大規模マルチモーダルモデルにおけるビジョンエンコーダーの付け替えと、日本語に強いモデル作成へ向けて

佐藤 諒、木下 彰、中田 乙一、金箱 裕介、 麻場 直喜
株式会社リコー
{ryo.sato4, akira.kinoshita, otoichi.nakata, yuusuke.kanebako,
naoki.asaba}@jp.ricoh.com

概要

本稿では画像とテキストを入力として文生成が可能なタイプの日本語向け大規模マルチモーダルモデル (LMM)開発について報告する。今回採用した開発手順は初めに LMM の構成要素であるモデルパーツの付け変えを行った後に追加で訓練を行う形態をとっている。この開発過程で行ったモデルの新しい付け替え(結合)方法やその手順、そこからさらに日本語向けにデータを用意し、追加訓練を行った内容について説明する。作成した LMM に対して日本語向けベンチマークで評価し、モデル精度とその変化についても公開する。

1 はじめに

近年、言語処理用の性能の高い大規模言語モデル(LLM)と画像処理用の Vision Transformer[1]や CLIP[2]系統のモデルを Vision Encoder として接続し、画像と言語に対応する大規模マルチモーダルモデル(LMM)の開発が盛んである[3]。しかしながら、作成されたモデルの多くは英語向けになっており、また訓練に使われるデータセットも英語と比べると日本語のものは少なく、日本語向けに特化した LMM の数は多くない。そのため、本研究では、メインは中国語向けではあるものの、精度の高い LMM である Qwen2-VL-7B-Instruct モデル[4]を利用し、日本語向に強い LLM である Llama3.1-Swallow-8B-v0.2 モデル[5]を結合して LMM 作成を行う。この手段を用いることで、Qwen2-VL と比べて日本語による安定生成が可能になり、また Qwen2-VL という LMM で採用した機能を持つモデルを作成することができる。この訓練前のモデル(結合モデル)作成の際には二つの別々に作られたモデルを融合するための手段が必要である。また、この結合モデルを作成した後は、日本語に強いモデルに仕上げるために、日本語データを追加で作成し、訓練する必要がある。その後、できたモデルに対して評価を行う。これらの課程で行ったことについて本稿では節を分けて順番に説明する。

2 節では結合モデルの構築方法について説明する。この結合に使われた元のモデルである Qwen2-VL、Llama3.1-Swallow は Appendix A で簡単に説明する。3 節 では、訓練に使用したデータセットの準備と学習過程について説明し、4 節では作成したモデルの精度とその考察について説明を行う。この一連の流れを図示したものが図 1 である。

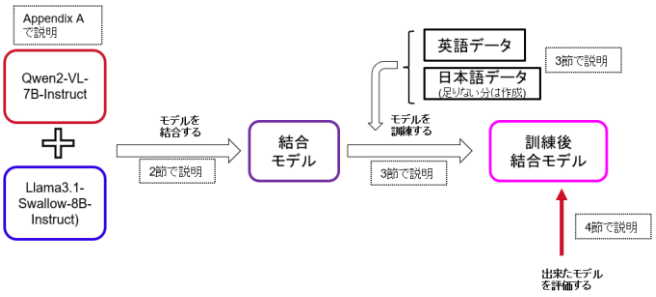

図 1 本稿の流れ

2 結合モデルの構築方法

この節では、Qwen2-VL モデルと Llama3.1-Swallow モデルを結合する方法について説明する。結合手順については図 2 に概略を図示してある。Vision Encoder 部には Qwen2-VL-7B-Instruct のモデルの

This work is licensed by the author(s) under CC BY 4.0
(https://creativecommons.org/licenses/by/4.0/).

