日本語 LLM 構築におけるコーパスクリーニングの網羅的評価

新里 顕大[1,2]　高瀬 翔[1]　清野 舜[1]　李 凌寒[1]　加藤 卓也[1]
水本 智也[1]　小林 滉河[1]　佐藤 潤一[1]　柴田 知秀[1]
[1]SB Intuitions 株式会社　[2]京都大学大学院 情報学研究科
{kenta.shinzato,sho.takase,shun.kiyono,ryokan.ri,takuya.kato,tomoya.mizumoto,
koga.kobayashi,junichi.sato,tomohide.shibata}@sbintuitions.co.jp

概要

本稿では日本語 LLM の事前学習用のコーパスをクリーニングすることがモデルの性能向上に有効であることを示す．評価実験として，クリーニングの条件を変え 1.3B パラメータの日本語 LLM を学習し，複数の質問応答タスクおよび事後学習後の自由記述質問応答における性能を比較した．その結果，計算資源が比較的限られている場合（250B トークンの学習）では，クリーニングによりモデルの性能が向上し，計算資源が十分な場合（1T トークンの学習）では，クリーニングにより性能維持，タスクによっては性能向上することを確認した．

1　はじめに

大規模言語モデル（Large Language Model，LLM）は汎用的な自然言語処理アプリケーションの基盤となっている [1, 2]．LLM の性能はパラメータ数と事前学習のデータ量に対数比例すると報告されており [3]，性能向上のためには大量のコーパスが必須である．

ウェブテキストを収集した Common Crawl[1)] が大規模コーパスとして主に使われているが，数多く含まれるワードサラダのような不自然なテキストを除去することの重要性が指摘されている [4, 5]．英語のコーパスクリーニングについては経験的な知見が蓄積されつつあるが，それら知見を得るには膨大な計算資源が必要なため，網羅的調査の障壁となっている．そこで日本語をはじめ，英語以外の言語については言語判定，文の長さによるフィルタリング，重複文除去のような，言語毎に設計した処理が不要な基本的な処理のみが利用されている [6, 7]．

このような状況を受けて本研究では，日本語 LLM 構築におけるクリーニング手法の効果を検証する．

1） commoncrawl.org

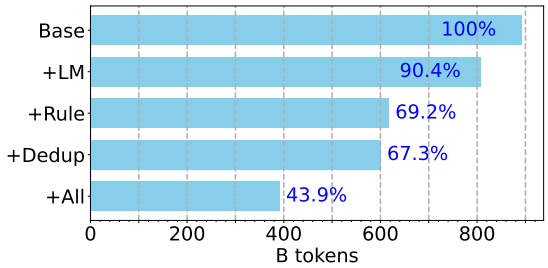

図 1　Base に各クリーニング手法を適用した際の学習データの量（含まれるトークン数）．なお，Base は Common Crawl に基本的な前処理を適用したコーパス．

検証する手法は，質の高い文書から N-gram 言語モデルを構築して Perplexity の高い文書を除去する手法（LM），記号の羅列や広告などに見られる人工的な定型句を多く含む文書を除去するルールベースの手法（Rule），類似した文書を除去する手法（Dedup）の 3 つである．これらを適用することで，質の高い文書の割合が増加し学習効率が高くなると予想される反面，学習データの量が半減する（図 1）ため，十分な計算資源を確保できる場合には LLM の性能が伸び悩む可能性もある．本稿の実験を通して，1. 計算資源が限られている場合，すなわち基本的な前処理を適用したコーパス（Base）全てを学習できるほどの資源がない場合，クリーニングを行った方が性能が良いこと，2. 計算資源が十分にある場合でもクリーニングにより性能は維持あるいは改善されることを示す．また，各クリーニング手法を分析し，それぞれの有効性についても議論する．

2　クリーニング手法

本研究で利用するクリーニング手法について説明する．なお CCNet を用いた基本処理と重複段落の除去を除き，クリーニングにはテキスト処理ライブラリである HojiChar [8][2)] を利用した．

2） github.com/HojiChar/HojiChar

表 1　Rule における処理の一覧.

