VisionTextDualEncoder and CLIP model training examples

The following example showcases how to train a CLIP-like vision-text dual encoder model using a pre-trained vision and text encoder.

Such a model can be used for natural language image search and potentially zero-shot image classification. The model is inspired by CLIP, introduced by Alec Radford et al. The idea is to train a vision encoder and a text encoder jointly to project the representation of images and their captions into the same embedding space, such that the caption embeddings are located near the embeddings of the images they describe.

Download COCO dataset (2017)

This example uses COCO dataset (2017) through a custom dataset script, which requires users to manually download the COCO dataset before training.

```
mkdir data
cd data
wget http://images.cocodataset.org/zips/train2017.zip
wget http://images.cocodataset.org/zips/val2017.zip
wget http://images.cocodataset.org/zips/test2017.zip
wget http://images.cocodataset.org/annotations/annotations_trainval2017.zip
wget http://images.cocodataset.org/annotations/image_info_test2017.zip
cd ..
```

Having downloaded COCO dataset manually you should be able to load with the `ydshieh/coc_dataset_script` dataset loading script:

```
COCO_DIR = "data"
ds = datasets.load_dataset("ydshieh/coco_dataset_script", "2017", data_dir=COCO_DIR)
```

Create a model from a vision encoder model and a text decoder model

Next, we create a VisionTextDualEncoderModel. The `VisionTextDualEncoderModel` class let's you load any vision and text encoder model to create a dual encoder. Here is an example of how to load the model using pre-trained vision and text models.

```
from transformers import (
    VisionTextDualEncoderModel,
    VisionTextDualEncoderProcessor,
    AutoTokenizer,
    AutoFeatureExtractor
)

model = VisionTextDualEncoderModel.from_vision_text_pretrained(
```

```
    "openai/clip-vit-base-patch32", "roberta-base"
)

tokenizer = AutoTokenizer.from_pretrained("roberta-base")
feat_ext = AutoFeatureExtractor.from_pretrained("openai/clip-vit-base-patch32")
processor = VisionTextDualEncoderProcessor(feat_ext, tokenizer)

# save the model and processor
model.save_pretrained("clip-roberta")
processor.save_pretrained("clip-roberta")
```

This loads both the text and vision encoders using pre-trained weights, the projection layers are randomly initialized except for CLIP's vision model. If you use CLIP to initialize the vision model then the vision projection weights are also loaded using the pre-trained weights.

Train the model

Finally, we can run the example script to train the model:

```
python examples/pytorch/contrastive-image-text/run_clip.py \
    --output_dir ./clip-roberta-finetuned \
    --model_name_or_path ./clip-roberta \
    --data_dir ./data \
    --dataset_name ydshieh/coco_dataset_script \
    --dataset_config_name=2017 \
    --image_column image_path \
    --caption_column caption \
    --remove_unused_columns=False \
    --do_train  --do_eval \
    --per_device_train_batch_size="64" \
    --per_device_eval_batch_size="64" \
    --learning_rate="5e-5" --warmup_steps="0" --weight_decay 0.1 \
    --overwrite_output_dir \
    --push_to_hub
```