日本語 LLM の事前学習データの品質・構成が性能に与える影響

高橋 未央　蔵内 雄貴　神山 歩相名　西田 京介
NTT 株式会社 人間情報研究所
{michika.takahashi, yuki.kurauchi, hosana.kamiyama, kyosuke.nishida}@ntt.com

概要

本研究では，大規模な日本語 Web コーパスを構築し，文書の教育的価値（Edu スコア）および可読性に基づく難易度を品質指標としたサンプリング戦略と，完全一致重複を含む学習データが LLM の下流タスク性能に与える影響を調査し，以下の知見を示す．(1) Edu スコアが高い文書の比率を高めると一般知識タスク性能は向上する一方で，総合性能は低下する．(2) 完全一致重複を含む学習データでは性能が低下する．(3) 単一の指標に基づく積極的な選別よりも，低品質データのみを除去した上でのランダムサンプリングが有効である．

1　はじめに

LLM の事前学習には大量のテキストデータが必要となるため，Web コーパスが広く用いられている．Web コーパスはデータ量の確保に有効な一方で，ノイズを含む文書や内容の価値が低い文書が多く混在するという問題を抱えている．

そのため近年では，ルールベースのフィルタに加え，文書内容の品質に基づくフィルタリングが提案されている．FineWeb [1] では，教育的価値を品質指標とした事前学習データの選別が LLM の性能向上に寄与したことが報告されている．また，難易度の高い語彙や文法を含むデータの混入が学習上のノイズとなり得ること [2] や，非ネイティブにとっての英文の読みやすさを表すスコアを品質指標として導入する試み [3, 4, 5] も報告されている．しかしながら品質を教育的価値で定義したキュレーションは，難易度分布に偏りを生じさせる可能性がある．また FineWeb2 [6] では，MinHash [7] による重複回数を品質の代理指標とし，高品質と推定されるデータをアップサンプリングした結果，性能向上したことが報告されている．しかし，アップサンプリングは高品質側データの露出確率を上げるだけでなく，同一文書が複数回サンプリングされる可能性も生む．実際にどの程度の重複が生じていたかは明確でなく，性能向上が「高品質データの露出確率の増加による分布変化」によるものか「重複そのもの」によるものかは切り分けられていない．学習データの重複削除が LLM の性能向上に有効であることは Lee [8] らをはじめとする複数の先行研究により報告されており，アップサンプリングと重複発生の影響を整理する必要がある．

以上のように，先行研究では教育的価値や読みやすさといった品質指標の導入，高品質データのアップサンプリングが下流タスク性能の向上に寄与することが報告されているが，研究対象は英語データが中心であり，日本語データにおける影響は十分に調査されていない．そこで本研究では，文書の品質指標として教育的価値（Edu スコア）および可読性に基づく難易度を設定し，事前学習データの品質分布の操作が下流タスク性能に与える影響を日本語データで検証する．

