TEXT SUMMARIZATION USING BART TRANSFORMER MODEL

BART - Bidirectional and Auto Regressive Transformers

---

2. WITHOUT FINE TUNING
3. WITH FINE - TUNING

LOADING THE DATASET

```
!pip install datasets
from datasets import load_dataset

df = load_dataset("knkarthick/dialogsum")
df
df['train'][1]['dialogue']
df['train'][1]['summary']
```

1. USING THE MODEL WITHOUT FINE TUNING

LOADING THE BART MODEL

```
from transformers import pipeline

text_summarizer = pipeline("summarization",
model="facebook/bart-large-cnn")
article_1 = df['train'][1]['dialogue']

text_summarizer(article_1, max_length=20, min_length=10, do_sample=False)
```

2. FINE - TUNING MODEL

```
from transformers import AutoTokenizer, AutoModelForSeq2SeqLM
from transformers import TrainingArguments, Trainer
tokenizer = AutoTokenizer.from_pretrained("facebook/bart-base")
model = AutoModelForSeq2SeqLM.from_pretrained("facebook/bart-base")
#tokenization

def preprocess_function(batch):
    source = batch['dialogue']
    target = batch["summary"]
    source_ids = tokenizer(source, truncation=True, padding="max_length",
max_length=128)
    target_ids = tokenizer(target, truncation=True, padding="max_length",
max_length=128)

    # Replace pad token id with -100 for labels to ignore padding in loss
computation
    labels = target_ids["input_ids"]
    labels = [[(label if label != tokenizer.pad_token_id else -100) for
label in labels_example] for labels_example in labels]

    return {
        "input_ids": source_ids["input_ids"],
        "attention_mask": source_ids["attention_mask"],
        "labels": labels
    }

df_source = df.map(preprocess_function, batched=True)
# Define training arguments
training_args = TrainingArguments(
    output_dir="/content",  # Replace with your output directory
    per_device_train_batch_size=8,
    num_train_epochs=2,  # Adjust number of epochs as needed
    remove_unused_columns=False
)
# Create Trainer object
trainer = Trainer(
    model=model,
    args=training_args,
    train_dataset=df_source["train"],
    eval_dataset=df_source["test"]
)

trainer.train()
# Evaluate the model
eval_results = trainer.evaluate()

# Print evaluation results
print(eval_results)

# Save the model and tokenizer after training
model.save_pretrained("/content/your_model_directory")
tokenizer.save_pretrained("/content/your_model_directory")
```

SUMMARIZING THE CUSTOM DATA USING SAVED MODEL AND TOKENIZER

```
from transformers import AutoTokenizer, AutoModelForSeq2SeqLM

# Load the trained model and tokenizer
tokenizer = AutoTokenizer.from_pretrained("/content/your_model_directory")
model =
AutoModelForSeq2SeqLM.from_pretrained("/content/your_model_directory")

# Function to summarize a blog post
def summarize(blog_post):
    # Tokenize the input blog post
    inputs = tokenizer(blog_post, max_length=1024, truncation=True,
return_tensors="pt")

    # Generate the summary
    summary_ids = model.generate(inputs["input_ids"], max_length=150,
min_length=40, length_penalty=2.0, num_beams=4, early_stopping=True)

    # Decode the summary
    summary = tokenizer.decode(summary_ids[0], skip_special_tokens=True)
    return summary
```

```python
# Example blog post
blog_post = """
Sunsets have an undeniable allure that captivates people all over the
world. There’s something about the way the sky shifts colors as the sun
sinks lower, painting the world in hues of pink, orange, and purple. It’s
a daily phenomenon that, while familiar, never fails to take our breath
away. Each sunset is a unique experience, influenced by the time of year,
the weather, and even the location from which it’s viewed. A sunset over
the ocean might look vastly different from one in the mountains or a
cityscape. These changes create a sense of wonder, as no two sunsets are
ever exactly the same.

For many, the sight of a sunset offers a moment of reflection and peace.
The slowing of the day’s pace mirrors a natural rhythm, allowing us to
pause, take a deep breath, and let go of the stresses that have built up
throughout the day. It reminds us that time moves on, regardless of what
we do, and that every ending brings the promise of a new beginning.
Whether watching alone or with others, sunsets offer a shared moment of
beauty, a reminder that even in the most ordinary of days, there’s
extraordinary beauty waiting to be appreciated.
"""

# Get the summary
summary = summarize(blog_post)
print("Summary:", summary)
```