PUPPET：タスク性能を維持しながら LLM として検出されやすくする学習フレームワーク

齋藤幸史郎 [1] 小池隆斗 [1] 金子正弘 [2,1] 岡崎直観 [1,3,4]
[1] 東京科学大学 [2] MBZUAI [3] 産業技術総合研究所 [4] NII LLMC
{koshiro.saito@nlp., ryuto.koike@nlp., okazaki@}comp.istc.ac.jp
masahiro.kaneko@mbzuai.ac.ae

概要

昨今，大規模言語モデル（Large Language Models; LLM）が生成する文を検出する需要が高まっており，代表的な手法の一つに透かし（Watermark）がある．生成文に埋め込まれた透かしトークンを調べることで高性能な検出が可能な一方で，そのトークンはランダムに選択されるため，その生成文の品質やタスク性能に課題がある．そこで，本研究では LLM のタスク性能を維持したまま，その LLM の生成文を検出しやすくするフレームワーク PUPPET を提案する．具体的には，生成文に対して検出器とタスク評価器の二つから報酬関数を設計し，LLM の強化学習を行う．評価実験の結果，我々のフレームワークで学習した LLM は文書要約タスクにおいてタスク性能を維持したまま，その検出が容易（再現率 6pt 向上）となったことを確認した．

1 はじめに

LLM の普及に伴い，学生が課題への回答を LLM に生成させる，悪意を持った人物が LLM を用いて Wikipedia に虚偽の記事を投稿する [1] 等の問題が見られるようになった．こうした LLM の不正利用を防ぐ一つの手段として，人間が書いた文章（人間文）と LLM などの生成器による文章（機械文）を識別する機械文検出タスクの重要性が増している．

機械文検出には，１）機械文検出器の性能向上，２）生成文の検出されやすさ向上，の二つの大きな方向性がある．一つ目の検出器の検出性能を高める既存研究として，様々なタスクに対する LLM の回答と人間による回答を学習データとして，RoBERTa [2] などの事前学習済みエンコーダを微調整する教師あり検出器がある [3]．その他，人間文と機械文における尤度 [4] やエントロピー [5] などの統計的な特徴量の違いに基づく検出器がある．こうした既存手法はロバスト性や検出性能に違いはあるものの [6]，検出対象の文が短い場合や [7]，人間が書く文と内容的に近い場合 [8] に検出が困難になるという共通の課題がある．

二つ目の生成文の検出されやすさを高める既存研究として，透かし（Watermark）[9] が代表的である．これは文生成過程において，ハッシュ関数によりランダムに選択された透かしトークンの生成確率を意図的に増加させ，その生成文に多く出現させる手法である．この手法は仕組み上，従来の教師あり検出器や統計的な特徴量の違いに基づく検出器よりも検出性能が高く，注目されている一方で，ハッシュ関数によりランダムに選択されるトークンが文脈に合致しないと，文章の流暢性や下流タスクでの性能劣化を引き起こすことが課題となっている [9, 10]．透かしは，一つ目の方向性で説明した短文や人間文と類似する機械文に対して検出率が低くなるという課題に加えて，透かしの手法に生成器の学習が含まれていないため，生成文の流暢性やタスク性能の劣化を抑えることができなかった．

そこで本研究では，検出器が推定する各クラス（機械文または人間文）の尤度を用いた検出器報酬と，劣化を防ぎたい観点の評価メトリクス（評価器報酬）を組み合わせた報酬を定義し，LLM の強化学習を行うことで，タスク性能を維持したまま LLM の生成文を検出されやすくする，柔軟で強力なフレームワーク **PUPPET** を提案する．評価実験の結果，機械文検出が難しいとされるタスクの一つである要約タスクにおいて，要約性能と文章の流暢性を維持しながら，検出されやすさ（再現率）を 6pt ほど向上させる学習に成功した．