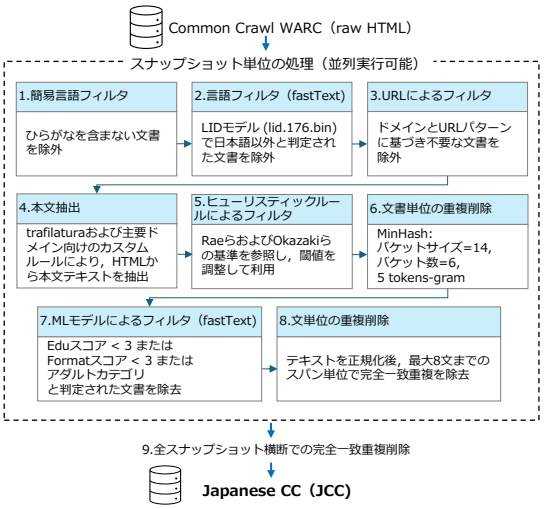

図 1　本研究の分析対象とした Japanese CC の構築フロー

2　日本語 Web コーパスの構築

本研究の分析のため，図 1 のフローを適用して日本語 Web コーパスを新たに構築した．

2.1 構築方法

概要 Common Crawl の WARC 形式ファイルをデータソースとして，CC-MAIN-2013-20 から CC-MAIN-2025-18 までの合計 108 スナップショットを対象に，Penedo ら [1] および Okazaki ら [9] を参考にして構築した．以降，本コーパスを Japanese CC（JCC）と呼ぶ．最終的に得られた文書数は約 8.6 億件，コーパス規模は約 2262 億 tokens[1)]である．

本文抽出 trafilatura [10] の抽出性能を補完するため，一部のドメインについては既定ルールで除去しきれない不要箇所を XPath で明示的に除外し，必要に応じて BeautifulSoup [11] による抽出も併用した．

文書単位の重複削除 Penedo ら [1, 6] は Jaccard 係数 0.75 の文書を約 80%の確率で検出する設定で MinHash による重複削除を実施している．本研究では，ボイラープレートテキストの除去を重視し，MinHash のハイパーパラメータを調整して Jaccard 係数 0.75 の文書を 90%以上，Jaccard 係数 0.8 を超える文書はほぼ 100%検出するよう設定した（図 1）．

ML モデルによるフィルタ 文書の教育的価値（Edu スコア），フォーマット品質（Format スコア），カテゴリを推定する fastText [12] 分類器をそれぞれ事前に訓練し（付録 A に補足情報を示す），図 1 の手順 7 で文書選別を行った．また，Edu スコアは後段の実験にてサンプリング制御にも用いた．

難易度のアノテーション Gohari ら [3] に倣い，日本語非ネイティブにとっての読みやすさを難易度の基準とし，難易度に応じて学習データを構築した．難易度の算出には Lee ら [14] の readability score 公式およびレベル尺度を用いた．このレベル尺度は日本語の学習段階に応じた文書の難易度分けであり，readability score が低いほどレベルが高い，すなわち語彙や文法の難易度が高いことを意味する．

2.2 分析

Edu・フォーマットスコアの分布 表 1 に JCC の Edu/Format スコアについて示す．JCC では全文書の 9 割以上が Edu/Format スコア 3 に集中していた．

難易度と Edu スコアの関係 JCC においては Edu スコアが高い文書ほど可読性に基づく難易度が高い傾向が観察されていた（図 2）．

1) 本研究におけるトークン数はすべて Gemma 7B [13] トークナイザを基準に算出した．

表 1 JCC における Edu スコア・Format スコアの分布

Score	Edu. Tokens (%)	Fmt. Tokens (%)
3	218 B (96.5%)	220 B (97.5%)
4	7 B (3.5%)	5 B (2.6%)
5	123 (0.0%)	3440 (0.0%)

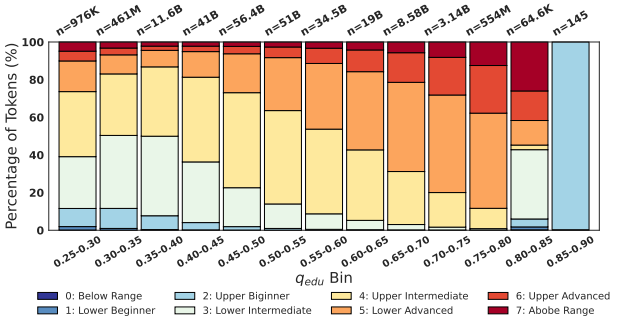

図 2 JCC における Edu スコアビンごとの文書難易度レベル分布．グラフ上部の値は該当ビンに含まれる合計トークン数を示す．

3 実験

本研究ではモデルと学習設定を固定し，学習データのみを変更してフルスクラッチ事前学習を行った．評価は学習途中のチェックポイントおよび最終チェックポイントに対して実施した．

