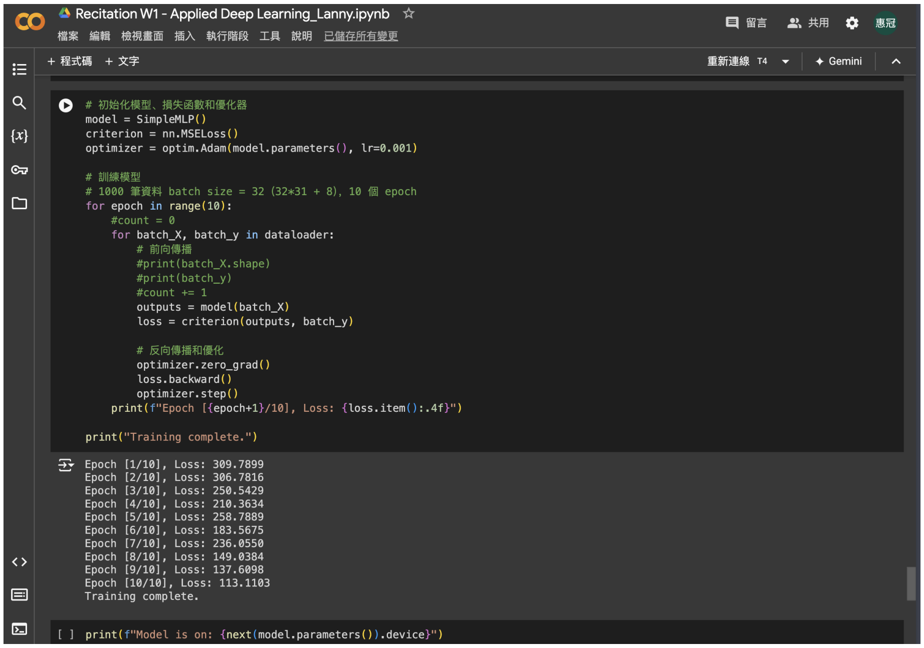
Homework 6 and 7

June 2024

1 Homework 6: Learn Basic PyTorch

The **pytorch tutorial notebook** provides several materials to learn PyTorch, Google Colab, and IPython display (ipd). In the last part of the notebook, there's a simple example to train an MLP on CPU. Below is the screenshot of my week 7 Colab training result.



Recitation W1 - Applied Deep Learning_Lanny.ipynb
檔案 編輯 檢視畫面 插入 執行階段 工具 說明 已儲存所有變更
留言
共用
惠冠
+ 程式碼 + 文字
重新連線 T4
Gemini

```python
# 初始化模型、損失函數和優化器
model = SimpleMLP()
criterion = nn.MSELoss()
optimizer = optim.Adam(model.parameters(), lr=0.001)

# 訓練模型
# 1000 筆資料 batch size = 32 (32*31 + 8), 10 個 epoch
for epoch in range(10):
    #count = 0
    for batch_X, batch_y in dataloader:
        # 前向傳播
        #print(batch_X.shape)
        #print(batch_y)
        #count += 1
        outputs = model(batch_X)
        loss = criterion(outputs, batch_y)

        # 反向傳播和優化
        optimizer.zero_grad()
        loss.backward()
        optimizer.step()
    print(f"Epoch [{epoch+1}/10], Loss: {loss.item():.4f}")

print("Training complete.")
```

```
Epoch [1/10], Loss: 309.7899
Epoch [2/10], Loss: 306.7816
Epoch [3/10], Loss: 250.5429
Epoch [4/10], Loss: 210.3634
Epoch [5/10], Loss: 258.7889
Epoch [6/10], Loss: 183.5675
Epoch [7/10], Loss: 236.0550
Epoch [8/10], Loss: 149.0384
Epoch [9/10], Loss: 137.6098
Epoch [10/10], Loss: 113.1103
Training complete.
```

```python
[ ] print(f"Model is on: {next(model.parameters()).device}")
```

Figure 1: Week 7 Colab Training Result

2 Homework 7: Learn Basic PyTorch

2.1 Overview

This homework involves fine-tuning a DistilBERT model for sequence classification on the IMDB dataset using the Hugging Face Transformers library. The training process incorporates LoRA (Low-Rank Adaptation) to enhance the model's performance. The model is trained to classify movie reviews as either positive or negative. Because my Colab GPU quota was used up and I think it's more convenient to run on IPython notebooks on my server, I didn't use Colab to do this part.