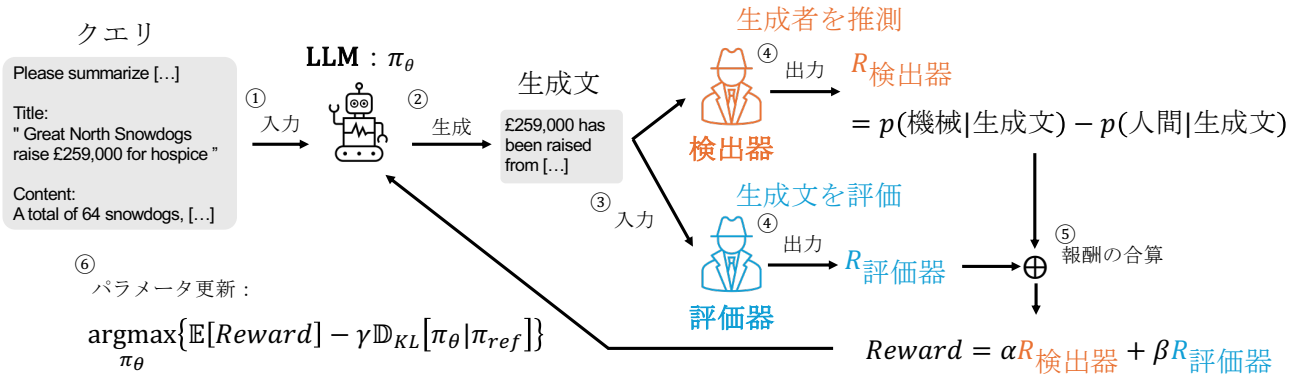

図 1　提案手法：PUPPET の概要図.

2　提案手法：PUPPET

図 1 に PUPPET フレームワークの概要図を示す．PUPPET では，検出器の予測値に基づく報酬とタスク評価器の評価値に基づく報酬を用いて LLM が検出器に検出されやすくなるよう，近傍方策最適化（Proximal Policy Optimization; PPO）[11] による強化学習を行う．大まかに，1）クエリの入力，2）回答の生成，3）検出器と評価器への回答の入力，4）検出器と評価器による報酬の計算，5）報酬の合算，6）パラメータ更新の 6 ステップから成る．

検出器の予測値に基づく報酬 $R_{\text{検出器}}$ は式 1 のように，LLM から生成された文（生成文）を検出器が機械文と判定する尤度 $p($ 機械 $|$ 生成文 $)$ から人間文と判定する尤度 $p($ 人間 $|$ 生成文 $)$ を引いたものと定義した．

$$R_{\text{検出器}} = p(\text{機械} \mid \text{生成文}) - p(\text{人間} \mid \text{生成文}) \quad (1)$$

タスクの評価メトリクスに対応する報酬 $R_{\text{評価器}}$ はタスクに応じて任意に定義してよく，各タスクの代表的な評価指標値をそのまま利用可能である．そうして得られた二つの報酬に対して重み α, β を用いて式 2 のように加重平均を取り，学習に用いる報酬とする．

$$\text{Reward} = \alpha R_{\text{検出器}} + \beta R_{\text{評価器}} \quad (2)$$

これらの係数は検出器報酬と評価器報酬の設計や，学習によって持たせたい LLM の性質に合わせて設定する．PPO の方策モデル π_θ（最終的に学習したい LLM のパラメータ）の更新時には，参照モデル π_{ref}（PPO 実施前の LLM のパラメータ）から離れすぎないように，π_θ と π_{ref} の確率分布のカルバックライブラー距離を重み γ で罰則項として導入したうえで，報酬を最大化（式 3）するように π_θ を更新する．

$$\arg\max_{\pi_\theta} \{\mathbb{E}[\text{Reward}] - \gamma\mathbb{D}_{KL}[\pi_\theta|\pi_{\text{ref}}]\} \quad (3)$$

方策モデルが参照モデルから乖離しすぎると，文法や意味が通らない文章が生成されるなど，言語生成能力の著しい劣化が引き起こされる [11].

