```
import torch
import numpy as np
import torch.nn as nn
import torch.optim as optim
from openpyxl.styles.builtins import output
from torch.nn import CrossEntropyLoss
from torch.utils.data import DataLoader, TensorDataset
from sklearn.metrics import accuracy_score, confusion_matrix
import matplotlib.pyplot as plt
from torchvision import datasets
from torchvision.transforms import ToTensor
import pandas as pd
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
import torchvision
import torchvision.transforms as transforms
import torchvision.models as models
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
!pip install datasets
```

```
Collecting datasets
  Downloading datasets-3.2.0-py3-none-any.whl.metadata (20 kB)
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-packages (from datasets) (3.16.1)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.10/dist-packages (from datasets) (1.26.4)
Requirement already satisfied: pyarrow>=15.0.0 in /usr/local/lib/python3.10/dist-packages (from datasets) (17.
Collecting dill<0.3.9,>=0.3.0 (from datasets)
  Downloading dill-0.3.8-py3-none-any.whl.metadata (10 kB)
Requirement already satisfied: pandas in /usr/local/lib/python3.10/dist-packages (from datasets) (2.2.2)
Requirement already satisfied: requests>=2.32.2 in /usr/local/lib/python3.10/dist-packages (from datasets) (2.
Requirement already satisfied: tqdm>=4.66.3 in /usr/local/lib/python3.10/dist-packages (from datasets) (4.67.1
Collecting xxhash (from datasets)
  Downloading xxhash-3.5.0-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (12 kB)
Collecting multiprocess<0.70.17 (from datasets)
  Downloading multiprocess-0.70.16-py310-none-any.whl.metadata (7.2 kB)
Collecting fsspec<=2024.9.0,>=2023.1.0 (from fsspec[http]<=2024.9.0,>=2023.1.0->datasets)
  Downloading fsspec-2024.9.0-py3-none-any.whl.metadata (11 kB)
Requirement already satisfied: aiohttp in /usr/local/lib/python3.10/dist-packages (from datasets) (3.11.11)
Requirement already satisfied: huggingface-hub>=0.23.0 in /usr/local/lib/python3.10/dist-packages (from datase
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-packages (from datasets) (24.2)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.10/dist-packages (from datasets) (6.0.2)
Requirement already satisfied: aiohappyeyeballs>=2.3.0 in /usr/local/lib/python3.10/dist-packages (from aiohtt
Requirement already satisfied: aiosignal>=1.1.2 in /usr/local/lib/python3.10/dist-packages (from aiohttp->data
Requirement already satisfied: async-timeout<6.0,>=4.0 in /usr/local/lib/python3.10/dist-packages (from aiohtt
Requirement already satisfied: attrs>=17.3.0 in /usr/local/lib/python3.10/dist-packages (from aiohttp->dataset
Requirement already satisfied: frozenlist>=1.1.1 in /usr/local/lib/python3.10/dist-packages (from aiohttp->dat
Requirement already satisfied: multidict<7.0,>=4.5 in /usr/local/lib/python3.10/dist-packages (from aiohttp->d
Requirement already satisfied: propcache>=0.2.0 in /usr/local/lib/python3.10/dist-packages (from aiohttp->data
Requirement already satisfied: yarl<2.0,>=1.17.0 in /usr/local/lib/python3.10/dist-packages (from aiohttp->dat
Requirement already satisfied: typing-extensions>=3.7.4.3 in /usr/local/lib/python3.10/dist-packages (from hug
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from reque
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests>=2.32.2-
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests>=2
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests>=2
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.10/dist-packages (from pandas-
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas->datasets)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.10/dist-packages (from pandas->dataset
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.8.
Downloading datasets-3.2.0-py3-none-any.whl (480 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 480.6/480.6 kB 12.8 MB/s eta 0:00:00
Downloading dill-0.3.8-py3-none-any.whl (116 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 116.3/116.3 kB 8.7 MB/s eta 0:00:00
Downloading fsspec-2024.9.0-py3-none-any.whl (179 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 179.3/179.3 kB 12.6 MB/s eta 0:00:00
Downloading multiprocess-0.70.16-py310-none-any.whl (134 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 134.8/134.8 kB 12.6 MB/s eta 0:00:00
Downloading xxhash-3.5.0-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (194 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 194.1/194.1 kB 17.4 MB/s eta 0:00:00
```

```
Installing collected packages: xxhash, fsspec, dill, multiprocess, datasets
  Attempting uninstall: fsspec
    Found existing installation: fsspec 2024.10.0
    Uninstalling fsspec-2024.10.0:
      Successfully uninstalled fsspec-2024.10.0
ERROR: pip's dependency resolver does not currently take into account all the packages that are installed. Thi
gcsfs 2024.10.0 requires fsspec==2024.10.0, but you have fsspec 2024.9.0 which is incompatible.
Successfully installed datasets-3.2.0 dill-0.3.8 fsspec-2024.9.0 multiprocess-0.70.16 xxhash-3.5.0
```

```
df = pd.read_csv('/content/drive/MyDrive/Mental project/dataset/Combined Data.csv')
print(df.shape)
df.head()
```

```
(53043, 3)
```