3.1 共通設定

モデルと学習設定 Penedo ら [6] を踏襲し，Llama [15] アーキテクチャの 1.46B パラメータモデルを用いた．学習設定と詳細を付録；表 4 に示す．

学習トークンと重複 各条件の学習データは Chinchilla [16] 最適である延べ約 29B tokens となるように構成した．意図的に重複を導入した条件以外では完全一致重複を含まない．

3.2 比較条件

本研究では，以下の 4 つの観点に基づき計 10 件の比較条件を設定した．

外部ベースライン JCC と同様に Common Crawl をソースとして構築された既存の公開日本語コーパスから，ランダムサンプリングしたものを用いた．

- **FineWeb2-ja**: FineWeb2 [6] の日本語サブセット
- **llmjpv4-L0**: llm-jp-v4 level0 [17]

高 Edu スコア文書のスコア分布を操作 Edu スコアを品質指標としたキュレーションの効果を検証した．JCC では全文書の 9 割以上が Edu/Format スコア

3 に集中するため，分類器の出力確率から計算した期待値を正規化した値 $q_{\text{edu}} = \mathbb{E}[\text{Edu}]/5$ を用いてサンプリング確率を制御した[2]．

- **JCC-Random**: ランダムサンプリング
- **JCC-EduBias**: q_{edu} をビン分割し，高いビンほどサンプリング確率が大きくなるように段階的に重み付け（最大 10 倍）
- **JCC-EduMix24**: 学習トークンの約 24%を，q_{edu} の高いビンから貪欲に収集し，残りはそれ以外のビンから分布比を保ってランダムに収集
- **JCC-EduTop**: q_{edu} の高い文書から貪欲に収集

高 Edu スコア文書のアップサンプリング　FineWeb2 の報告を踏まえ，「高品質データであれば，重複[3]を導入しても性能が維持または向上し得る」という仮説を検証した．

- **JCC-Top2ep**: q_{edu} 上位データから約 14.5B tokens 相当を構成し，2epoch 学習
- **JCC-UpDup24**: 学習トークンの約 24%を q_{edu} 上位データから最大 8 回の重みづけをした重複を意図的に導入し，残りは分布比を保ってランダムに収集

難易度の分布の操作　図 2 に示す通り Edu スコアが高い文書ほど難易度が高い傾向が観察されたことから，文書の難易度を品質指標としたキュレーションの効果を検証した．

- **JCC-ReadableTop**: 難易度レベルの低い（読みやすい）順に貪欲に収集
- **JCC-ReadL4Rand**: 難易度レベル 4 の文書集合からランダムに収集

3.3 評価

Penedo ら [6] による early-signal task の定義に基づき選定された日本語ベンチマーク[4]を用いた．

タスクタイプ　表 2 に示す通り，一般知識（GK），読解（RC），常識推論（CR），自然言語理解（NLU）の 4 タイプを利用した．特に GK タイプは比較条件に応じて性能傾向が異なるため詳細に分析する．

タスク設定　事前学習初期段階のモデルを多肢選択タスクで評価する際，選択肢記号を補完させる Multi-Choice Formulation（MCF）方式では結果が不安定になることが報告されている [23]．そこで本研究では，選択肢テキストを補完させる Cloze Formulation（CF）方式を採用した．

表 2　評価に使用したベンチマーク

ベンチマーク名	タスクタイプ	メトリック
JMMLU [19]	GK	Acc (PMI)
Belebele [20]	RC	Acc (Token)
CommonSenseQA [21]	CR	Acc (Token)
X-Winograd [22]	NLU	Acc (Token)
JSQuAD [21]	RC	F1