Vision transformer 部の重みを使い(重みをロードし)、LLM 部には Llama3.1-Swallow-8B-Instruct の重みを使う。Adapter 層部には Qwen2-VL-7B-Instruct の Adapter 層部の重みの一部を使用する。これらの構成で今回作成するモデルを Lwen2-VL と呼ぶ。

この節の構成は 2.1 LLM 部の扱い、2.2 Vision Encoder 部の扱い、2.3 Adapter 層の扱いの順になっており、結合モデルの作成の手順を順に説明する。これらの手順を取ることで、Adapter 層を一からスクラッチで用意して訓練し、調整するのと比べて、Qwen2-VL という LMM 化の工夫を取り入れることが可能となり、訓練前の初期モデルをより精度良い状態で準備することができると期待される。

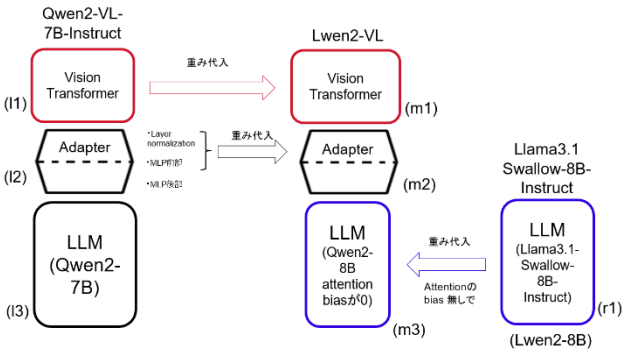

図 2 結合 LMM モデル設計

2.1 LLM 部の扱い

LLM として Qwen2 と Llama3.1-Swallow のアーキテクチャには類似点が多い[Appendix A]。そのため、Qwen2-VL の LLM 部を Llama3.1-Swallow にするための前段階として、Qwen2 のアーキテクチャ範囲で Llama3.1-Swallow のモデルの重みをロードする。これらのアーキテクチャには以下のような大きな違いが存在する。

・attention 部の bias 項の違い(式(A1),式(A2))

・位置 embedding の違い

これらの違いを踏まえた上で Qwen2 のアーキテクチャで Llama3.1-Swallow のモデルの重みをロードする。

最初に Qwen2 のアーキテクチャで 8B のスクラッチモデルを用意する。この状態で attention 部の bias 項を除いて Llama3.1-Swallow-8B-Instruct の重みをロードする。重みをロードしたのちに、ランダムな値になっている attention 部の bias 項に 0 ベクトルを代入する。こうすることで位置 embedding が異なるが、LLM として Llama3.1-Swallow-8B-Instruct の重みを持つ Qwen2-8B モデルができる[図 2 の(r1)]。この LLM を Lwen2-8B と呼ぶ。この Lwen2-8B は位置 embedding が異なるが故に多少の精度劣化はあるものの、通常のテキスト推論を行う上で問題がない。また、後ほど追加訓練を行うため、位置 embedding 変更により必要となるキャリブレーション効果はそこで吸収する。

さらにこの作成した Lwen2-8B を Qwen2-VL アーキテクチャで使用するために、Vision Transformer は Qwen2-VL-7B-Instruct のアーキテクチャで、LLM 部は Lwen2-8B と同形のアーキテクチャで尚且つ Adapter 層は Lwen2-8B の hidden size に合わせた状態の Qwen2-VL-8B のモデルをスクラッチで用意し、そのモデルの LLM 部に Lwen2-8B のモデルの重みを代入する[図 2 の(r1)から(m3)]。このモデルを Lwen2-VL-8B と呼ぶ。このようにモデルを準備することで、Qwen2-VL の LMM 用の位置 embedding をそのまま使うことができ、またモデルコードのアップデートに伴う変更に対応しやすいという実用上のメリットがある。

2.2 Vision Encoder 部の扱い

Vision Encoder 部は性能の良い Qwen2-VL-7B-Instruct のものを使う。2.1 で用意した Lwen2-VL-8B は Vision Transformer のアーキテクチャが、Qwen2-VL-7B-Instruct と同形であるため、そのままモデルの重みをロードすることが可能である。[図 2 の(l1)から(m1)]

This work is licensed by the author(s) under CC BY 4.0
(https://creativecommons.org/licenses/by/4.0/).