PUPPET フレームワークによって，人間が見ても分からない（＝生成文自体の質を劣化させない）が検出器には強く反応する機微を LLM に学習させ，文章の流暢性や下流タスクでの性能を劣化させずに検出されやすさを向上させることを期待している．

3　実験設定

3.1　検出データセットの構築

検出器の構築（§ 3.2）と LLM の学習（§ 3.3）に使うデータセットの構築手法について説明をする．

本研究では，検出器が従来苦手とされている条件設定（想定される回答の幅が狭く [8] かつ出力が短い [7]）を持つタスクとして，要約タスクを対象にする．データセットには，要約タスクにおいて代表的な XL-Sum[1] [12] を採用した．この XL-Sum は英語や日本語を含む 44 言語の BBC ニュースを収集し，ニュース記事，タイトル，記事要約を収録したデータセットである．

本研究では，XL-Sum において重複する記事があるもの，文字数が 1,500 文字を超える記事，収集のミスにより要約の長さが 3 単語以下のものは除外した．残った事例のうち，2,000 件ずつを検出器の学習用，LLM の学習用，評価用にランダムに選んだ．LLM 生成文の検出実験には，要約文のみを用い，XL-Sum に収録されている要約文を人間文とした．XL-Sum には人間によって書かれた文しか収録されていないので，Llama-3.1-8B-Instruct に記事の要約を生成させ，機械文とした．ここで，LLM 検出器が容易に見つけやすい特徴（ショートカット），例

1）https://huggingface.co/datasets/csebuetnlp/xlsum

表 1　学習前後における LLM の被検出性能，生成文の流暢性，要約性能の比較（人間文に対する検出結果は 3.2 で述べた値と同様．[] 内の値は未学習の設定との比較で赤は向上，青は劣化を示し，PPL と BARTScore についてはそれぞれ Mann-Whitney の U 検定と Paired T 検定を行い，有意（有意水準 5%）である差に † を付した．PPL は中央値，BARTScore は平均値を表示．太文字は最高値を表す．）

			検出のされやすさ		文章の流暢性	要約性能
	報酬	(α, β)	再現率 [%] ↑	F1 値 [%] ↑	PPL ↓	BARTScore ↑
未学習	-	-	80.03	86.40	26.42	0.650
学習	検出器報酬	$(1, 0)$	**89.13 [△9.10]**	**91.71 [△5.32]**	32.10 [△5.68†]	0.642 [▽0.008†]
	+評価器報酬	$(1, 0.5)$	87.83 [△7.80]	90.98 [△4.59]	32.67 [△6.25†]	0.642 [▽0.008†]
		$(1, 1)$	85.88 [△5.85]	89.87 [△3.48]	23.43 [▽2.98†]	0.651 [△0.001]
		$(1, 2)$	85.18 [△5.15]	89.46 [△3.08]	**23.08 [▽3.33†]**	**0.654 [△0.004]**

えば文長の違い [6] や Llama-3.1-8B-Instruct に起因する誤り（文法的な誤り [13] やそもそも指示に追従できていない）は学習データから排除したい．そのため，機械文生成時のプロンプトには，人間文と機械文の文長の分布が近くなるように文字数指定を行い，さらに生成文 (要約) の質を高めるため one-shot 事例を追加した．なお，one-shot 事例を選ぶ際には，解かせるクエリが含まれるデータセットと重複しない同数の異なるデータセットを用意し，そこからタイトルの意味的類似度が最も近い[2]サンプルを選んだ [14]．プロンプトや one-shot 事例の実装方法に関する詳細は付録 B を参照されたい．

3.2　検出器の構築

PUPPET フレームワークで用いる検出器は，検出器報酬さえ計算できればどのようなものでも良い．本稿では提案手法の実現可能性を担保するために，xlm-RoBERTa-base をベースモデルとした簡易的な検出器を構築した．

