修士論文

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

主指導教員 白井 清昭

北陸先端科学技術大学院大学
先端科学技術研究科
（情報科学）

令和８年３月

Abstract

In recent years, large language models (LLMs) have demonstrated remarkable performance across a wide range of natural language processing tasks, including natural language understanding and generation. In general, the application of LLMs follows a two-stage procedure consisting of pre-training on large-scale corpora and subsequent fine-tuning to adapt the model to specific downstream tasks. Along with this progress, parameter-efficient fine-tuning methods have been proposed, enabling the adaptation of LLMs to downstream tasks even in environments with limited computational resources, thereby further facilitating the practical use of LLMs. Nevertheless, a fundamental challenge remains, as fine-tuning typically requires a large amount of labeled data. In supervised fine-tuning, each downstream task demands a labeled dataset of sufficient quantity and quality, the construction of which incurs substantial human and time costs. This issue is particularly pronounced in highly specialized domains such as medicine, law, and finance, where large-scale public datasets are sparse and annotation often requires expert involvement, making data collection itself a major bottleneck.

Against this backdrop, self-generated data-based approaches, in which LLMs generate training data for their own learning, have recently attracted increasing attention. Previous studies, including Self-Instruct, have shown that using pseudo-labeled data generated by the model itself can improve the performance of a downstream task without relying on human annotation. However, much of the existing work has primarily focused on the design of data generation processes, while comparatively little attention has been paid to how individual generated samples should be evaluated and selected after generation, or how different selection strategies influence the performance of downstream tasks. In particular, systematic investigations comparing the effects of data quality assessment and filtering methods for self-generated data across multiple downstream tasks remain limited.

The goal of this study is to clarify under what conditions fine-tuning with self-generated labeled data produced by LLMs is effective, as well as to identify its inherent limitations. In particular, we focus on the quality assessment and selection of generated samples after data generation, and systematically analyze how different filtering strategies, such as generation probability filtering and LLM-as-a-judge filtering, affect both the quality of training data and the downstream task performance of LLMs.

The proposed method consists of three stages: (1) generation of labeled samples, (2) filtering of self-generated samples, and (3) fine-tuning using the filtered samples. First, the LLM is prompted with a natural language description of the target downstream task, and generates pseudo-labeled samples that satisfy the the

task-specific requirement of a pair of an input and output. To ensure the diversity of generated samples, different keywords are provided in the prompts, instructing the model to generate samples related to each keyword.

Next, confidence-based filtering strategies are applied to remove samples containing incorrect labels, semantic inconsistencies, or redundant information. Depending on characteristics of the task, three types of filtering methods are employed: (1) similarity filtering, which evaluates the semantic correspondence between two texts within a sample, (2) generation probability filtering, which relies on the output's generation probability predicted by the LLM, and (3) LLM-as-a-judge filtering, in which the LLM itself assesses the validity of each sample with respect to the task definition. These methods evaluate sample quality from complementary perspectives, including semantic consistency, textual fluency, and relevance to the task.

Finally, the filtered self-generated data are used to fine-tune the LLM using LoRA. By adopting this parameter-efficient fine-tuning approach, our method enables effective adaptation to downstream tasks under limited computational resources. The proposed framework is applicable to a wide range of downstream tasks, including both classification and generation tasks.

To evaluate the effectiveness of the proposed method, experiments on multiple downstream tasks, including both classification and generation tasks, were conducted. Specifically, three classification tasks were considered, namely Recognizing Textual Entailment, Sentiment Analysis, and Natural Language Inference in a legal-domain. In addition, a text generation from structured data, called End-to-End Natural Language Generation (E2E NLG), was considered as a generation task. A zero-shot inference with a pre-trained LLM was used as the baseline, and compared with inference using LLMs fine-tuned on self-generated labeled data, i.e., our proposed methods. For the latter, we evaluated fine-tuning without filtering and fine-tuning with three filtering strategies: similarity filtering, generation probability filtering, and LLM-as-a-judge filtering. Across all tasks, the same LLM (Llama-3) was used consistently for sample generation, quality evaluation, and fine-tuning.

Experimental results showed that fine-tuning with self-generated data consistently improved the performance over the baseline across all tasks. Moreover, in many cases, applying filtering to self-generated samples further enhanced the performance of the downstream tasks. For example, in the Sentiment Analysis task, the accuracy of the pre-trained model was 0.54. The accuracy was increased to 0.82 by fine-tuning the LLM using the self-generated labeled dataset without filtering, and further improved to 0.91 when LLM-as-a-judge filtering was applied. For the generation task, the proposed method achieved an improvement of 0.036

in BERTScore F1 over the baseline.

A comparison of filtering strategies revealed that LLM-as-a-judge filtering was the most effective for tasks with relatively explicit input-output correspondences, such as Sentiment Analysis and E2E NLG. In contrast, for tasks requiring sentence-level reasoning or domain-specific knowledge, including Recognizing Textual Entailment and legal-domain NLI, similarity-based filtering yielded better performance, while the effectiveness of LLM-as-a-judge filtering and generation probability filtering was limited. These results indicated that the quality of self-generated data has a substantial impact on the performance of downstream tasks, and that the optimal filtering strategy is highly dependent on the characteristics of the task.

概要

近年，大規模言語モデル（Large Language Models; LLMs）は自然言語理解や生成を含む多様な自然言語処理タスクにおいて高い性能を示している．一般に，LLMを利用する際の手続きは，大規模コーパスを用いた事前学習と，特定の下流タスクへの適応を目的としたファインチューニングの二段階から構成される．

近年では，計算資源が比較的乏しい環境でも LLM の下流タスクへの適応を可能とするパラメータ効率的なファインチューニング手法が提案され，LLM の利用を更に促進する要因となっている．しかしながら，ファインチューニングでは大量のラベル付きデータが必要という課題は依然として残されている．教師あり学習に基づくファインチューニングでは，下流タスクごとに十分な量と品質を備えたラベル付きデータが必要となるが，その構築には多大な人的・時間的コストが伴う．特に，医療，法律，金融といった専門性の高いドメインでは，大規模な公開データセットが限られているだけでなく，アノテーションに専門家の関与が不可欠であるため，ラベル付きデータの収集自体が大きな障壁となっている．

このような背景のもと，近年では LLM 自身にデータを生成させ，それを学習に用いる自己生成データに基づくアプローチが注目を集めている．Self-Instruct をはじめとする先行研究では，モデル自身が生成した疑似的なラベル付きデータを用いることで，人手アノテーションに依存せずに下流タスク性能を向上させることが示されている．一方で，これらの研究の多くは，合成データを「どのように生成するか」という生成プロセスの設計に主眼を置いており，生成後に得られた個々のサンプルをどのような基準で評価・選別すべきか，またその選別手法の違いが下流タスク性能にどのような影響を与えるかについては，体系的な検証が十分に行われていない．特に，複数の異なるタスクを対象に，自己生成データの品質評価やフィルタリング手法が LLM のファインチューニングに与える効果を比較した研究は限られている．

本研究では，LLM が自己生成したラベル付きデータを用いたファインチューニングが，どのような条件下で有効に機能し，またどのような限界を持つのかを明らかにすることを目的とする．特に，生成後のサンプルに対する品質評価ならびにその選別に着目し，生成確率による品質評価や LLM による品質評価など，異なるフィルタリング手法が学習データの品質および下流タスクに対する LLM の性能に与える影響を体系的に分析する．

本研究の提案手法は，(1) 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) フィルタリング後のサンプルを用いたファインチューニング，の三段階から構成される．まず，対象とする下流タスクの定義を自然言語で記述したプロンプトを LLM に与え，入力と出力の対応関係がタスクの条件を満たす疑似的なラベル付きサンプルを生成する．この際，異なるキーワードをプロンプトに与え，そのキーワードに関連したサンプルの生成を LLM に指示することで，サンプルの多様性を確保する．

次に，生成されたサンプルに含まれるラベルの誤り，意味的不整合，あるいは

冗長なデータを除去するため，自己生成サンプルの信頼度に基づくフィルタリングを行う．本研究では，タスク特性に応じて三種類のフィルタリング手法を用いる．具体的には，サンプル内の 2 つのテキストの意味的対応関係を評価する「類似度フィルタリング」,「出力テキストの生成確率に基づく生成確率フィルタリング」，および LLM 自身にタスクの定義に沿っているかという観点からサンプルの妥当性を評価させる「LLM 評価フィルタリング」である．これらは，意味的一貫性，テキストの自然さ，タスク適合性といった異なる観点からサンプルの品質を評価する．

最後に，フィルタリング後の自己生成データを用いて，LoRA による LLM のファインチューニングを行う．パラメータ効率的な LoRA を適用することで限られた計算資源下でも LLM の下流タスクへの適応を可能にする．なお，本手法は分類タスクおよび生成タスクを含む多様な下流タスクに適用可能である．

提案手法の有効性を検証するため，分類タスクおよび生成タスクを含む複数の下流タスクに対して評価実験を行った．具体的には，含意関係認識，感情分析，法律ドメインの自然言語推論の 3 つの分類タスクと，構造化データからのテキスト生成タスク (End-to-End Natural Language Generation; E2E NLG) を評価対象の下流タスクとした．事前学習済み LLM を用いた zero-shot 推論をベースラインとし，自己生成データを用いてファインチューニングした LLM による zero-shot 推論と比較した．後者については，フィルタリングなしの場合および三種類のフィルタリング手法（類似度フィルタリング，生成確率フィルタリング，LLM 評価フィルタリング）を比較した．すべてのタスクにおいて，同一の LLM (Llama-3) を用いてサンプル生成・評価・学習を行った．

実験結果から，自己生成データを用いたファインチューニングは，すべてのタスクにおいてベースラインの性能を向上させることが確認された．また，多くのタスクにおいて，自己生成サンプルのフィルタリングは LLM の性能を更に向上させた．例えば，感情分析タスクにおいては，事前学習のみの LLM の正解率が 0.54 であるのに対し，フィルタリングなしのファインチューニングでは 0.82，LLM 評価フィルタリングを適用したときは 0.91 となり，正解率が大幅に向上した．生成タスクについては，提案手法により BERTScore F1 が 0.036 ポイント向上した．フィルタリング手法を比較すると，感情分析タスクや E2E NLG タスクのように，入力と出力の対応関係が比較的明確なタスクでは，LLM 評価に基づくフィルタリングが最も高い性能を示した．一方で，含意関係認識や法律ドメインの自然言語推論タスクでは，類似度に基づくフィルタリングが有効であり，LLM 評価や生成確率に基づく手法の効果は限定的であった．これらの結果から，自己生成データの品質が下流タスク性能に与える影響は大きく，また最適なフィルタリング手法はタスクの性質に依存することが明らかとなった．

目 次

第 1 章　はじめに **1**
1.1 背景 1
1.2 目的 2
1.3 本論文の構成 2

第 2 章　関連研究 **3**
2.1 インストラクションチューニングと合成データ生成 3
2.2 自己教師あり学習・自己チューニング 7
2.3 合成データのノイズの影響 8
2.4 本研究の特徴 10

第 3 章　提案手法 **11**
3.1 概要 11
3.2 下流タスク 12
3.2.1 含意関係認識 13
3.2.2 感情分析 13
3.2.3 契約文書を対象とした自然言語推論 13
3.2.4 構造化データからのテキスト生成 14
3.3 ラベル付きサンプルの生成 15
3.3.1 RTE タスクのサンプル生成 15
3.3.2 SA タスクのサンプル生成 15
3.3.3 ContractNLI タスクのサンプル生成 17
3.3.4 E2E NLG タスクのサンプル生成 18
3.4 生成サンプルのフィルタリング 18
3.4.1 類似度フィルタリング 18
3.4.2 生成確率フィルタリング 20
3.4.3 LLM 評価フィルタリング 21
3.5 ファインチューニング 24
3.6 下流タスクへの適用 24

第 4 章　評価 **26**
4.1 実験設定 26
4.2 含意関係認識タスク 29

4.3 感情分析タスク 32
4.4 法律ドメインの自然言語推定タスク 34
4.5 テキスト生成タスク 39
4.6 タスク間比較に基づく考察 42

第 5 章 おわりに **44**
5.1 本研究のまとめ 44
5.2 今後の課題 45

付 録 A 参考データ **50**
A.1 RTE・ContractNLI タスクのキーワード 50
A.2 SA タスクのキーワード 50
A.3 E2E NLG タスクのキーワード一覧 50

図 目 次

3.1 提案手法の概要 12
3.2 プロンプトと生成サンプル例 (RTE) 16
3.3 プロンプトと生成サンプル例 (SA) 16
3.4 プロンプトと生成サンプル例 (ContractNLI) 17
3.5 プロンプトと生成サンプル例 (E2E NLG) 19
3.6 信頼性評価プロンプト 23
3.7 RTE タスクのプロンプト 24
3.8 SA タスクのプロンプト 24
3.9 ContractNLI タスクのプロンプト 25
3.10 E2E NLG タスクのプロンプト 25

4.1 類似度フィルタリングによる信頼度スコアの分布 (RTE タスク) . . 30
4.2 生成確率フィルタリングにおける信頼度スコアの分布 (RTE タスク) 30
4.3 LLM 評価フィルタリングにおける信頼度スコアの分布 (RTE タスク) 31
4.4 生成確率フィルタリングにおける信頼度スコアの分布 (SA タスク) . 33
4.5 LLM 評価フィルタリングにおける信頼度スコアの分布 (SA タスク) 33
4.6 類似度フィルタリングによる信頼度スコアの分布 (ContractNLI タスク，entailment クラス) 36
4.7 類似度フィルタリングによる信頼度スコアの分布 (ContractNLI タスク，contradiction クラス) 36
4.8 類似度フィルタリングによる信頼度スコアの分布 (ContractNLI タスク，neutral クラス) 37
4.9 生成確率フィルタリングにおける信頼度スコアの分布 (ContractNLI タスク) 37
4.10 LLM 評価フィルタリングにおける信頼度スコアの分布 (ContractNLI タスク) 38
4.11 生成確率フィルタリングにおける信頼度スコアの分布 (E2E NLG タスク) 40
4.12 LLM 評価フィルタリングにおける信頼度スコアの分布 (E2E NLG タスク) 40

表 目 次

4.1 評価データセットのサンプル数 27
4.2 評価データセットのラベル分布 27
4.3 RTE タスクにおけるフィルタリング前後のサンプル数 29
4.4 RTE タスクにおける評価結果 29
4.5 SA タスクにおけるフィルタリング前後のサンプル数 32
4.6 感情分析（SA）タスクにおける評価結果 32
4.7 ContractNLI タスクにおけるフィルタリング前後のサンプル数 . . . 35
4.8 ContractNLI タスクの評価結果 35
4.9 E2E NLG タスクにおけるフィルタリング前後のサンプル数 39
4.10 E2E NLG タスクの評価結果 40
4.11 生成したサンプル数 42
4.12 分類タスクにおける性能比較 43
4.13 E2E NLG タスクにおける性能比較 (再掲) 43