2.3 Adapter 層の扱い

Adapter 層[図 2 の(12)]は Layer normalization 層、MLP 層前部、MLP 層後部で構成されているが、MLP 層後部以外は Lwen2-VL-8B と Qwen2-VL-7B-Instruct が同形である。そのため、同形部分についてはそのまま Qwen2-VL-7B-Instruct のモデルの重みをロードする[図 2 の(12)から(m2)]。Adapter 層内にある MLP 層後部は LLM の hidden size に合わせた変換が入る。この層による変換は入力ベクトルを$\boldsymbol{x}_{\mathrm{mlp}}$, 出力ベクトル$\boldsymbol{y}_{\mathrm{mlp}}$としたときに式(3)であらわされる。

$$\boldsymbol{y}_{\mathrm{mlp}} = W_{\mathrm{mlp}}\boldsymbol{x}_{\mathrm{mlp}} + \boldsymbol{b}_{\mathrm{mlp}} \quad (3)$$

W_{mlp}の次元：$h_{\mathrm{LLM}} \times h_{\mathrm{LMM}}$

$\boldsymbol{x}_{\mathrm{mlp}}$の次元: h_{LMM}

$\boldsymbol{y}_{\mathrm{mlp}}$の次元: h_{LLM}

ここでh_{LLM}は LLM の、h_{LMM}は LMM の hidden size である。h_{LLM}については Qwen2-VL-7B-Instruct と Lwen2-VL-8B で異なっており、ここの変換は LLM 内 tokenizer の id と紐づいてテキストトークンのベクトル表現を作る Token Embedded 層 の hidden size に一致させるものとなっている。この Token Embedded 層の状況に合わせて、MLP 層後部はスクラッチ状態よりなるべく初期値をよくした状態で用意したい。その準備計算は Appendix B に載せてある。

3 データセット準備と学習課程

2 節で用意した LMM である Lwen2-VL-8B を使い、各課程ごとに学習データと学習対象パラメータを選び訓練を行う[表 1]。

表 1 学習データセットと訓練対象パラメーター一覧

学習データ	データ説明	学習対象パラメータ
Chart QA データ[6]	グラフ画像に対する質問に yes/no で回答する。	Adapter 層
画像説明タスクデータ[7]	物体画像を詳細に説明する。	Adapter 層+LLM の query, key, value
OCR データ[9][10]	画像内のテキストに対して OCR する。	Adapter 層+LLM の query, key, value, MLP 層
グラフ質問回答タスクデータ[11]	グラフを読み取り回答する。	LMM 全体(Vision Transformer + Adapter 層 + LLM)

3.1 Adapter 層の調整の学習データ

まずは最初の学習課程として公開データセットである Chart QA データ[6]を利用し Adapter 層の学習を行う。Adapter 層は Vision Encoder と LLM をつなぐ役割があり、Vision Encoder から LLM に送られる image トークンベクトルがテキストトークンと同等のベクトル空間で表現されるよう調整する必要がある。そこで画像の概略が判断できるように yes または no の回答で簡単に学習できる Chart QA データセットを利用する。

3.2 Adapter 層＋LLM の学習データ(画像説明タスクデータ)

2 番目の学習課程として Adapter 層の学習のみならず、LLM の query, key, value までを訓練対象のパラメータとしつつ、画像説明タスクを学習する[7]。これにより、image トークンベクトルがより正確にテキストトークンと同じ空間内マッピングされることを狙い、また、画像の情報を受け取った状態で説明する能力を LLM に持たせることを狙う。このデータセットは、元は英語データセットであるが、Qwen2.5-32B-Instruct[8]で翻訳したものも同時に利用し、英語だけでなく日本語でも画像を説明可能な状態にする。

3.3 Adapter 層+LLM の学習データ(OCR データ)

3 番目の学習課程として節 3.2 の効果に加えて文字の認識能力を高めるために OCR データを用いて学習する[9] [10]。このデータは 2 種類あり、物体画

This work is licensed by the author(s) under CC BY 4.0
(https://creativecommons.org/licenses/by/4.0/).

像メインで中に文字が少量含まれるデータセット[9]と文字が多く含まれる文字主体のデータセット[10]を利用する。この課程でのLLMの学習対象のパラメータはquery, key, valueのみならず、MLP層まで拡大する。このように学習対象パラメータを拡大していく理由は元モデルであるQwen2-VLがパラメータ全体を学習対象としているのを参考としている[4]。

3.4 LMM全体の学習データ(グラフ質問回答タスクデータ)

最後の学習課程としてグラフ質問回答データ を学習し、より実践的なタスク向けにチューニングする。ここで使うデータは日本語で適したものを大量に必要とするためLLMを利用し作成した合成グラフデータで行う[11]。学習対象のパラメータはLMM全体とし、最終的な性能ができる限り高い状態になることを狙う。

