Mastering LLMs

Analytics Vidhya

Day 20: Tokenizing Chat Inputs for Different LLMs

```python
# Install transformers library if not already installed
# pip install transformers

from transformers import AutoTokenizer

# Define different models to compare
models = {
    "Blenderbot": "facebook/blenderbot-3B",
    "Mistral": "mistralai/Mistral-7B-Instruct",
    "Gemma": "google/gemma-7b",
    "LLaMA 3": "meta-llama/Llama-3-8B"
}

# Define a sample conversation
messages = [
    {"role": "user", "content": "How do chat templates work?"},
    {"role": "assistant", "content": "Chat templates help LLMs give coherent responses."},
    {"role": "user", "content": "How do I use them?"}
]

# Store tokenized outputs
tokenized_outputs = {}

for model_name, model_path in models.items():
    try:
        # Load tokenizer for each model
        tokenizer = AutoTokenizer.from_pretrained(model_path)

        # Apply chat template (if supported)
        if hasattr(tokenizer, "apply_chat_template"):
            encoded_input = tokenizer.apply_chat_template(messages, return_tensors="pt")
        else:
            # Basic tokenization as fallback
            concatenated_text = " ".join([msg["content"] for msg in messages])
            encoded_input = tokenizer(concatenated_text, return_tensors="pt")

        tokenized_outputs[model_name] = encoded_input
    except Exception as e:
        tokenized_outputs[model_name] = f"Error: {str(e)}"

# Display results
for model, tokenized in tokenized_outputs.items():
    print(f"\n=== {model} Tokenized Output ===")
    print(tokenized)
```

What This Code Does:

- Loads Tokenizers for Blenderbot, Mistral, Gemma, and LLaMA 3.

- Prepares a Chat Message Sequence with User Queries and Assistant Responses.

- Applies Chat Templates (if available) to structure messages properly.

- Tokenizes the Messages and prints the structured output.

Stay Tuned for **Day 21** of

Mastering LLMs