2.2 Requirements

- Python 3.8+
- PyTorch
- Transformers
- Datasets
- PEFT (Parameter Efficient Fine-Tuning)

2.3 Setup

1. Install dependencies:

```
pip install torch transformers datasets peft
```

2.4 Dataset

The IMDB dataset is used for training and validation. The dataset is loaded and processed using the `datasets` library.

2.5 Training

2.5.1 Model and Tokenizer

The base model used is `distilbert-base-cased`. The tokenizer is initialized using the same model checkpoint.

2.5.2 Data Preparation

The dataset is truncated to the first 50 tokens for each review to speed up processing. The dataset is then tokenized and prepared in batches of 16 examples.

2.5.3 DataLoader

DataLoaders are created for training and evaluation datasets.

2.5.4 LoRA Configuration

A LoRA configuration is applied to the model to optimize performance. The configuration includes parameters like rank number, alpha (scaling factor), dropout probability, and target modules.

2.5.5 Training Arguments

The training arguments are defined to control various aspects of the training process such as batch size, learning rate, evaluation strategy, and more.

2.5.6 Trainer

The Hugging Face `Trainer` is used to manage the training process. It handles the training loop, evaluation, and other functionalities.

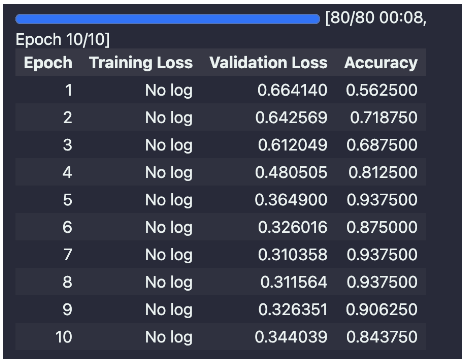
2.5.7 Compute Metrics

A custom function `compute_metrics` is defined to compute the accuracy of the model during evaluation.

2.5.8 Training the Model

The training process is initiated using the `trainer.train()` method.

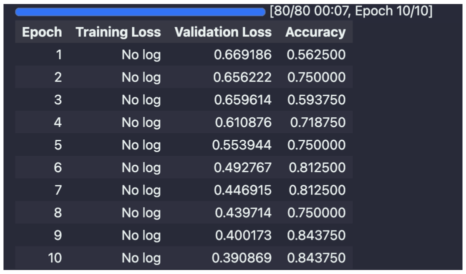
2.6 Result



[80/80 00:08,
Epoch 10/10]

Epoch	Training Loss	Validation Loss	Accuracy
1	No log	0.664140	0.562500
2	No log	0.642569	0.718750
3	No log	0.612049	0.687500
4	No log	0.480505	0.812500
5	No log	0.364900	0.937500
6	No log	0.326016	0.875000
7	No log	0.310358	0.937500
8	No log	0.311564	0.937500
9	No log	0.326351	0.906250
10	No log	0.344039	0.843750

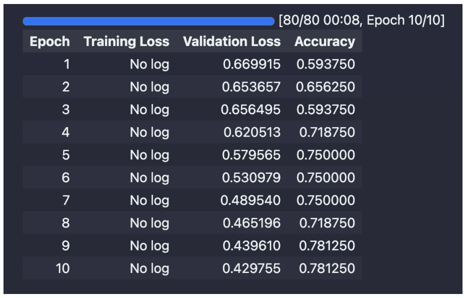
Figure 2: Training Result Configuration 1



[80/80 00:07, Epoch 10/10]

Epoch	Training Loss	Validation Loss	Accuracy
1	No log	0.669186	0.562500
2	No log	0.656222	0.750000
3	No log	0.659614	0.593750
4	No log	0.610876	0.718750
5	No log	0.553944	0.750000
6	No log	0.492767	0.812500
7	No log	0.446915	0.812500
8	No log	0.439714	0.750000
9	No log	0.400173	0.843750
10	No log	0.390869	0.843750

Figure 3: Training Result Configuration 2



[80/80 00:08, Epoch 10/10]

Epoch	Training Loss	Validation Loss	Accuracy
1	No log	0.669915	0.593750
2	No log	0.653657	0.656250
3	No log	0.656495	0.593750
4	No log	0.620513	0.718750
5	No log	0.579565	0.750000
6	No log	0.530979	0.750000
7	No log	0.489540	0.750000
8	No log	0.465196	0.718750
9	No log	0.439610	0.781250
10	No log	0.429755	0.781250

Figure 4: Training Result Configuration 3