4 訓練結果と考察

3節のLwen2-VL各課程におけるモデルを評価した結果と生成例を表2に示す。比較として日本語向けLMM である llava-calm2-siglip[12]と元モデルであるQwen2-VL-7B-Instructを載せてある[表2]。

表2 JDocQA ベンチマーク 評価比較

モデル	対応節	JDocQA スコア
Lwen2-VL-8B α	4.1	0.6015
Lwen2-VL-8B β	4.2	0.8421
Lwen2-VL-8B γ	4.3	0.3985
Lwen2-VL-8B δ	4.4	0.8346
Qwen2-VL-7B-Instruct	-	0.6692
llava-calm2-siglip	-	0.7970

ベンチマークとしてJDocQAのスコアを採用してある[13]。JDocQAはLMMの日本語向けベンチマークであり、中でも「はい/いいえ」で解答する形式のもの(JDocQA Binary)を選んだ。これにより、比較に使うLMMのファインチューニング状況にそれほどよらず解答しやすくなるよう配慮した。このJDocQAは一画像に多量の文章、項目、セクション図を含んでおり、LMMの能力測定において十分な難易度となっているはずである。

評価の結果、各訓練課程を経たLwen2-VLの持つスコアは各モデルに行った訓練の効果を反映したものになっている。途中OCR訓練をすること(モデル γ)で一旦精度が落ちているのは、OCR訓練により文字を出力するような回答形式になってしまっているためである。また、今回の学習期間の範囲で最終課程まで訓練したモデル δ は、llava-calm-siglip とQwen2-VL-7B-Instructの同程度または上回った精度まで到達している。精度比較にも利用した元モデルであるQwen2-VLはLMMの形態をとってから1.4Tもの多量の画像+テキストトークンの学習をしている。そのためLwen2-VLにおいてもこれよりも後続の訓練を追加することでさらなる精度の向上ができると期待できる。それは今後の課題とする。

5 おわりに

本論文では, 日本語向けの大規模マルチモーダルモデル(LMM)開発を行った。学習前の初期モデルはQwen2-VL-7B-Instructから Vision Encoderを採用し、LLMとしてはLlama3.1-Swallow-8B-Instructから採用し構成し、そのモデルの作成手順について説明した。訓練データはChart QAデータ、画像説明データ、OCRデータ、グラフ質問回答データという画像の認識から実際の画像処理タスクへ向けて学習し、それにより日本語でグラフ読み取りが可能なモデル開発が行えることを確かめた。

6 謝辞

この成果は、NEDO(国立研究開発法人新エネルギー・産業技術総合開発機構)の助成事業(JPNP20017)の結果得られたものです。

参考文献

[1] Alexey Dosovitskiy, et al, An image is worth 16X16 words: Transformers for image recognition at scale. In ICLR, 2021.

[2] Alec Radford, et al, Learning Transferable Visual Models From Natural Language

This work is licensed by the author(s) under CC BY 4.0
(https://creativecommons.org/licenses/by/4.0/).

Supervision, in Proceedings of the 38 th International Conference on Machine Learning, PMLR 139, 2021.

[3] Deyao Zhu, et al, MiniGPT-4: Enhancing Vision-Language Understanding with Advanced Large Language Models In ICLR, 2023.

[4] Peng Wang, et al, Qwen2-VL: Enhancing Vision-Language Model's Perception of the World at Any Resolution. arXiv preprint arXiv:2409.12191, 2024, Qwen/Qwen2-VL-7B-Instruct, https://huggingface.co/Qwen/Qwen2-VL-7B-Instruct

[5] Kazuki Fujii, et al, Continual Pre-Training for Cross-Lingual LLM Adaptation: Enhancing Japanese Language Capabilities, arXiv preprint arXiv:2404.17790, 2024, tokyotech-llm/Llama-3.1-Swallow-8B-Instruct-v0.2, https://huggingface.co/tokyotech-llm/Llama-3.1-Swallow-8B-v0.2

[6] HuggingFaceM4, multimodal team, 2024, https://huggingface.co/datasets/HuggingFaceM4/ChartQA/tree/main/data

[7] Aaron Gokaslan, et al, CommonCanvas: An Open Diffusion Model Trained with Creative-Commons Images, Computer Vision Foundation, 2024, https://huggingface.co/datasets/common-canvas/commoncatalog-cc-by