教師あり検出器のベースモデルには，機械文検出の先行研究 [6, 4, 15, 16] において広く用いられている xlm-RoBERTa-base[3] [17] を選んだ．学習では，§3.1 で作成した検出器の学習用データの 80%を訓練セット，残り 20%を検証セットとした．評価セットは，訓練セット・検証セットとは被りのないように，§3.1 で作成した評価用のデータ 2,000 件とした．検出器の学習の実装のため，`transformers.Trainer` を用いた（詳細なハイパーパラメータは 付録 A を参照）．結果として，評価セットにおいて人間文に対して 94.75%，機械文に対して 80.0%の再現率を示す検出器を構築できた．

2)　最近傍の検索には `faiss.IndexFlatL2` を用いた．
3)　https://huggingface.co/FacebookAI/xlm-roberta-base

3.3　二つの報酬を用いた学習

PUPPET フレームワークにおける評価器報酬は自由に設計することができる．本研究では $R_{\text{評価器}}$ として BARTScore [18] を採用した．これは，要約タスクにおいて，LLM を用いない評価指標の中で最も人間の評価と相関が高いと報告されている [19] からである．検出器報酬と評価器報酬を合算したものは，式 4 のように表される．検出器報酬の係数 α と評価器報酬の係数 β は次の三通りで実験した: $(\alpha, \beta) = (1, 0.5), (1, 1), (1, 2)$．学習セットは § 3.1 の LLM の学習用の 2,000 件を用いた（したがって，検出器学習用のデータと LLM 学習用のデータが混ざることはない）．

$$Reward = \alpha R_{\text{検出器}} + \beta BARTScore \tag{4}$$

また，本研究ではタスク性能の劣化に加え，生成される文章の流暢性の劣化も調べるため，パープレキシティ（PPL）も計測した．

4　結果

表 1 に LLM の検出されやすさ，およびタスクの性能の変化を示した．ここから大きく二つのことが分かった．

検出器報酬の学習による被検出性の向上　表内の再現率と F1 値の差分に着目する．全ての α, β の組み合わせにおいて，未学習時に比べて LLM の検出されやすさが向上しており，最大で再現率が 9pt，F1 値が 5pt ほど向上している．

評価器報酬による流暢性の改善，要約性能の維持　表内の PPL と BARTScore の差分に着目する．検出器報酬のみの設定や報酬比 $(1, 0.5)$ の設定では，PPL の劣化幅に関する p 値は 5.75×10^{-4}, 1.43×10^{-3},

BARTScore の劣化幅に関する p 値は 3.52×10^{-14}, 3.48×10^{-17} といずれも 0.05 を下回っており，有意に劣化している．一方で，報酬比 $(1, 1)$ や $(1, 2)$ の設定では PPL の改善幅に関する p 値は 3.70×10^{-6}, 9.02×10^{-8} と 0.05 を下回り有意に改善，BARTScore の変化幅に関する p 値は 0.63, 0.07 と 0.05 を上回ることから維持ができている．

また，表 1 から評価器報酬の係数 β と負の PPL や BARTScore の間に正の相関（相関係数はそれぞれ，0.98，0.92 程度）が確認された．反対に，検出されやすさ（再現率，F1 値）と負の PPL や BARTScore の間にはトレードオフの関係（相関係数はいずれも-0.9 を下回る）が確認された．このことから，PUPPET フレームワークでは，報酬係数 α, β の調整により，LLM の検出のされやすさとタスク性能のトレードオフを調節できることが分かった．

5 分析

5.1 学習による生成文の変化

PUPPET フレームワークによる学習で得られた LLM の生成文はなぜ検出されやすくなったのか．ここでは，先行研究に倣って品詞タグ [3] や文長の変化 [6] を分析し，生成文の変化を調べる．

