスク，ドメインに特化したタスクを含む 4 つの下流タスクにおいて提案手法を評価し，その有効性を実験的に確認した．今後の課題として，様々な LLM を用いた検証実験や，生成サンプルの信頼度評価の精緻化などが挙げられる．

参考文献

[1] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. arXiv preprint arXiv:2106.09685, 2021. ICLR 2022.

[2] Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-instruct: Aligning language models with self-generated instructions. In **Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)**, pp. 13484–13508. Association for Computational Linguistics, 2023.

[3] Wei Jie Yeo, Teddy Ferdinan, Przemyslaw Kazienko, Ranjan Satapathy, and Erik Cambria. Self-training large language models through knowledge detection. In **Findings of the Association for Computational Linguistics: EMNLP 2024**, pp. 15148–15162. Association for Computational Linguistics, 2024.

[4] Mengjie Ren, Boxi Cao, Hongyu Lin, Cao Liu, Xianpei Han, Ke Zeng, Guanglu Wan, Xunliang Cai, and Le Sun. Learning or self-aligning? Rethinking instruction fine-tuning. In **Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)**, pp. 6090–6105. Association for Computational Linguistics, 2024.

[5] Ahmed Alajrami, Xingwei Tan, and Nikolaos Aletras. Fine-tuning on noisy instructions: Effects on generalization and performance. In **Proceedings of the 2025 Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics (AACL 2025)**. Association for Computational Linguistics, 2025.

[6] Ondrej Dusek, Jekaterina Novikova, and Verena Rieser. Evaluating the state-of-the-art of end-to-end natural language generation: The E2E NLG challenge. **Computer Speech & Language**, Vol. 59, pp. 123–156, January 2020. arXiv preprint arXiv:1901.07931.

[7] all-minilm-l6-v2. https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2 (accessed 2026-01-07).

[8] Recognizing textual entailment – GLUE benchmark. https://gluebenchmark.com/tasks (accessed 2025-12-30).

[9] Stanford sentiment treebank v2 (SST2). https://www.kaggle.com/datasets/atulanandjha/stanford-sentiment-treebank-v2-sst2 (accessed 2025-12-30).

[10] ContractNLI. https://stanfordnlp.github.io/contract-nli/ (accessed 2025-12-30).

[11] Ondřej Dušek, David M Howcroft, and Verena Rieser. Semantic noise matters for neural natural language generation. In **Proceedings of the 12th International Conference on Natural Language Generation (INLG 2019)**, pp. 421–426, Tokyo, Japan, 2019.

[12] The E2E challenge dataset. https://github.com/tuetschek/e2e-dataset (accessed 2025-12-30).

[13] AI@Meta Llama Team. The Llama 3 herd of models. In **Technical Report, Meta**, pp. 1–92, Menlo Park, USA, 2024.

A　プロンプト

RTE，SA，ContractNLI，E2E NLG タスクのサンプル生成に用いたプロンプト，ならびに生成されたサンプルの例を図 2, 3, 4, 5 にそれぞれ示す．

(プロンプト)
RTE task requires to recognize, given two text fragments, whether the meaning of one text is entailed (can be inferred) from the other text. Give 1 example of text1 containing the word '{{keyword}}' in this task. Text 1 and Text 2 must be at least 20 words and must be natural sentences.
text1:

(生成例: keyword="Research")
"New research suggests that the key to a sustainable future lies in reducing our carbon footprint." text2: "Studies have shown that reducing carbon footprint can lead to a more sustainable future."

図 2: プロンプトと生成サンプル例 (RTE)

(プロンプト)
SST2 task requires to classify the sentiment of a given text as positive or negative. Give 1 example of a text containing the word '{{keyword}}' with (positive|negative) sentiment. The text must be at least 20 words and must be a natural sentence.
text:

(生成例: keyword="Historical_function")
I was completely swept up in the cinematic experience of "Schindler's List" as a historical drama that masterfully humanizes the atrocities of the Holocaust.

図 3: プロンプトと生成サンプル例 (SA)

(プロンプト)
ContractNLI task requires to determine the relationship between a premise and a hypothesis. Give 1 example of premise containing the word '{{keyword}}' where the hypothesis is {{label}} to the premise. Premise and hypothesis must be at least 20 words and must be natural sentences. Format: premise: [text] hypothesis: [text] label: {{label}}.
Example:
premise: {{premise}}
hypothesis: {{hypothesis}}
label: {{label}}
Now generate a new example:

(生成例: keyword="Oceanography", label="1"(neutral))
premise: "Oceanography", as a science, is an interdisciplinary field of study that deals with the physical and biological aspects of the Earth's ocean and its interactions with the atmosphere and the sea floor.
hypothesis: Oceanography is a branch of geology.
label: 1

図 4: プロンプトと生成サンプル例 (ContractNLI)

B　ファインチューニングの詳細

LoRA のランクを 8，スケーリングファクターを 32，ドロップアウト率を 0.05 に設定した．

(プロンプト)
You are generating E2E NLG training data (Meaning Representation → Text).
Task
Given a Meaning Representation (MR) with restaurant/venue attributes, generate a natural text description.
Keyword to incorporate
"{{city}}"
MR Fields (use ALL of these fields):
- name: Restaurant/venue name (MUST include "{{city}}" in the name, e.g., "{{city}} Cafe", "The {{city}} Restaurant")
- eatType: One of [restaurant, coffee shop, pub]
- food: One of [Japanese, Chinese, English, French, Italian, Fast food, Indian]
- priceRange: One of [cheap, moderate, high, less than £20, £20-25, more than £30]
- customerRating: One of [1 out of 5, 3 out of 5, 5 out of 5, low, average, high]
〈省略〉
OUTPUT Requirements:
- Write exactly ONE paragraph (2-3 sentences, 30-50 words).
- Mention ALL fields from the MR naturally.
- Do NOT add any information not in the MR.
- Do NOT use bullet points or lists.
- Write fluent, natural English.
Output Format (STRICTLY follow this format):
{"name": "...", "eatType": "...", "food": "...", "priceRange": "...", "customerRating": "...", "area": "...", "familyFriendly": "...", "near": "..."}
text [Your natural text description here]
Generate exactly one example now. Output ONLY the json and text blocks, nothing else.

(生成例: city="Tokyo")
{""name"": ""Tokyo Sushi Bar"", ""eatType"": ""restaurant"", ""food"": ""Japanese"", ""priceRange"": ""high"", ""customerRating"": ""5 out of 5"", ""area"": ""city centre"", ""familyFriendly"": ""no"", ""near"": ""the train station""}
Tokyo Sushi Bar is a high-end restaurant located in the heart of the city centre, offering exquisite Japanese cuisine. With a 5-star rating, it is a popular spot for foodies who want to indulge in premium sushi. However, it's not family-friendly, so it's best suited for a night out with friends.

図 5: プロンプトと生成サンプル例 (E2E NLG)

更新対象の層は Query projection(q_proj) と Value projection(v_proj) の 2 層とした．学習パラメータは，学習率を 1×10^{-4}，エポック数を 50，バッチサイズを 8 に設定した．また，アーリーストッピングを適用し，監視指標を train_loss，忍耐度 (patience) を 10，最小改善量 (min_delta) を 0.001 に設定した．これらの設定は，RTE，SA，ContractNLI，E2E NLG の全てのタスクで共通である．