A.1 RTE・ContractNLI タスクで使用したキーワードの一部 50
A.2 SA タスクで使用した映画ジャンル 50
A.3 SA タスクで使用した映画のアスペクト 51
A.4 E2E NLG タスクで使用した都市名の一部 51

第1章　はじめに

1.1　背景

近年，大規模言語モデル（Large Language Models; LLMs）は，自然言語理解や生成を含む多様な自然言語処理タスクにおいて卓越した性能を示している．一般に，言語モデルの利用は，大量のテキストコーパスを用いた事前学習と，事前学習済み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二段階から構成される．事前学習によって獲得された汎用的な言語知識を，下流タスクに適した形で活用するためには，ファインチューニングが不可欠であると広く認識されている．

一方で，近年の LLM はパラメータ数が非常に大きく，従来のフルパラメータ更新によるファインチューニングは計算資源の観点から実用的でない場合が多い．この問題に対して，LoRA（Low-Rank Adaptation）[11] をはじめとする Parameter Efficient Fine-Tuning（PEFT）と呼ばれる手法が提案され，モデル全体を更新することなく，下流タスクへの効率的な適応が可能となった [17, 31]．PEFT により計算資源やメモリ使用量の制約は大きく緩和され，LLM を用いた下流タスク適応の実用性は大きく向上している．

しかしながら，計算効率の問題が改善された一方で，ファインチューニングにおけるラベル付きデータへの依存という本質的な課題は依然として残されている．一般に，ファインチューニングは教師あり学習として行われるため，下流タスクごとに十分な量の高品質なラベル付きデータが必要となる．このようなデータの構築には，多大な時間的・人的コストが伴い，実運用上の大きな障壁となっている [12]．特に，医療や金融，法律といった専門性の高いドメインにおいては，公開されている大規模なラベル付きデータセットが限られているだけでなく，アノテーション作業に専門家の関与が不可欠である．そのため，十分な量のラベル付きデータを用意すること自体が困難であり，結果として LLM の性能を十分に引き出せない状況が生じている．LLM を下流タスクに適応させるためのラベル付きデータをいかに確保するかは，依然として重要な研究課題である．

1.2 目的

本研究では，ラベル付きデータの構築コストが高いという課題に対処するため，LLM 自身に下流タスクのラベル付きデータを生成させ，その自己生成データを用いて LLM をファインチューニングするアプローチに着目する．LLM は事前学習によって多様な言語知識や推論能力を内包していると考えられるが，これらの知識は必ずしも下流タスクに直接適用可能な形で利用されているわけではない．

本研究では，LLM が事前学習によって獲得している知識を，下流タスクに対応した「入力–出力ペア」という明示的な形式で引き出し，それを学習データとして再利用するという枠組みを採用する．すなわち，LLM 自身に疑似的なラベル付きデータを生成させることで，人手によるアノテーションなしに LLM を下流タスクに適応させることを目指す．

本研究の目的は，このような自己生成データに基づくファインチューニングが，どのような条件下で有効に機能し，またどのような限界を持つのかを明らかにすることである．具体的には，複数の自然言語処理タスクを対象として，自己生成データを用いたファインチューニングによる性能変化を評価し，タスクの性質や生成データの特性が学習効果に与える影響を考察する．

1.3 本論文の構成

本論文の構成は以下の通りである．2 章では，インストラクションチューニングや自己生成データを用いた学習，ならびに自己教師あり学習に関する関連研究を整理し，既存研究の到達点と本研究の位置づけを明確にする．3 章では，ラベル付きデータが存在しない，あるいは極めて限定的な状況を想定し，自己生成データの生成・フィルタリング・学習からなる提案手法の枠組みを説明する．4 章では，複数の下流タスクを対象とした評価実験を通じて，フィルタリングをはじめとする提案手法の有効性について検証し，考察を行う．最後に，5 章では，本研究のまとめと今後の課題について述べる．

第2章　関連研究

本章では，本研究に関連する先行研究を整理し，自己生成データを用いた下流タスク適応に関する研究動向を概観する．近年，大規模言語モデル（LLM）においては，ラベル付きデータの収集コストを低減する手段として，モデル自身がデータを生成し学習に用いる手法が広く検討されている．2.1 節において，インストラクションチューニングおよび合成データ生成に関する代表的な研究を整理する．続いて 2.2 節では，自己教師あり学習および自己チューニングに基づく手法を紹介する．2.3 節では，合成データに含まれるノイズや品質劣化がモデル性能に与える影響について議論する．最後に，これらの先行研究を踏まえ，本研究の特徴と位置付けを明確にする．

2.1　インストラクションチューニングと合成データ生成

近年，ラベル付きデータのコストを低減しつつ大規模言語モデル（LLM）の下流タスク適応性能を高めるために，モデル自身がデータを生成して学習に用いる研究が活発に行われている．特に「インストラクションチューニング（Instruction Tuning）」においては，人手で収集された命令・入出力データに依存せずにモデル能力を引き出す手法が注目されている．この節では，代表的な研究として Self-Instruct を中心に，合成データ生成の一般的フレームワークや変種的アプローチについて整理する．

Self-Instruct　Self-Instruct は，自己生成された命令データを用いて大規模言語モデル（LLM）を指示に従うモデルへと学習させる代表的なフレームワークである [27]．従来のインストラクションチューニングでは，人手で作成された命令・応答ペアを用いることが一般的であったが，その収集には多大な人的コストを要し，命令の多様性にも制約があった．Self-Instruct はこの課題に対し，LLM 自身に命令（instruction），入力，出力を生成させることで，大規模かつ多様なインストラクションチューニングのためのデータセットを構築する点に特徴がある．

Self-Instruct における合成データ生成の過程は，以下のように抽象化して表現できる．まず，プロンプト集合 $\{p_i\}_{i=1}^{N}$ を入力として，LLM g_θ により命令および

対応する入出力ペアを生成し，合成データ集合 D_{syn} を構築する．

$$D_{\text{syn}} = \bigcup_{i=1}^{N} g_\theta(p_i). \tag{2.1}$$

ここで，g_θ は事前学習済みの LLM を表し，各プロンプト p_i に対して複数の命令・応答サンプルを生成する．

生成された合成データには，重複サンプルや不適切な形式のデータが含まれる可能性があるため，Self-Instruct では一定のフィルタリング処理を施す．この処理は一般に，サンプル (x, y) に対して品質スコア $s(x, y)$ を計算し，所定の閾値 τ を満たすもののみを採用する操作として表される．

$$D_{\text{filtered}} = \{(x, y) \mid s(x, y) \geq \tau\}. \tag{2.2}$$

ここで，$s(x, y)$ は形式的妥当性や重複度合いなどに基づく指標である．フィルタリングは主として低品質または冗長なサンプルの除去を目的としている．

Self-Instruct は，初期のシード命令から開始し，命令生成，インスタンス生成，データ選別を反復的に行うブートストラップ型の学習フレームワークであり，この一連の流れは LLM を用いた合成データ生成パイプラインの原型としても位置付けられる．実際に，Self-Instruct を GPT-3 に適用した実験では，合成データのみを用いた学習であっても，従来の手法と同程度の性能を持つモデルを学習できることが示されている．

一方で，Self-Instruct におけるフィルタリングは，主に重複除去や形式的な品質確認に焦点が当てられており，フィルタリング基準の違いが下流タスク性能にどのような影響を与えるかについては，体系的な検証は行われていない．特に，生成データの品質評価と下流タスクにおける性能向上との関係性は十分に明らかにされていない．

DataGen　DataGen は，LLM を用いて多様かつ高品質なテキストデータセットを自動生成するための統一的なフレームワークである [13]．合成データ生成においてしばしば課題となる，生成データの多様性，制御可能性，真実性，および品質の一貫性といった点に対して，包括的に対処することを目的としている．

DataGen では，属性ガイド付き生成モジュールやグループ単位でのチェック機構を導入することで，生成データの分布を制御しつつ多様性を拡張する．さらに，コードベースの検証手法や検索拡張生成（Retrieval-Augmented Generation; RAG）を組み合わせることで，生成されたラベルや記述内容の正確性および事実性を担保する設計が採用されている．これにより，ユーザが指定した制約や要件を反映した高品質な合成データを柔軟に生成できる点に特徴がある．

大規模な実験を通じて，各構成モジュールが生成データの品質向上に寄与することが示されており，生成データをベンチマーク構築やデータ拡張用途に活用で

きる可能性が報告されている．この点で DataGen は，Self-Instruct のような特定形式の指示データ生成に留まらず，タスク非依存な合成データ生成フレームワークとしての一般化を目指した研究と位置付けられる．

一方で，DataGen は主として「どのように高品質な合成データを設計・生成するか」という生成プロセスそのものに焦点を当てており，生成されたデータが具体的な下流タスクにおいてどの程度性能向上に寄与するか，あるいは生成後のデータ選別や品質評価の違いが下流タスクに対する LLM の性能にどのような影響を与えるかについて，タスク横断的な詳細分析は行われていない．

Response Tuning　Response Tuning は，従来の指示チューニングにおいて前提とされてきた instruction conditioning（命令条件付け）を明示的に用いず，モデルの応答分布そのものを学習対象とするアプローチである [7]．従来手法では，命令文と応答文の対応関係を条件付けとして与えることでモデルの出力挙動を制御することが一般的であった．これに対し Response Tuning では，命令条件を除去した状態で応答テキストのみを学習することで，応答空間の分布を直接的に調整する．

実験結果から，このような学習設定においても，命令条件付きで学習したモデルと同程度の応答性能や有用性が得られる場合があることが示されており，事前学習済み LLM は明示的な命令条件がなくともある程度構造化された応答分布を内在的に獲得している可能性が示唆されている．この点で Response Tuning は instruction–response ペアを必ずしも必要としないファインチューニングの方向性を提示する研究として位置付けられる．

一方で，Response Tuning の論文では主に応答分布の学習可能性や表現能力の観点から議論が行われており，自己生成された個々のサンプルの品質をどのように評価・選別すべきか，あるいは生成データの品質差が特定の下流タスク性能にどのような影響を与えるかといった点については，体系的な検討はなされていない．

REInstruct　REInstruct は，大規模なラベルなしテキストコーパスを起点として，高品質な指示付きデータを自動的に構築する手法である [9]．本手法では，まずラベルなしテキストの中から，指示データとして有用であると考えられる文書をヒューリスティックによって選択する．次に，選択されたテキストに対して命令文および対応する応答文を生成する．さらに応答の再構成や洗練を行うことで，品質の高い instruction–response ペアを構築する．

実験では，ごく少量のシードデータと大量の合成データを組み合わせて学習したモデルが，オープンソースの評価ベンチマークにおいて，既存の指示チューニング手法と同程度の性能を示すことが報告されている．これにより，ラベルなしデータを効果的に活用することで，人手による大規模な指示データ収集を行わずとも，一定水準の下流タスク性能を達成できる可能性が示された．

REInstruct の特徴は，専用の LLM によって指示文と応答を自由に大量生成す

る手法とは異なり，ラベルなしのテキストデータそのものを入力として，そこに内在するタスク構造や暗黙的な指示を抽出・再構成する点にある．具体的には，既存の文書や文脈から，それに対応する instruction–input–output 形式のデータを生成することで，新たな知識を創出するのではなく，元テキストに基づいた指示データを構築する．このように，合成データの情報源を既存テキストに限定することで，外部に高品質な指示データが事前に存在しない場合や，利用可能なデータ源が特定ドメインのテキストに限られる環境においても，比較的信頼性の高い指示データを生成できる点が REInstruct の利点である．

一方で，REInstruct は主に合成データ構築プロセスそのものの設計に焦点を当てており，生成後のデータに対してどのような選別基準やフィルタリング手法を適用すべきか，またそれらの違いが下流タスク性能にどのような影響を与えるかについては，体系的な比較検証は行われていない．

まとめ　ここまで紹介した研究は，いずれもラベル付きデータの収集コストが高いという制約を緩和しつつ， LLM の汎化性能を向上させることを目的として，LLM 自身やラベルなしデータを起点としたデータ生成・学習手法を検討している．Self-Instruct は，自己生成データを用いた指示チューニングの先駆的手法として，合成データのみで高いゼロショット性能を達成可能であることを示した．DataGen は，この流れを発展させ，特定のタスクに限定されない合成データ生成の統一的設計原則を提示することで，タスク横断的なデータ生成の可能性を示している．また，Response Tuning や REInstruct は，命令条件やデータ生成の起点を再考することで，指示付きデータに依存しない学習やラベルなしデータの有効活用という観点から，合成データを用いた学習の新たな方向性を提示している．

しかし，これらの研究の多くは，合成データを「どのように生成するか」という生成手法の設計に主眼を置いており，生成後に得られた個々のサンプルをどのような基準で評価・選別すべきか，またその違いが下流タスク性能にどのような影響を与えるかについては，体系的な比較検証が十分に行われていない．実際，Self-Instruct や REInstruct においてもフィルタリングは重要な構成要素として導入されているものの，その設計は主に重複除去や形式的妥当性の確認に留まっており，生成確率や意味的一貫性など，異なる基準に基づくフィルタリング戦略の違いが，下流タスク性能に与える影響を詳細に分析する枠組みは示されていない．

本研究は，以上の点を踏まえ，合成データの生成方法そのものではなく，生成後のサンプル選別・フィルタリングという観点から，既存手法を補完する立場に位置づけられる．特に，複数の下流タスクに対して，生成確率や LLM による意味的評価などといった同一のフィルタリング手法を適用し，それらが学習データの品質及び下流タスクに対する LLM の性能に与える影響を明らかにする点に特徴がある，

2.2　自己教師あり学習・自己チューニング

近年，LLM の学習において，ラベル付きデータの不足や収集コストの高さといった制約を背景に，モデル自身が生成したデータや内部評価を学習データとして活用する自己教師あり学習および自己チューニング手法が注目を集めている．これらの手法は，外部知識から与えられる正解ラベルに依存せず，モデルが自ら生成した出力やその整合性・不確実性を手がかりとして学習データの選択や再学習を行う点に特徴がある．

本研究で扱う自己生成データに基づくファインチューニングも，このような自己チューニングの枠組みと密接に関連しているが，特に「生成後のサンプルをどのように評価・選別するか」という点に焦点を当てている点で，既存研究とは異なる観点を持つ．本節では，自己訓練・自己改善を扱う代表的な研究を概観し，それぞれがどの段階（生成，評価，選別，再学習）に主眼を置いているかを整理することで，本研究の位置づけを明確にする．