5.1.1 品詞タグの 2-gram の変化

未学習，検出器報酬のみで学習した後，報酬係数 $(1, 1)$ で学習した後について，それぞれの生成文（1,998 件）の品詞タグの 2-gram を調べ[4)]，各 2-gram の頻度についてスピアマンの順位相関を計算した．未学習時と検出器報酬のみでの学習後，未学習時と報酬比 $(1, 1)$ での学習後は，ともに 0.984 という高い相関係数を示した．また，検出器報酬のみでの学習後と報酬比 $(1, 1)$ での学習後の順位相関係数は 0.985 で，先ほどと同等の高さであった．この高い相関の要因は，英語で書かれた文章に共通する事項かもしれないが，少なくとも PUPPET フレームワークを用いた学習を経ても，LLM が生成する文章の品詞タグの分布の変化は僅かであることが示された．この分析の詳細については，付録 C を参照されたい．

5.1.2 文長分布の変化

未学習，検出器報酬のみでの学習した後，報酬係数 $(1, 1)$ での学習した後について，それぞれの生成文（1,998 件）の文字数（記号を除く）を調べ，ヒストグラムとして図 2 に示した．この図を見ると，検出器報酬のみで学習した場合は未学習の時と文長の分布が酷似しているが，報酬係数 $(1, 1)$ の設定で学習した場合は，文字数の多い方に分布が離れていた．このことから，評価器報酬（BARTScore）を学習に加えると文長が長くなり，人間文や未学習時の機械文の文長分布から乖離していることが分かる．ただし，検出器報酬のみの学習よりも報酬係数 $(1, 1)$ の方が LLM としての検出されやすさが低下していることから，文長分布が人間文から離れたことが検出に有利に働いていることはないと考えている．これは検出器を学習する際に機械文の文長分布を人間文の分布に近づけた（付録 B）ことによって，検出器がより本質的な（ショートカットではない）特徴量を学習していることの現れかもしれない．

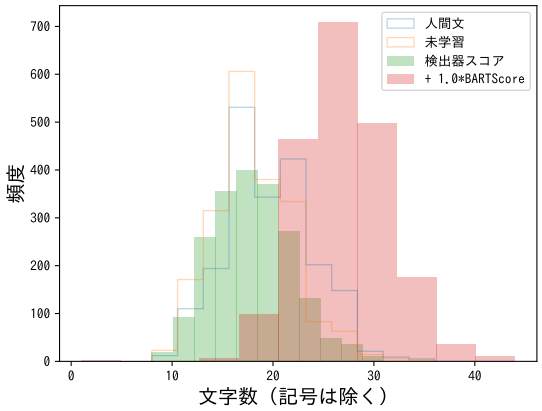

図 2 学習前後における文長分布の変化．

6 結論

本研究では，文章の流暢性や下流タスクでの性能を劣化させずに LLM の検出されやすさを向上させることを目標とし，検出器報酬と評価器報酬を用いた学習フレームワーク PUPPET の有効性を検証した．実験結果から，このフレームワークによって文章の流暢性と要約タスクの性能を維持しながら LLM としての検出されやすさを向上できることを確認した．PUPPET フレームワークは，高精度な LLM 検出手法として注目を集めている透かし（Watermark）と同じく，LLM 生成文を検出されやすくするアプローチでありながら，透かしとは異なり，任意の評価メトリクスを報酬として学習に組み込み，特定の性能の劣化を抑えながら検出されやすさを向上できるという独自の強みがある．

4) 品詞タグの抽出には spacy の en_core_web_sm を用いた．

謝辞

本研究成果は，国立研究開発法人情報通信研究機構（NICT）の委託研究（22501）により得られたものです．

参考文献

[1] WikiProject Cleanup. https://en.wikipedia.org/wiki/Wikipedia:Cleanup.

[2] 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, 2020.

[3] Qintong Li, Leyang Cui, Wei Bi, Zhilin Wang, Longyue Wang, Linyi Yang, Shuming Shi, and Yue Zhang. MAGE: Machine-generated text detection in the wild. In **Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)**, pp. 36–53, 2024.

[4] Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D. Manning, and Chelsea Finn. DetectGPT: Zero-shot machine-generated text detection using probability curvature, 2023.