	Unnamed: 0	statement	status
0	0	oh my gosh	Anxiety
1	1	trouble sleeping, confused mind, restless hear...	Anxiety
2	2	All wrong, back off dear, forward doubt. Stay ...	Anxiety
3	3	I've shifted my focus to something else but I'...	Anxiety
4	4	I'm restless and restless, it's been a month n...	Anxiety

Next steps: Generate code with df   View recommended plots   New interactive sheet

```
df = df[['statement', 'status']]
```

```
df.head()
```

	statement	status
0	oh my gosh	Anxiety
1	trouble sleeping, confused mind, restless hear...	Anxiety
2	All wrong, back off dear, forward doubt. Stay ...	Anxiety
3	I've shifted my focus to something else but I'...	Anxiety
4	I'm restless and restless, it's been a month n...	Anxiety

Next steps: Generate code with `df` View recommended plots New interactive sheet

```
df.isnull().sum()
```

	0
statement	362
status	0

dtype: int64

```
df.dropna(inplace=True)
```

```
df.isnull().sum()
```

```
              0
statement     0
   status     0

dtype: int64
```

```
df.duplicated().sum()
```

```
1588
```

```
df.drop_duplicates(inplace=True, keep='first')
```

```
df.duplicated().sum()
```

```
0
```

```
df2 = df.sample(6000).reset_index(drop=True)
df2.shape
```

```
(6000, 2)
```

```
df2['status'].value_counts()
```

status	count
Normal	1827
Depression	1777
Suicidal	1269
Anxiety	428
Bipolar	309
Stress	283
Personality disorder	107

dtype: int64

```
from imblearn.over_sampling import RandomOverSampler

ros = RandomOverSampler(random_state=42)
X_resampled, y_resampled = ros.fit_resample(df2[['statement']], df2['status'])


df_resambled = pd.concat([X_resampled, y_resampled], axis=1)
df_resambled.head()
```

	statement	status
0	I will just introduce a bit.I am 21 years old....	Suicidal
1	Ikea king â	Normal
2	Health Anxiety, what has worked for me - Hopef...	Anxiety
3	blame society Blame moviesBlames stereotypesBl...	Depression
4	Are you a liberal activist?	Normal

Next steps: Generate code with df_resambled View recommended plots New interactive sheet

```
df_resambled['status'].value_counts()
```

	count
status	
Suicidal	1827
Normal	1827
Anxiety	1827
Depression	1827
Bipolar	1827
Stress	1827
Personality disorder	1827

dtype: int64

```
from sklearn.preprocessing import LabelEncoder
```

```
le = LabelEncoder()
df_resambled['status'] = le.fit_transform(df_resambled['status'])
df_resambled.head()
```

```
df_resambled['status'].value_counts()
```