知識検出に基づく自己訓練　Yeo らは，LLM が自ら生成した予測結果に基づいて自己ラベル付けと選択的学習を行う自己訓練フレームワークを提案している [29]．本手法では，外部の参照モデルや人手ラベルを用いず，モデル出力間の内部整合性（reference-free consistency）を指標として信頼度の高いサンプルを抽出し，再学習に用いる．このような自己選別機構により，不確実な予測や誤りを含むサンプルの影響を抑制しつつ，生成結果の安定性や一貫性を向上させることができる．評価実験では，ハルシネーションの抑制や外部分布に対する性能低下の緩和が報告されており，ラベル付きデータが存在しない状況下における LLM による自己改善の有効性を示す具体例として位置づけられる．

一方で，この研究におけるサンプル選別は，主にモデル内部の整合性に基づく単一の評価基準に依存しており，異なる信頼度指標やフィルタリング戦略を比較した分析は行われていない．本研究は，この点を拡張し，自己生成サンプルに対する複数の選別基準を下流タスク横断で比較する点に新規性を持つ．

自己改善能力の限界分析　Song らは，LLM の自己改善能力を理論的・統計的観点から分析する研究である [25]．本研究では，モデルによる出力生成と，それに対する自己検証スコアとの乖離を「generation-verification gap」として定義し，このギャップが自己改善の成否に関係する可能性を示している．さらに，モデル規模や計算資源といった要因が，自己改善ループの安定性や収束特性にどのように影響するかを分析することで，自己チューニングが有効に機能する条件や限界についても議論している．このような分析は，自己生成・自己評価に基づく学習手法を設計する上で，理論的な指針を与える点で重要である．

一方，自己改善の可否や条件分析に主眼を置いており，生成された個々のサンプルをどのような基準で選別すべきか，またその違いが下流タスク性能にどの程

度影響するかについては，実験的な比較は行われていない．

外部教師なしの自律学習　Ji らは，LLM が外部教師なしで自律的に知識を獲得・更新できる可能性について報告している [14]．具体的には，モデルがテキストとの相互作用を通じて自身の知識ギャップを検出し，その補完を目的とした生成・再学習を行う枠組みを提案している．自己生成と自己評価を組み合わせた包括的な自己改善ループが提案されており，教師なし環境における自己学習の可能性を示す点で意義が大きい．一方で，生成された学習サンプルの品質管理や，サンプル選別戦略の違いが性能に与える影響については，詳細な分析は行われていない．

まとめ　本節で紹介した研究は，モデル自身が生成した情報や内部評価を用いて，ラベル付きデータに依存せずモデルの性能向上を図る自己教師あり学習・自己チューニングの枠組みを提示している．これらの研究は，自己生成，自己評価，再学習というループの有効性を示す点で共通しているが，その多くは自己改善の成立条件や全体設計に主眼を置いている．

一方で，自己生成されたサンプルをどのような基準で評価・選別するか，また選別手法の違いが下流タスクの性能にどの程度影響するかについては，タスク横断的な比較は十分に行われていない．本研究は，この点に着目し，自己チューニングの一構成要素としてのサンプルフィルタリング手法に特に着目し，複数の下流タスクにおいてその影響を体系的に分析することを目的とする．

2.3　合成データのノイズの影響

合成データを用いた学習は，ラベル付きデータの収集コストを削減しつつ，LLM の下流タスク性能を向上させる手法として広く用いられている．一方で，合成データには，誤ラベル，不自然な表現，分布の偏りといった様々なノイズが内在する可能性があり，それらがモデルの一般化性能や頑健性に与える影響については慎重な検討が必要である．

近年では，合成データに含まれるノイズなどの不具合が必ずしも一様に性能低下を引き起こすわけではなく，条件によっては汎化性能の向上や頑健性の改善に寄与し得ることも報告されている．本節では，合成データのノイズ，多様性，品質といった観点から，モデル学習への影響を分析した代表的な研究を概観する．これらの知見を通じて，本研究が着目する「自己生成サンプルの選別」が果たす役割を位置づける．

インストラクションチューニングにおけるノイズの影響評価　Alajrami らは，インストラクションチューニングに用いられるデータに意図的なノイズを付与し，その影響を体系的に評価している [6]．具体的には，ストップワードの削除や語順の

入れ替えといった表層的な乱れを含む命令データを用いてファインチューニングを行い，MMLU，BBH，GSM8K などの標準ベンチマークにおける性能変化を分析している．その結果，一定条件下では，ノイズを含む命令データで学習したモデルが，クリーンなデータのみを用いた場合と同等，あるいはそれ以上の性能を示すことが報告されている．この知見は，合成データに含まれるノイズが，必ずしも学習を阻害する要因ではなく，モデルの頑健性や分布外一般化を促進する可能性があることを示唆している．

一方で，ノイズが主に入力命令の表層的な変形に限定されており，生成サンプル全体の意味的妥当性やタスク適合性をどのように評価・制御すべきかについては，踏み込んだ議論は行われていない．本研究は，この点に着目し，自己生成データに含まれるノイズを生成後のフィルタリングによって制御するアプローチを採る．

合成データの多様性と一般化　Xiao らは，インストラクションチューニングにおける過学習の問題に対し，データ多様性を高めるための正則化手法である SFTMix を提案している [28]．この手法では，訓練データ中の異なるサンプルを線形補間する Mixup 操作を導入し，合成的な教師信号を用いてモデルを学習させる．Mixup 操作は，訓練データ分布を連続的に拡張する効果を持ち，高信頼度サンプルへの過度な適合を抑制しつつ，低信頼度サンプルの一般化性能を向上させる．Mixup 操作は次式で表される：

$$\tilde{x} = \lambda x_1 + (1 - \lambda) x_2 \tag{2.3}$$

ここで，λ は補間重みであり，データ分布の平滑化を通じてロバストな学習を促進する．

SFTMix は，合成データの多様性を意図的に設計することが下流タスクの性能向上に寄与することを示しているが，生成された個々のサンプルの品質を明示的に評価・選別する枠組みは想定していない．これに対し，本研究は，生成後のサンプルを選別する立場からデータ品質と性能の関係を分析する点で Xiao らの研究を補完するものである．

一般化インストラクションチューニング（GLAN）にみる多様性創出　Generalized Instruction Tuning（GLAN）は，体系化された知識構造を利用して，広範な分野にわたる命令データを生成する枠組みを提案した研究である [16]．GLAN は，シラバス設計や階層的な知識分解を通じて，特定の種例や既存データセットへの依存を抑えながら，多様かつ包括的な命令集合を構築することを目指している．このような多様性創出の設計は，モデルが幅広い下流タスクに対して安定した性能を示すための重要な要因であると考えられている．一方で，GLAN においても，生成された命令データの品質を下流タスク性能と関連付けて詳細に分析する試みは限定的である．

これに対し，本研究は，このような大規模・多様な合成データ生成を前提としつつ，生成後のサンプル選別という観点から，合成データの品質制御を実験的に

検討する点に特徴がある．

まとめ　本節では，合成データに含まれるノイズや多様性が，モデルの一般化性能および頑健性に与える影響を分析した研究を概観した．Alajrami らの研究は，ノイズが必ずしも性能低下を招くとは限らず，条件によっては汎化性能を向上させ得ることを示している．また，SFTMix や GLAN のような研究は，合成データの多様性を意図的に設計することが安定した学習に寄与する可能性を示唆している．

一方で，これらの研究の多くは，ノイズや多様性を「与える側」の設計に主眼を置いており，生成後のサンプルをどのような基準で選別・除去すべきか，またその違いが下流タスク性能にどの程度影響するかについては，体系的な比較は十分に行われていない．本研究は，この点に着目し，自己生成データに対するフィルタリング手法の違いを複数の下流タスクにおいて比較することで，合成データの品質と下流性能との関係を明らかにすることを目的とする．

2.4　本研究の特徴

以上の関連研究は，合成データや自己生成データを用いることで，ラベル付きデータへの依存を低減しつつ，LLM の性能向上を図る多様なアプローチを提示している．これらの研究では，合成データ生成手法の設計，自己改善ループの構築，あるいはノイズや多様性が学習に与える影響の分析など，それぞれ異なる観点から手法の有効性が検討されている．一方で，多くの研究は，性能向上そのものを主目的としており，生成されたサンプルをどのような基準で選別するか，またその選別方法の違いが下流タスク性能にどの程度影響するかについては，タスク横断的な比較が十分になされていない．特に，モデル自身が生成したデータのみを用いる設定において，フィルタリング条件とタスク特性との関係を体系的に分析した研究は限られている．

これに対し，本研究は，モデル自身が生成したラベル付きデータのみを用いる状況を想定し，生成・選別・学習の各段階を明示的に分離した枠組みの下で，自己生成サンプルのフィルタリング手法が下流タスク性能に与える影響を実験的に検証する．分類タスクおよび生成タスクを含む複数の下流タスクに対して評価を行うことで，合成データの品質と学習効果との関係を明らかにする．

第3章　提案手法

本研究では，LLM によって下流タスクのラベル付きデータを生成させ，その自己生成データを基に LLM をファインチューニングすることで，ラベル付きデータを用いずに LLM を下流タスクに適応させる手法を提案する．本章では，その提案手法の詳細を述べる．まず，3.1 節において提案手法の全体像を示す．3.2 節では評価に用いる下流タスクについて説明する．3.3 節では，ラベル付きサンプルの生成方法を述べる．3.4 節では，フィルタリング手法について詳述する．3.5 節では，得られた自己生成データを用いたファインチューニングの方法について述べる．3.6 節では，ファインチューニングを行ったモデルを下流タスクへ適用する方法について述べる．

3.1　概要

本手法は，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 が自己生成したデータを用いて LLM をファインチューニングする．従来の Instruction Tuning や自己学習（self-training）に基づく手法は，ある程度の人手で作成されたラベル付きデータや高品質な生成データの存在を前提としている場合が多く，ほぼ完全な教師なし学習の状況下における汎化性能・安定性の評価は十分ではない．これに対し，本手法はラベル付きデータの生成・選別ならびに自動構築したラベル付きデータを用いた LLM のファインチューニングを通じて，自己生成データの品質と下流タスクに対する LLM の性能との関係を分析する．

提案手法の概要を図 3.1 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．「LLM によるラベル付きサンプルの生成」では，タスクの定義に従って LLM に対してプロンプトを与え，入力–出力ペアを生成する．「自己生成サンプルのフィルタリング」では，生成確率や LLM による自己評価に基づく指標を用いて，タスクに対する妥当性が低いと判断されるサンプルを除外する．「自己生成サンプルを用いた LLM のファインチューニング」では，フィルタリング後の自己生成サンプルを用いて LLM をファインチューニングする．本手法は，分類タスクおよび生成タスクのいずれも対象としており，タスク定義が自然言語によって記述できる場合を想定している．ま

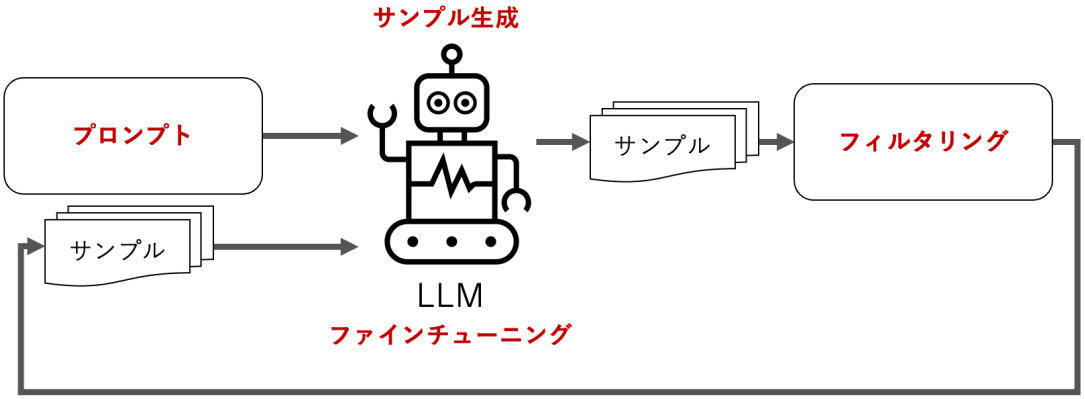

図 3.1: 提案手法の概要

た，LLM が事前学習によって当該タスクに関連する知識をある程度保持していることを前提とする．

3.2 下流タスク

自然言語処理における下流タスクは，出力形式の観点から，入力文に対して離散的なラベルを予測する分類タスクと，入力に基づいて自由形式のテキストを生成する生成タスクに大別される．また，対象とするデータの性質に応じて，ニュース記事やレビューなどの一般ドメインを対象としたタスクと，法律や医療など専門知識を要する特定ドメインを対象としたタスクに分類することもできる．LLM を下流タスクへ適応させる手法の有効性は，これらの異なるタスク形式やドメインにおいて一様に成立するとは限らない．特に，自己生成データを用いる場合，その有効性はタスクの性質や出力構造，およびドメイン特有の制約に強く依存する可能性がある．

そこで本研究では，提案手法が幅広い下流タスクに適用可能であるかを検証するため，タスク形式，推論の有無，およびドメインの異なる複数の下流タスクを対象とした評価を行う．具体的には，一般ドメインの分類タスクとして含意関係認識（Recognizing Textual Entailment; RTE）および感情分析（Sentiment Analysis; SA），専門ドメインにおける分類タスクとして契約文書を対象とした自然言語推論（Contract Natural Language Inference; ContractNLI）を対象とする．これらに加え，生成タスクとして構造化データからテキストを生成する End-to-End Natural Language Generation（E2E NLG） を対象とする．これら 4 つのタスクは，分類タスクと生成タスクの双方を含むとともに，一般ドメインおよび専門ドメインを網羅しており，提案手法の有効性を検証するにあたり，様々な下流タスクの中での代表的なケーススタディであると位置付けられる．以下では，各下流タスクの設定および特徴について順に説明する．

3.2.1 含意関係認識

含意関係認識（Recognizing Textual Entailment; RTE）は，前提（premise）と仮説（hypothesis）からなる文対が与えられたとき，仮説が前提から論理的に導かれるか否かを判定する分類タスクである．

本タスクは，RTE チャレンジ [10] を起源とし，その後 GLUE ベンチマーク [26] にも採用されている．本研究では，出力ラベルが *entailment* または *non-entailment* である二値分類として扱う．例えば，前提が「*A man is playing a guitar.*」，仮説が「*A person is performing music.*」である場合，仮説は前提から自然に導かれるため，正解ラベルは *entailment* となる．一方，前提が「*A man is playing a guitar.*」，仮説が「*A man is cooking dinner.*」である場合，前提からは導かれないため，*non-entailment* と判定される．

