```
!pip -q install -U transformers datasets accelerate evaluate scikit-learn torch sentence-transformers
```

```
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 84.1/84.1 kB 3.3 MB/s eta 0:00:00
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 9.5/9.5 MB 88.4 MB/s eta 0:00:00
```

```
!pip install -U transformers
```

```
Requirement already satisfied: transformers in /usr/local/lib/python3.12/dist-packages (4.56.1)
Requirement already satisfied: filelock in /usr/local/lib/python3.12/dist-packages (from transformers) (3.19.1)
Requirement already satisfied: huggingface-hub<1.0,>=0.34.0 in /usr/local/lib/python3.12/dist-packages (from transformers) (0.34.4)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.12/dist-packages (from transformers) (2.0.2)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.12/dist-packages (from transformers) (25.0)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.12/dist-packages (from transformers) (6.0.2)
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.12/dist-packages (from transformers) (2024.11.6)
Requirement already satisfied: requests in /usr/local/lib/python3.12/dist-packages (from transformers) (2.32.4)
Requirement already satisfied: tokenizers<=0.23.0,>=0.22.0 in /usr/local/lib/python3.12/dist-packages (from transformers) (0.22.0)
Requirement already satisfied: safetensors>=0.4.3 in /usr/local/lib/python3.12/dist-packages (from transformers) (0.6.2)
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.12/dist-packages (from transformers) (4.67.1)
Requirement already satisfied: fsspec>=2023.5.0 in /usr/local/lib/python3.12/dist-packages (from huggingface-hub<1.0,>=0.34.0->tran
Requirement already satisfied: typing-extensions>=3.7.4.3 in /usr/local/lib/python3.12/dist-packages (from huggingface-hub<1.0,>=0.
Requirement already satisfied: hf-xet<2.0.0,>=1.1.3 in /usr/local/lib/python3.12/dist-packages (from huggingface-hub<1.0,>=0.34.0->
Requirement already satisfied: charset_normalizer<4,>=2 in /usr/local/lib/python3.12/dist-packages (from requests->transformers) (3
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.12/dist-packages (from requests->transformers) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.12/dist-packages (from requests->transformers) (2.5.0)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.12/dist-packages (from requests->transformers) (2025.8.
```

```
!nvidia-smi
```

```
Sat Sep 13 13:32:07 2025
+-----------------------------------------------------------------------------------------+
| NVIDIA-SMI 550.54.15              Driver Version: 550.54.15      CUDA Version: 12.4     |
|-----------------------------------------+------------------------+----------------------+
| GPU  Name                 Persistence-M | Bus-Id          Disp.A | Volatile Uncorr. ECC |
| Fan  Temp   Perf          Pwr:Usage/Cap |           Memory-Usage | GPU-Util  Compute M. |
|                                         |                        |               MIG M. |
|=========================================+========================+======================|
|   0  Tesla T4                       Off |   00000000:00:04.0 Off |                    0 |
| N/A   52C    P8             10W /   70W |       0MiB /  15360MiB |      0%      Default |
|                                         |                        |                  N/A |
+-----------------------------------------+------------------------+----------------------+

+-----------------------------------------------------------------------------------------+
| Processes:                                                                              |
|  GPU   GI   CI        PID   Type   Process name                              GPU Memory |
|        ID   ID                                                               Usage      |
|=========================================================================================|
|  No running processes found                                                             |
+-----------------------------------------------------------------------------------------+
```

```
from google.colab import files
uploaded = files.upload()
```

Choose Files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
!pip -q install -U transformers datasets accelerate evaluate sentence-transformers
import transformers, datasets, torch
print("transformers:", transformers.__version__)
print("datasets:", datasets.__version__)
!nvidia-smi
```

```
transformers: 4.56.1
datasets: 4.0.0
Sat Sep 13 14:01:32 2025
+-----------------------------------------------------------------------------------------+
| NVIDIA-SMI 550.54.15              Driver Version: 550.54.15      CUDA Version: 12.4     |
|-----------------------------------------+------------------------+----------------------+
| GPU  Name                 Persistence-M | Bus-Id          Disp.A | Volatile Uncorr. ECC |
| Fan  Temp   Perf          Pwr:Usage/Cap |           Memory-Usage | GPU-Util  Compute M. |
|                                         |                        |               MIG M. |
|=========================================+========================+======================|
|   0  Tesla T4                       Off |   00000000:00:04.0 Off |                    0 |
| N/A   39C    P8              9W /   70W |       2MiB /  15360MiB |      0%      Default |
|                                         |                        |                  N/A |
```

