



Agentic_AI_Lab1

Student Name-Simran Shikha

System id-20223431909

Roll no-2301010849

CSE-F(2023-27)

STEP-BY-STEP WORKING OF THE CODE(lab-1)

◆ 1. Environment Setup

```
!pip install git+https://github.com/huggingface/transformers.git@main
```

```
!pip install -q datasets
```

◆ What this does:

- Installs the **latest Transformers library** from Hugging Face
- Installs the **datasets** library to load ready-made datasets

Required for:

- Loading pre-trained models
 - Loading image-caption datasets
-

◆ 2. Load the Image Captioning Dataset

```
from datasets import load_dataset
```

```
dataset = load_dataset("ybelkada/football-dataset", split="train")
```

◆ What happens here:

- Downloads a **football image dataset**
- Each data item contains:
 -  an **image**
 -  a **text caption**

Dataset structure:

```
dataset[i]["image"] # image
```

```
dataset[i]["text"] # caption
```

◆ 3. Inspect Dataset Samples

```
dataset[0]["text"]
```

```
dataset[0]["image"]
```

◆ **Purpose:**

- To **verify data**
- Ensures images and captions are correctly loaded

📌 This step is mainly for **understanding and debugging**

◆ **4. Create a Custom PyTorch Dataset**

```
from torch.utils.data import Dataset, DataLoader
```

A custom class (already defined in notebook):

◆ **What this class does:**

- Converts **raw images + captions** into model-ready format
- Uses **processor** to:
 - Resize images
 - Tokenize text
 - Convert everything into tensors

📌 Output format:

```
{  
    "pixel_values": image tensor,  
    "input_ids": tokenized caption,  
    "attention_mask": mask  
}
```

This is **required for training with PyTorch**.

◆ **5. Load Processor and Model**

```
from transformers import AutoProcessor, BlipForConditionalGeneration
```

```
processor = AutoProcessor.from_pretrained("Salesforce/blip-image-captioning-base")
```

```
model = BlipForConditionalGeneration.from_pretrained("Salesforce/blip-image-captioning-base")
```

◆ **Processor:**

- Handles **both image + text**

- Prepares input exactly how BLIP expects

◆ **Model:**

- A **pre-trained BLIP image captioning model**
 - Already trained on large datasets
 - We fine-tune it on football images
-

◆ **6. Create DataLoader**

```
train_dataset = ImageCaptioningDataset(dataset, processor)  
train_dataloader = DataLoader(train_dataset, shuffle=True, batch_size=2)
```

◆ **What this does:**

- Wraps dataset into batches
- batch_size=2 → trains on 2 images at a time
- shuffle=True → improves learning

💡 Needed for efficient training

◆ **7. Training the Model**

```
optimizer = torch.optim.AdamW(model.parameters(), lr=5e-5)
```

◆ **Optimizer:**

- Updates model weights
 - AdamW is best for transformer models
-

⌚ **Training Loop**

```
for epoch in range(2):  
    for batch in train_dataloader:  
        outputs = model(**batch)  
        loss = outputs.loss  
        loss.backward()  
        optimizer.step()  
        optimizer.zero_grad()
```

◆ **What happens internally:**

1. Model predicts captions
2. Calculates **loss** (difference from true caption)
3. Backpropagation updates weights
4. Model slowly learns football-specific captions

📌 This is **fine-tuning**, not training from scratch

◆ **8. Inference (Testing the Model)**

```
example = dataset[0]
```

```
image = example["image"]
```

◆ **Selects a sample image**

◆ **Generate Caption**

```
inputs = processor(images=image, return_tensors="pt")
```

```
outputs = model.generate(**inputs)
```

```
generated_caption = processor.decode(outputs[0], skip_special_tokens=True)
```

```
print(generated_caption)
```

◆ **What happens:**

1. Image is processed
2. Model predicts caption tokens
3. Tokens converted back to human-readable text

📌 Output:

📄 Automatically generated image caption

◆ **9. Additional Image Checks**

```
dataset[1]["image"]
```

```
dataset[3]["image"]
```

Used to visually verify dataset images.