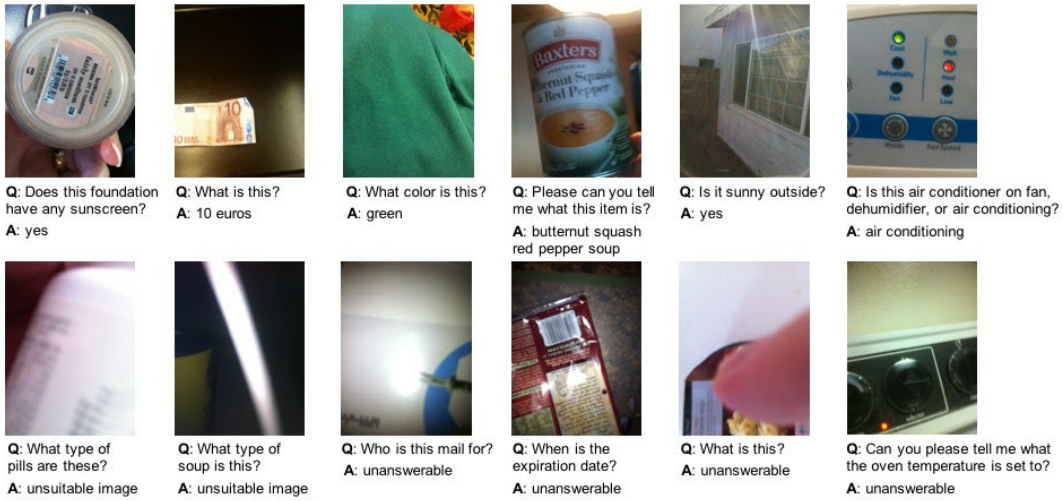
Vizwiz における VQA 予測モデルの検討

tom11111111

1. Vizwiz Dataset

- **視覚障がい者が撮影した画像であり、綺麗な画像データではない [1]**



→ 画像のデータ分布が特殊であるため、画像エンコーダーは別のデータセットで事前学習されたモデルが好ましい。

- **データセットの回答（最頻値）**

出現頻度の高い順に並べると 3 割の回答が unanswerable であり、その次に Yes/No の回答である。

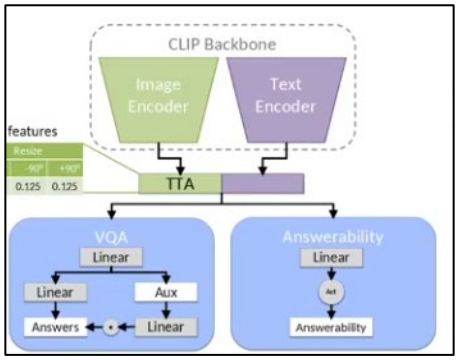
> [('unanswerable', **55613**), ('no', 5225), ('yes', 4337), ('white', 2511), ('grey', 2097), ('black', 2032), ('blue', 1716), ('red', 1087), ('brown', 787), ('pink', 748), ('green', 703), ('keyboard', 672), ('purple', 566), ('nothing', 516), ('soup', 507), ('dog', 479), ('laptop', 476), ('yellow', 425), ('ph1', 392), ('food', 352), ('tan', 352), ('lotion', 339), ('orange', 330), ('cell ph1', 281), ('chicken', 279), ('pepsi', 279), ('corn', 279), ('coffee', 277), ('0', 274), ('remote', 267), ('coca cola', 262), ('shampoo', 253), ('beans', 245), ('wine', 245), ('remote control', 241), ('computer', 234), ('can', 227), ('soda', 227), ('bottle', 225), ('green beans', 225), ('tv', 224), ('chair', 211), ('beer', 208), ('table', 207), ('book', 205), ('beige', 202), ('computer screen', 198), ('pen', 194), ('black white', 184), ('cup', 184)…]

→クラス分類タスクとして扱うにはデータが偏りすぎている。
また、Aliki et al [11]によると、Vizwiz のとき無闇にデータ拡張を行っても精度が悪化する可能性が示唆されている。