RTE タスクは，文間の意味関係を正確に捉える推論能力を必要とするため，モデルの意味理解および論理的一貫性を評価する下流タスクとして広く用いられている．

3.2.2 感情分析

感情分析（Sentiment Analysis; SA）は，与えられたテキストが表す感情的極性を判定する分類タスクであり，自然言語処理における代表的な文分類問題の一つである．本研究では，文が肯定的な感情を表すか否かを判定する二値分類設定を採用する．例えば,「*This movie was absolutely wonderful.*」という文は肯定的な評価を含むため *positive* と分類される．一方で,「*The plot was boring and predictable.*」という文は否定的な感情を表しており，*negative* と判定される．

感情分析は，Pang らによって映画レビューを対象とした極性分類タスクとして体系化され [22]，その後，多様なデータセットおよび設定で研究が進められてきた．本研究で用いる設定は，Stanford Sentiment Treebank に基づく SST-2 タスク [24] に対応する．SST-2 は GLUE ベンチマーク [26] にも採用されている．

感情分析タスクは，入力文と出力ラベルとの対応関係が比較的明確であり，2 つの文間の関係に関する推論を必要としない一方で，語彙的・文脈的な感情表現を正確に捉える能力が求められる．そのため，自己生成データを用いた学習が安定して機能するかどうかを検証する基礎的な下流タスクとして適している．

3.2.3 契約文書を対象とした自然言語推論

ContractNLI は，契約文書を対象とした専門ドメインの自然言語推論（Natural Language Inference; NLI）タスクである．自然言語推論タスクは，一般に，前提（premise）と仮説（hypothesis）の組を入力とし，仮説が前提との間にどのような意味的関係を持つかを判定する分類タスクとして定式化される．典型的には，仮説が前

提から必然的に導かれる場合を *entailment*，前提と矛盾する場合を *contradiction*，いずれにも該当しない場合を *neutral* とする三値分類が用いられる．ContractNLI は，この自然言語推論の枠組みを契約書文書に特化して適用したタスクであり，入力として契約文書の条文（premise）と，契約内容に関する仮説（hypothesis）が与えられる．モデルは，仮説が契約文書の内容から必然的に導かれるか，契約文書の記述と矛盾するか，あるいは文書中に十分な情報が存在しないかを判定し，それぞれ *entailment*，*contradiction*，*neutral* のいずれかに分類する．

ContractNLI データセット [15] は，Koreeda と Manning によって提案されたデータセットであり，実際の契約書に含まれる条項を対象として，法的な条件，義務，権利，例外規定などに関する推論を要求する．この点で，一般ドメインの自然言語推論タスクとは異なり，専門的な単語や契約特有の表現に加えて，文書全体の構造や前提条件を踏まえた解釈が必要となる．例えば，契約書中に「*The service provider may terminate the agreement with 30 days notice.*」という条項が含まれている場合，「*The service provider can immediately terminate the contract at any time.*」という仮説は，契約文書に明示された通知期間の条件と矛盾するため，*contradiction* と判定される．このように，ContractNLI では，明示的に記述された条件や制約を正確に捉え，それらを満たすか否かを判断する能力が求められる．

契約文書は一般に文書長が長く，専門的な用語や定型表現，複雑な条件構造を含む．そのため，入力テキストと出力ラベルの対応関係が必ずしも明確ではなく，自己生成データに基づく学習においては，生成されたラベルの妥当性や一貫性が性能に大きく影響すると考えられる．このタスクは，自己生成データを用いたファインチューニングが専門ドメインにおいてどの程度機能するのかを検証したり，その限界を検証したりすることの重要なケーススタディとなる．言い換えれば，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 ContractNLI を対象タスクとして選択する．

3.2.4 構造化データからのテキスト生成

End-to-End Natural Language Generation（E2E NLG）は，構造化された意味表現（Meaning Representation; MR）を入力として，対応する自然言語文を生成する生成タスクである．入力は属性-–値の集合として与えられ，モデルは与えられた情報を過不足なく反映したテキストを出力することが求められる．本タスクは，分類タスクとは異なり，出力が自由形式のテキストである点に特徴がある．

Novikova らは E2E NLG Challenge と呼ばれるタスクを提案し，そのためにデータセットを構築・公開している [21]．このタスクでは主にレストラン情報を対象とした意味表現からの文生成を扱う．例えば，`name[The Eagle]`，`food[French]`，`area[city centre]` という意味表現が与えられた場合，“The Eagle is a French restaurant located in the city centre.” のような文を生成することが期待される．

本研究では，自己生成データを用いたファインチューニング手法が，分類タスクだけでなく，出力が自然言語文である生成タスクにおいても有効に機能するかを検証するため，E2E NLG を評価対象として採用する．特に，生成タスクにおいては，自己生成データの品質が出力文の流暢性や情報の正確性に直接影響するため，提案手法の汎用性および安定性を評価する上で重要なケーススタディとなる．

3.3 ラベル付きサンプルの生成

本節では，下流タスク毎に LLM 自身にラベル付きサンプルを生成させる手法について述べる．既に述べたように，本研究では，下流タスクに対応する人手ラベル付きデータが存在しない，あるいは極めて限られている状況を想定する．LLM に対してタスク定義と生成指示を与えることで，疑似的なラベル付きデータを自動生成する．

生成される各サンプルは，入力とそれに対応する正解出力（ラベルまたは生成文）から構成され，後続のフィルタリングおよびファインチューニングに用いられる．サンプル生成の基本方針は全タスクで共通であるが，タスクの形式（分類／生成）やドメインの違いに応じて，プロンプト設計および出力形式を定める．以下では，各下流タスクについて，具体的な生成例を示しながら説明する．

3.3.1 RTE タスクのサンプル生成

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．キーワードの一部を付録 A.1 に示す．

プロンプトと生成されたサンプルの例を図 3.2 に示す．{{keyword}} にはキーワードを 1 つ埋める．生成したサンプルには entailment のラベルを付与する．一方，non-entailment のラベルについては，LLM によって生成された前提と仮説をランダムに組み合わせることで作成する．

3.3.2 SA タスクのサンプル生成

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．

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

(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 3.2: プロンプトと生成サンプル例 (RTE)

キーワードとして映画のジャンル (Action, Historical など)[2] とアスペクト (camera angle, function など)[3] の組み合わせを与える．予備的な検討において，ジャンルのみ，あるいはアスペクトのみをキーワードとして与えた場合，生成されるサンプルの多様性および十分なサンプル数を確保することが困難であった．そこで，本研究ではこれら二つの要素を組み合わせたものをキーワードとして用いる．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．ジャンルとアスペクトの一覧を付録 A.2 に示す．

プロンプトと生成されたサンプルの例を図 3.3 に示す．この図に示した生成例で与えたキーワードは「Historical_function」であるが，Historical は映画のジャンル，function は映画のアスペクトである．

(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 3.3: プロンプトと生成サンプル例 (SA)

[2] https://en.wikipedia.org/wiki/List_of_genres#Film_and_television_genres
[3] https://nofilmschool.com/braveheart-theyll-never-take-our-freedom

3.3.3 ContractNLI タスクのサンプル生成

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[4]を用いる．ただし，RTE タスクで用いたキーワードリストは 727 個であったのに対し，ContractNLI タスクではそのうちランダムに選択した 478 個のキーワードを用いる．RTE は二値分類タスクであるのに対し，ContractNLI は分類クラス数が 3 であるマルチクラス分類タスクである (entailment, contradiction, neutral)．提案手法ではキーワードと分類クラスのそれぞれの組に対してサンプルを生成するが，ContractNLI タスクは RTE タスクよりも多くのサンプル生成の試行が必要である．しかし，実装した環境では LLM によるサンプル生成に要する計算時間が大きく，727 個のキーワード全てを使うとかなりの時間を要する．LLM のファインチューニングに必要最低限と思われるサンプル数を確保しつつ，実験を効率的に進めるため，使用するキーワード数を 478 個に減らした．

プロンプトと生成されたサンプルの例を図 3.4 に示す．

(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 3.4: プロンプトと生成サンプル例 (ContractNLI)

[4] https://en.wikipedia.org/wiki/Wikipedia:Contents/Categories

3.3.4 E2E NLG タスクのサンプル生成

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プルを生成する．都市名は，国連加盟国 193 か国の首都リスト[5]，各国の主要都市（人口上位都市，経済・文化的重要都市）[6]，ISO 3166-1 国コード標準[7]，および国土地理院の世界の首都一覧[8]から取得し，1,972 個の都市名リストを用意した．都市名の一部を付録 A.3 に示す．プロンプトと生成されたサンプルの例を図 3.5 に示す．

3.4 生成サンプルのフィルタリング

LLM によって自己生成されたサンプルには，ラベルの誤りや意味的な不整合，さらには内容が類似した冗長なサンプルが含まれる可能性がある．これらの低品質なサンプルをそのまま学習に用いた場合，ファインチューニング後のモデルの性能低下を引き起こす恐れがある．そこで本研究では，自己生成サンプルの信頼度を評価し，信頼度が低いと判断されるサンプルを除外するフィルタリングを行う．

本研究における信頼度とは，生成された入力–出力ペアが当該タスクの定義に照らして妥当である可能性を指す．正解ラベルが存在しない条件下では，この信頼度を直接評価することは困難であるため，文埋め込みといった外部知識や LLM の内部情報などを用いて近似的に評価する．

具体的には，(1) 入力が 2 つの文であるタスクにおけるサンプル自体の品質の評価，(2) 出力の生成確率によるラベルや出力テキストの評価，(3) 生成したモデル自身による入力–出力ペアの評価，3 種類のフィルタリング手法を用いる．

以下，これら 3 種類のフィルタリング手法について，それぞれの評価指標および適用方法を順に説明する．

3.4.1 類似度フィルタリング

RTE タスクや NLI タスクでは，前提と仮説の意味的関係そのものがラベル定義となっているため，両者の意味的類似度は生成サンプルの信頼度を評価する重要な指標となる．

RTE タスクにおいて，正解ラベルが *entailment* の場合，前提と仮説は意味的に強く関連している必要がある．したがって，生成された前提–仮説ペアの類似度が著しく低い場合，含意関係が成立していない可能性が高いと判断し，類似度が

[5] https://www.un.org/en/about-us/member-states
[6] https://worldpopulationreview.com/world-cities
[7] https://www.iso.org/iso-3166-country-codes.html
[8] https://www.gsi.go.jp/KOKUJYOHO/SEKAI/sekai_syuto.htm

(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]
- area: One of [city centre, riverside]
- familyFriendly: One of [yes, no]
- near: A nearby landmark (e.g., "Burger King", "the train station", "the city park")
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 3.5: プロンプトと生成サンプル例 (E2E NLG)

閾値 α_1 以下のサンプルを除外する．一方，*non-entailment* の場合には，前提と仮説が意味的に独立していることが多いため，類似度が閾値 α_2 以上のサンプルを除外する．

一方， ContractNLI タスクでは，ラベルが *entailment*, *contradiction*,*neutral* の三値で定義されている．このタスクでは，前提と仮説が，*entailment* では高い意味的類似度，*contradiction* では低い類似度，*neutral* では中程度の類似度を示す傾向があると考えられる．そこで，*entailment* については類似度が閾値 β_1 以下のサンプルを，*contradiction* については類似度が閾値 β_2 以上のサンプルを，*neutral* については閾値が β_3 以下もしくは β_4 以上 (ただし $\beta_3 < \beta_4$) のサンプルを除外するフィルタリングを行う．上記のフィルタリングにおいて，閾値 α_i と β_i は，ラベル毎に全体の 20% のサンプルが除外されるように設定する．

前提と仮説の類似度は，それぞれをベクトル表現に変換し，2 つのベクトルのコサイン類似度で測る．文のベクトル表現については，タスクの特性に応じて異なる表現を用いる．RTE タスクでは，一般ドメインの比較的短い文を対象としており，単語の重なりが文間の意味的関係を十分に反映すると考えられるため，TF-IDF を重みとする単語ベクトルを採用する．一方，ContractNLI タスクでは，法律文書特有の長文かつ形式的な表現や言い換えが多く含まれるため，表層的な単語の一致では意味的関係を適切に捉えられない．そのため，Sentence Transformer [1] を用いて文を分散表現に変換する．

なお，この類似度フィルタリングは，前提—仮説の意味的関係が明確に定義されている RTE および ContractNLI にのみ適用し，SA タスクおよび E2E NLG タスクには適用しない．

3.4.2 生成確率フィルタリング

LLM によって自己生成されたサンプルにおいて，生成確率が低いテキストは，文法的あるいは意味的に不自然である場合や，プロンプトで与えたタスク定義から逸脱している場合が多い．したがって，LLM によって算出されるテキストの生成確率は，モデル自身がそのテキストをどの程度自然かつ妥当であると判断しているかを表す指標とみなすことができる．この性質に基づき，本研究では出力テキストの生成確率をサンプルの信頼度を評価する尺度として用いる．

具体的には，式 (3.1) に示すように，生成されたテキスト s に含まれる各単語の次単語予測確率の平均を信頼度スコアとして定義する．

$$Score(s) = \frac{1}{N} \sum_{i=1}^{N} P_{\mathrm{LLM}}(w_i \mid \mathrm{context}) \tag{3.1}$$

ここで，P_{LLM} はサンプル生成に用いた LLM 自身によって推定される次単語の予測確率であり，N はテキスト s に含まれる単語数を表す．単語確率の平均を用いることで，文長の違いによる影響を緩和している．

このスコアが閾値 T_p 以上のサンプルを保持し，それ未満のサンプルは除外する．閾値 T_p は，全体のおよそ 1 割から 3 割のサンプルが除外されるよう，タスク毎に経験的に設定した．具体的には，RTE では 0.85，SA では 0.7，ContractNLI では 0.7，E2E NLG では 0.85 とした．

この生成確率フィルタリングは，類似度フィルタリングとは異なり，タスク固有の制約に依存しないため，すべてのタスクに共通して適用可能である．

3.4.3 LLM 評価フィルタリング

自己生成されたサンプルの信頼度を評価するために，サンプルを生成した LLM 自身にそのサンプルの妥当性を評価させ，その信頼度が低いサンプルを除外する．この LLM による自己評価に基づくフィルタリングを「LLM 評価フィルタリング」と呼ぶ．本手法では，生成テキストの流暢さや自然さそのものではなく，生成サンプルにおける入力と出力の関係が下流タスクの要件を満たしているかどうかを LLM に評価させる．

LLM は，タスク定義や入出力間の意味的対応関係に関する知識を内部表現として保持しており，サンプル生成時と同一の知識分布および推論能力に基づいて，生成結果がタスクの制約を満たしているかを再評価できると仮定する．この仮定に基づき，LLM 自身による評価は，タスク適合性の観点からサンプルの信頼度を測る指標として機能すると考えられる．