status	count
6	1827
3	1827
0	1827
2	1827
1	1827
5	1827
4	1827

dtype: int64

```
from nltk.corpus import stopwords
import nltk
from nltk.tokenize import word_tokenize
import string

nltk.download('punkt_tab')
nltk.download('stopwords')

def cleaned_text(text):
    text = text.lower()
    token = word_tokenize(text)
    stop_words = set(stopwords.words('english'))
    filtered_sentence = [word for word in token if word not in stop_words and string.punctuation and word.isalnum()]
    return " ".join(filtered_sentence)

text = "I am learning NLP ### AA @@@ !!! , any one can HEPL ME OUT ??? "
cleaned_text(text)
```

```
[nltk_data] Downloading package punkt_tab to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt_tab.zip.
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
'learning nlp aa one hepl'
```

```
X = df_resambled['statement'].apply(cleaned_text)
y = df_resambled['status']
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
from transformers import BertTokenizer, BertForSequenceClassification, AutoTokenizer, AutoModelForSequenceClassificati

tokenizer = BertTokenizer.from_pretrained('bert-base-uncased')
train_tokenizer = tokenizer(list(X_train), padding=True, truncation=True, max_length=128)
test_tokenizer = tokenizer(list(X_test), padding=True, truncation=True, max_length=128)
```

```
/usr/local/lib/python3.10/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/to
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(
tokenizer_config.json: 100%                                  48.0/48.0 [00:00<00:00, 1.19kB/s]

vocab.txt: 100%                                         232k/232k [00:00<00:00, 2.90MB/s]

tokenizer.json: 100%                                      466k/466k [00:00<00:00, 5.74MB/s]

config.json: 100%                                       570/570 [00:00<00:00, 16.2kB/s]
```

```
from datasets import Dataset

train_dataset = Dataset.from_dict({'input_ids': train_tokenizer['input_ids'], 'attention_mask': train_tokenizer['atter
test_dataset = Dataset.from_dict({'input_ids': test_tokenizer['input_ids'], 'attention_mask': test_tokenizer['attentic
```

```
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
device
```

```
device(type='cuda')
```

```
num_labels = len(df_resambled['status'].unique())
model = BertForSequenceClassification.from_pretrained('bert-base-uncased', num_labels=num_labels)

training_args = TrainingArguments(
    output_dir="./results",          # Output directory for results
    evaluation_strategy="epoch",     # Evaluate once per epoch
    save_strategy="epoch",          # Save model at the end of each epoch to match evaluation strategy
    learning_rate=2e-5,              # Learning rate
    per_device_train_batch_size=16,  # Batch size for training
    per_device_eval_batch_size=16,   # Batch size for evaluation
    num_train_epochs=5,                # Increase number of epochs
    weight_decay=0.01,                 # Strength of weight decay
    logging_dir="./logs",              # Directory for logging
    logging_steps=10,                  # Log every 10 steps
    lr_scheduler_type="linear",        # Use linear learning rate scheduler with warmup
    warmup_steps=500,                  # Number of warmup steps for learning rate scheduler
    load_best_model_at_end=True,       # Load the best model at the end of training
    metric_for_best_model="eval_loss", # Monitor eval loss to determine the best model
    save_total_limit=3,                # Limit the number of checkpoints to save
    gradient_accumulation_steps= 2     # Simulate larger batch size if GPU memory is limite
)

trainer = Trainer(
    model = model.to(device),
    args=training_args,
    train_dataset=train_dataset,
    eval_dataset=test_dataset,
)

trainer.train()
```

```
model.safetensors: 100%                                    440M/440M [00:07<00:00, 72.2MB/s]
Some weights of BertForSequenceClassification were not initialized from the model checkpoint at bert-base-uncased
You should probably TRAIN this model on a down-stream task to be able to use it for predictions and inference.
/usr/local/lib/python3.10/dist-packages/transformers/training_args.py:1575: FutureWarning: `evaluation_strategy` i
  warnings.warn(
wandb: WARNING The `run_name` is currently set to the same value as `TrainingArguments.output_dir`. If this was no
wandb: Using wandb-core as the SDK backend.  Please refer to https://wandb.me/wandb-core for more information.
wandb: Logging into wandb.ai. (Learn how to deploy a W&B server locally: https://wandb.me/wandb-server)
wandb: You can find your API key in your browser here: https://wandb.ai/authorize
wandb: Paste an API key from your profile and hit enter, or press ctrl+c to quit: ··········
wandb: Appending key for api.wandb.ai to your netrc file: /root/.netrc
Tracking run with wandb version 0.19.1
Run data is saved locally in /content/wandb/run-20250112_111429-0hlq2aov
Syncing run ./results to Weights & Biases (docs)
View project at https://wandb.ai/rickydoan144/huggingface
View run at https://wandb.ai/rickydoan144/huggingface/runs/0hlq2aov
                                    [1600/1600 22:09, Epoch 5/5]

Epoch  Training Loss  Validation Loss
  1      2.058000       0.965442
  2      1.025200       0.476821
  3      0.407700       0.316571
  4      0.346100       0.285863
  5      0.239400       0.295584

TrainOutput(global_step=1600, training_loss=1.1807581675052643, metrics={'train_runtime': 1361.134,
'train_samples_per_second': 37.583, 'train_steps_per_second': 1.175, 'total_flos': 3365012567904000.0,
'train_loss': 1.1807581675052643, 'epoch': 5.0})
```

```
from sklearn.metrics import classification_report, confusion_matrix

predictions, labels, _ = trainer.predict(test_dataset)
```

```
predictions_label = np.argmax(predictions, axis=1)
print(classification_report(labels, predictions_label, target_names= le.classes_))
```