[8] Qwen Team, Qwen2.5 Technical Report, arXiv preprint arXiv:2412.15115, 2025, Qwen/Qwen2.5-32B-Instruct, https://huggingface.co/Qwen/Qwen2.5-32B-Instruct,

[9] Amanpreet Singh, et al, TextOCR: Towards large-scale end-to-end reasoning for arbitrary-shaped scene text, CVPR, 2021, https://textvqa.org/textocr/dataset/

[10] Kim, Geewook, et al, OCR-free Document Understanding Transformer, European Conference on Computer Vision (ECCV), 2022, naver-clova-ix/synthdog-en, https://huggingface.co/datasets/naver-clova-ix/synthdog-en, naver-clova-ix/synthdog-ja, https://huggingface.co/datasets/naver-clova-ix/synthdog-ja

[11] 合成データ作成の工程として 1.大規模言語モデルを活用して、図表(表、円、折れ線、棒、フローチャート等)を作成するための文字情報を生成する工程、2.生成した文字情報を画像化する工程、3.生成した文字情報を基に大規模言語モデルに質問を生成させる工程、を経て作成したもの.

[12] Aozora Inagaki, cyberagent/llava-calm2-siglip, 2024, https://huggingface.co/cyberagent/llava-calm2-siglip

[13] 大南英理, 栗田修平, 宮西大樹, 渡辺太郎, JDocQA: 図表を含む日本語文書質問応答データセットによる大規模言語モデルチューニング 第 30 回 年 次 大 会, 2024. https://www.anlp.jp/proceedings/annual_meeting/2024/pdf_dir/C3-5.pdf, jlli/JDocQA-binary, https://huggingface.co/datasets/jlli/JDocQA-binary

[14] Joshua et al, GQA: Training Generalized Multi-Query Transformer Models from Multi-Head Checkpoints, EMNLP, 2023.

[15] Jianlin Su, et al, RoFormer: Enhanced transformer with Rotary Position Embedding, Neurocomputing, 568, 1, 2024, 127063.

[16] Llama Team, AI @ Meta, The Llama 3 Herd of Models, https://www.llama.com/

[17] Liang Zhao, et al, Length Extrapolation of Transformers: A Survey from the Perspective of Positional Encoding, arXiv preprint arXiv:2312.17044, 2024.

This work is licensed by the author(s) under CC BY 4.0
(https://creativecommons.org/licenses/by/4.0/).

Appendix A 結合に使う元モデル

今回のモデルはLMMとして性能のいいQwen2-VL-7B-Instruct[4]と日本語向けに継続学習されたLlama3.1-Swallow-8B-v0.2[5]モデルを結合して作成する。そのためにこれら二つモデルについて簡単に説明する。

A.1 Qwen2-VLとQwen2

Qwen2-VL-7B-InstructモデルはQwen2-7Bモデルを LLM部分として持つ。Vision Encoder部分では32層構造のVision Transformerを持つ。またその二つを結合するadapter層が存在しており、Layer normalization層とMLP層前部、MLP層後部で構成される。このVision Transformer、Adapter層、LLMの三部構成はよく採用されているLMM形態である。この形態に加え、Qwen2-VLモデルではMultimodal-RoPEが位置embeddingとして採用されており、これにより、2次元である画像に対してはimageトークンの位置が縦横、それぞれ割り当てられ、より正確な位置情報を保持したまま計算が可能となっている。また、解像度については、画像サイズの上限、下限を設定しなければ、(2×14)^2 pixelのパッチサイズを1単位として、画像を分割し、その数分のimageトークンが割り振られ、テキストと同列のコンテキスト長内で処理される仕組みとなっている。LLM部であるQwen2-7Bに関しては、Llama3.1アーキテクチャと基本的に類似した構成になっている。Attention部については、Grouped Query attention[14]が採用されている。Attention部に入るベクトルを$\boldsymbol{x}$としたときに、query, key, valueベクトルに変換する式は式(A1)のようにbias項を持った状態になっている。

$$\boldsymbol{q} = W_q\boldsymbol{x} + \boldsymbol{b}_q \qquad \text{(A1)}$$

Qwen2-VLに入れ込む前のQwen2の位置embeddingに関してはRoPE[15]を採用している。

A.2 Llama3.1-Swallow

Meta社のLlama3.1モデル[16]を日本語向けに継続学習し、インストラクトチューニングを施したモデルであるLlama3.1-Swallow-8B-Instruct-v0.2 はAttention部にQwen2と同様にGrouped Query attentionが採用されている。入力ベクトル$\boldsymbol{x}$を変query, key, valueベクトルに変換する際には(2)式のようにbias項を伴わない変換になっている。