メトリック　先行研究 [6, 23] に従い，JMMLU では選択肢テキストの条件付き尤度を非条件付き尤度で補正する PMI-based accuracy（付録；C.1.2）を用いた．それ以外の多肢選択タスクでは，選択肢テキストの token-normalized 対数尤度に基づく accuracy（付録；C.1.1）を用いた．JSQuAD では生成回答と正解の語[5]重複に基づく F1 スコアで評価した．

aggregate score　複数ベンチマークの結果を統合するため，各ベンチマークのスコアを式 1 で変換し，その平均を aggregate score[25, 26] とした．

$$\text{new\_score} = \max\left(0, \frac{\text{score} - \text{random\_baseline}}{1 - \text{random\_baseline}}\right) \quad (1)$$

4 結果

どの設定がベストであるか？　図 3 に，各実験条件の最終チェックポイントにおける aggregate score を示す．全タスク総合での aggregate score は JCC-Random が最も高い結果となり，Edu スコアや難易度の分布を極端に操作した条件はいずれも JCC-Random の性能を上回らなかった．

一方で，GK タスクにおいては JCC-EduTop の性能が最も高い結果となった．

Edu スコアに基づく品質構成の操作は性能向上に有効か？　Edu スコアの分布を上位側に寄せるキュレーションは全体的な性能向上に寄与しない．一方で，JCC は構築時に Edu スコアが 2 以下のデータが削除されている．JCC-Random と llmjpv4-L0 の性能（図 3）および品質分布（図 4）の比較から，Edu スコアが極端に低い文書の除去は有効であると言える．一方，GK タスクでは Edu スコアが高い文書の比率が増えるほど性能向上する傾向が観察された[6]．

2）　Format スコアは正規化後も分布の集中が強いため，サンプリング基準には用いなかった．
3）　本研究では，完全に同一な文書が複数回サンプリングされる設定を「重複導入」と呼ぶ．
4）　x-codah，x-csqa [18] は Penedo ら [6] が使用したベンチマークであるが，翻訳品質が低いため本研究では不採用とした．
5）　単語分割には MeCab [24] を用いた．
6）　参考：JCC-EduBias，JCC-EduMix24，JCC-EduTop における $q_{\text{edu}} > 0.6$ ビンの割合はそれぞれ 0.24，0.31，0.95 である．

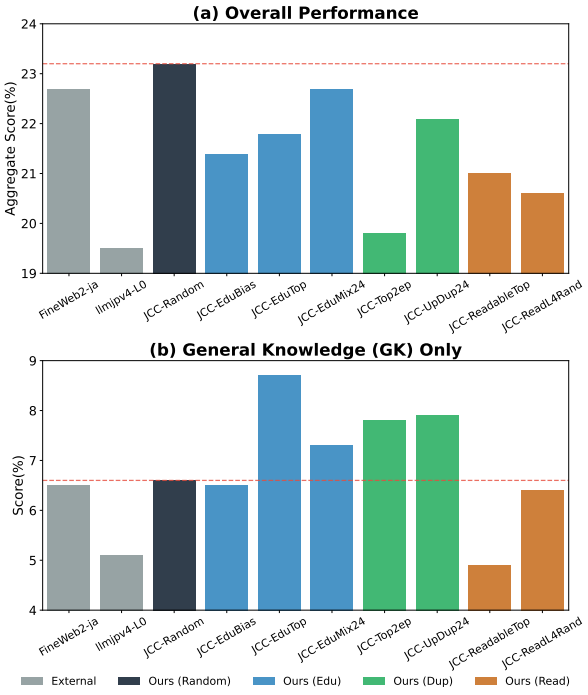

図 3　最終チェックポイントにおける性能

高品質文書のアップサンプリングは有効か？　本研究の設定においては，Edu スコアに基づく高品質文書の重複導入は全体的な性能向上に寄与しない．

JCC-Random と比較して，同一文書の再出現を許容した全ての条件で aggregate score が低下した．さらに，学習トークンの 24% を q_{edu} 上位ビン由来のトークンで構成した条件同士を比べると，重複なしの JCC-EduMix24 に対して，重複ありの JCC-UpDup24 は性能が劣った．これは，q_{edu} 分布の変化とは独立に，完全一致重複の導入が性能劣化に繋がる可能性を示唆する．