文長フィルタ：100-200,000 字の範囲外の文書を除去
制御文字などの非表示文字の除去
単一文字を 200 回以上繰り返している文書の除去
特定の文字列パターン (日付/URL/Copyright/ナビゲーションメニュー/HTML タグ) 等に合致する行の削除
複数行にわたり先頭が一致する場合，1 つを残して削除
句読点を含む記号，空白，絵文字，その他特殊な文字を一定の割合以上含む文書を除去
成人向け/広告キーワードに合致する文書の確率的除去
文書内での N-gram の頻度を元に，繰り返しの多い文書を除去
形態素解析し，名詞の割合が 0.8 以上の文書を除去

2.1　基本処理: CCNet による処理

Common Crawl の前処理として，コーパスのクリーニングを行うツールセットである CCNet [7] を用い，重複段落除去と言語判定を適用した．重複段落除去では Common Crawl のスナップショットを 5GB に分割し，分割された各ファイル内で重複している段落[3]を除去する．これにより定型文や日付からなる段落などが除去できる．言語判定には fastText [9] を用いる．fastText は文字 N-gram を素性とし，与えられた文書がどの言語であるかについて，0 から 1 の値のスコアを付与する．ここでは日本語のスコアが 0.5 以上の文書を抽出する．

2.2　LM: N-gram 言語モデルによる処理

基本処理だけでは除去できない，日本語としては不自然な文書が存在する．例えば「こたつ掛け布団スウェード調パッチワーク円形冬…」のような広告に見られるキーワードの羅列や「全球冏雅思…」のような言語判定をすり抜けた外国語などがこれにあたる．こうした文書を除去するため，質の高い日本語文書で学習した N-gram 言語モデルで Perplexity を計測し，その値が高い文書を除去する．言語モデルの実装として，Kenser-Ney Smoothing [10] を適用した 2-gram の言語モデルを，MeCab [11] で単語分割を行った Wikipedia の日本語記事で学習した[4]．これを用いて計算した Perplexity において，コーパス内上位 10%を質が低い文書として除去する．

2.3　Rule: ルールベースの処理

表 1 に示したルールを用いてクリーニングを行う．例えば名詞の割合が 0.8 以上の文書を除去することで「ブレスレット ダウンコート マフラー…」というような名詞の連続からなる文書を除去できる．

3)　実装上はファイル上の 1 行を 1 段落とみなして処理する．
4)　N-gram 言語モデルの実装には KenLM [12] を用いた．

2.4　Dedup: 類似文書・重複文書除去

同一のテンプレートから生成された文書など，ウェブテキストには類似した文書が散見される．Lee ら [4] はこのような文書を除去することで LLM の学習効率が向上すると報告している．Lee らと同様，本研究でも MinHash LSH [13] を用いて類似した文書を重複とみなし除去を行う．具体的には文書中の 5-gram 集合についてハッシュ関数を適用し，文書間の Jaccard 係数が設定した閾値を超えた場合に重複とみなして除去した[5]．閾値は既存研究 [14] を参考に設定した．この処理により，例えば付録 A に示したような文書ペアが除去される．

また，Penedo ら [5] が指摘しているように，Common Crawl は同一のウェブページをクロールしていることもあるため，取得元 URL が同一である文書も重複とみなして除去した．

3　実験設定

3.1　事前学習用コーパス

実験用の大規模なコーパスとして，Common Crawl の 2021 年 11 月から 2023 年 6 月までのスナップショットをダウンロードした．また近年では，Multilingual C4 (mC4) [6, 15] をはじめとして，クリーニング済みの Common Crawl [16, 17] も配布されている．本稿では A) ダウンロードした Common Crawl に 2.1 節で説明した基本処理を適用したコーパス，B) mC4 内の日本語コーパスを組み合わせたものを Base と呼び，そこに 2 節で紹介したクリーニング手法を適用していく．なお mC4 は 2.1 節の基本処理と同様に，言語判定や重複除去が行われており，加えて，200 文字以上の文を 3 文以上含む文書のみを抽出するという処理がなされている．