LLM 評価フィルタは，タスクごとに評価観点を明示したプロンプトを与え，生成出力の妥当性を 5 段階（1–5）で評価させる．本研究では，分類ヘッドを持たない生成モデルを用いているため，評価値 $s \in \{1, 2, 3, 4, 5\}$ に対応する数値トークンの生成確率を評価に用いる．

各サンプル x に対して，出力の先頭の位置における数値トークン s の確率 $p(s \mid x)$ を算出し，最も高い生成確率を持つ数値を当該サンプルの評価値として採用する．この処理は以下のように定式化される．

$$\hat{s} = \arg \max_{s \in \{1,\ldots,5\}} p(s \mid x) \tag{3.2}$$

フィルタリング条件は 4 つの下流タスクすべてで共通とし，$\hat{s} \geq 3$ のサンプルのみを学習に使用し，$\hat{s} \leq 2$ のサンプルは除外する．

各タスクにおける信頼性評価用プロンプトを図 3.6 に示す．タスクの名称や，分類タスクと生成タスクとで評価基準の説明が異なるが，基本的な指示は 4 つのタスクで同じである．

サンプルの生成と評価を同一の LLM で行うことにより，生成時の表現空間や暗黙的な制約と整合した基準でサンプルを検査できる点が本手法の特徴である．また，生成確率フィルタリングが主に言語的自然さを評価するのに対し，LLM 評価フィルタリングは入出力関係の妥当性や制約充足を評価する点で補完的である．ま

た，下流タスク毎にその評価基準をプロンプトに与えることで，分類タスク・生成タスクのいずれにも共通して適用可能である．

Rate the quality of this RTE example on a scale of 1-5, where 1 is very poor and 5 is excellent.
Consider the clarity of the premise-hypothesis relationship, logical consistency, and overall quality.
Premise: "{{text1}}"
Hypothesis: "{{text2}}"
Label: {{label}}
Rating:

(a) RTE

Rate the quality of this SST2 example on a scale of 1-5, where 1 is very poor and 5 is excellent.
Consider the clarity of sentiment expression, naturalness of language, and overall quality.
Text: "{{text}}"
Label: {{label}}
Rating:

(b) SA

Rate the quality of this ContractNLI example on a scale of 1-5, where 1 is very poor and 5 is excellent.
Consider the clarity of the premise-hypothesis relationship, logical consistency, and overall quality.
Premise: "{{premise}}"
Hypothesis: "{{hypothesis}}"
Label: {{label}}
Rating:

(c) ContractNLI

Rate the quality of this E2E NLG example on a scale of 1-5, where 1 is very poor and 5 is excellent.
Consider whether the generated text accurately represents all the information in the meaning representation, the fluency and naturalness of the text, and overall quality.
Meaning Representation: {{mr}}
Generated Text: "{{text}}"
Rating:

(d) E2E NLG

図 3.6: 信頼性評価プロンプト

3.5 ファインチューニング

フィルタリング後の自己生成ラベル付きデータを用いて，LoRA（Low-Rank Adaptation）[11] によるファインチューニングを行う．LoRA は，事前学習済み LLM の全パラメータを更新することなく，重み行列を低ランク行列で近似し，そのパラメータのみを学習することで，計算効率と学習の安定性を両立する手法である．本研究では，複数の下流タスクに対して同一の学習設定で比較実験を行う必要があるため，学習に使用する計算機の性能がそれほど高くない環境下でも安定かつ再現性の高い学習が可能な LoRA を採用した．

3.6 下流タスクへの適用

LLM のファインチューニング後，それを用いて下流タスクを解く．その際には zero-shot のプロンプトを用いる．各タスクのプロンプトを図 3.7〜図 3.10 に示す．分類タスクにおいても，使用する LLM は分類器ではなく自己回帰型の生成モデルである．したがって，分類タスクにおいて，出力ラベルに対応する数値トークンの生成確率 $p(y \mid x)$ を用い，$\arg\max_y p(y \mid x)$ により最も高い生成確率を持つ数値を予測ラベルとして採用する．

> The purpose of the RTE task is identifying the relation between a premise and a hypothesis is ”entailment” or ”not entailment”. If the relation between the premise and the hypothesis is ”entailment”, the answer is 0. Else, if the relation between the premise and the hypothesis is ”not entailment”, the answer is 1. Now, the premise ”{text1}” and the hypothesis ”{text2}” are entered. Which is the answer, 0 or 1:

図 3.7: RTE タスクのプロンプト

> The purpose of the SST2 task is to classify the sentiment of a given text as positive or negative. If the sentiment is positive, the answer is 1. If the sentiment is negative, the answer is 0. Now, the text ”{text}” is entered. Which is the answer, 0 or 1:

図 3.8: SA タスクのプロンプト

The purpose of the ContractNLI task is to classify the relationship between a premise and a hypothesis as "entailment", "neutral", or "contradiction". If the premise entails the hypothesis, the answer is 0. If the premise is neutral to the hypothesis, the answer is 1. If the premise contradicts the hypothesis, the answer is 2. Now, the premise "{premise}" and the hypothesis "{hypothesis}" are entered. Which is the answer, 0, 1, or 2:

図 3.9: ContractNLI タスクのプロンプト

Generate a natural language description for the following restaurant/venue attributes.

Attributes:
{mr}

Description:

図 3.10: E2E NLG タスクのプロンプト

第4章　評価

本章では，生成した自己ラベル付きデータを用いた提案手法の有効性を，複数の下流タスクにおける実験を通して評価する．まず 4.1 節では，全タスクに共通する実験設定として，使用モデル，学習方法，評価指標，および比較手法を示す．続く 4.2 節から 4.5 節では，分類タスクおよび生成タスクを含む各下流タスクについて，個別の実験設定と結果を報告し，それぞれの観点から考察を行う．最後に 4.6 節では，各下流タスクの結果を踏まえ，フィルタリング手法の有効性について考察する．

4.1　実験設定

本節では，提案手法の評価に用いた実験設定について述べる．本研究で扱うすべての下流タスクに共通する設定として，使用モデル，ファインチューニング条件，評価指標および比較手法を示す．

使用モデル　既に述べたように，下流タスクを解くための基盤となる LLM として Llama-3.2-3B-Instruct [20] を使用する．本研究では，LLM が下流タスクに対する指示追従能力を有することを前提とするため，事前にインストラクションチューニングが施された Instruct 版モデルを採用する．本モデルは約 3B パラメータを有する．提案手法では，同じ LLM を下流タスクのサンプル生成と LLM 評価フィルタリングにおけるサンプルの信頼評価に用いる．

ファインチューニング設定　既に述べたように，本研究では，LoRA[11] を用いてファインチューニングを行う．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．更新対象の層は Transformer における Query projection（`q_proj`）および Value projection（`v_proj`）の 2 層とした．これらの層はアテンション機構の挙動に直接関与するため，下流タスクへの適応において効率的かつ安定した学習が可能であると考えられる．学習率は 1×10^{-4}，エポック数は 50，バッチサイズは 8 に設定した．また，自己生成データによる過学習を防ぐため，アーリーストッピングを適用した．監視指標には `train_loss` を用い，忍耐度（patience）を 10，最小改善量（min_delta）を 0.001 とした．これ

らのハイパーパラメータは，RTE，SA，ContractNLI，E2E NLG のすべての下流タスクにおいて共通に用いた．

評価データセット　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[4]，SST-2[5] データセット，ContractNLI タスクについては ContractNLI データセット [2]，E2E NLG タスクについては E2E Challenge Dataset [3] をそれぞれ用いる．本研究では，下流タスクのラベル付きデータが利用できない状況を想定し，訓練データおよびバリデーションデータは使用せず，テストデータのみを用いて評価を行う．各評価データセットのサンプル数を表 4.1 に示す．また，各評価データセットにおけるラベルごとのサンプル数とその割合を表 4.2 に示す．なお，E2E NLG は生成タスクであるため，分類ラベルは存在しない．

表 4.1: 評価データセットのサンプル数

Dataset	#Test Samples
RTE [4]	3,000
SST-2 [5]	1,821
ContractNLI [2]	1,991
E2E NLG [3]	1,847

表 4.2: 評価データセットのラベル分布

Task	Label	Count	Ratio
RTE	entailment	1500	50.0%
	not_entailment	1500	50.0%
SST2	positive	911	50.0%
	negative	910	50.0%
ContractNLI	Contradiction	903	45.4%
	Neutral	878	44.1%
	Entailment	210	10.5%

評価指標　評価指標は下流タスクに応じて設定する．分類タスクでは主に正解率を用い，ラベル分布に偏りが存在するタスク (具体的には ContractNLI タスク) については，各クラスを等しく評価するために macro F1 スコアを併用する．

生成タスクについては，自然言語生成の評価に一般的に用いられる複数の自動評価指標を用いる．本研究では，n-gram に基づく表層的一致度を評価する指標と，意味的類似性に基づく指標の両方を用いることで，生成されたサンプルの品質を多面的に評価する．

n-gram に基づく評価指標として，Corpus BLEU [23]，ROUGE-L F1 [18]，および METEOR [8] を用いる．BLEU は，生成文と参照文の n-gram の一致率に基づき，文集合全体における翻訳・生成品質を評価する指標である．また，ROUGE-L は，最長共通部分列（Longest Common Subsequence; LCS）に基づいて生成文と参照文の系列的な一致度を測定する指標である．METEOR は，n-gram 一致に加え，語形変化や同義語対応を考慮することで，BLEU よりも柔軟な表層一致評価を行う指標である．

一方，意味的類似性に基づく評価指標として，BERTScore F1 [30] を用いる．BERTScore は，事前学習済み言語モデルの文脈埋め込みを用いて，生成文と参照文のトークン間の意味的類似度を計算する指標であり，表層的な単語の一致に依存しない意味レベルでの評価を可能にする．本研究では，BERTScore の先行研究で広く用いられている設定に従い，RoBERTa-large [19] を事前学習済み言語モデルとして使用した．

生成タスクである E2E NLG タスクでは，同一の意味内容が複数の異なる表現で実現され得るため，n-gram に基づく指標のみでは生成品質を十分に評価できない場合があると考えられる．そのため，本研究では，表層的一致度を測る n-gram に基づく指標と，意味的妥当性を捉える意味類似度に基づく指標を併用することで，生成結果を包括的に評価する．

比較手法　実験では提案手法を含めて以下の手法を比較する．

- **Baseline (zero-shot)**: ファインチューニングを行わず，事前学習済み大規模言語モデルに zero-shot プロンプトを与えて下流タスクを解かせる設定．
- **FT w/o Filtering**: モデル自身が生成したラベル付きデータをそのまま用いてファインチューニングを行う手法．生成サンプルに対するフィルタリングは行わない．
- **FT w/ Similarity Filtering**: 生成したサンプルに対し類似度フィルタリングを適用した後，ファインチューニングを行う手法．
- **FT w/ Probability Filtering**: 生成確率フィルタリングに基づいて自己生成サンプルを選別し，ファインチューニングを行う手法．
- **FT w/ LLM-based Filtering**: LLM 評価フィルタリングを用いて自己生成サンプルを選別した後，ファインチューニングを行う手法．

ファインチューニングを行わず，事前学習済みモデルをそのまま用いる zero-shot 推論をベースラインとする．これに加え，自己生成データを用いてファインチューニングを行う提案手法について，フィルタリングを行わない場合と，類似度フィルタリング，生成確率フィルタリング，LLM 評価フィルタリングを適用した場合を比較する．

4.2 含意関係認識タスク

LLM によって生成された含意関係認識タスクのサンプル数および各フィルタリング手法適用後のサンプル数を表 4.3 に示す．表 4.3 より，適用するフィルタリング手法によって，最終的に保持されるサンプル数が異なることが分かる．また，類似度フィルタリングでは，他の手法と比べてより多くのサンプルが除外されている．これらの違いが，下流タスクの分類性能にどのような影響を与えるかについては，後述する評価結果に基づいて詳しく考察する．

表 4.3: RTE タスクにおけるフィルタリング前後のサンプル数

	#Samples
フィルタリング前	1,454
Similarity Filtering	1,162
Probability Filtering	1,242
LLM-based Filtering	1,277

表 4.4 に，含意関係認識 (RTE) におけるベースラインならびに提案手法の評価結果を示す．また，図 4.1 における自己生成サンプルの信頼度スコア（前提と仮説の類似度）の分布を，図 4.2 に生成確率フィルタリングにおける自己生成サンプルの信頼度スコア（生成確率）の分布を，図 4.3 に LLM 評価フィルタリングにおける信頼度スコア（LLM による評価スコア）の分布をそれぞれ示す．類似度によるフィルタリングでは，entailment のサンプルのみフィルタリングを行い，フィルタリング後のデータから負例サンプルを作成しているため，図 4.1 は entailiment のサンプルのみの分布表となっている．

表 4.4: RTE タスクにおける評価結果

手法	Accuracy
Baseline (zero-shot)	0.4693
FT w/o Filtering	0.5235
FT w/ Similarity Filtering	**0.5632**
FT w/ Probability Filtering	0.5307
FT w/ LLM-based Filtering	0.4908

表 4.4 から分かるように，ベースラインである zero-shot 推論と比較して，自己生成したラベル付きサンプルを用いてファインチューニングを行った場合，フィルタリングを行わない設定（FT w/o Filtering）でも正解率が 0.469 から 0.524 へと向上している．この結果は，下流タスクに関する知識を LLM 自身から引き出し，それを学習データとして再利用する本研究の基本的なアプローチが，RTE タスクに対しても一定の有効性を持つことを示している．

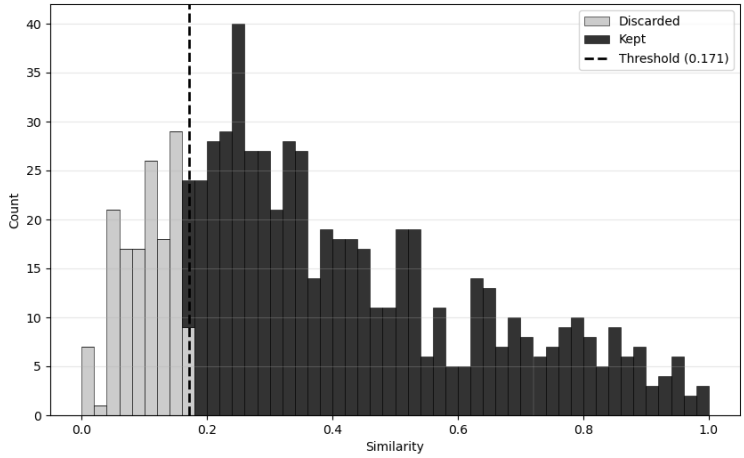

図 4.1: 類似度フィルタリングによる信頼度スコアの分布 (RTE タスク)