GK タスクのみに注目すると，重複なしの JCC-EduMix24 に対して，重複ありの JCC-UpDup24 は性能が高い．さらに，JCC-Top2ep は 2 epoch 相当の重複を含むにも関わらず，JCC-UpDup24 との性能差は小さかった．これは，GK タスク性能は重複の有無よりも Edu スコアの高いデータ比率が支配的である可能性を示す．

FineWeb2 [6] と知見の整合性はあるか？　本研究で用いた FineWeb2-ja では，MinHash 基準の近似重複は一定量観測されたが，完全一致重複は確認されなかった．したがって，FineWeb2 におけるアップサンプリングの効果は，高品質な同一文書の反復学習によるものではなく，近似重複を含む形で高品質文書の露出確率を高めたことによる分布変化に起因すると推測される．

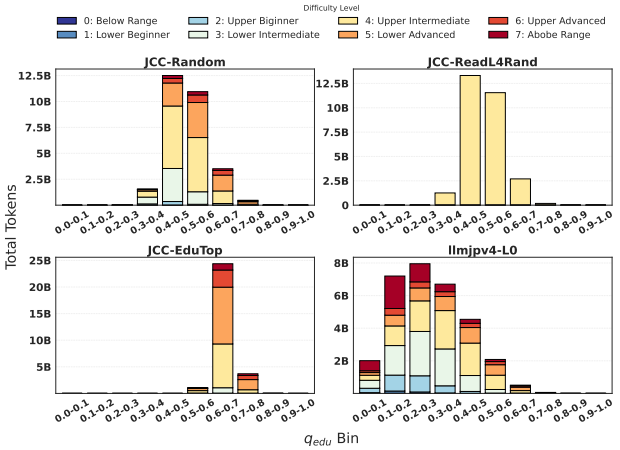

図 4　実験条件ごとの品質分布．q_{edu} ビンごとの難易度構成比を示す．その他の条件については付録；図 5 を参照．

難易度に基づく品質構成の操作は性能向上に有効か？　難易度分布を操作した条件はいずれも JCC-Random の性能を上回らず，全タスクで性能が低下した．特に，難易度の低い分布に偏らせると q_{edu} 分布も低スコア側に寄り，GK タスク性能が大きく低下した．なお，難易度をレベル 4 に固定した JCC-ReadL4Rand は，q_{edu} 分布が JCC-Random とほぼ同一であるが（図 4），全タスクで性能が低下した．

分布偏りのドメイン分布への影響　各条件の URL ドメイン分布を比較すると，高 Edu スコア偏重は学術系テキストへの集中を招き，これが GK タスク以外での性能低下の一因と考えられる．難易度固定が生む偏りは比較的小さいものの，特定のサイトが減少する傾向が見られた．単一指標への過度な依存は，LLM の学習に有用な情報の損失に繋がることが示唆される．

5　おわりに

本研究では日本語 LLM の事前学習データ構築における知見を示した．

本研究の重要性　Web コーパスのキュレーション方法に関する研究は英語中心に議論されており，日本語では未だ十分な検証がなされていない．本研究は，大規模日本語 Web コーパスの構築から取り組み，教育的価値・文書難易度に基づくデータ品質および構成操作の効果と限界を評価した．主に，教育的スコアと一般知識タスク性能の強い関連と，単一の指標に基づく極端なキュレーションが総合的な下流タスク性能向上にとって悪影響を与える可能性に関する知見を示した．本成果は，日本語を重視した大規模言語モデル開発の発展に貢献する．

参考文献

[1] Guilherme Penedo, Hynek Kydlícek, Loubna Ben Allal, Anton Lozhkov, Margaret Mitchell, Colin A. Raffel, Leandro von Werra, and Thomas Wolf. The fineweb datasets: Decanting the web for the finest text data at scale. In **NeurIPS**, 2024.