```
+---------------------------------------+------------------------+----------------------+

+-----------------------------------------------------------------------------------------+
| Processes:                                                                              |
|  GPU   GI   CI        PID   Type   Process name                             GPU Memory |
|        ID   ID                                                              Usage      |
|=========================================================================================|
|  No running processes found                                                             |
+-----------------------------------------------------------------------------------------+
```

```
# -*- coding: utf-8 -*-
import os, numpy as np, pandas as pd, torch, joblib
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import accuracy_score, precision_recall_fscore_support
from datasets import load_dataset
from transformers import (AutoTokenizer, AutoModelForSequenceClassification,
                          TrainingArguments, Trainer, DataCollatorWithPadding)

# ===== إعدادات =====
CSV_PATH   = "sentiment_synthetic_final.csv"   # تأكد من الاسم
MODEL_NAME = "distilbert-base-uncased"         # لو عندك عربي كثير لاحقًا جرّب "xlm-roberta-base"
MAX_LENGTH = 128
NUM_EPOCHS = 4
LR_LAST2   = 2e-4
BATCH_TRAIN, BATCH_EVAL = 16, 32               # قلّل BATCH_TRAIN إذا ظهر OOM إلى 8/4
OUTPUT_DIR = f"{MODEL_NAME.replace('/','_')}_last2_finetuned_old"

assert os.path.exists(CSV_PATH), f"الملف غير موجود: {CSV_PATH}"

# ===== 1) تحميل وتنظيف =====
df = (pd.read_csv(CSV_PATH, encoding="utf-8-sig")
        .dropna(subset=["phrase","sentiment"]))
df = df[(df["phrase"].astype(str).str.strip()!="") &
        (df["sentiment"].astype(str).str.strip()!="")].copy()

le = LabelEncoder()
df["label"] = le.fit_transform(df["sentiment"].astype(str))
print("Classes:", list(le.classes_), "| counts:", dict(df["sentiment"].value_counts()))
print("Shape:", df.shape)

# ===== 2) تقسيم stratify عبر sklearn =====
train_df, test_df = train_test_split(
    df[["phrase","label"]],
    test_size=0.2,
    stratify=df["label"],
    random_state=42
)
train_df.to_csv("train.csv", index=False)
test_df.to_csv("test.csv", index=False)
ds = load_dataset("csv", data_files={"train":"train.csv","test":"test.csv"})

# ===== 3) Tokenizer =====
tok = AutoTokenizer.from_pretrained(MODEL_NAME)

def tokenize(batch):
    return tok(batch["phrase"], padding=False, truncation=True, max_length=MAX_LENGTH)

ds_tok = ds.map(tokenize, batched=True, remove_columns=["phrase"])
ds_tok.set_format(type="torch", columns=["input_ids","attention_mask","label"])

# ===== 4) نموذج + فك آخر طبقتين =====
model = AutoModelForSequenceClassification.from_pretrained(MODEL_NAME, num_labels=len(le.classes_))

# جمّد كل الطبقات
for p in model.base_model.parameters():
    p.requires_grad = False

# فك آخر طبقتين (DistilBERT فيه 6 طبقات)
for layer in model.base_model.transformer.layer[-2:]:
    for p in layer.parameters():
        p.requires_grad = True

# رأس التصنيف يتدرّب دائمًا
if hasattr(model, "pre_classifier"):
    for p in model.pre_classifier.parameters():
        p.requires_grad = True
```

```
for p in model.classifier.parameters():
    p.requires_grad = True

data_collator = DataCollatorWithPadding(tok)

def compute_metrics(eval_pred):
    logits, labels = eval_pred
    preds = np.argmax(logits, axis=1)
    acc = accuracy_score(labels, preds)
    p, r, f1, _ = precision_recall_fscore_support(labels, preds, average="weighted", zero_division=0)
    return {"accuracy": acc, "precision_w": p, "recall_w": r, "f1_w": f1}

# ===== 5) إعدادات تدريب بدون evaluation_strategy (متوافق قديم) =====
has_cuda = torch.cuda.is_available()
bf16 = False  # الإصدارات القديمة غالبًا لا تدعم bf16 بسهولة

args = TrainingArguments(
    output_dir=OUTPUT_DIR,
    num_train_epochs=NUM_EPOCHS,
    learning_rate=LR_LAST2,
    per_device_train_batch_size=BATCH_TRAIN if has_cuda else 8,
    per_device_eval_batch_size=BATCH_EVAL if has_cuda else 16,
    gradient_accumulation_steps=1 if has_cuda else 2,
    weight_decay=0.01,
    warmup_ratio=0.1,