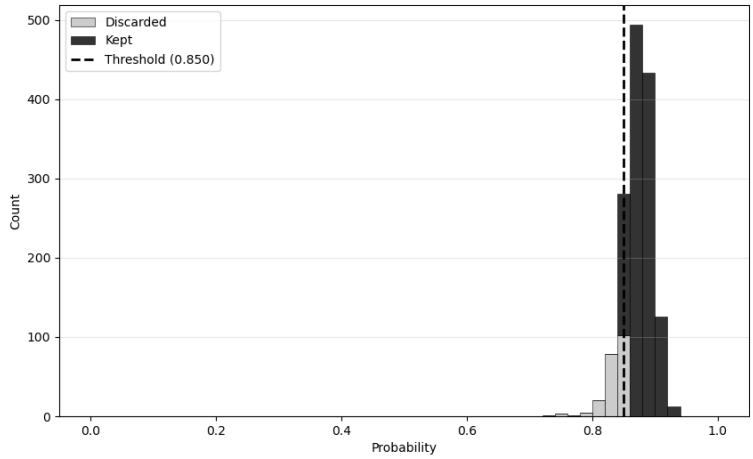

図 4.2: 生成確率フィルタリングにおける信頼度スコアの分布 (RTE タスク)

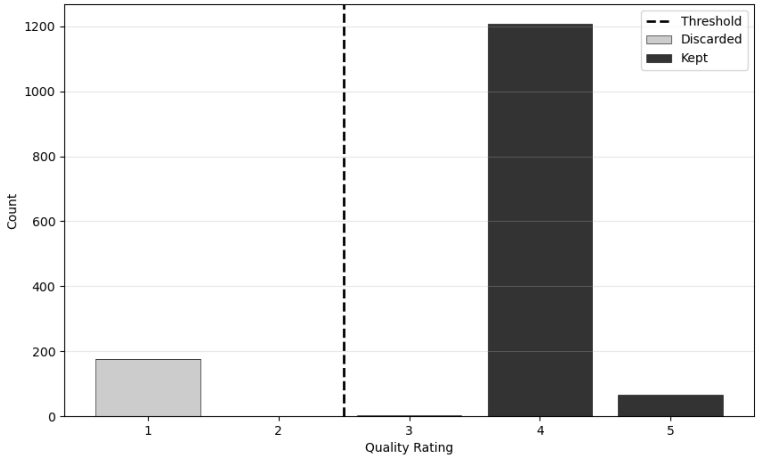

図 4.3: LLM 評価フィルタリングにおける信頼度スコアの分布 (RTE タスク)

フィルタリング手法の違いに着目すると，類似度フィルタリングを適用した場合に最も高い正解率（0.563）が得られた．RTE タスクでは，entailment ラベルが付与されたサンプルにおいて，前提と仮説の意味的整合性が重要であるため，両者の類似度が高いサンプルを選別することが，ラベル誤りや不自然な文対の除去に有効に働いたと考えられる．図 4.1 に示す類似度分布を見ると，類似度が 0 から 1 までのサンプルが比較的均等に存在することがわかる．2 割のサンプルを除外するというフィルタリングの設計にしたがい，閾値 0.171 以下のサンプルを除外している．このことから，類似度フィルタリングは，意味的対応が弱いサンプルを完全に分離するものではないものの，学習に有用でないサンプルを部分的に除去することで，結果としてモデルの分類性能が向上した可能性が示唆される．

一方，生成確率フィルタリングを用いた場合，フィルタリングなしの設定と比べて大きな性能向上は見られなかった．生成確率は主に文の流暢性やモデルの生成しやすさを反映する指標であり，文間の意味関係そのものを直接評価するものではないため，RTE タスクにおいては効果が限定的であった可能性がある．この点は，図 4.2 に示す生成確率分布からも確認できる．多くのサンプルが高確率領域に集中していることから，生成確率がサンプル間の質的差異を十分に反映していないことが示唆される．

また，LLM による自己評価を用いたフィルタリングでは，正解率がベースラインと同程度に留まり，他のフィルタリング手法と比べて有効性は確認できなかった．RTE では，微妙な含意関係や暗黙的な前提に基づく推論が求められるため，サンプル生成と同一のモデルによる自己評価では，ラベルの妥当性を十分に判別できなかった可能性がある．この傾向は，図 4.3 に示す LLM 評価スコア分布にも現れている．多くのサンプルが LLM によって信頼度が高い (4 以上) と評価されており，低評価サンプルがあまり存在しないことから，同一モデルによる自己評

価では誤りサンプルを効果的に識別できないことが示唆される．

以上より，RTE タスクにおいては，自己生成データを用いたファインチューニング自体は有効であるものの，その性能向上の度合いはフィルタリング手法に強く依存することが分かった．特に，タスクに特化した信頼度指標を用いたフィルタリングが重要であり，文間の意味関係が本質となる RTE では，類似度に基づくフィルタリングが最も適していた．

4.3 感情分析タスク

LLM によって生成された感情分析 (SA) タスクのサンプル数および各フィルタリング手法適用後のサンプル数を表 4.5 に示す．表 4.5 より，SA タスクにおいて，適用するフィルタリング手法によって最終的に保持されるサンプル数が大きく異なることが分かる．特に，LLM 評価に基づくフィルタリングでは，他の手法と比べて多くのサンプルが除外されている．次に，2 つのフィルタリング手法の違いがファインチューニング後の LLM の性能にもたらす影響について考察する．

表 4.5: SA タスクにおけるフィルタリング前後のサンプル数

	#Samples
フィルタリング前	3,480
Probability Filtering	3,108
LLM-based Filtering	2,243

表 4.6 に SA タスクにおけるベースラインならびに提案手法の評価結果を示す．また，図 4.4 に生成確率フィルタリングに使用した信頼度スコアである生成確率の分布を，図 4.5 に LLM 評価フィルタリングにおける信頼度スコア (LLM による評価スコア) の分布をそれぞれ示す．

表 4.6: 感情分析（SA）タスクにおける評価結果

手法	Accuracy
Baseline (zero-shot)	0.5356
FT w/o Filtering	0.8234
FT w/ Probability Filtering	0.6881
FT w/ LLM-based Filtering	**0.9071**

表 4.6 をみると，ベースラインである zero-shot 推論と比較して，自己生成したラベル付きサンプルを用いたファインチューニングは，フィルタリングを行わない場合でも正解率が 0.536 から 0.823 へと大きく向上している．この結果は，感情分析のように入力文と出力ラベルの対応関係が比較的明確なタスクにおいて，LLM

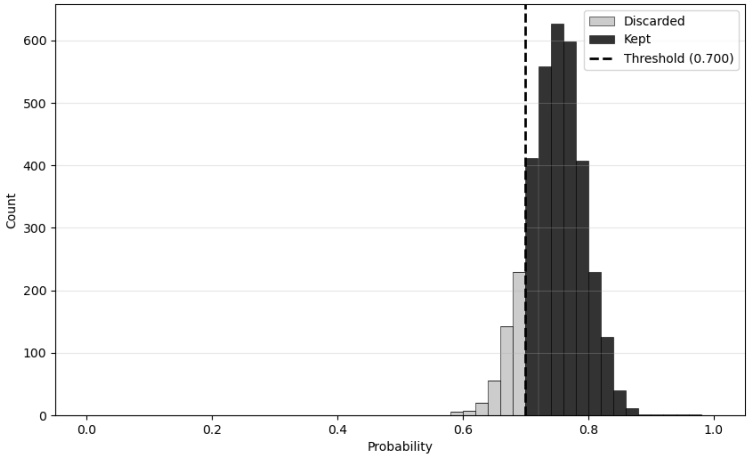

図 4.4: 生成確率フィルタリングにおける信頼度スコアの分布 (SA タスク)

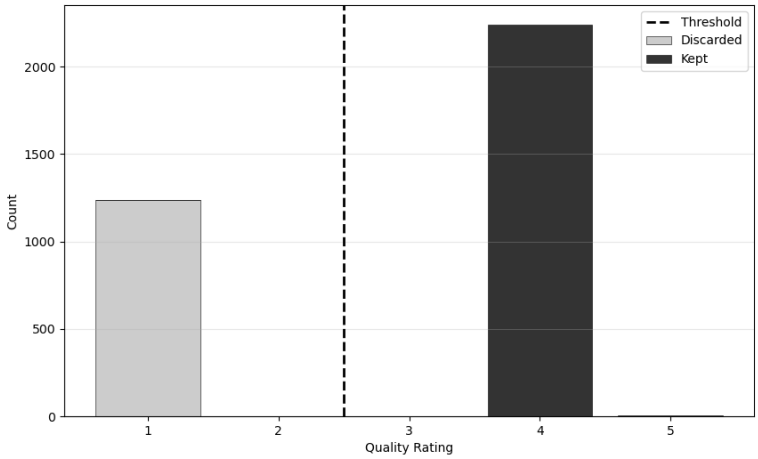

図 4.5: LLM 評価フィルタリングにおける信頼度スコアの分布 (SA タスク)

自身から生成した疑似ラベル付きデータを用いたファインチューニングが下流タスクの性能向上に大きく寄与することを示している．

フィルタリング手法の違いに注目すると，LLM 評価によるフィルタリングを適用した場合に最も高い正解率（0.907）が得られた．SA タスクでは，生成された文が特定の感情極性を適切に表現しているかどうかを判断することが重要であり，この判断は文の流暢性や表層的な特徴よりも，文全体の意味内容に基づく評価が有効である．そのため，タスク要件に即した観点からサンプルの妥当性を評価する LLM ベースのフィルタリングが特に効果的に機能したと考えられる．このことは，図 4.5 に示す評価スコア分布において，高評価と低評価のサンプルが明確に分離していることと整合しており，LLM による自己評価が SA タスクにおいて有効なサンプル選別指標として機能することを裏付けている．また，表 4.5 に示したように，LLM 評価フィルタリング後のサンプル数が他の条件よりも少ないことから，フィルタリングによるサンプル品質の向上が SA タスクに対する LLM の性能向上の主たる要因と考えられる．

一方，生成確率フィルタリングを用いた場合，フィルタリングを行わない設定と比べて正解率が低下した．生成確率はモデルが当該ラベルをどれだけ確信を持って生成しているかを表す指標であるが，LLM は zero shot でサンプルを生成しているため，必ずしも正しいラベルに対して高い確率が割り当てられるとは限らない．図 4.4 に示す生成確率分布を見ると，多くのサンプルが高確率領域に集中しており，サンプル間の質的差異を十分に捉えることができていないと考えられる．そのため，SA タスクにおいては，生成確率のような信頼度指標よりも，タスク適合性を直接評価する LLM 評価フィルタリング手法の方が有効である．

以上より，感情分析タスクにおいては，自己生成データを用いたファインチューニングが有効であり，特に LLM 自身によるタスク指向の評価を用いたフィルタリングが性能向上に大きく寄与することが分かった．この結果は，本研究で採用した自己生成・自己評価という枠組みが，入力と出力の対応関係が明確な分類タスクにおいて，強い適応能力を持つことを示している．

4.4 法律ドメインの自然言語推定タスク

LLM によって生成された ContractNLI タスクのサンプル数および各フィルタリング手法適用後のサンプル数を表 4.7 に示す．表 4.7 より，フィルタリング手法によって保持される自己生成サンプル数に大きな差があることが分かる．類似度フィルタリングおよび生成確率フィルタリングでは，フィルタリング前と比べて一定数のサンプルが除外されているのに対し，LLM-based Filtering では，ほぼ全てのサンプルが保持されている．この結果は，各フィルタリング手法が想定している「信頼度」の基準が異なり，サンプル選別の厳しさにも違いがあることを示している．以降では，これらのフィルタリング手法の違いが，LLM の分類性能にどのように反映されるかを詳しく考察する．

表 4.7: ContractNLI タスクにおけるフィルタリング前後のサンプル数

	#Samples
フィルタリング前	1,434
Similarity Filtering	1,145
Probability Filtering	1,032
LLM-based Filtering	1,422

表 4.8 に，ContractNLI タスクにおける評価結果を示す．表 4.2 に示したように，使用した ContractNLI のテストデータセット [2] にはラベル分布に大きな偏りがあるため，評価指標としては正解率だけでなく，クラス間のバランスを考慮した macro F1 スコアも採用している．また，図 4.6〜図 4.8 に entailment，contradiction，neutral のそれぞれの分類クラスを持つ自己生成サンプルに対して，類似度フィルタリングで算出された信頼度スコア (前提と仮説の類似度) の分布を示す．さらに，図 4.9 に生成確率フィルタリングにおける信頼度スコア（生成確率）の分布を，図 4.10 に LLM 評価フィルタリングにおける信頼度スコア（LLM による評価スコア）の分布をそれぞれ示す．

表 4.8: ContractNLI タスクの評価結果

	Acc.	macro F1
Baseline	0.4385	0.2125
FT フィルタリング前	0.3651	0.2422
FT Similarity Filtering	0.3134	**0.2734**
FT Probability Filtering	0.2983	0.2577
FT LLM-based Filtering	**0.4134**	0.2376

Acc. は正解率，macro F1 はクラス不均衡を考慮したマクロ平均 F1 スコアを表す．

まず，表 4.8 をみると，全体的な傾向として，自己生成したラベル付きサンプルを用いたファインチューニングにより，ベースラインと比較して macro F1 はいずれの設定においても向上している．この結果は，専門ドメインかつマルチクラスという難易度の高いタスクにおいても，自己生成データが一定の下流タスク適応効果を持つことを示している．

一方で，正解率については，ファインチューニング後に低下する設定も見られ，評価指標によって手法の優劣が大きく異なる結果となった．このような乖離は，ContractNLI におけるラベル不均衡に起因すると考えられる．正解率は多数派クラスの予測性能に強く影響されるため，少数派クラスの性能向上が必ずしも正解率の改善として反映されない．そのため，本タスクにおいては，macro F1 に基づく評価の方が，モデルの実質的な性能を適切に評価していると考える．

フィルタリング手法の違いに注目すると，類似度フィルタリングを適用した場

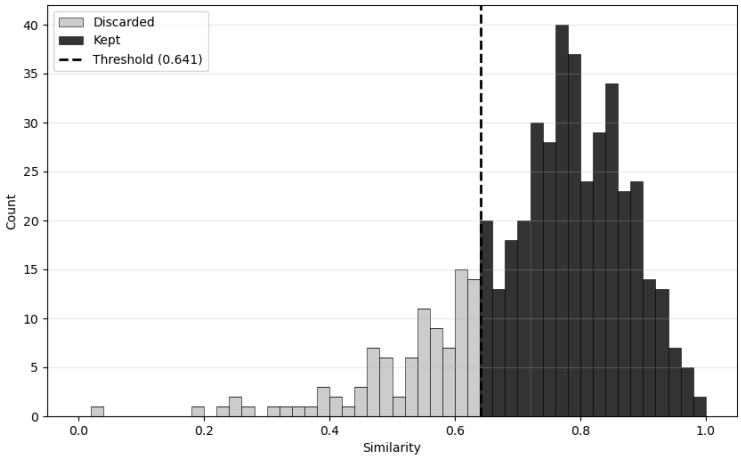

図 4.6: 類似度フィルタリングによる信頼度スコアの分布 (ContractNLI タスク, entailment クラス)