[2] Mohammad Amin Ghanizadeh and Mohammad Javad Dousti. Towards data-efficient language models: A child-inspired approach to language learning. **arXiv preprint arXiv:2503.04611**, 2025.

[3] Hajar Emami-Gohari, Swanand Ravindra Kadhe, et al. Gneissweb: Preparing high quality data for llms at scale. **arXiv preprint arXiv:2502.14907**, 2025.

[4] Richard L. Mueller. EFALW readability score. https://www.rlmueller.net/Readability.htm, 2012.

[5] Rachel McAlpine. From plain english to global english. https://www.angelfire.com/nd/nirmaldasan/journalismonline/fpetge.html, 2006.

[6] Guilherme Penedo, Hynek Kydlícek, Vinko Sabolcec, Bettina Messmer, Negar Foroutan, Amir Hossein Kargaran, Colin Raffel, Martin Jaggi, Leandro von Werra, and Thomas Wolf. Fineweb2: One pipeline to scale them all - adapting pre-training data processing to every language. **arXiv preprint arXiv:2506.20920**, 2025.

[7] Andrei Z. Broder. On the resemblance and containment of documents. In **SEQUENCES**, pp. 21–29. IEEE, 1997.

[8] Katherine Lee, Daphne Ippolito, Andrew Nystrom, Chiyuan Zhang, Douglas Eck, Chris Callison-Burch, and Nicholas Carlini. Deduplicating training data makes language models better. In **ACL**, pp. 8424–8445, 2022.

[9] Naoaki Okazaki, Kakeru Hattori, Hirai Shota, Hiroki Iida, Masanari Ohi, Kazuki Fujii, Taishi Nakamura, Mengsay Loem, Rio Yokota, and Sakae Mizuki. Building a large japanese web corpus for large language models. **arXiv preprint arXiv:2404.17733**, 2024.

[10] Adrien Barbaresi. Trafilatura: A web scraping library and command-line tool for text discovery and extraction. In **ACL (demo)**, pp. 122–131. Association for Computational Linguistics, 2021.

[11] Leonard Richardson. Beautiful soup 4 documentation. https://www.crummy.com/software/BeautifulSoup/bs4/doc/. Accessed: 2025-12-23.

[12] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomás Mikolov. Enriching word vectors with subword information. **Trans. Assoc. Comput. Linguistics**, Vol. 5, pp. 135–146, 2017.

[13] Gemma Team. Gemma: Open models based on gemini research and technology. **arXiv preprint arXiv:2403.08295**, 2024.

[14] Jaeho Lee. 日本語教育のための文章難易度に関する研究. 早稲田日本語教育学, No. 21, pp. 1–16, 2016.

[15] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models. **arXiv preprint arXiv:2302.13971**, 2023.

[16] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. Training compute-optimal large language models. **arXiv preprint arXiv:2203.15556**, 2022.

[17] LLM-jp. LLM-jp Corpus v4. https://gitlab.llm-jp.nii.ac.jp/datasets/llm-jp-corpus-v4, optnote = Accessed: 2025-12-24, 2025.

[18] Bill Yuchen Lin, Seyeon Lee, Xiaoyang Qiao, and Xiang Ren. Common sense beyond english: Evaluating and improving multilingual language models for commonsense reasoning. In **ACL/JJCNLP**, pp. 1274–1287, 2021.

[19] Ziqi Yin, Hao Wang, Kaito Horio, Daisuke Kawahara, and Satoshi Sekine. Should we respect LLMs? a cross-lingual study on the influence of prompt politeness on LLM performance. In **SICon**, pp. 9–35, 2024.