各クリーニング手法を適用した際の，コーパス内のトークン数の変化を図 1 に示す．トークン数は，Base に SentencePiece [18] の Unigram 言語モデルを適用して構築したトークナイザで計数しており，このトークナイザは事前学習モデル構築でも利用する．Base の時点では 900B トークン程度含まれているが，LM，Rule，Dedup のすべての処理を適用した場合（図 1 における+All）では 400B トークン程度と，Base の半分以下のサイズとなる．

5)　MinHash LSH の実装には datasketch を用いた：github.com/ekzhu/datasketch

3.2 事前学習モデルの学習設定と評価指標

本研究では GPT [1, 19, 20] と同様，Transformer [21] を用いて LLM を構築する．様々な条件で実験を行うため，パラメータ数は比較的軽量な 1.3B とした．具体的には GPT-NeoX [22] のリポジトリにある，1.3B パラメータの設定を使用する[6]．各バッチに含まれるトークン数は 4M トークンとし，更新回数を変化させることによって事前学習に利用するコーパス量を変動させる．

事前学習モデルの評価は zero-shot および few-shot [1] の設定で，下記 3 つのデータセットを用いて行う．モデルに与えたプロンプトと生成時のハイパーパラメータ等の詳細設定は付録 C に記載する．

AI 王 クイズの問題文を入力とし，その解答を出力する．AI 王公式配布データセット Version 2.0[7] 中の開発セットを用い，正答とモデル出力の完全一致率で評価する．プロンプトに few-shot 事例は含めず，zero-shot の設定で評価を行う．

JSQuAD Wikipedia の段落とその内容に関する質問文を入力とし，解答を段落から抽出する [23]．開発セットにおける完全一致率で評価する．学習セットからサンプルした 3 つの事例を含めて各事例のプロンプトを構築し，3-shot の評価を行う．

JCommonSenseQA (JCQA) 常識推論能力を評価する問題文と 5 つの選択肢から解答を選択する [23]．開発セットにおける正答率を 2-shot で評価する．

3.3 事後学習モデルの学習設定と評価指標

事前学習コーパスの量や質と LLM の学習効率の関係を議論した研究の多くは，事前学習モデルの評価のみにとどまっている [4, 24] が，実用的には事後学習後の性能も重要である．本研究では，実応用での性能の一端として，事前学習した LLM について Instruction Tuning [25] を行い，モデルが人間の指示をどの程度理解し，適切に返答できるか確認する．学習データには理研が公開している 2,903 件の日本語インストラクションデータ [26] を用いた．

事後学習モデルの評価には Rakuda Benchmark [8] を使用する．Rakuda Benchmark は自由記述型の質問に言語モデルが回答するタスクで，日本の歴史，地理，政治，社会の各 10 問，合計 40 問からなる．2 つのモデルの回答をペアとし，優れた回答を GPT-4 [27] に選択させることで 2 モデルごとの比較を行う．

6） github.com/EleutherAI/gpt-neox/blob/main/configs/1-3B.yml
7） sites.google.com/view/project-aio/dataset
8） github.com/yuzu-ai/japanese-llm-ranking

表 2 計算資源が限られている場面での正解率．

		平均値	AI 王	JSQuAD	JCQA
Base	250B tokens	45.6	30.8	**57.8**	48.2
+All	250B tokens	**48.9**	**34.9**	57.0	**54.9**

表 3 計算資源が十分にあり，Base を 1 周学習できる場面での正解率．

		平均値	AI 王	JSQuAD	JCQA
Base	1T tokens	54.2	37.6	62.7	**61.9**
+All	500B tokens	52.1	37.9	59.1	59.2
+All	1T tokens	**54.7**	**40.0**	**62.9**	61.3

4 実験結果と考察

実験を通して，1. 計算資源が限られている場合，2. 計算資源が十分であり，Base コーパスを 1 周学習できる場合におけるクリーニングの効果を検証する．また，2 節で紹介した各手法の効果を検証する．

4.1 事前学習におけるクリーニング効果

まず計算資源が限られており，モデルの学習に利用可能なトークン数に制限がある場面を考え，250B トークンまでの学習を行った[9]．結果を表 2 に記す．+All のコーパスで学習したモデルは Base で学習したモデルよりも AI 王，JCQA で性能が良く，また，JSQuAD でも同等の性能を達成していることが分かる．このことから，計算資源が限られている場合には様々なクリーニングを行い，学習コーパスの質を高めることが重要であると言える．