    # لا يوجد evaluation_strategy/save_strategy في القديم:
    do_eval=True,
    logging_steps=100,
    eval_steps=500,       # قيّم كل 500 خطوة (عدّلها لو تبغى)
    save_steps=500,       # احفظ كل 500 خطوة
    save_total_limit=2,   # احتفظ بآخر 2 تشيكبوينت فقط

    fp16=True if has_cuda else False,
    report_to="none",
)

trainer = Trainer(
    model=model,
    args=args,
    train_dataset=ds_tok["train"],
    eval_dataset=ds_tok["test"],
    tokenizer=tok,
    data_collator=data_collator,
    compute_metrics=compute_metrics,
)

trainer.train()
metrics = trainer.evaluate()
print("\nFinal metrics:",
      {k: (f"{v*100:.2f}%" if k!='eval_loss' else round(v,4)) for k,v in metrics.items()})

# ===== 6) حفظ النموذج والـTokenizer والـLabels =====
trainer.save_model(OUTPUT_DIR)      # يحفظ آخر حالة (ليست بالضرورة الأفضل في القديم)
tok.save_pretrained(OUTPUT_DIR)
joblib.dump({"label_encoder": le}, f"{OUTPUT_DIR}/label_encoder.pkl")
print("Saved ->", OUTPUT_DIR)
```

```
Classes: ['negative', 'neutral', 'positive'] | counts: {'neutral': np.int64(4764), 'positive': np.int64(4436), 'negative': np.int64
Shape: (13534, 3)
```

```
import torch, joblib
from transformers import AutoTokenizer, AutoModelForSequenceClassification

# 1) حمّل الموديل والـ tokenizer والـ label encoder
MODEL_DIR = "distilbert-base-uncased_last2_finetuned_old"   # غيّر المسار إذا مختلف
tok = AutoTokenizer.from_pretrained(MODEL_DIR)
model = AutoModelForSequenceClassification.from_pretrained(MODEL_DIR)
le = joblib.load(f"{MODEL_DIR}/label_encoder.pkl")["label_encoder"]

# 2) دالة للتنبؤ بجملة واحدة أو أكثر
def predict_sentiment(texts):
    if isinstance(texts, str):
        texts = [texts]
    enc = tok(texts, padding=True, truncation=True, max_length=128, return_tensors="pt")
    with torch.no_grad():
        logits = model(**enc).logits
    preds = logits.argmax(dim=1).cpu().numpy()
    return le.inverse_transform(preds)

# 3) مثال عملي:
sentence = "I really sad this product, it is not work."
print("Sentence:", sentence)
print("Predicted Sentiment:", predict_sentiment(sentence)[0])
```

```
Sentence: I really sad this product, it is not work.
Predicted Sentiment: negative
```

Step	Training Loss
100	0.801500

```
# ضغط مجلد الموديل
!zip -r model_last2.zip distilbert-base-uncased_last2_finetuned_old

# تنزيله إلى جهازك
from google.colab import files
files.download("model_last2.zip")

# أيضًا إذا تبي تنزّل الـ CSVs
files.download("sentiment_synthetic_final.csv")
files.download("train.csv")
files.download("test.csv")
```

1000	0.154600
1100	0.166100
1200	0.152100
1300	0.155900
1400	0.139300
1500	0.126500
1600	0.106000
1700	0.098900
1800	0.100400
1900	0.090500
2000	0.121800
2100	0.078100
2200	0.065200
2300	0.058000
2400	0.056800
2500	0.069300
2600	0.079300
2700	0.058300

[85/85 00:01]

```
Final metrics: {'eval_loss': 0.2979, 'eval_accuracy': '92.76%', 'eval_precision_w': '92.81%', 'eval_recall_w': '92.76%', 'eval_f1_w
```

```
Saved to distilbert-base-uncased_last2_finetuned_old
  adding: distilbert-base-uncased_last2_finetuned_old/ (stored 0%)
  adding: distilbert-base-uncased_last2_finetuned_old/model.safetensors (deflated 8%)
  adding: distilbert-base-uncased_last2_finetuned_old/label_encoder.pkl (deflated 35%)
  adding: distilbert-base-uncased_last2_finetuned_old/config.json (deflated 49%)
  adding: distilbert-base-uncased_last2_finetuned_old/training_args.bin (deflated 54%)
```