Week 4 Practical Exercise of Lesson 3: NER With Transformers [60 Mins]

Your task

Objective: Adapt a pre-trained transformer for Named-Entity Recognition, run batch inference on sample texts, filter and align sub-tokens, and record structured JSONL output for later metric computation.
Create a Google Colab python notebook and implement the following steps:

1 Environment Setup

```
pip install transformers torch datasets pandas time

import json, time

import pandas as pd

from transformers import pipeline
```

2 Initialize the NER Pipeline

```
ner = pipeline("token-classification",model="dbmdz/bert-large-cased-finetuned-conll03-english",aggregation\_strategy="simple" # merges B/I tags into spans)
```

Test on one sentence:

```
print(ner("Barack Obama was born in Honolulu on August 4, 1961."))
```

3 Prepare Sample Texts

Load or define a small list of 20–30 varied sentences. For example:

```
samples = [

"Google announced the Pixel 6 on October 19, 2021 in New York City.",

"The FDA approved Moderna's COVID-19 vaccine in December 2020.",

"Amazon’s HQ2 is split between Virginia and New York.",

\# … add 20 more


]
```

4 Batch Inference & Latency Measurement

Measure per-sentence latency and process in batches of 8:

```
start = time.perf_counter()

all_preds = ner(samples, batch_size=8)

elapsed = time.perf_counter() - start

print(f"Avg latency: {elapsed/len(samples):.3f}s per sentence")
```

5 Filter & Align Predictions
Keep only spans with score 0.5 and ensure correct aggregation:

```
filtered = []

for text, preds in zip(samples, all_preds):

    rec = {"text": text, "predictions": []}

    for e in preds:

        if e["score"] >= 0.5:

                rec["predictions"].append({"entity":e["entity_group"],"start":e["start"],"end":
e["end"],"text":e["word"],"score": round(e["score"], 3)})

    filtered.append(rec)
```

6 Save to JSONL for Evaluation
Write each record as one JSON line:

```
with open("ner_output.jsonl", "w") as f:

    for rec in filtered:

        f.write(json.dumps(rec) + "\textbackslash{}n")
```

Verify by loading into a DataFrame:

```
df = pd.read_json("ner_output.jsonl", lines=True)

df.head()
```

7 Inspection & Reflection

1. Scan the first few JSON records to ensure spans align with full words.

2. Manually check one tricky sentence (e.g., containing hyphens or apostrophes).
3. Note any false positives or misses you'd want to address with threshold tuning or a different checkpoint.

Deliverable: A notebook or script containing the above steps, printed latency results, a preview of the JSONL file (as a DataFrame), and a brief (3–5 sentence) commentary on the quality of your entity predictions and any adjustments you'd make before formal evaluat