次に Base を 1 周学習できる程度には計算資源が十分ある状況を考える．図 1 のとおり，Base と+All はそれぞれ約 900B，400B のトークンを含んでいる．ここでは議論を簡単にするために，1T トークンと 500B トークンをそれぞれのコーパス 1 周分として扱い，学習を行う．また+All を 1T トークン，すなわち Base1 周と同等の計算資源を用いて学習を行ったモデルも比較する．表 3 に結果を記す．表 3 より，それぞれのコーパスを 1 周した場合，すなわち，Base の 1T トークンと+All の 500B トークンを比較した場合には Base の 1T トークンの方が性能が高いことが分かる．しかしながら，日本に特有の知識を必要とする AI 王では+All の 500B トークンは Base の 1T トークンと同等の性能を達成している．

9） 250B トークンの学習を 1 度行うには 128 枚の A100 (80GB) を用いて約 2 日（= 5300GPU 時間）を要した．

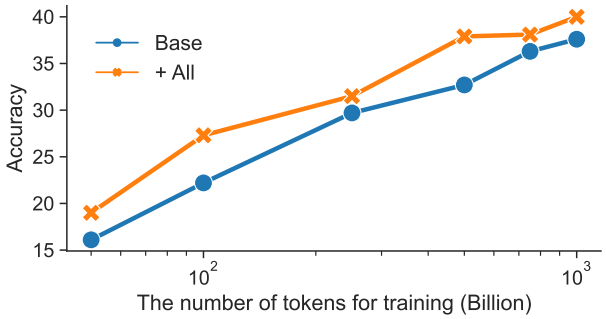

図 2　1T トークンまで学習させた場合の Base と+All の AI 王による性能比較．Base は約 1 周，+All は約 2 周学習させた．+All のユニークなデータ量はクリーニングによって Base の半分以下となっているが，学習に用いるトークン数が同じ場合には常に Base よりも性能が高い．

表 4　事後学習後の性能比較．各セルは「左の列のモデルの勝利数-上の行のモデルの勝利数」を示す．括弧内は引き分けを表す．

		Base 1T	+All 500B	+All 1T
Base	1T tokens	–	15-22 (3)	16-22 (2)
+All	500B tokens	**22-15 (3)**	–	15-21 (4)
+All	1T tokens	**22-16 (2)**	**21-15 (4)**	–

また+All を 1T トークン学習した場合には Base の 1T トークンよりもわずかに高い性能を達成している．この結果は，各種クリーニングに伴い学習コーパスが減少し，コーパスを 1 周以上する状況であっても，学習に用いる計算資源が同等であれば性能に毀損はないことを示唆している．

また，1T トークンまでの学習において，各トークン数での AI 王の正解率を図 2 に示した．この図からも，同等の計算資源を用いれば+All は Base よりも常に高い性能を達成することが分かる[10]．

4.2　事後学習への影響

Base を 1T トークン学習したモデルおよび+All を 500B トークン，1T トークン学習したモデルについて事後学習を行い，Rakuda Benchmark で評価した．結果を表 4 に示す．

この表から，+All を事前学習に用いたモデルは，Base を事前学習に用いたモデルよりも高い性能を達成していると分かる．すなわち，様々なクリーニングを適用して質を高めたコーパスでの事前学習は，事後学習後のモデルの性能も高めると言える．

10）学習に用いたトークン数に性能が対数比例していることからパラメータ数が不足して性能上昇が停滞していることはないと考えられる．付録 B にパラメータ数を 3.6B に増やした場合でも同傾向の結果が得られることを示す．

表 5　各クリーニング手法を適用した場合の性能比較

	平均値	AI 王	JSQuAD	JCQA
Base	45.6 -	30.8	57.8	48.2
+LM	46.8 (+1.2)	31.1	58.8	50.6
+Rule	50.9 (+5.3)	38.1	61.9	52.7
+Dedup	43.0 (−2.6)	29.6	52.2	47.3
+All	48.9 (+3.3)	34.9	57.0	54.9

