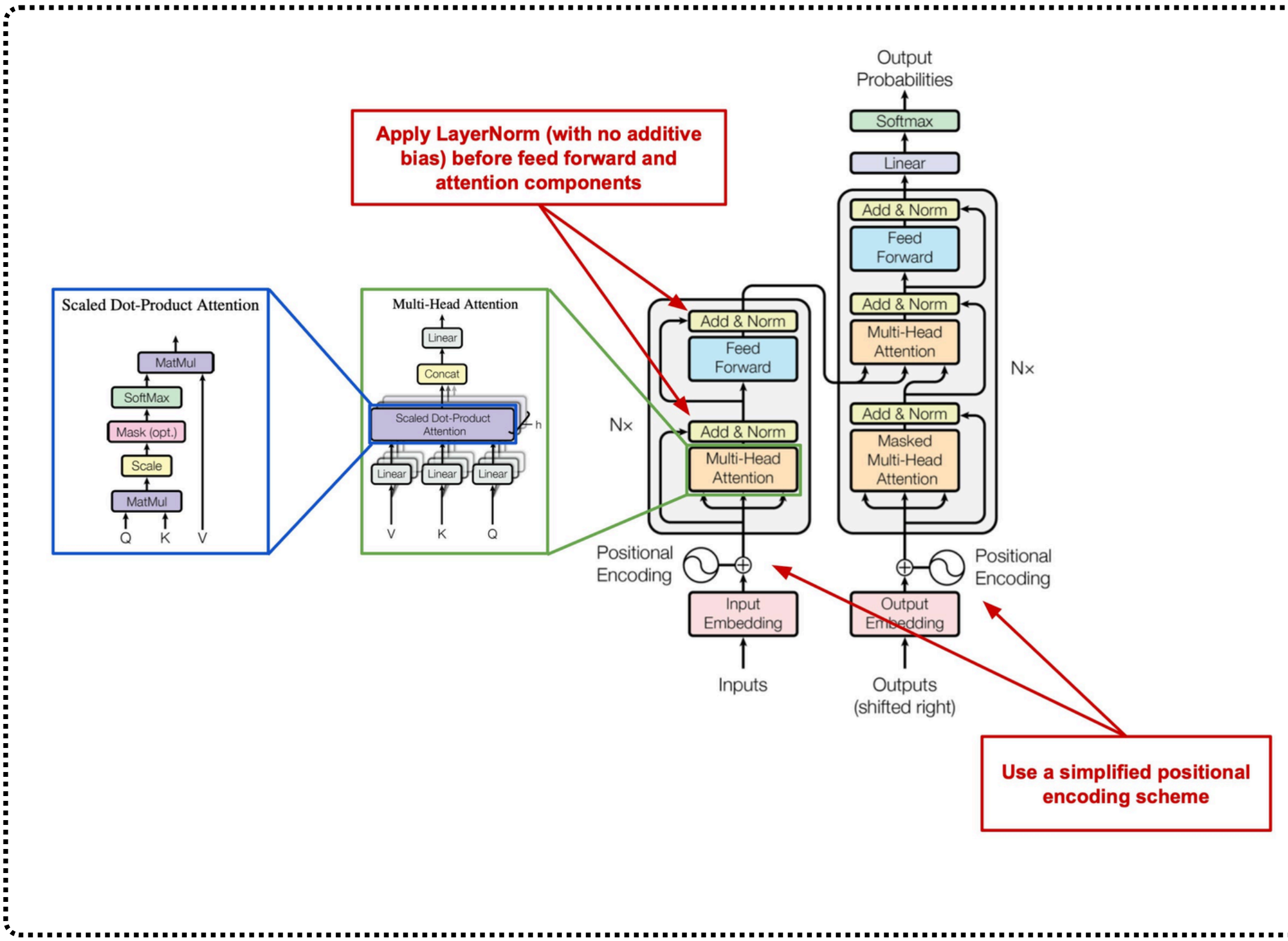
Analytics Vidhya

Mastering LLMs

Day 16: T5(Encoder Decoder Model) for Writing Product Reviews



In this post, we fine-tune a **T5 (encoder-decoder) model** to **generate product reviews** based on a product's title and star rating. Using a subset of the Amazon electronics review dataset, we preprocess and tokenize the data, train the model for text-to-text generation, and test it by generating realistic product reviews. The lesson demonstrates how T5's generative capabilities can be leveraged for practical applications and provides insights into fine-tuning and inference techniques.

```python
from transformers import T5Tokenizer, T5ForConditionalGeneration, Trainer, TrainingArguments,
DataCollatorForSeq2Seq
from datasets import load_dataset
import torch

def preprocess_data(examples):
    """
    Prepare input-output format for training T5 model.
    Input: "review: <title> <star_rating>"
    Output: "<headline>. <review_body>"
    """
    inputs = [f"review: {title} {rating} stars" for title, rating in zip(examples['title'],
examples['star_rating'])]
    targets = [f"{headline}. {body}" for headline, body in zip(examples['headline'],
examples['review_body'])]
    return {"input_text": inputs, "target_text": targets}

def load_and_prepare_data():
    """
    Load and process the dataset.
    """
    dataset = load_dataset("amazon_polarity", split="train")
    dataset = dataset.rename_columns({"title": "title", "content": "review_body", "label":
"star_rating"})
    dataset = dataset.filter(lambda x: len(x["review_body"]) > 20 and x["star_rating"] in [1, 2, 3, 4,
5])
    dataset = dataset.map(preprocess_data, remove_columns=['title', 'review_body', 'star_rating'])
    dataset = dataset.train_test_split(test_size=0.1, stratify_by_column="star_rating")
    return dataset

def fine_tune_t5():
    """
    Fine-tune T5 model for product review generation.
    """
    model_name = "t5-base"
    tokenizer = T5Tokenizer.from_pretrained(model_name)
    model = T5ForConditionalGeneration.from_pretrained(model_name)

    dataset = load_and_prepare_data()
    tokenized_dataset = dataset.map(lambda x: tokenizer(x['input_text'], truncation=True,
max_length=128, padding="max_length"), batched=True)
    tokenized_dataset = tokenized_dataset.map(lambda x: tokenizer(x['target_text'], truncation=True,
max_length=128, padding="max_length"), batched=True)
```

```python
    training_args = TrainingArguments(
        output_dir="./t5-product-reviews",
        evaluation_strategy="epoch",
        learning_rate=3e-5,
        per_device_train_batch_size=8,
        per_device_eval_batch_size=8,
        num_train_epochs=3,