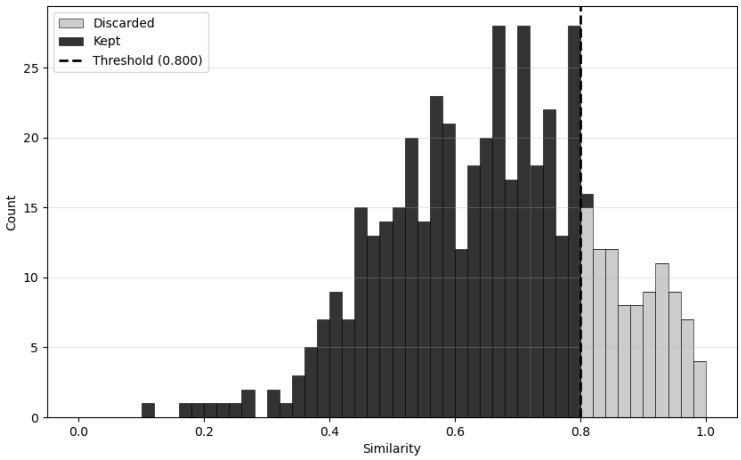

図 4.7: 類似度フィルタリングによる信頼度スコアの分布 (ContractNLI タスク, contradiction クラス)

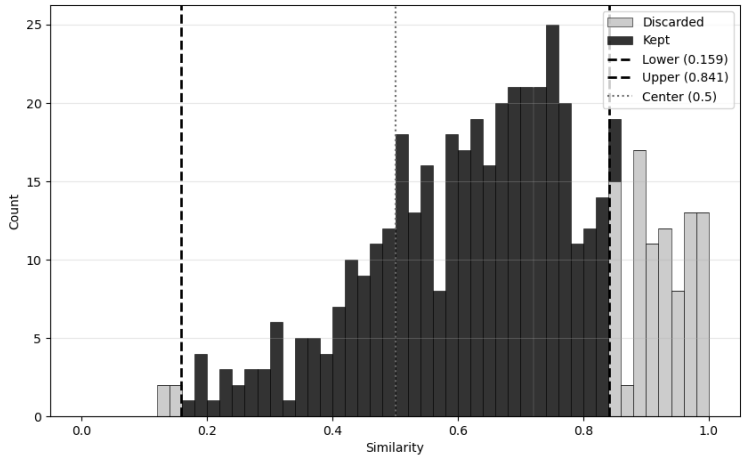

図 4.8: 類似度フィルタリングによる信頼度スコアの分布 (ContractNLI タスク, neutral クラス)

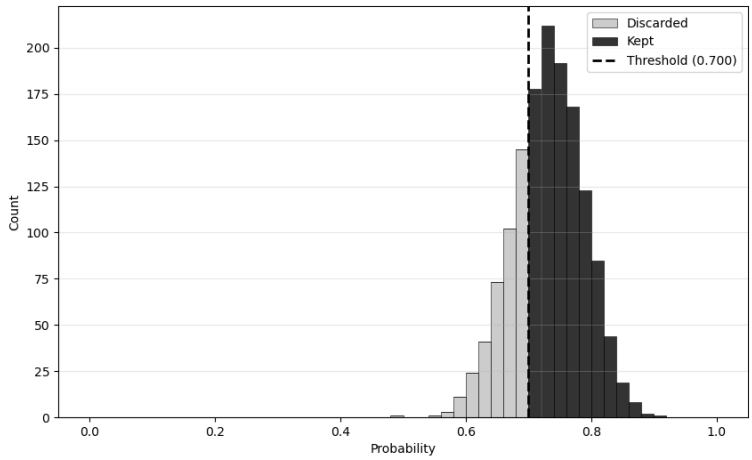

図 4.9: 生成確率フィルタリングにおける信頼度スコアの分布 (ContractNLI タスク)

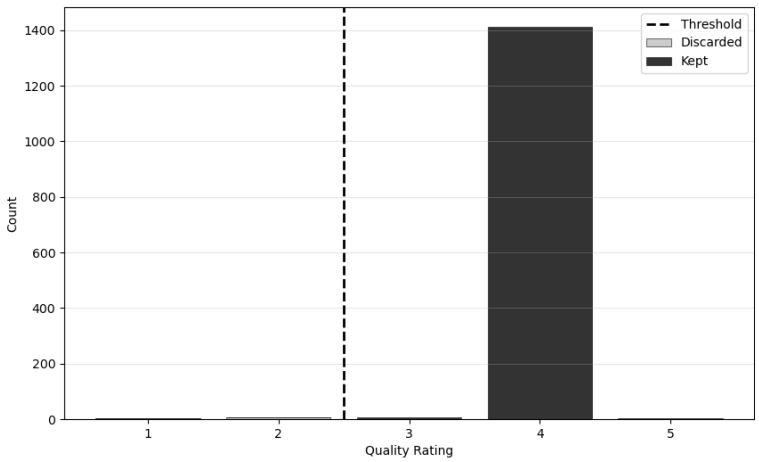

図 4.10: LLM 評価フィルタリングにおける信頼度スコアの分布 (ContractNLI タスク)

合に最も高い macro F1（0.273）が得られた．ContractNLI では，前提と仮説の対応関係が契約条項に基づく限定的かつ明示的な意味対応に依存するため，両文の意味的対応関係が明確なサンプルを選別することが，学習に有効に働いた可能性がある．本研究で用いた類似度フィルタリングは，前提と仮説の文脈表現間の類似度に基づいてサンプルを選別するものであり，推論関係が曖昧なペアを除外する効果を持つ．この結果は，専門ドメインにおける自然言語推論タスクでは，前提と仮説の意味的対応関係が明確であることが，自己生成サンプルの品質の高さの必要条件となっていることを示唆している．この傾向は，図 4.6〜図 4.8 に示す類似度分布において，ラベルごとに分布の特徴が異なることからも裏付けされる．Entailment ラベルでは 0.7 以上の高い類似度領域にサンプルが集中している一方，Neutral ラベルは中程度の類似度帯に広く分布しており，Contradiction ラベルの分布は Neutral と同じ，もしくは Entailment と Neutral の中間的な概形を持つ．このような分布の違いは，前提と仮説の意味的関係の強さが予測するべき分類ラベルと密接に関連していることを示しており，類似度に基づくフィルタリングが推論関係の曖昧なサンプルを効果的に除外できている可能性を示唆している．

一方で，LLM 評価によるフィルタリングは，正解率の面では比較的高い値を示したものの，macro F1 の改善は限定的であった．これは，汎用 LLM が契約文書特有の法的含意や，微妙な否定・条件表現を十分に捉えきれず，少数派クラスに属するサンプルの妥当性評価が必ずしも適切でなかった可能性を示している．すなわち，専門ドメインにおいては，LLM による自己評価が，必ずしもタスクに即した信頼度指標として機能しない場合がある．図 4.10 に示す LLM 評価スコア分布を見ると，ほとんどのサンプルの評価点が 4 であり，サンプル間の質的差異が十分に捉えられていないことが分かる．以上から，LLM 評価によるフィルタリン

グは，ContractNLI のような専門ドメインにおいては有効な選別指標として機能しにくいと言える．

また，生成確率によるフィルタリングでは，正解率・macro F1 ともに大きな改善は見られなかった．この結果は生成確率が高いサンプルが必ずしも推論タスクに有用な学習サンプルであるとは限らないことを示唆する．また，図 4.9 に示す生成確率分布では，多くのサンプルが高確率領域に集中しており，生成確率に基づくフィルタリングがサンプル間の有用性の差を十分に反映できていない可能性がある．

以上の結果から，ContractNLI のような専門ドメインかつマルチクラス・不均衡タスクにおいては，自己生成データによるファインチューニングは一定の効果を持つものの，その効果は他の一般的な分類タスク (RTE タスク，SA タスク) と比べて限定的であると言える．また，汎用的な信頼度評価よりも，タスク固有の特性を考慮した類似度フィルタリングが有効であり，専門タスクへの適応には，よりタスク依存的なサンプル選別が必要であることが示された．

4.5 テキスト生成タスク

LLM によって生成された E2E NLG タスクのサンプル数および各フィルタリング手法適用後のサンプル数を表 4.9 に示す．表 4.9 より，E2E NLG タスクにおいて，生成確率フィルタリングおよび LLM 評価フィルタリングのいずれを適用した場合も，フィルタリング前と比べて除外されるサンプル数はほとんどないことが分かる．また，フィルタリング手法間で保持されるサンプル数に大きな差は見られず，本設定においては，いずれの手法も自己生成サンプルの大部分を学習に用いている．フィルタリング後のサンプル数にほとんど差がないとはいえ，以降の実験では，2 つのフィルタリング手法が下流タスク性能にどのような影響を与えるかを詳しく分析する．

表 4.9: E2E NLG タスクにおけるフィルタリング前後のサンプル数

	#Samples
フィルタリング前	1,972
Probability Filtering	1,965
LLM-based Filtering	1,958

表 4.10 に，E2E NLG タスクにおけるベースラインならびに提案手法の評価結果を示す．また，図 4.11 に生成確率フィルタリングににおける信頼度スコアである生成確率の分布を，図 4.12 に LLM 評価フィルタリングにおける信頼度スコア (LLM による評価スコア) の分布をそれぞれ示す．

まず，全体的な傾向として，自己生成したラベル付きサンプルを用いたファインチューニングにより，すべての評価指標において，ベースラインである zero-shot

表 4.10: E2E NLG タスクの評価結果

	BLEU	ROUGE-L	METEOR	BERTScore
Baseline	0.0458	0.2138	0.4045	0.8768
FT フィルタリング前	0.0874	0.2949	0.4693	0.9116
FT Probability Filtering	0.0881	0.2914	0.4712	0.9116
FT LLM-based Filtering	**0.0910**	**0.2990**	**0.4743**	**0.9126**

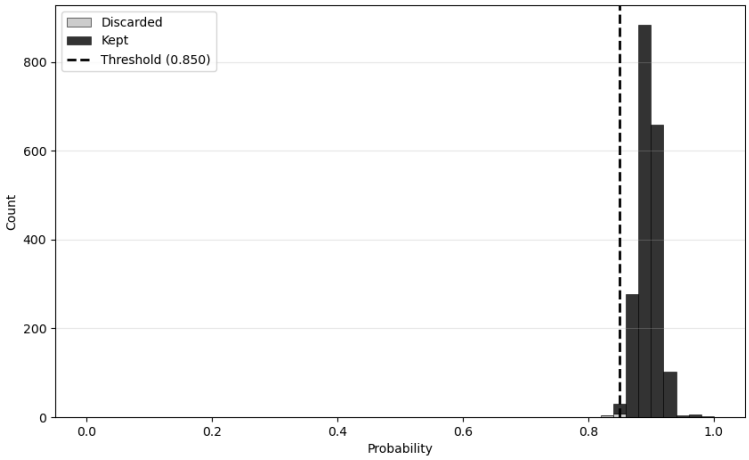

図 4.11: 生成確率フィルタリングにおける信頼度スコアの分布 (E2E NLG タスク)

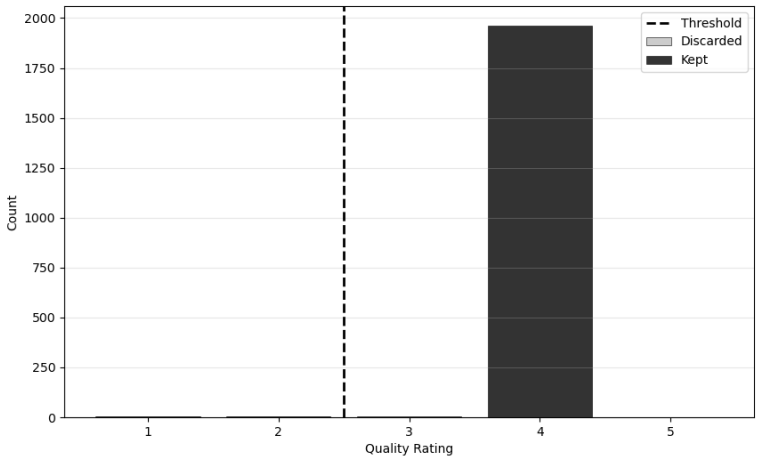

図 4.12: LLM 評価フィルタリングにおける信頼度スコアの分布 (E2E NLG タスク)

推論から性能が向上していることが確認できる．この結果は，生成タスクにおいても，LLM 自身が生成した疑似教師データを用いることで，下流タスクへの適応が可能であることを示している．

評価指標に注目すると，BLEU や ROUGE-L といった表層一致に基づく指標では，全体として比較的低いスコアに留まっている一方で，BERTScore は高い値を示している．これは，モデルが生成した文が入力 MR と単語や構文の面では一致しない場合でも，意味内容としては類似していることを示唆している．E2E NLG のように，入力に対する妥当な出力が多様に存在するタスクにおいては，意味的類似性を評価できる指標の方がモデル性能を適切に反映していると考えられる．

フィルタリング手法の違いに着目すると，LLM 評価によるフィルタリングを適用した場合に，すべての評価指標において最も高い性能が得られた．E2E NLG タスクでは，生成文が入力 MR に含まれる属性–値の組を過不足なく含んでいるかどうかが重要であり，このようなタスク要件は，主に文の流暢性を評価する生成確率フィルタリングでは十分に評価できない．そのため，入力と出力の対応関係を直接評価する LLM ベースの信頼度評価が，高品質な学習サンプルの選別に有効であったと考えられる．図 4.12 に示す，LLM 評価スコアを確認すると，ほとんどのサンプルの評価スコアが 4 となっている．他の分類タスクでもスコアが 4 のサンプルが多いが，E2E NLG タスクの方がサンプル間のスコアのばらつきがより小さい．これは，E2E NLG タスクにおいては，入力 MR に対する妥当な出力の多様性が高く，LLM による妥当性判断が比較的寛容に働くことを示唆している．

一方で，生成確率によるフィルタリングを用いた場合，フィルタリングなしの設定と比べて性能向上は限定的であった．生成確率は，モデルにとって生成しやすい一般的な表現を高く評価する傾向があり，必ずしも MR の内容を忠実に反映した文を選別できるとは限らない．また，図 4.11 では，生成確率が高い領域にほぼ全サンプルが集中しており，生成確率に基づく指標がサンプル間の有用性の差を十分に反映できていないと言える．