4.3　各クリーニング手法の効果検証

2 節で紹介した各手法が事前学習にもたらす効果について検証する．Base に対して LM，Rule，Dedup をそれぞれ個別に適用し，得られたコーパスで 250B トークンの学習を行い，比較する．結果を表 5 に示す．LM と Rule は共に評価値を向上させており，コーパスの質を高めた効果が事前学習モデルの性能向上につながっていることがわかる．特に Rule は性能を大きく向上させており，日本語に特化したクリーニングの有効性を示している．

一方で，Dedup はすべてのデータセットでの性能が下がっている．この結果は既存研究の知見 [4] に反するが，2 つの原因が考えられる．ひとつは，類似文書の閾値の設定が低く，多くの有用な文書を除去してしまっている可能性である．もうひとつは，URL を用いた重複文書除去において，同一 URL だが内容が異なる文書を除去している悪影響である．Dedup で有用な文書を除去することなく，性能を向上させるようなクリーニングを行うことができれば+All の性能もより高くなると考えられるため，原因究明に努めたい．

5　おわりに

日本語 LLM の事前学習コーパスにおける日本語に応じたクリーニングの効果を，事前学習・事後学習の実験を通じて検証した．基本処理に加えて様々なクリーニングを適用しコーパスの質を高めることは，計算資源が限られている場合はモデルの性能を向上させ，十分に計算資源がある場合でも性能を維持または改善させることを示した．

4.3 節で記したさらなる検証に加え，今後は他言語，特に英語コーパスを事前学習に利用する効果も検証したい．英語コーパスを事前学習に追加することで，言語横断的な知識獲得が期待されるが，AI 王のような日本に特有の知識を問うタスクでも効果的であるかは不明瞭であり，より良い日本語 LLM 構築には何が必要かを明らかにしていきたい．

参考文献

[1] Tom Brown, Benjamin Mann, Nick Ryder, et al. Language Models are Few-Shot Learners. In **Advances in Neural Information Processing Systems**, Vol. 33, pp. 1877–1901. Curran Associates, Inc., 2020.

[2] Hugo Touvron, Thibaut Lavril, Gautier Izacard, et al. LLaMA: Open and Efficient Foundation Language Models, 2023.

[3] Jared Kaplan, Sam McCandlish, Tom Henighan, et al. Scaling Laws for Neural Language Models, 2020.

[4] Katherine Lee, Daphne Ippolito, Andrew Nystrom, et al. Deduplicating Training Data Makes Language Models Better. In **Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)**, pp. 8424–8445. Association for Computational Linguistics, 2022.

[5] Guilherme Penedo, Quentin Malartic, Daniel Hesslow, et al. The RefinedWeb Dataset for Falcon LLM: Outperforming Curated Corpora with Web Data, and Web Data Only, 2023.

[6] Linting Xue, Noah Constant, Adam Roberts, et al. mT5: A Massively Multilingual Pre-trained Text-to-Text Transformer. In **Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies**, pp. 483–498. Association for Computational Linguistics, 2021.

[7] Guillaume Wenzek, Marie-Anne Lachaux, Alexis Conneau, et al. CCNet: Extracting High Quality Monolingual Datasets from Web Crawl Data. In **Proceedings of the Twelfth Language Resources and Evaluation Conference**, pp. 4003–4012. European Language Resources Association, 2020.

[8] Kenta Shinzato. HojiChar: The text processing pipeline, 2023.

[9] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of Tricks for Efficient Text Classification. In **Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers**, pp. 427–431. Association for Computational Linguistics, 2017.

[10] Reinhard Kneser and Hermann Ney. Improved backing-off for M-gram language modeling. In **1995 International Conference on Acoustics, Speech, and Signal Processing**, Vol. 1, pp. 181–184 vol.1, 1995.

[11] Taku Kudo, Kaoru Yamamoto, and Yuji Matsumoto. Applying Conditional Random Fields to Japanese Morphological Analysis. In **Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing**, pp. 230–237. Association for Computational Linguistics, 2004.

[12] Kenneth Heafield. KenLM: Faster and Smaller Language Model Queries. In **Proceedings of the Sixth Workshop on Statistical Machine Translation**, pp. 187–197. Association for Computational Linguistics, 2011.