        weight_decay=0.01,
        save_strategy="epoch",
    )

    data_collator = DataCollatorForSeq2Seq(tokenizer=tokenizer, model=model)

    trainer = Trainer(
        model=model,
        args=training_args,
        train_dataset=tokenized_dataset["train"],
        eval_dataset=tokenized_dataset["test"],
        tokenizer=tokenizer,
        data_collator=data_collator,
    )

    trainer.train()
    model.save_pretrained("./t5-product-reviews")
    tokenizer.save_pretrained("./t5-product-reviews")

def generate_review(title, star_rating):
    """
    Generate a product review based on input title and star rating.
    """
    model_name = "./t5-product-reviews"
    tokenizer = T5Tokenizer.from_pretrained(model_name)
    model = T5ForConditionalGeneration.from_pretrained(model_name)

    input_text = f"review: {title} {star_rating} stars"
    inputs = tokenizer(input_text, return_tensors="pt")
    outputs = model.generate(**inputs, max_length=128, num_beams=5, no_repeat_ngram_size=3,
early_stopping=True)

    response = tokenizer.decode(outputs[0], skip_special_tokens=True)
    return response

if __name__ == "__main__":
    fine_tune_t5()
    print(generate_review("Wireless Headphones", 5))
    print(generate_review("USB Cable", 3))
    print(generate_review("Smartphone Charger", 1))
```

We've provided a Python script for fine-tuning a T5 model to generate product reviews based on a product title and star rating. Here's a breakdown of what the code does:

Code Explanation

1. **Data Preprocessing (preprocess_data):**
   - Formats the data into an input-output text structure suitable for T5.
   - Example input: "review: Wireless Headphones 5 stars"
   - Example output: "Great sound quality. I love these headphones!"

2. **Loading and Preparing Data (load_and_prepare_data):**
   - Loads the Amazon product review dataset.
   - Filters relevant data (long enough reviews and valid star ratings).
   - Splits the data into training and testing sets.

3. **Model Fine-Tuning (fine_tune_t5):**
    - Loads a pre-trained T5 model and tokenizer.
    - Tokenizes input and target text.
    - Sets up training arguments like batch size, learning rate, and epochs.
    - Uses Hugging Face's Trainer to train the model.

4. **Generating Reviews (generate_review):**
    - Loads the fine-tuned model.
    - Generates a review based on a product title and star rating.
    - Implements parameters like beam search and repetition control for better output quality.

5. **Execution Flow:**
    - The script fine-tunes the model and generates example reviews for various product titles with different star ratings.

Analytics Vidhya

Stay Tuned for **Day 17** of

Mastering LLMs