In [ ]:
```
import torch
from torch.utils.data import Dataset, DataLoader
from transformers import BertTokenizer, BertForSequenceClassification, AdamW
import numpy as np
from tensorflow.keras.datasets import imdb
```

```
/root/miniconda3/envs/vllm_darren/lib/python3.12/site-packages/tqdm/auto.py:21: T
qdmWarning: IProgress not found. Please update jupyter and ipywidgets. See http
s://ipywidgets.readthedocs.io/en/stable/user_install.html
  from .autonotebook import tqdm as notebook_tqdm
2025-05-21 01:34:37.387150: I tensorflow/core/util/port.cc:153] oneDNN custom ope
rations are on. You may see slightly different numerical results due to floating-
point round-off errors from different computation orders. To turn them off, set t
he environment variable `TF_ENABLE_ONEDNN_OPTS=0`.
2025-05-21 01:34:37.396201: E external/local_xla/xla/stream_executor/cuda/cuda_ff
t.cc:467] Unable to register cuFFT factory: Attempting to register factory for pl
ugin cuFFT when one has already been registered
WARNING: All log messages before absl::InitializeLog() is called are written to S
TDERR
E0000 00:00:1747816477.406628 2521008 cuda_dnn.cc:8579] Unable to register cuDNN
factory: Attempting to register factory for plugin cuDNN when one has already bee
n registered
E0000 00:00:1747816477.409755 2521008 cuda_blas.cc:1407] Unable to register cuBLA
S factory: Attempting to register factory for plugin cuBLAS when one has already
been registered
W0000 00:00:1747816477.418069 2521008 computation_placer.cc:177] computation plac
er already registered. Please check linkage and avoid linking the same target mor
e than once.
W0000 00:00:1747816477.418081 2521008 computation_placer.cc:177] computation plac
er already registered. Please check linkage and avoid linking the same target mor
e than once.
W0000 00:00:1747816477.418083 2521008 computation_placer.cc:177] computation plac
er already registered. Please check linkage and avoid linking the same target mor
e than once.
W0000 00:00:1747816477.418084 2521008 computation_placer.cc:177] computation plac
er already registered. Please check linkage and avoid linking the same target mor
e than once.
2025-05-21 01:34:37.420935: I tensorflow/core/platform/cpu_feature_guard.cc:210]
This TensorFlow binary is optimized to use available CPU instructions in performa
nce-critical operations.
To enable the following instructions: AVX2 AVX512F AVX512_VNNI AVX512_BF16 FMA, i
n other operations, rebuild TensorFlow with the appropriate compiler flags.
```

参数设置&设置数据集

In [ ]:
```
MAX_LEN = 256          # 最大文本长度
BATCH_SIZE = 16        # 批大小
EPOCHS = 3             # 训练轮数

# 加载Keras中的IMDB数据集
(x_train, y_train), (x_test, y_test) = imdb.load_data(num_words=20000)

# 获取单词到索引的映射并构建反向映射
word_index = imdb.get_word_index()
reverse_word_index = dict(
    [(i + 3, word) for (word, i) in word_index.items()])
```

```
reverse_word_index[0] = "<pad>"
reverse_word_index[1] = "<sos>"
reverse_word_index[2] = "<unk>"

# 将整数序列解码为文本（过滤特殊符号）
def decode_review(ids):
    return ' '.join([reverse_word_index.get(i, '?') for i in ids if i >= 3])

# 转换所有样本为文本
train_texts = [decode_review(seq) for seq in x_train]
test_texts = [decode_review(seq) for seq in x_test]

# 初始化BERT分词器
tokenizer = BertTokenizer.from_pretrained('bert-base-uncased')

# 文本编码函数
def encode_texts(texts, tokenizer, max_len):
    input_ids = []
    attention_masks = []
    for text in texts:
        encoded = tokenizer.encode_plus(
            text,
            add_special_tokens=True,
            max_length=max_len,
            padding='max_length',
            truncation=True,
            return_attention_mask=True,
            return_tensors='pt'
        )
        input_ids.append(encoded['input_ids'])
        attention_masks.append(encoded['attention_mask'])
    input_ids = torch.cat(input_ids, dim=0)
    attention_masks = torch.cat(attention_masks, dim=0)
    return input_ids, attention_masks

# 编码训练集和测试集
train_input_ids, train_attention_masks = encode_texts(train_texts, tokenizer, MA
test_input_ids, test_attention_masks = encode_texts(test_texts, tokenizer, MAX_L

# 转换为PyTorch张量
train_labels = torch.tensor(y_train, dtype=torch.long)
test_labels = torch.tensor(y_test, dtype=torch.long)

# 定义数据集类
class IMDBDataset(Dataset):
    def __init__(self, input_ids, attention_masks, labels):
        self.input_ids = input_ids
        self.attention_masks = attention_masks
        self.labels = labels

    def __len__(self):
        return len(self.labels)

    def __getitem__(self, idx):
        return {
            'input_ids': self.input_ids[idx],
            'attention_mask': self.attention_masks[idx],
            'labels': self.labels[idx]
        }
```

```
# 创建DataLoader
train_dataset = IMDBDataset(train_input_ids, train_attention_masks, train_labels
train_loader = DataLoader(train_dataset, batch_size=BATCH_SIZE, shuffle=True)

test_dataset = IMDBDataset(test_input_ids, test_attention_masks, test_labels)
test_loader = DataLoader(test_dataset, batch_size=BATCH_SIZE)
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-dataset
s/imdb.npz
17464789/17464789 ━━━━━━━━━━━━━━━━━━━━ 2s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-dataset
s/imdb_word_index.json
1641221/1641221 ━━━━━━━━━━━━━━━━━━━━ 1s 0us/step
```

