

報告書

1 今週の進捗

- wikipedia データの前処理 (不必要な情報の削除)
- llm-jp-1.8b-instruct4 の継続事前学習 (CPT) および教師あり学習 (SFT)

2 wikipedia データの前処理 (不必要な情報の削除)

Wikipedia から収集したアニメ関連記事を継続事前学習に適したコーパスへ変換するため、下記の前処理をした。

1. **Markdown / MediaWiki 構文の除去**見出し (#), 強調記号 (*, **, "), HTML コメント, 定義リスト記法 (:, ::), 箇条書きに由来する不要記号 (・, *, -) などの構造情報を削除した。
2. **wikilink の除去** [ラベル](リンク "wikilink") 形式を認識し, ラベルのみを残してリンク情報はすべて削除した。また, (～ "wikilink") のような残存要素も正規表現により除去した。
3. **脚注および番号記述の削除** [3], \[4\] などの脚注番号, ならびに 1. や 2. のような脚注一覧由来の行を除去し, 文章内容のみを残した。
4. **改行に挟まれた孤立ハイフンの正規化**行間に単独で置かれた “-” を検出し, 前後の行を「A - B」の形式に連結することで, 本来一文であったテキストを復元した。
5. **リンクエスケープの解除** Markdown に由来する \(<, \), \!, * などのエスケープ文字を正規化し, 通常の文字に戻した。
6. **ノイズセクションの除去**「脚注」「注釈」「出典」「参考文献」等のセクションは事前学習上の有用性が低いため丸ごと削除した。
7. **Unicode 正規化**テキスト全体に対して NFKC 正規化を行い, 表記揺れを抑制した。
8. **空行および不要スペースの整理**連続する改行を適宜圧縮し, 行末の不要スペースを削除することで, モデル入力に適したテキスト構造へ整形した。
9. **一覧ページの除外**「作品一覧」「アニメ一覧」「話数一覧」「OVA 一覧」「放送一覧」「ニュース一覧」など, 記述内容が列举中心で文脈情報が乏しいページは学習データから除外し, 別フォルダに退避させた。ただし「登場人物一覧」は作品理解に重要であるため保持した。

以上の前処理により, 記号類・構造情報・辞書的列举などのノイズを取り除き, 自然文として有用な内容のみを残した高品質なコーパスを構築した。

3 llm-jp-1.8b-instruct4 の継続事前学習 (CPT) および教師あり学習 (SFT)

llm-jp-1.8b-instruct4 に対して, 素のモデル, CPT のみ, SFT のみ, CPT + SFT の 4 つで推論し, 比較した。

3.1 訓練データ

CPT の訓練データは, Wikipedia から集めたアニメに関する記事の約 18,000 件に対して前処理を施したマークダウンファイルである.

また, SFT の訓練データは, 以前作成したアニメに関する質問と回答のペアの QA データセットである. 表 1 に例を示す.

表 1: 訓練データセットの例

質問: ぽにぽにだっしゅ! の主人公は誰ですか?	出力: ぽにぽにだっしゅ! の主人公はレベッカ宮本です。
質問: ぽにぽにだっしゅ! の主人公は誰ですか?	出力: ぽにぽにだっしゅ! の主人公は、10 歳で MIT を卒業した天才少女、レベッカ宮本です。
質問: 「ぽにぽにだっしゅ!」で主人公として描かれているキャラクターは誰ですか?	出力: ぽにぽにだっしゅ! の主人公はレベッカ宮本です。
質問: 「ぽにぽにだっしゅ!」で主人公として描かれているキャラクターは誰ですか?	出力: ぽにぽにだっしゅ! の主人公は、10 歳で MIT を卒業した天才少女、レベッカ宮本です。
質問: レベッカ宮本が主人公を務める作品は何ですか?	出力: レベッカ宮本が主人公を務める作品は「ぽにぽにだっしゅ!」です。
質問: レベッカ宮本が主人公を務める作品は何ですか?	出力: 彼女が主人公の作品は「ぽにぽにだっしゅ!」です。この作品で彼女は天才少女教師として描かれています。
質問: レベッカ宮本は、どの作品で主人公として登場しますか?	出力: レベッカ宮本が主人公を務める作品は「ぽにぽにだっしゅ!」です。
質問: レベッカ宮本は、どの作品で主人公として登場しますか?	出力: 彼女が主人公の作品は「ぽにぽにだっしゅ!」です。この作品で彼女は天才少女教師として描かれています。

3.2 学習手法と対象パラメータ

CPT では, パラメータ効率の良いファインチューニング手法である Low-Rank Adaptation (LoRA) を採用した. LoRA は, 元のモデルの重みを凍結したまま, 特定の線形層に低ランク行列分解を利用したアダプター層を追加し, そのアダプターの重みのみを学習する. これにより, 少ない計算リソースで大規模モデルの挙動を適応させることが可能となる.

また, SFT では LoRA を用いず, フルモデルを学習している. CPT + SFT については CPT の LoRA をマージした後, SFT をしている.

本研究では, モデルの挙動に大きく影響を与える以下の層を学習対象 (target_modules) として設定した.

- **q_proj, k_proj, v_proj, o_proj:** これらは Self-Attention 機構を構成する主要な線形層である. Query, Key, Value, Output の射影をそれぞれ担当し, 文中のどの情報に注目するかという注意パターンを学習する.
- **gate_proj, up_proj, down_proj:** これらは MLP ブロック (本モデルでは MoE の各 Expert 層) を構成する線形層である. Attention からの出力を処理し, モデルが持つ知識や表現力を更新する役割を担う.

3.3 実験設定と結果

3.3.1 学習パラメータ

表 4 に CPT の主要なハイパーパラメータを示す.

表 3: CPT の学習ハイパーパラメータ

パラメータ	値
モデル	llm-jp/llm-jp-3.1-1.8b-instruct4
LoRA ランク (r)	16
LoRA アルファ (alpha)	32
LoRA ドロップアウト率	0.05
バッチサイズ (per device)	2
勾配蓄積ステップ	2
エポック数	1
学習率	2e-5
オプティマイザ	AdamW (8-bit)

表 4 に SFT の主要なハイパーパラメータを示す.

表 4: SFT の学習ハイパーパラメータ

パラメータ	値
モデル	llm-jp/llm-jp-3.1-1.8b-instruct4
バッチサイズ (per device)	2
勾配蓄積ステップ	8
エポック数	3
学習率	2e-5
オプティマイザ	AdamW (8-bit)

4 今後の課題

- 訓練データの前処理の調整
- 他のモデルでの試行

参考文献