表 4.9 に示すように，生成確率フィルタリングおよび LLM 評価フィルタリングのいずれにおいても，除外されるサンプル数は少なかった．しかし，LLM 評価フィルタリングのみが全ての指標においてファインチューニング後のモデルの性能を向上させた．このことから，LLM による自己評価の方が品質の良いサンプルの選別に適していること，たとえ少量でも低品質のサンプルを除外することが E2E NLG タスクでは効果的であったことがわかる．

以上の結果から，自己生成データを用いたファインチューニングは，分類タスクに限らず，生成タスクにおいても有効に機能することが確認された．特に，入力と出力の意味的対応関係を重視する生成タスクにおいては，LLM 自身によるタスク指向の評価を用いたフィルタリングが，学習に用いる自己生成サンプルの選別において一定の有効性を示した．このことは，本研究で提案した自己生成・自己評価の枠組みが，多様な下流生成タスクに適用可能であることを示唆している．

4.6 タスク間比較に基づく考察

本節では，前節までに示した各下流タスクの実験結果を踏まえ，自己生成サンプルに対するフィルタリング手法の有効性について，タスクによる違いを考察する．表 4.11 に各下流タスクにおける生成サンプル数と各フィルタリング後の生成適用後のサンプル数を再掲する．また，表 4.12 と表 4.13 に各下流タスクにおける評価結果を再掲する．

表 4.11: 生成したサンプル数

		フィルタリング後		
	サンプル数	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

まず，全体的な傾向として，自己生成サンプルを用いたファインチューニングは，分類タスク・生成タスクの別を問わず，zero-shot ベースラインと比較して性能を向上させることが確認された．この結果は，下流タスクに関する知識が事前学習済み LLM に潜在的に内在しており，自己生成データを介したファインチューニングによって，その知識を下流タスクに適応させることが可能であることを示唆している．

フィルタリング手法に着目すると，LLM 評価によるフィルタリングは，SA タスクおよび E2E NLG タスクにおいて一貫して高い性能向上を示した．これらのタスクでは，入力と出力の対応関係が比較的明確であり，サンプルの妥当性を LLM が判断しやすかった可能性がある．また，表 4.11 からもわかるように，SA タスクでは LLM 評価フィルタリングによって多くのサンプルが削除された一方で，最終的な性能は最も高くなっている．一方で，RTE や ContractNLI のように，文間推論や専門的知識を必要とするタスクでは，LLM 評価フィルタリングによる性能改善は限定的であった．これらのタスクでは，自己生成されたサンプル自体に含まれる推論誤りや知識の欠落を LLM が捉えることができなかったと考えられる．特に ContractNLI のような専門ドメインタスクでは，法的表現や暗黙的前提に関する誤りを LLM が十分に自己検出できなかった可能性がある．

生成確率に基づくフィルタリングは，サンプルがモデルによってどれだけ生成されやすいか，もしくはサンプルの流暢性を指標としているものの，タスク定義に沿った入出力関係の正しさを直接評価するものではない．そのため，特に SA タスクにおいては，フィルタリングなしの手法と比較して性能が低下する結果となった．このことから，ファインチューニング用データの選別においては，サンプルがタスクの定義に適合しているかを考慮することが重要であると考えられる．

表 4.12: 分類タスクにおける性能比較

手法	RTE	SST-2	ContractNLI	
	Acc	Acc	Acc	F1-macro
Baseline	0.4693	0.5356	0.4385	0.2125
FT フィルタリング前	0.5235	0.8234	0.3651	0.2422
FT Similarity Filtering	**0.5632**	–	0.3134	**0.2734**
FT Probability Filtering	0.5307	0.6881	0.2983	0.2577
FT LLM-based Filtering	0.4908	**0.9071**	**0.4134**	0.2376

表 4.13: E2E NLG タスクにおける性能比較 (再掲)

手法	BLEU	ROUGE-L	METEOR	BERTScore F1
Baseline	0.0458	0.2138	0.4045	0.8768
FT フィルタリング前	0.0874	0.2949	0.4693	0.9116
FT Probability Filtering	0.0881	0.2914	0.4712	0.9116
FT LLM-based Filtering	**0.0910**	**0.2990**	**0.4743**	**0.9126**

類似度フィルタリングは，全てのタスクに適用できる手法ではないものの，RTE タスクの正解率や ContractNLI のマクロ F1 において最も高い性能を示した．これは，入力と出力の意味的関係が明確なペアを選別することが，推論型タスクにおいて有効である可能性を示唆している．

分類タスクと生成タスクを比較すると，本研究で扱った生成タスクである E2E NLG においては，LLM による意味ベースのフィルタリングが，参照文との意味的類似度を評価する BERTScore の向上に寄与する傾向が確認された．一方，分類タスクにおいては，タスクごとに有効なフィルタリング手法が異なり，単一の手法が全てのタスクに一様に有効であるとは限らないことが明らかになった．

以上より，自己生成データによるファインチューニングは，複数の下流タスクに対して一定の有効性を示した一方で，その効果の大きさや有効なフィルタリング手法はタスクの性質に依存することが明らかになった．

第5章　おわりに

本章では，本研究のまとめと今後の課題について述べる．

5.1　本研究のまとめ

本研究では，ラベル付きデータが存在しない，あるいは極めて限定的な状況下において，大規模言語モデル（LLM）が自己生成したラベル付きサンプルを用いたファインチューニングの有効性を検証した．特に，自己生成データに対するフィルタリング手法に着目し，複数の下流タスクにおいて，フィルタリング条件の違いが下流下流タスクに対するモデルの性能に与える影響を実証的に分析した．本研究では，サンプル生成・サンプル選別・モデル学習の各段階を明示的に分離した枠組みを採用することで，自己生成データの品質とファインチューニング後のモデルの性能との関係を考察した．

実験結果から，自己生成データを用いたファインチューニングは，分類タスクおよび生成タスクの別を問わず，zero-shot ベースラインと比較して一定の性能向上をもたらすことが確認された．この結果は，事前学習済み LLM が下流タスクに関連する知識を潜在的に保持しており，自己生成データを介した学習によって LLM の下流タスクを解く能力を向上させる可能性を示唆したものであった．一方で，フィルタリング手法の効果はタスクに強く依存しており，単一のフィルタリングが全てのタスクにおいて一貫して有効であるとは限らないことも明らかとなった．特に，入力と出力の対応関係が比較的明確なタスクでは，LLM による意味的妥当性評価に基づくフィルタリングが有効に機能する一方，文間推論や専門知識を要するタスクでは，その効果は限定的であった．

本研究は，自己生成データを用いたファインチューニングにおいて，フィルタリング手法とタスク特性の関係を明示的に整理した点に意義がある．本研究で得られた知見は，ラベル付きデータに依存しない下流タスク適応手法の設計に対して参考となる情報を提供するものであり，今後の自己学習および自己チューニング研究の基盤となることが期待される．

一方，本研究にはいくつかの制約も存在する．まず，評価は限られた数の下流タスクおよび LLM に基づいて行われており，得られた知見が全てのタスクや LLM に一般化可能であるとは限らない．また，自己生成データに対するフィルタリング手法として，類似度，生成確率，LLM 評価といった比較的単純な方法を採用し

ており，より高度な推論的整合性や外部知識との整合性を直接評価する仕組みは含まれていない．さらに，本研究では自己生成データのみを用いた学習設定に焦点を当てており，少量の人手データとの併用や半教師あり設定については検討していない．

5.2 今後の課題

今後の課題として，まずフィルタリング基準の自動化および適応的選択が挙げられる．タスクの性質や生成データの分布に応じて，複数の生成サンプルの品質評価基準を組み合わせる，あるいは動的に切り替えることで，より安定したデータ選別が可能になると考えられる．また，専門ドメインタスクにおいては，外部知識やルールベースの検証を組み込んだフィルタリング手法の導入も有望である．さらに，自己生成データと人手データを段階的に組み合わせる学習設定や，自己生成・自己評価ループを反復的に適用する枠組みの検討も，自己生成データに基づく下流タスク適応の理解を深める上で重要な方向性である．

参考文献

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

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

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

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

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

[6] 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.

[7] Seokhyun An and Hyounghun Kim. Response tuning: Aligning large language models without instruction, 2024. Submitted to ICLR 2025.

[8] Satanjeev Banerjee and Alon Lavie. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *ACL Workshop*, 2005.

[9] Shu Chen, Xinyan Guan, Yaojie Lu, Hongyu Lin, Xianpei Han, and Le Sun. Reinstruct: Building instruction data from unlabeled corpus. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 6840–6856. Association for Computational Linguistics, 2024.

[10] Ido Dagan, Oren Glickman, and Bernardo Magnini. The PASCAL recognising textual entailment challenge. In *Machine Learning Challenges. Evaluating*

Predictive Uncertainty, Visual Object Classification, and Recognising Textual Entailment, Vol. 3944 of *Lecture Notes in Computer Science*, pp. 177–190. Springer, 2006.

[11] 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.

[12] Chen Huang, Yang Deng, Wenqiang Lei, Jiancheng Lv, and Ido Dagan. Selective annotation via data allocation: These data should be triaged to experts for annotation rather than the model. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 301–320. Association for Computational Linguistics, 2024.

[13] Yue Huang, Siyuan Wu, Chujie Gao, Dongping Chen, Qihui Zhang, Yao Wan, Tianyi Zhou, Chaowei Xiao, Jianfeng Gao, Lichao Sun, and Xiangliang Zhang. Datagen: Unified synthetic dataset generation via large language models. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2025.

[14] Ke Ji, Junying Chen, Anningzhe Gao, Wenya Xie, Xiang Wan, and Benyou Wang. Llms could autonomously learn without external supervision, 2024.

[15] Yuta Koreeda and Christopher Manning. ContractNLI: A dataset for document-level natural language inference for contracts. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pp. 1907–1919, Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[16] Haoran Li, Qingxiu Dong, Zhengyang Tang, Chaojun Wang, Xingxing Zhang, Haoyang Huang, Shaohan Huang, Xiaolong Huang, Zeqiang Huang, Dongdong Zhang, Yuxian Gu, Xin Cheng, Xun Wang, Si-Qing Chen, Li Dong, Wei Lu, Zhifang Sui, Benyou Wang, Wai Lam, and Furu Wei. Synthetic data (almost) from scratch: Generalized instruction tuning for language models, 2024.

[17] Baohao Liao, Yan Meng, and Christof Monz. Parameter-efficient fine-tuning without introducing new latency. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 4242–4260. Association for Computational Linguistics, 2023.

[18] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *ACL Workshop*, 2004.

[19] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

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

[21] Jekaterina Novikova, Ondřej Dušek, and Verena Rieser. The E2E dataset: New challenges for end-to-end generation. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pp. 201–206, Saarbrücken, Germany, 2017. Association for Computational Linguistics.

[22] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs up? Sentiment classification using machine learning techniques. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 79–86, 2002.

[23] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *ACL*, 2002.

[24] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 1631–1642. Association for Computational Linguistics, 2013.

[25] Yuda Song, Hanlin Zhang, Carson Eisenach, Sham Kakade, Dean Foster, and Udaya Ghai. Mind the gap: Examining the self-improvement capabilities of large language models. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2025.

[26] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pp. 353–355. Association for Computational Linguistics, 2018.

[27] 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.

[28] Yuxin Xiao, Shujian Zhang, Wenxuan Zhou, Marzyeh Ghassemi, and Sanqiang Zhao. Sftmix: Elevating language model instruction tuning with mixup recipe, 2024. Submitted to ICLR 2025 (Withdrawn).

[29] 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. 15033–15045. Association for Computational Linguistics, 2024.

[30] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert. In *ICLR*, 2020.

[31] Xiongtao Zhou, Jie He, Yuhua Ke, Guangyao Zhu, Víctor Gutierrez Basulto, and Jeff Z. Pan. An empirical study on parameter-efficient fine-tuning for multimodal large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 10057–10084. Association for Computational Linguistics, 2024.

付 録A　参考データ

A.1　RTE・ContractNLI タスクのキーワード

RTE タスクで使用したキーワード（Wikipedia のカテゴリ名）の一部を表 A.1 に示す．また，ContractNLI タスクで使用したキーワードは，RTE タスクと同じキーワードリストの中から抽出して使用している．

表 A.1: RTE・ContractNLI タスクで使用したキーワードの一部

Research	Mountains	Empires	Statistics	Scientists
Encyclopedias	Oceans	Agriculture	Logic	Politicians
Music	Medicine	Education	Biology	Writers
Literature	Nutrition	Politics	Physics	Philosophy
Architecture	Historiography	Mathematics	Astronomy	Ethics
Buddhism	Christianity	Islam	Economics	Law
Sociology	Artificial intelligence	Robotics	Engineering	

A.2　SA タスクのキーワード

SA タスクで使用した映画のジャンルとアスペクトの一覧を表 A.2 および表 A.3 に示す．

表 A.2: SA タスクで使用した映画ジャンル

Action	Adventure	Animation	Art	Biography
Comedy	Crime	Documentary	Drama	Experimental
Fantasy	Film noir	Historical	Horror	Horror noir
Military	Musical	Mystery	Romance	Science fiction
Steampunk	Space opera	Dystopian	Utopian	Thriller
Western	Post-apocalyptic	Tech noir	New Wave	Fantastique

A.3　E2E NLG タスクのキーワード一覧

E2E NLG タスクで使用した都市名の一覧を表 A.4 に示す．

表 A.3: SA タスクで使用した映画のアスペクト

shallow focus	deep focus	deep space	dialogue overlap	diegesis
diegetic sound	non-diegetic sound	post-synchronization	direct sound	focal length
focus	front projection	rear projection	frontality	function
process shot	reestablishing shot	re-framing	rhetorical form	rhythm
rotoscope	scene	screen direction	segmentation	sensor
sequence	shot	shot/reverse shot	side lighting	soft lighting
sound bridge	sound over	sound perspective	space	special effects
story	montage	editing	continuity editing	match cut
jump cut	dissolve	fade in	fade out	cross-cutting
flashback	flash-forward	camera angle	lighting	theme
character	dialogue	pacing	tension	tone
visual style	symbolism	mise-en-scène		

表 A.4: E2E NLG タスクで使用した都市名の一部

Tokyo	Bangkok	Mumbai	Shanghai	Dubai
London	Berlin	Paris	Barcelona	Moscow
Cairo	Nairobi	Lagos	Cape Town	New York
Los Angeles	Mexico City	Toronto	Chicago	Sao Paulo
Buenos Aires	Rio de Janeiro	Lima	Sydney	Melbourne
Auckland	Istanbul	Singapore	Seoul	Amsterdam