加载BERT模型

In [ ]:
```
device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')
model = BertForSequenceClassification.from_pretrained(
    'bert-base-uncased',
    num_labels=2
).to(device)

# 设置优化器
optimizer = AdamW(model.parameters(), lr=2e-5)
```

```
Some weights of BertForSequenceClassification were not initialized from the model
checkpoint at bert-base-uncased and are newly initialized: ['classifier.bias', 'c
lassifier.weight']
You should probably TRAIN this model on a down-stream task to be able to use it f
or predictions and inference.
/root/miniconda3/envs/vllm_darren/lib/python3.12/site-packages/transformers/optim
ization.py:640: FutureWarning: This implementation of AdamW is deprecated and wil
l be removed in a future version. Use the PyTorch implementation torch.optim.Adam
W instead, or set `no_deprecation_warning=True` to disable this warning
  warnings.warn(
```

训练

In [ ]:
```
for epoch in range(EPOCHS):
    model.train()
    total_loss = 0
    for batch in train_loader:
        optimizer.zero_grad()
        input_ids = batch['input_ids'].to(device)
        attention_mask = batch['attention_mask'].to(device)
        labels = batch['labels'].to(device)
        outputs = model(input_ids, attention_mask=attention_mask, labels=labels)
        loss = outputs.loss
        total_loss += loss.item()
        loss.backward()
        optimizer.step()
    avg_loss = total_loss / len(train_loader)
    print(f'Epoch {epoch+1}, Loss: {avg_loss:.4f}')
```

```
Epoch 1, Loss: 0.2704
Epoch 2, Loss: 0.1422
Epoch 3, Loss: 0.0677
```

评估模型

In [ ]:
```python
model.eval()
total_correct = 0
total_samples = 0
for batch in test_loader:
    input_ids = batch['input_ids'].to(device)
    attention_mask = batch['attention_mask'].to(device)
    labels = batch['labels'].to(device)
    with torch.no_grad():
        outputs = model(input_ids, attention_mask=attention_mask)
    logits = outputs.logits
    preds = torch.argmax(logits, dim=1)
    total_correct += (preds == labels).sum().item()
    total_samples += labels.size(0)

accuracy = total_correct / total_samples
print(f'Test Accuracy: {accuracy:.4f}')
```

```
Test Accuracy: 0.9126
```

用训练好的模型推理真实影评

In [ ]:
```python
# 预测单条影评文本的函数
def predict_review_sentiment(text, model, tokenizer, max_len=256):
    model.eval()
    encoded = tokenizer.encode_plus(
        text,
        add_special_tokens=True,
        max_length=max_len,
        padding='max_length',
        truncation=True,
        return_attention_mask=True,
        return_tensors='pt'
    )
    input_id = encoded['input_ids'].to(device)
    attention_mask = encoded['attention_mask'].to(device)

    with torch.no_grad():
        output = model(input_id, attention_mask=attention_mask)
        logits = output.logits
        pred = torch.argmax(logits, dim=1).item()

    sentiment = "Positive" if pred == 1 else "Negative"
    return sentiment

# 示例影评进行测试
sample_reviews = [
    "The movie was absolutely wonderful, I loved every moment of it!",
    "I hated this film. It was a complete waste of time and money.",
    "It had some good moments, but overall it was boring and too long."
]
```

```python
for i, review in enumerate(sample_reviews):
    sentiment = predict_review_sentiment(review, model, tokenizer)
    print(f"Review {i+1}: {sentiment}")
```

```
Review 1: Positive
Review 2: Negative
Review 3: Negative
```