[5] Thomas Lavergne, Tanguy Urvoy, and François Yvon. Detecting fake content with relative entropy scoring. In **Proceedings of the ECAI'08 Workshop on Uncovering Plagiarism, Authorship and Social Software Misuse**, 2008.

[6] Laida Kushnareva, Tatiana Gaintseva, Dmitry Abulkhanov, Kristian Kuznetsov, German Magai, Eduard Tulchinskii, Serguei Barannikov, Sergey Nikolenko, and Irina Piontkovskaya. detection in mixed AI-human texts. In **First Conference on Language Modeling**, 2024.

[7] Sandra Mitrović, Davide Andreoletti, and Omran Ayoub. ChatGPT or human? detect and explain. explaining decisions of machine learning model for detecting short chatGPT-generated text. arXiv:2301.13852, 2023.

[8] Zhenpeng Su, Xing Wu, Wei Zhou, Guangyuan Ma, and Songlin Hu. Hc3 plus: A semantic-invariant human chatgpt comparison corpus. arXiv:2309.02731, 2024.

[9] John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. A watermark for large language models. arXiv:2301.10226, 2024.

[10] Pierre Fernandez, Antoine Chaffin, Karim Tit, Vivien Chappelier, and Teddy Furon. Three bricks to consolidate watermarks for large language models. arXiv:2308.00113, 2023.

[11] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. arXiv:1707.06347, 2017.

[12] Tahmid Hasan, Abhik Bhattacharjee, Md. Saiful Islam, Kazi Mubasshir, Yuan-Fang Li, Yong-Bin Kang, M. Sohel Rahman, and Rifat Shahriyar. XL-sum: Large-scale multilingual abstractive summarization for 44 languages. In **Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021**, pp. 4693–4703, 2021.

[13] Weixin Liang, Mert Yuksekgonul, Yining Mao, Eric Wu, and James Zou. GPT detectors are biased against non-native english writers. **Patterns**, Vol. 4, No. 7, p. 100779, 2023.

[14] Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. What makes good in-context examples for GPT-3? In **Proceedings of Deep Learning Inside Out (DeeLIO 2022): The 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures**, pp. 100–114, 2022.

[15] Zhouxing Shi, Yihan Wang, Fan Yin, Xiangning Chen, Kai-Wei Chang, and Cho-Jui Hsieh. Red teaming language model detectors with language models. **Transactions of the Association for Computational Linguistics**, Vol. 12, pp. 174–189, 2024.

[16] Yuxia Wang, Jonibek Mansurov, Petar Ivanov, jinyan su, Artem Shelmanov, Akim Tsvigun, Osama Mohammed Afzal, Tarek Mahmoud, Giovanni Puccetti, Thomas Arnold, Chenxi Whitehouse, and et al. Semeval-2024 task 8: Multidomain, multimodel and multilingual machine-generated text detection. In **Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024)**, pp. 2041–2063, 2024.

[17] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. In **Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics**, pp. 8440–8451, 2020.

[18] Weizhe Yuan, Graham Neubig, and Pengfei Liu. Bartscore: Evaluating generated text as text generation. In **Advances in Neural Information Processing Systems**, Vol. 34, pp. 27263–27277, 2021.

[19] Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. G-eval: NLG evaluation using gpt-4 with better human alignment. In **Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing**, pp. 2511–2522, 2023.