2. Models

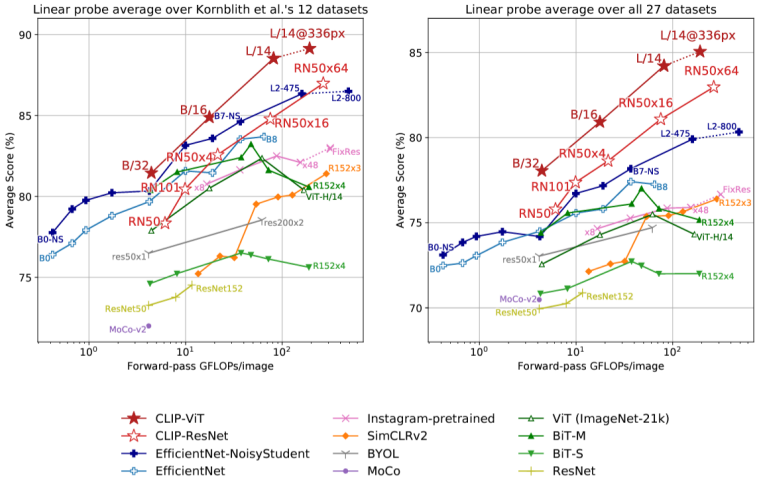
今回は大きく分けて、二種類のモデルを検討した。一つ目は(a) clip ベースでクラス分類を行うモデルであり、もう一つがテキスト生成を行う(b) Vision Language Model である。Valid データとして train データの 5%を用いた。

(a) clip ベースモデル (main_clip_val.py, main_open_clip_val.py)

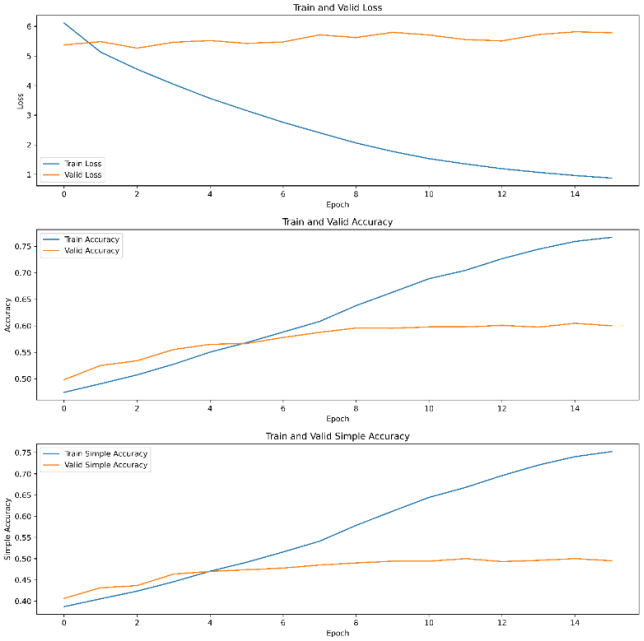


参考文献[2][3]においては右図のように clip の事前学習モデルの後ろに 2 つに分岐する線形層を用いて VQA のクラス分類を行うモデルを用いた（Answerability の部分は用いていない）。

