```
import os
os.environ["TRANSFORMERS_NO_TF"] = "1"


# 📦 Imports


import pandas as pd
import torch
import random
from datasets import load_dataset
from transformers import (
    BertTokenizer,
    BertForSequenceClassification,
    Trainer,
    TrainingArguments
)
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, f1_score


# 📥 Load dataset


dataset = load_dataset("amazon_polarity", split="train[:10000]")
df = pd.DataFrame(dataset)
df.rename(columns={"label": "sentiment", "content": "text"}, inplace=True)
```

```
/usr/local/lib/python3.11/dist-packages/huggingface_hub/utils/_auth.py:94: UserWarning:
The secret `HF_TOKEN` does not exist in your Colab secrets.
To authenticate with the Hugging Face Hub, create a token in your settings tab (https://huggingface.co/settings/tokens), set it as secret in your Google Colab and restart your session.
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to access public models or datasets.
  warnings.warn(
README.md: 100%                                                  6.81k/6.81k [00:00<00:00, 646kB/s]
train-00000-of-00004.parquet: 100%                               260M/260M [00:01<00:00, 189MB/s]
train-00001-of-00004.parquet: 100%                               258M/258M [00:04<00:00, 30.6MB/s]
train-00002-of-00004.parquet: 100%                               255M/255M [00:01<00:00, 196MB/s]
train-00003-of-00004.parquet: 100%                               254M/254M [00:01<00:00, 191MB/s]
test-00000-of-00001.parquet: 100%                                117M/117M [00:00<00:00, 164MB/s]
Generating train split: 100%                                     3600000/3600000 [00:15<00:00, 286970.78 examples/s]
Generating test split: 100%                                      400000/400000 [00:00<00:00, 457348.37 examples/s]
```

```
# 🔧 Simulate aspects
aspects = ['battery', 'display', 'performance', 'price', 'design']
df["aspect"] = df["text"].apply(lambda x: random.choice(aspects))


# 🔁 Convert to 3-class sentiment
neutral_idx = df.sample(frac=0.2, random_state=42).index
df.loc[neutral_idx, "sentiment"] = 1  # neutral
df.loc[df["sentiment"] == 1, "sentiment"] = 2  # convert original positive to 2


# 📝 Format: "[aspect]: review"
df["input_text"] = df["aspect"] + ": " + df["text"]
df = df.sample(n=3000, random_state=42).reset_index(drop=True)


# ✂️ Split dataset
train_texts, val_texts, train_labels, val_labels = train_test_split(
    df["input_text"].tolist(),
    df["sentiment"].tolist(),
    test_size=0.2,
    random_state=42
)
```

```
# 🔠 Tokenize
tokenizer = BertTokenizer.from_pretrained("bert-base-uncased")
train_encodings = tokenizer(train_texts, truncation=True, padding=True, max_length=128)
val_encodings = tokenizer(val_texts, truncation=True, padding=True, max_length=128)
```

```
tokenizer_config.json: 100%  48.0/48.0 [00:00<00:00, 5.15kB/s]
vocab.txt: 100%  232k/232k [00:00<00:00, 3.74MB/s]
tokenizer.json: 100%  466k/466k [00:00<00:00, 3.11MB/s]
config.json: 100%  570/570 [00:00<00:00, 61.5kB/s]
```

```
# 🔗 Create Dataset class
class SentimentDataset(torch.utils.data.Dataset):
    def __init__(self, encodings, labels):
        self.encodings = encodings
        self.labels = labels
    def __getitem__(self, idx):
        item = {k: torch.tensor(v[idx]) for k, v in self.encodings.items()}
        item["labels"] = torch.tensor(self.labels[idx])
        return item
    def __len__(self):
        return len(self.labels)

train_dataset = SentimentDataset(train_encodings, train_labels)
val_dataset = SentimentDataset(val_encodings, val_labels)
```

```
# 🧠 Load model
model = BertForSequenceClassification.from_pretrained("bert-base-uncased", num_labels=3)
```

```
Xet Storage is enabled for this repo, but the 'hf_xet' package is not installed. Falling back to regular HTTP download. For better performance, install the package with: `pip install huggingface_hub[hf_xet]` or `pip install hf_xet`
WARNING:huggingface_hub.file_download:Xet Storage is enabled for this repo, but the 'hf_xet' package is not installed. Falling back to regular HTTP download. For better performance, install the package with: `pip install huggingface_hub[hf_xe
model.safetensors: 100%  440M/440M [00:02<00:00, 186MB/s]
Some weights of BertForSequenceClassification were not initialized from the model checkpoint at bert-base-uncased and are newly initialized: ['classifier.bias', 'classifier.weight']
You should probably TRAIN this model on a down-stream task to be able to use it for predictions and inference.
```

```
# ⚙️ TrainingArguments
training_args = TrainingArguments(
    output_dir="./results",
    num_train_epochs=3,
    per_device_train_batch_size=8,
    per_device_eval_batch_size=8,
    logging_dir="./logs",
    do_eval=True,  # runs evaluation after training
    logging_steps=10
)
```

```
# 📊 Define compute_metrics (accuracy and F1)
def compute_metrics(eval_pred):
    predictions, labels = eval_pred
    preds = predictions.argmax(-1)
    return {
        "accuracy": accuracy_score(labels, preds),
        "f1": f1_score(labels, preds, average="weighted")
    }
```

```
# 🧠 Create Trainer instance
trainer = Trainer(
    model=model,
    args=training_args,
    train_dataset=train_dataset,
    eval_dataset=val_dataset,
    compute_metrics=compute_metrics
)
```

```
# 🚀 Train the model
trainer.train()
```

wandb: Using wandb-core as the SDK backend. Please refer to https://wandb.me/wandb-core for more information.
wandb: WARNING If you're specifying your api key in code, ensure this code is not shared publicly.
wandb: WARNING Consider setting the WANDB_API_KEY environment variable, or running `wandb login` from the command line.
wandb: No netrc file found, creating one.
wandb: Appending key for api.wandb.ai to your netrc file: /root/.netrc
wandb: Currently logged in as: raja-n (raja-n-northeastern-university) to https://api.wandb.ai. Use `wandb login --relogin` to force relogin
Tracking run with wandb version 0.19.9
Run data is saved locally in /content/wandb/run-20250417_030940-wc03y551
Syncing run **./results** to Weights & Biases (docs)
View project at https://wandb.ai/raja-n-northeastern-university/huggingface
View run at https://wandb.ai/raja-n-northeastern-university/huggingface/runs/wc03y551
[900/900 03:10, Epoch 3/3]

Step	Training Loss
10	0.536100
20	0.401600
30	0.248500
40	0.614300
50	0.443600
60	0.589500
70	0.497700
80	0.536500
90	0.446700
100	0.459300
110	0.480300
120	0.449800
130	0.402200
140	0.519800
150	0.536900
160	0.503000
170	0.469700
180	0.443300
190	0.337600
200	0.564400
210	0.503000
220	0.433000
230	0.480500
240	0.460700
250	0.453300
260	0.466600
270	0.490100
280	0.519200
290	0.390300
300	0.356900
310	0.544200
320	0.414100
330	0.479700
340	0.481900
350	0.375900