TANSFORMER ARCHITECTURE ASSIGNMENTS

Assignment 1 – HF transformer pipeline to do generation

Task:

Use Huggingface transformer pipeline code as a reference (01_transformer_basic_pipeline.py) to generate text given a prompt.

For example, with the prompt:
"Best place to visit in France is"

It should produce a full response text.

1. Run the code in colab
2. Find out with the help of AI chatbots which LLM can be used which is as powerful as it can get, to run on the colab free resources
3. Load different LLMs and compare responses (at least 3)

Submit:

Link to completed notebook with the outputs in it

Assignment 2 – Next token prediction using barebones transformer model

Task:

Use Huggingface GPT2LMHeadModel and use it to predict the next token, given an input

Understand the steps such as tokenization, probability distribution (logits) and interpreting them

1. Run the code in colab or your laptop
2. Change the code as follows:
    a. Add the highest probability next token to the input
    b. Add it to the input sequence
    c. Make it predict the next token

    Repeat the above in a loop to predict 10 next tokens

    Print the final output sequence

    Repeat the above for at least 2 different prompts
3. Replace GPT2LMHeadModel with another model that is more powerful and can still run on the hardware you have (laptop or colab)
    Use assistance from AI chatbot like claude or gemini or copilot to get suggestions on which models would work well in your runtime environment
4. Compare responses (at least 3 models and 2 prompts each)

Submit:

Link to completed notebook with the outputs in it

Assignment 3 – Encoder-Decoder architecture

Task:

Run the 04_transformer_encoder_decoder_translation.ipynb in colab or VSCode/local

Enhance the code with the following:

1. Change the sentence to test out longer inputs or idioms (`"How are you?"` → `"Comment ça va ?"`).

2. Compare **beam search vs greedy** `model.generate(..., num_beams=5)`.

3. Inspect **attention weights**: set `output_attentions=True` when calling `model(...)`.

4. Replace MarianMT with **T5** (`t5-small`) and prepend the task prefix `"translate English to French: "` to the input.

Submit:

Link to completed notebook with the outputs in it

HOMEWORK/MODULE ASSIGNMENT

Assignment 4 – Sentiment classification with a mix of structured and unstructured data as input

Task:

Analyse and understand the code in 11_transformer_patient_senti.py

Answer the following questions by adding comments to the code:

1) How is the model (gpt2 in this case, which is pre-trained) learning to predict sentiment based on new training data?
   Which technique is it using? Re-training? Fine tuning? Transfer Learning?
2) In typical LLM training approaches, transfer learning like above is not effective. Why? How come it is working for above use-case?
3) The data has both structured data (categorical/numeric features such as # days admitted surgery status and treatment type), and unstructured data (patient notes text) present in the dataset. How can the model effectively find the patterns across both structured and unstructured data? For example, how can model learn that phrases like "sharp pain" in patient notes combined with "surgery: yes" in the numeric features typically correspond to higher pain levels.

Explain via comments in the code and do any modifications necessary to make the above pattern learning more effective

4) Is attention mechanism available in the GPT model being leveraged/useful in this use-case in the way it is implemented? How?
5) If not, do the modifications to the code so that it uses the attention mechanism (in context learning) to more effectively predict the sentiment.