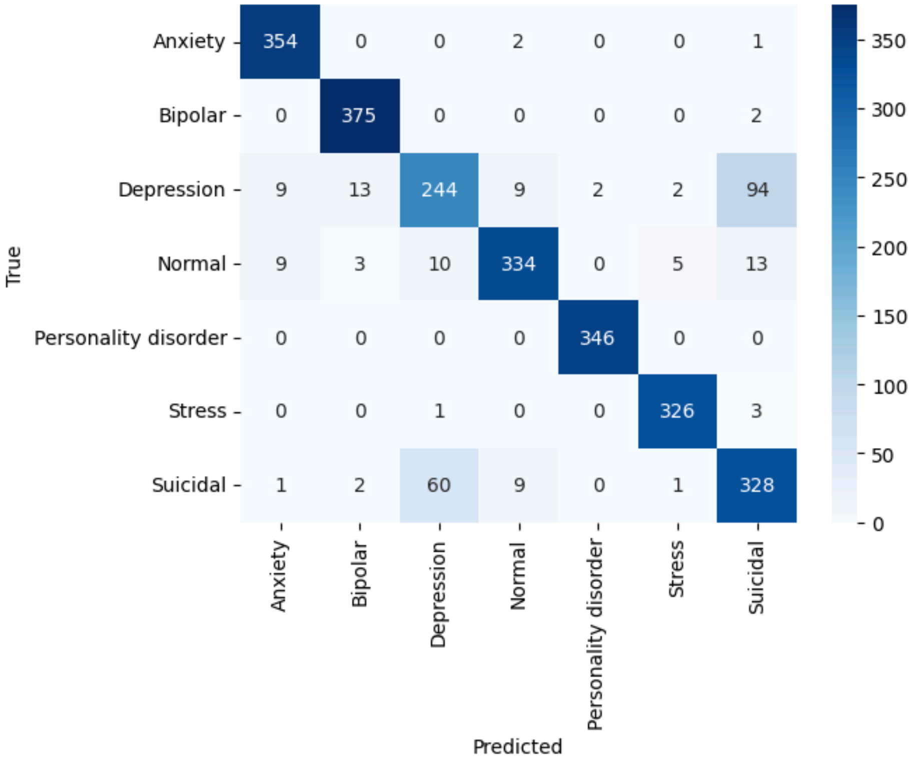
```
                      precision    recall  f1-score   support

             Anxiety       0.95      0.99      0.97       357
             Bipolar       0.95      0.99      0.97       377
          Depression       0.77      0.65      0.71       373
              Normal       0.94      0.89      0.92       374
Personality disorder       0.99      1.00      1.00       346
              Stress       0.98      0.99      0.98       330
            Suicidal       0.74      0.82      0.78       401

            accuracy                           0.90      2558
           macro avg       0.91      0.91      0.90      2558
        weighted avg       0.90      0.90      0.90      2558
```

```
cm = confusion_matrix(labels, predictions_label)
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=le.classes_, yticklabels=le.classes_)
plt.xlabel('Predicted')
plt.ylabel('True')
```

```
Text(50.72222222222221, 0.5, 'True')
```



	Anxiety	Bipolar	Depression	Normal	Personality disorder	Stress	Suicidal
Anxiety	354	0	0	2	0	0	1
Bipolar	0	375	0	0	0	0	2
Depression	9	13	244	9	2	2	94
Normal	9	3	10	334	0	5	13
Personality disorder	0	0	0	0	346	0	0
Stress	0	0	1	0	0	326	3
Suicidal	1	2	60	9	0	1	328

```
trainer.save_model('/content/drive/MyDrive/Mental project/model_bert_mental')
tokenizer.save_pretrained('/content/drive/MyDrive/Mental project/model_bert_mental')
```

```
model = AutoModelForSequenceClassification.from_pretrained('/content/drive/MyDrive/Mental project/model_bert_mental2')
# tokenizer = AutoTokenizer.from_pretrained('/content/drive/MyDrive/Mental project/model_bert_mental')
tokenizer = AutoTokenizer.from_pretrained('bert-base-uncased')
```

```
from joblib import dump

dump(le, '/content/drive/MyDrive/Mental project/label_encoder.joblib')
```

```
['/content/drive/MyDrive/Mental project/label_encoder.joblib']
```

```
model
```

Show hidden output

```
def predict_sentiment(text):
    text_cleaned = cleaned_text(text)
    inputs = tokenizer(text_cleaned, padding=True, truncation=True, max_length=128, return_tensors="pt")
    # Move input tensors to the same device as the model
    inputs = {key: val.to(device) for key, val in inputs.items()}
    outputs = model(**inputs)
```