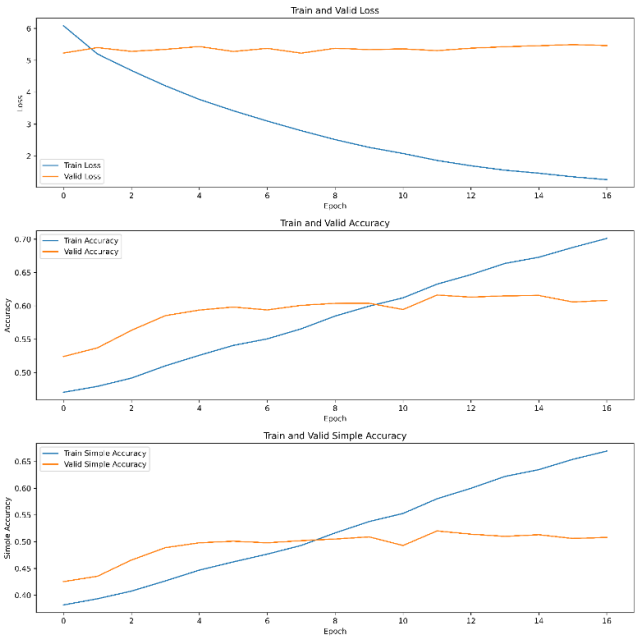
Clip の事前学習モデルについては、下図[4]に示すように OpenAI の clip で高精度の L/14@336px を用いたモデル(**main_clip_val.py**)と、clip のオープンソースである openclip の表[5]において、平均パフォーマンスが上から二番目の ViT-H-14-quickgelu を用いたモデル(**main_open_clip_val.py**)の二種類を用いた。 このモデルの選定理由については、 パラメータ数が許容範囲であり、FLOPs が一番目のモデルの 3 分の 1 程度と計算が軽いためである。