[20] Lucas Bandarkar, Davis Liang, Benjamin Muller, Mikel Artetxe, Satya Narayan Shukla, Donald Husa, Naman Goyal, Abhinandan Krishnan, Luke Zettlemoyer, and Madian Khabsa. The belebele benchmark: a parallel reading comprehension dataset in 122 language variants. In **ACL**, pp. 749–775, 2024.

[21] Kentaro Kurihara, Daisuke Kawahara, and Tomohide Shibata. JGLUE: japanese general language understanding evaluation. In **LREC**, pp. 2957–2966, 2022.

[22] Niklas Muennighoff, Thomas Wang, Lintang Sutawika, Adam Roberts, Stella Biderman, Teven Le Scao, M. Saiful Bari, Sheng Shen, Zheng Xin Yong, Hailey Schoelkopf, Xiangru Tang, Dragomir Radev, Alham Fikri Aji, Khalid Almubarak, Samuel Albanie, Zaid Alyafeai, Albert Webson, Edward Raff, and Colin Raffel. Crosslingual generalization through multitask finetuning. In **ACL**, pp. 15991–16111, 2023.

[23] Yuling Gu, Oyvind Tafjord, Bailey Kuehl, Dany Haddad, Jesse Dodge, and Hannaneh Hajishirzi. OLMES: A standard for language model evaluations. In **NAACL (Findings)**, pp. 5005–5033, 2025.

[24] Taku Kudo, Kaoru Yamamoto, and Yuji Matsumoto. Applying conditional random fields to japanese morphological analysis. In **EMNLP**, pp. 230–237, 2004.

[25] Clementine Fourrier, Nathan Habib, Alina Lozovskaya, Konrad Szafer, and Thomas Wolf. Open llm leaderboard v2. https://huggingface.co/spaces/open-llm-leaderboard/, 2024.

[26] Jeffrey Li, Alex Fang, et al. Datacomp-lm: In search of the next generation of training sets for language models. In **NeurIPS**, 2024.

[27] Fan Zhou, Zengzhi Wang, Qian Liu, Junlong Li, and Pengfei Liu. Programming every example: Lifting pre-training data quality like experts at scale. In **ICML**, 2025.

[28] Bo Adler, Niket Agarwal, et al. Nemotron-4 340b technical report. **arXiv preprint arXiv:2406.11704**, 2024.

A　ML モデル

以下 3 つのラベルを付与するため，fastText 分類器を訓練した．

- **Edu スコア**: 「学生の学習に有効な文章は LLM 学習にも有効」という仮定の下，FineWeb[1] を参考に付与．
- **Format スコア**: 文章の書式・体裁の品質を，Zhou ら [27] を参考に付与．
- **アダルトカテゴリ**: 判定結果を付与．

A.1　訓練データの作成

図 1 の手順 6 まで完了した文書からランダムに約 40 万件を抽出し，Nemotron-4-340B-Instruct-hf-FP8 [28] に適用してアノテーションを実施した．

作成したデータを 学習: テスト=4:1 に分割し，3 つの fastText 分類器を学習した．各モデルの分類精度を表 3 に示す．

表 3　fastText 分類器の設定と性能

モデル	ラベル	macro F1
Edu スコア推定	0–5（6 クラス）	0.75
Format スコア推定	0–5（6 クラス）	0.84
カテゴリ推定	4 クラス	0.82

B　学習設定

実験時に利用した LLM の学習設定を表 4 に示す．

表 4　モデルおよび学習設定

Parameter	Value
モデル	
Architecture	Llama (1.46B)
Hidden Layers	14
Dimentions	2048
Attention Heads	32 (KV: 32)
Tokenizer	Gemma 7B
学習設定	
Sequence Length	2048
Global Batch Size	1024
Optimizer	AdamW ($\beta_1 = 0.9, \beta_2 = 0.95$)
Weight Decay	0.1
Learning Rate	3.0×10^{-4} (min: 3.0×10^{-5})
LR Schedule	Cosine Decay (Warmup 500)

