vlm

pretrain

1.tokenizer

2.dataset

基于 llava，原理文章很多，且较容易。本文主要负责代码调试，明晰细节。

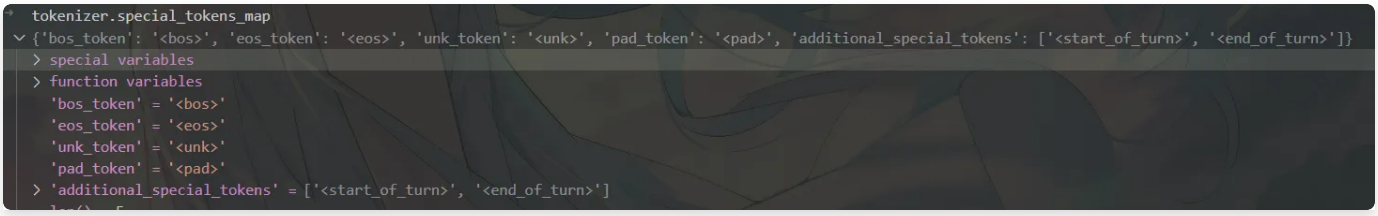
本文项目基于 llava：https://github.com/dvlab-research/MGM

pretrain

1.tokenizer

断点位于以下代码处：`if model_args.version == "v0":`

查看 tokenizer 的 special token，由于 unk 和 pad 都有，因此不需要做什么处理



除此之外，以及 dataset 部分，如果需要自定义模型几乎不需要修改什么

2.dataset

断点打在 dataset 的 `__getitem__` 函数

```python
image.shape: torch.Size([3, 768, 768])
```

进入 `preprocess_multimodal` 函数

原数据: `[[{'from': 'human', 'value': 'What is in the photo?\n<image>'}, {'from': 'gpt', 'value': 'the box chain purse black'}]]`

返回: `[[{'from': 'human', 'value': '<image>\nWhat is in the photo?'}, {'from': 'gpt', 'value': 'the box chain purse black'}]]`

`preprocess_multimodal` 函数只将 `\n<image>` 移至最前为 `<image>\n`

进入 `preprocess` 函数

进入 `preprocess_gemma` 函数

1.应用模版

```python
print(conv)
Conversation(system='', roles=('user', 'model'), messages=[], offset=0, sep_style=<SeparatorStyle.GEMMA: 7>, sep='', sep2='<eos>', version='gemma', skip_next=False)

# 指定身份
print(roles)
{'human': 'user', 'gpt': 'model'}

print(source)
[{'from': 'human', 'value': '<image>\nWhat is in the photo?'}, {'from': 'gpt', 'value': 'the box chain purse black'}]

# conv.messages包含了纠正身份的对话
print(conv)
Conversation(system='', roles=('user', 'model'), messages=[['user', '<image>\nWhat is in the photo?'], ['model', 'the box chain purse black']], offset=0, sep_style=<SeparatorStyle.GEMMA: 7>, sep='', sep2='<eos>', version='gemma', skip_next=False)
```

应用对话模版：`conversations.append(conv.get_prompt())`

进入：`elif self.sep_style == SeparatorStyle.GEMMA:`

拼接后的对话为

```python
print(ret)
<start_of_turn>user
<image>
What is in the photo?<end_of_turn>
<start_of_turn>model
the box chain purse black<end_of_turn>
<eos>
```

2.tokenize 部分：`if has_image:`

进入函数`tokenizer_image_token`

```python
print(prompt_chunks)
[[2, 106, 1645, 108], [2, 108, 1841, 603, 575, 573, 2686, 235336, 107, 108, 106, 2516, 108, 1175, 3741, 9488, 45189, 2656, 107, 108, 1]]

tokenizer("<start_of_turn>user\n").input_ids
[2, 106, 1645, 108]

tokenizer("What is in the photo?<end_of_turn>\n<start_of_turn>model\nthe box chain purse black<end_of_turn>\n<eos>").input_ids
[2, 1841, 603, 575, 573, 2686, 235336, 107, 108, 106, 2516, 108, 1175, 3741, 9488, 45189, 2656, 107, 108, 1]
```

相当于在`<image>`处进行了拆分

通过`insert_separator`函数

```python
input_ids
[2, 106, 1645, 108, -200, 108, 1841, 603, 575, 573, 2686, 235336, 107, 108, 106, 2516, 108, 1175, 3741, 9488, 45189, 2656, 107, 108, 1]
```

相当于在`prompt_chunks`间插入了`image token id` -200，同时删除第二个 `chunk` 的句首 token

返回的 shape 为：`torch.Size([1, 25])`，复制一份作为`targets`

3.mask

```python
input_ids
tensor([[     2,    106,   1645,    108,   -200,    108,   1841,    603,
          575,
            573,   2686, 235336,    107,    108,    106,   2516,    108,
         1175,
           3741,   9488,  45189,   2656,    107,    108,      1]])

targets
tensor([[ -100,  -100,  -100,  -100,  -100,  -100,  -100,  -100,  -100,  -
100,
           -100,  -100,  -100,  -100,  -100,  -100,  -100,  1175,  3741,  9
488,
          45189,  2656,   107,   108,     1]])

tokenizer("the box chain purse black<end_of_turn>\n<eos>").input_ids
[2, 1175, 3741, 9488, 45189, 2656, 107, 108, 1]
```

只保留`the box chain purse black<end_of_turn>\n<eos>`这一部分

插值将 `image` 变为`torch.Size([3, 336, 336])`