そのときの学習曲線を以下に示す。



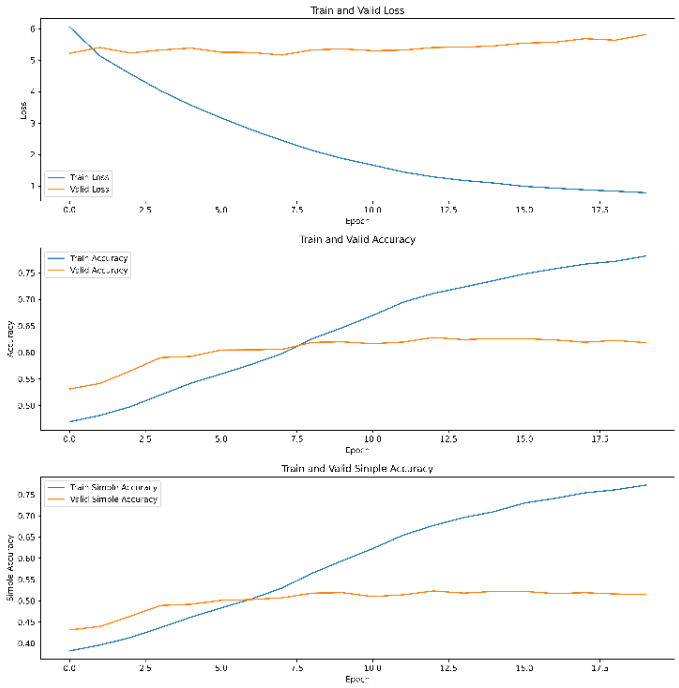
L/14@336px



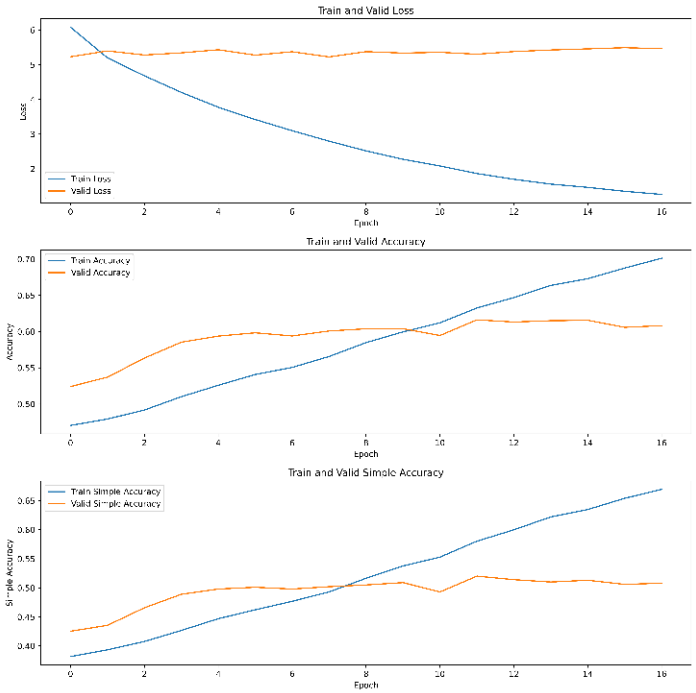
ViT-H-14-quickgelu

これらを比較すると、どちらも valid の loss は減少しておらず、VQA accuracy が **0.60** 程度で停滞している。ViT-H-14-quickgelu は L/14@336px よりパフォーマンスが良いモデルではあるが、Vizwiz データセットにおいて精度の改善は見られなかった。従って、より良い精度を得るためにはクラス分類問題ではなく、テキスト生成問題として扱う必要があることがわかった。
今回このモデルを用いた場合の learderboard 最高スコアは 0.6165 であった。

また、データ拡張(randomcrop,Horizontalflip,Verticalflip)を行った結果と比較した。
データ拡張なし(左)に比べて、データ拡張あり(右)の精度は改善しなかった。



データ拡張なし(clip)



データ拡張あり(clip)

(b) **Vision Language Model**

ここでは、Vizwiz2024 のリーダーボード[6]の第 1 位(SLCV[7])と第 3 位(PaliGemma-3B[8])に着目する。SLCV のモデルは DeepSeek-VL-7B と CogAgent-18B の 2 種類のモデルを用いたアンサンブルである。いずれも LoRA+を用いたファインチューニングを行っている。 CogAgent-18B に関しては実行環境の VRAM 容量を超えるため、今回は使わないこととした。DeepSeek-VL-7B のファインチューニングを検討した。　しかし、訓練や生成に時間がかかる(訓練:1poch 3 時間, submission の生成: 3 時間)ことと、空白””や”ununun”などの無意味な文字列が生成される問題を解決する時間がなかったため断念した。

PaliGemma-3B については画像サイズが 224px のものと 448px のものがあり、　個別のデータセットに対してファインチューニングされたモデルが多数存在する。そのため、224px のものを用いて lora を行った。lora のコードについては参考文献[9]を参考にした。google/paligemma-3b-mix-224 や google/paligemma-3b-ft-vizwizvqa-224 については ZeroShot にて valid の VQA accuracy が 0.8441 と 0.8501 であった。そのため、これらのモデルの訓練データには Vizwiz データセットが含まれている

と判断し、今回はこれらを用いないこととした。その上で別の VQA データセットの PaliGemmaFT モデルを比較した。この際、モデルの重みは 4 ビット量子化したものを用いて，計算は bfloat16 を用いることで消費メモリを削減している。

スコアは、valid の VQA accuracy であり、左から zeroshot, epoch1, 2, 3, 4 と続く。

FT モデル	VQAv2-224	OKVQA-224	TextVQA-224	OCRVQA-224
Zeroshot	0.3473	0.3053	**0.3537**	0.2477
Epoch1	**0.5912(7)**	0.5519	0.5196	0.2478
Epoch2	**0.7121(6)**	0.6142(8)	0.5277	0.4941
Epoch3	**0.7296(4)**	0.6044	0.5330	
Epoch4			0.5157	

すると、VQAv2 のファインチューニングモデル(google/paligemma-3b-ft-vqav2-224)で lora を行ったときの結果が最も優れている。 そのため、より精度の高い google/paligemma-3b-ft-vqav2-448 を用いて、lora と lora+[10]を用いた peft の結果を比較した。いずれも行列のランクは 8 であり、 Lora+においては一方の低ランク行列における学習率の倍数は 16.0 とした。

	LORA (**main_paligemma_FT448.py**)		LORA+(**main_paligemma_FT448_loraplus.py**)	
	VQA accuracy	loss	VQA accuracy	loss
Epoch 1	**0.7348(3)**	0.6470	**0.7439(2)**	0.6380
Epoch 2	0.7331	0.4137	0.7406	0.3221
Epoch 3	**0.7441(5)**	0.2415	0.7356	0.1523
Epoch 4	0.7343	0.1243		