[13] Andrei Z. Broder. On the resemblance and containment of documents. In **Proceedings. Compression and Complexity of SEQUENCES 1997 (Cat. No.97TB100171)**, pp. 21–29, 1997.

[14] Leo Gao, Stella Biderman, Sid Black, et al. The Pile: An 800GB Dataset of Diverse Text for Language Modeling, 2020.

[15] Colin Raffel, Noam Shazeer, Adam Roberts, et al. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. **Journal of Machine Learning Research**, Vol. 21, No. 140, pp. 1–67, 2020.

[16] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, et al. Unsupervised Cross-lingual Representation Learning at Scale. In **Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics**, pp. 8440–8451. Association for Computational Linguistics, 2020.

[17] Julien Abadji, Pedro Ortiz Suarez, Laurent Romary, and Benoît Sagot. Towards a Cleaner Document-Oriented Multilingual Crawled Corpus, 2022.

[18] Taku Kudo and John Richardson. SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing. In **Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations**, pp. 66–71. Association for Computational Linguistics, 2018.

[19] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018.

[20] Alec Radford, Jeff Wu, Rewon Child, et al. Language Models are Unsupervised Multitask Learners. 2019.

[21] Ashish Vaswani, Noam Shazeer, Niki Parmar, et al. Attention is All you Need. In **Advances in Neural Information Processing Systems**, Vol. 30. Curran Associates, Inc., 2017.

[22] Sidney Black, Stella Biderman, Eric Hallahan, et al. GPT-NeoX-20B: An Open-Source Autoregressive Language Model. In **Proceedings of BigScience Episode #5 – Workshop on Challenges & Perspectives in Creating Large Language Models**, pp. 95–136. Association for Computational Linguistics, 2022.

[23] Kentaro Kurihara, Daisuke Kawahara, and Tomohide Shibata. JGLUE: Japanese General Language Understanding Evaluation. In **Proceedings of the Thirteenth Language Resources and Evaluation Conference**, pp. 2957–2966. European Language Resources Association, 2022.

[24] Niklas Muennighoff, Alexander M. Rush, Boaz Barak, et al. Scaling Data-Constrained Language Models. In **Thirty-Seventh Conference on Neural Information Processing Systems**, 2023.

[25] Jason Wei, Maarten Bosma, Vincent Zhao, et al. Finetuned Language Models are Zero-Shot Learners. In **International Conference on Learning Representations**, 2021.

[26] 関根聡, 安藤まや, 後藤美知子, 鈴木久美, 河原大輔, 井之上直也, 乾健太郎. Ichikara-instruction: LLM のための日本語インストラクションデータの構築. 言語処理学会 第 30 回年次大会, 2024.

[27] OpenAI, Josh Achiam, Steven Adler, et al. GPT-4 Technical Report, 2023.

[28] Deepak Narayanan, Mohammad Shoeybi, Jared Casper, et al. Efficient large-scale language model training on GPU clusters using megatron-LM. In **Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis**, SC '21, pp. 1–15. Association for Computing Machinery, 2021.

電子部品・半導体通販のマルツオンライン > 電気部品 > キット (Digi-Key) > 静電気対策キット > FLD SRVC KIT BL 2 PCKT 24X24 の概要

高角サッカー選手相手に得点の映像素材\nHigh angle drone shot of a soccer player dribbling past the defenders and scoring on the apposing team\n高角サッカー選手相手に得点 : ストック動画・映像\nサッカーボール 動画\n対戦試合 動画\nスタジアム 動画\nスポーツトレーニング 動画\n競争 動画\nプレーする

図 3　LM で除去される文書の例.

Kapeli Dash のダウンロード - Kapeli Dash プログラムに関する情報 - OTFE\nKapeli Dash アプリケーションの可能性次に掲げるリストは、Kapeli Dash プログラムがファイルのデータ編集および変換の両方をサポートするファイル拡張子のリストです。特定の拡張子が Kapeli Dash プログラムでサポートされている場合でも、(以下省略)

Puyo Tools のダウンロード - Puyo Tools プログラムに関する情報 - OTFE\nPuyo Tools アプリケーションの可能性次に掲げるリストは、Puyo Tools プログラムがファイルのデータ編集および変換の両方をサポートするファイル拡張子のリストです。特定の拡張子が Puyo Tools プログラムでサポートされている場合でも、 (以下省略)

図 4　Dedup で除去される文書ペアの例.