C　評価

C.1　設定

多肢選択問題データセットを

$$\mathcal{D} = \{(S_n, q_n, \{o_{n,k}\}_{k=1}^{K}, a_n)\}_{n=1}^{N}$$

とする．ここで n は設問番号，k は選択肢番号，S_n は文脈，q_n は設問文，$o_{n,k}$ は n 番目の設問に対する k 番目の選択肢テキスト，a_n は正解選択肢のインデックスを表す．

C.1.1　Token-normalized accuracy

以下のテンプレートにより文脈文字列 c_n を構築する：

$$c_n = S_n \oplus \texttt{"\textbackslash n\textbackslash n"} \oplus \text{“問題: ”} \oplus q_n \oplus \texttt{"\textbackslash n"} \oplus \text{“回答: ”} \quad (2)$$

選択肢 $o_{n,k}$ のトークン列を $t = (t_1, \ldots, t_L)$ として，c_n を条件としたトークン生成の対数尤度をトークン数で正規化したスコアを

$$s_{n,k} = \frac{1}{L} \sum_{i=1}^{L} \log P_\theta(t_i \mid c_n, t_{<i}) \quad (3)$$

と定義し，予測ラベルを以下で定める．

$$\hat{a}_n = \underset{k \in \{1,\ldots,K\}}{\operatorname{argmax}} \; s_{n,k} \quad (4)$$

C.1.2　PMI-based accuracy

PMI に基づく補正のため，以下のテンプレートにより条件付き文脈 $c_n^{(c)}$ と非条件付き文脈 $c^{(u)}$ を構築する：

$$c_n^{(c)} = \text{“問題: ”} \oplus q_n \oplus \texttt{"\textbackslash n"} \oplus \text{“回答: ”}, \quad c^{(u)} = \text{“回答: ”} \quad (5)$$

選択肢 $o_{n,k}$ の PMI スコアを

$$s_{n,k} = \log P_\theta(o_{n,k} \mid c_n^{(c)}) - \log P_\theta(o_{n,k} \mid c^{(u)}) \quad (6)$$

と定義し，予測ラベルを以下で定める．

$$\hat{a}_n = \arg \max_{k \in \{1,\ldots,K\}} s_{n,k} \quad (7)$$

D　実験データの品質分布

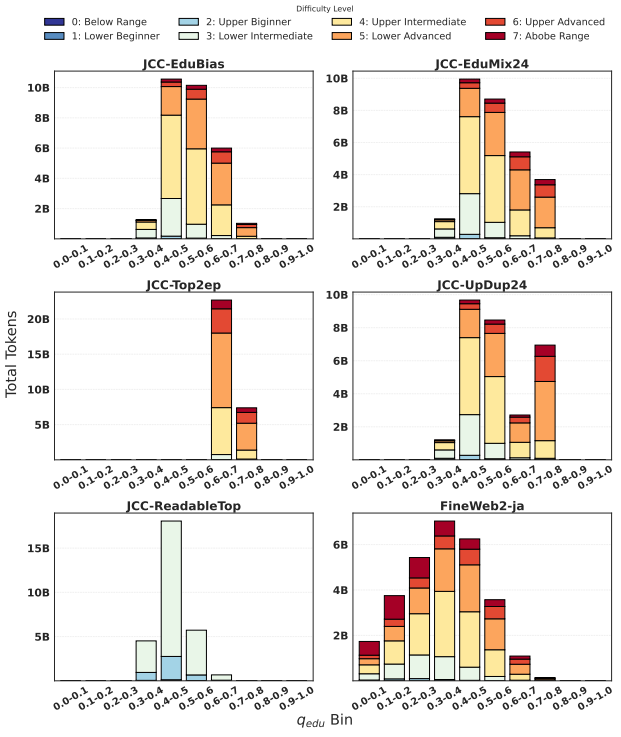

図 5　実験条件ごとの品質分布．q_{edu} ビンごとの難易度構成比を示す．