両者を比較すると loss の減少速度から、LORA+の方が LORA よりも学習が早くなることを実際に確認した。しかし、VQA accuracy は loss に従って単調に増加しているわけではないため、計測する段階によって精度の良し悪しは変化する。

次に LORA+において、低ランク行列のランク r を r=8 と r=16 としたときの比較を行った。

	r=8 (**main_paligemma_FT448_loraplus.py**)		r=16(**main_paligemma_FT448_loraplus_r16.py**)	
	VQA accuracy	loss	VQA accuracy	loss
Epoch 1	**0.7439**	0.6380	0.7284	0.6255
Epoch 1+5000			**0.7485(1)**	0.319
Epoch1+10000			0.7289	
Epoch 2	0.7406	0.3221		
Epoch 3	0.7356	0.1523		
Epoch 4				

最終的な Leaderboard の結果は黄色で塗られた 5 つのモデル(1)~(5)での予測結果を用いて、多数決を行った(ensemble.py)。

model	Test score
(1)	0.74607
(2)	0.73469
(3)	0.74037
(4)	0.72836
(5)	0.72729
(1) + (2) + (3) ensemble	0.7548
(1) + (2) + (3) +(4) ensemble	0.76152
(1) + (2) + (3) +(4) +(5) ensemble	0.76227
(1) + (2) + (3) +(4) +(5) + (6) ensemble	0.76486
(1) + (2) + (3) +(4) +(5) + (6) + (7) ensemble	**0.76613**
(1) + (2) + (3) +(4) +(5) + (6) + (7) + (8)	**0.77247**

3. Conclusion

今回は Vizwiz データセットの answer を予測できるモデルを検討した。 PaliGemma-3b のみをファインチューニングしたモデルを用いて、リーダーボードスコア **0.77247** を達成した。当初の計画では、Vizwiz2024 における一位のモデルのように、PaliGemma-3b, DeepSeek-vl-7b, InternLM-XComposer2.5(8B)(参考文献[12]における TextVQA5 位のモデル)を駆使して、アンサンブルを取るつもりだったが、そこまでには至らなかった。

4. Reference

[1] VizWiz Visual Question Answering (https://vizwiz.org/tasks-and-datasets/vqa/)

[2] Fabian Deuser, Konrad Habel, Philipp J. Rösch, & Norbert Oswald. (2022). Less Is More: Linear Layers on CLIP Features as Powerful VizWiz Model.

[3] Visual Question Answering (https://github.com/yousefkotp/Visual-Question-Answering)

[4] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, & Ilya Sutskever. (2021). Learning Transferable Visual Models From Natural Language Supervision.

[5] OpenClip Result https://github.com/mlfoundations/open_clip/blob/main/docs/openclip_results.csv

[6] Vizwiz2024 Leaderboard https://eval.ai/web/challenges/challenge-page/2185/leaderboard/5394

[7] **2024 VizWiz Grand Challenge Workshop: VQA Presentation from SLCV** https://www.youtube.com/watch?v=Z3_QyH6zzJ8

[8] PaliGemma https://huggingface.co/docs/transformers/main/en/model_doc/paligemma#transformers.PaliGemmaForConditionalGeneration

[9]PaliGemma ファインチューニング例
https://github.com/huggingface/notebooks/blob/main/examples/paligemma/Fine_tune_PaliGemma.ipynb
[10] lora_plus https://github.com/nikhil-ghosh-berkeley/loraplus
[11] Aliki Anagnostopoulou, Mareike Hartmann, & Daniel Sonntag. (2023). Towards Adaptable and Interactive Image Captioning with Data Augmentation and Episodic Memory.
[12] OpenVLM leaderboard https://huggingface.co/spaces/opencompass/open_vlm_leaderboard
(I accessed all sites on July 17, 2024.)