A　LM，Dedup の除去サンプル

LM，Dedup で除去される文書の例を図 3，4 に記載する.

B　3.6B パラメータモデルでの比較

4 節よりもパラメータ数が多いモデルでの比較を行う. 具体的には 3.6B パラメータのモデルを Base，+All で学習し，比較を行う. 層の数などのハイパーパラメータは Narayanan らに従った [28].

Base，+All それぞれで 250B トークン学習した際の結果を表 6 に示す. 4 節での実験と同様に，計算資源が限られているとして，学習を 250B トークンに限った場合には+All で学習したモデルの方が高い性能を達成している. 学習に用いたトークン数と AI 王での正解率の推移を図 5 に示す. この図から，3.6B パラメータにおいても，学習に費やした計算資源が同等であれば，様々なクリーニングを適用し，質の高いコーパスで学習したほうが常に高い性能を達成できることが分かる.

C　事前学習モデルの評価に用いたプロンプト

図 6，7 と 8 に事前学習モデルの評価に用いたプロンプト例を記載する. いずれも解答部分を鉤括弧でくくり，モデルには“」”まで出力させることで生成箇所を抽出している.

生成時のハイパーパラメータは基本的に貪欲法を採用した. AI 王のみ候補数 5 のビームサーチを採用し，repetition penalty を 3.5 と設定した.

表 6　3.6B パラメータでの比較.

		平均値	AI 王	JSQuAD	JCQA
Base	250B tokens	58.5	43.9	67.3	**64.3**
+All	250B tokens	**61.3**	**50.2**	**69.7**	64.2

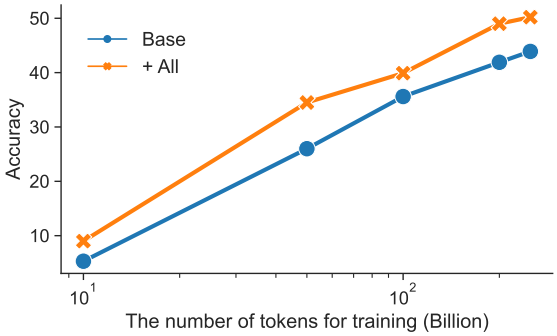

図 5　3.6B パラメータのモデルについて，学習トークン数と AI 王の正解率の推移.

映画『ウエスト・サイド物語』に登場する 2 つの少年グループといえば、シャーク団と何団？答えは**「ジェット団」**

図 6　AI 王のプロンプトと正答例.

質問に対する回答を文章から一言で抜き出してください.

文章
造語
造語（ぞうご）は、新たに語（単語）を造ることや、既存の語を組み合わせて新たな意味の語を造ること、また、そうして造られた語である。新たに造られた語については、新語または新造語とも呼ばれる。
質問：新たに造られた語のことを新語または何という？
回答：「新造語」

...

文章
グスタフ・マーラー
グスタフ・マーラー（Gustav Mahler，1860 年 7 月 7 日 - 1911 年 5 月 18 日）は、主にオーストリアのウィーンで活躍した作曲家、指揮者。交響曲と歌曲の大家として知られる。
質問：グスタフ・マーラーの誕生日は？
回答：**「1860 年 7 月 7 日」**

図 7　JSQuAD のプロンプトと正答例.

正しい答えは何でしょう？

0.「世界」
1.「写真集」
2.「絵本」
3.「論文」
4.「図鑑」
問題：主に子ども向けのもので、イラストのついた物語が書かれているものはどれ？
回答：「絵本」

...

0.「掲示板」
1.「パソコン」
2.「マザーボード」
3.「ハードディスク」
4.「まな板」
問題：電子機器で使用される最も主要な電子回路基板の事をなんと言う？
回答：**「マザーボード」**

図 8　JCQA プロンプトと正答例.