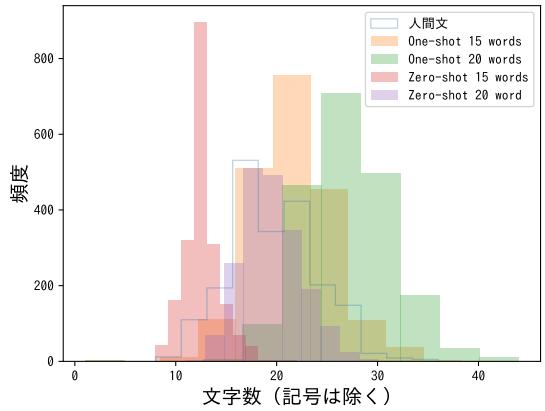

図 3　プロンプト毎の人間文との文長分布の比較．（人間文と機械文は共に valid split の 1,998 件）

A　ハイパーパラメータ

表 2　訓練時パラメタ

Name	Value
Batch Size	3
Num Epochs	1
Random Seed	42
LR	1.41e-4

表 3　推論時パラメタ

Name	Value
min_length	-1
top_k	0.0
top_p	1.0
do_sample	True
max_new_tokens	300

学習時に `trl.PPOTrainer` に渡した `trl.PPOConfig` に含まれる値と，推論時に `trl.PPOTrainer.generate` に渡していた入力文を除く引数のうち，執筆時点のデフォルト値とは異なる値を用いたものをそれぞれ表 2 と表 3 にまとめた．なお，それぞれのデフォルト値については公式のドキュメント[5] を参照．

B　プロンプト

文長の違い [6] や LLM 由来の誤り（e.g. 文法的な誤り [13] やそもそも指示に追従できていない）は検出器のショートカットとして学習されかねない [6]．そこで我々は以下の工夫をプロンプトに凝らした．

one-shot 事例を考慮して要約するプロンプト　one-shot 事例の書き方を参考にしながら，2nd Article のみ要約をするプロンプトを作成した（図 4）．

文長分布を揃える文字数指定　プロンプト内の文字数指定をいくつか試した（図 3）ところ，zero-shot で”within 20 words”と指定したものが最も人間文の分布に近く，one-shot で”within 15 words”が次点に近いと分かる．本研究では one-shot の方が人間文との峻別が難しい生成が得られた（次段落参照）ので

5）Hugging Face. ”TRL. Trainer”., Hugging Face. ”Trainsformers. Utilities for Generation”.

Prompt (Input)

```
Please summarize the 2nd article within 15 words. You may use
the 1st summary as an example.
You must start your summary with 'Summary:¥n'


# 1st Article (Example)
Example Title:
"{one-shot title}"

Example Content:
{one-shot content}

Example Summary:
{one-shot summary}


# 2nd Article
Title:
"{title}"

Content:
{content}

Summary:
(Your summary on 2nd article here)
```

図 4　最終的に採用したプロンプト

図 4 の通り，one-shot の”within 15 words”を採用した．

人間文らしい生成　生成文の質が悪く，一見して LLM が書いたとバレるデータセットを用いては検出の意義が薄くなってしまう．そこで，生成文がなるべく人間らしく（＝人間文との峻別が難しく）なるように one-shot 事例の選び方を探索した．一般に，解かせるサンプルに近いサンプルを one-shot 事例とする方が生成文の質が上がると報告されている [14] ことから，各サンプルについて近いサンプルを one-shot 事例として取ってくる動的な one-shot を採用した．そして，1）比較対象：記事 vs. タイトル，2）データの入手先：使用セットと同じ vs. 違う，3）近さ（L2 ノルム）を測る埋め込みのモデル：Llama-3.1-8B-Inst vs. xlm-RoBERTa-base の 2^3 通りを検証した．その結果，使用セット外のサンプルのタイトルを Llama-3.1-8B-Inst で埋め込んだ時の最近傍サンプルを取ってくることが最も機械文の検出率（再現率）を下げると判明し，それを採用した．

C　頻出品詞タグ 2-gram の詳細

§ 5.1.1 で、学習前後で比較したときに生成文の品詞タグの分布の変化が僅かであることを述べたが，未学習と報酬比 $(1, 0), (1, 1)$ の品詞タグ 2-gram 頻出上位 10 件を比較したところいずれの報酬比でも未学習に対して名詞→接置詞と助動詞→動詞が減り，名詞→句読点が増えていた．しかし，それ以外には特に一貫した変化はなく，定性的な分析は難しい．