$$\boldsymbol{q} = W_q\boldsymbol{x} \qquad \text{(A2)}$$

位置embeddingに関しては”llama3”というNTK aware 補完[17]タイプのロングコンテキストに対応した手法が採用されている。

Appendix B LLM変更に伴うAdapter層の変換

Vision EncoderにたいしてLLMを付け替えたときにその大きさにあったAdapter層をスクラッチとして用意し、訓練するのが従来手法であるが、本研究ではそれとは異なり、Adapter層を可能な限り使いまわし、hidden sizeの違いは変換Layerを追加で挟むことによって対応する。この変換layerにより、Adapter層の形状の違いを吸収するよう設計する。変換Layerに関してはVision EncoderとLLMの組み合わせ変更前後のLLMのテキストembedding空間情報を利用して計算する。

Embedding空間の情報を利用して計算するための準備について説明する。まずVision Encoderの処理として、画像が入力として入り、それに対応したimageトークンが出力される。このimageトークンをLLMで意味のあるものとして処理できるように、Adapter層によって、テキストトークンのembedding基底に合うように変換される。このテキスト tokenのembeddingはLLMの保持する語彙の違いからLLMごとに異なっており、LLMごとに異なるAdapter層での変換を必要とする。そこでLLMを取り換えた際に異なる座標分の変換を変換Layerで担うという構成にする。この変換Layerは、取り換え前後のLLMのテキストトークンembedding基底をもとに計算される。取り換え前のモデルをA（ここではQwen2-7B)、取り換え後のモデルをB（ここではLlama3.1-Swallow-8B-Instruct）とし、Adapter Aの変換をR_{A}、Adapter Bの変換をR_{B}、変換レイヤーの変換をCとすると、AのLLMの基底($\boldsymbol{e}_{\mathrm{A}}$)とBのLLMの基底($\boldsymbol{e}_{\mathrm{B}}$)の関係は線形変換によって、

$$R_{\mathrm{B}} = C \times R_{\mathrm{A}} \qquad \text{(B1)}$$

$$\boldsymbol{e}_{\mathrm{B}} = C \times \boldsymbol{e}_{\mathrm{A}} \qquad \text{(B2)}$$

となる。ここで、$\boldsymbol{e}_{\mathrm{A}}$と$\boldsymbol{e}_{\mathrm{B}}$が分かれば、変換レイヤーの変換$C$を求めることができる。しかしながら、直接$\boldsymbol{e}_{\mathrm{A}}$と$\boldsymbol{e}_{\mathrm{B}}$を求めることは難しい。そのため、代わりとして同一の単語から作られるembeddingを変換前後のLLMのTokenizerとToken embedded層により作成し、その関係から、変換レイヤーの変換Cを求める。まず同一単語を入力単語としてTokenizerに入れ、対応する複数トークンのテキストembeddingを得る。次にこの複数のテキストembeddingを平均化して入力単語のembeddingを得る。この処理をモデルA、Bどちらに対しても行い、それから得られる同一単語の表現であるembedding間の関係からCを求めることができる。得られる入力単語のembeddingをT_{A}, T_{B}とすると、これらは

$$T_{\mathrm{A}} = C \times T_{\mathrm{B}} \qquad \text{(B3)}$$

の関係がある。Cを求めるために複数の単語に対してこの処理を回せば数値的に変換レイヤーの変換Cを求めることができる。

Qwen2-VL-7B-InstructのAdapterのMLP層後部と変換レイヤーは共に線形変換であることから、一層の線形変換で表現できるため、その一層化した線形変換をLwen2-VL-BのAdapter層のMLP層後部の初期値として採用する。この計算により考慮できるのはLLMのトークンの線形関係のみであり、Token embedded層に続くLLM内層まで浸透し、処理されるべき画像の特徴量を考慮した非線形効果は通常のLMM訓練課程で修正することとする。

This work is licensed by the author(s) under CC BY 4.0
(https://creativecommons.org/licenses/by/4.0/).