Title - FAQ Categorizer

Text Classification by Fine-tuning Language Model

Section - 1: Data Loading

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

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

```
!pip install opencv-python
```

Show hidden output

```
!pip install simpletransformers
import pandas as pd
import re
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from simpletransformers.classification import ClassificationModel, ClassificationArgs

data = pd.read_csv('/content/drive/MyDrive/NLP_MINI/nlp_faq_dataset_cleaned.csv')
```

Show hidden output

```
print("Dataset Info:")
print(data.info())
print("\nClass Distribution:")
print(data['Labels'].value_counts())

train_data, val_data = train_test_split(data, test_size=0.2, random_state=42)
```

```
Dataset Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1024 entries, 0 to 1023
Data columns (total 2 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Questions  1024 non-null   object
 1   Labels     1024 non-null   object
dtypes: object(2)
memory usage: 16.1+ KB
None

Class Distribution:
Labels
General Inquiry         128
Account Management      128
Payment Issues          128
Troubleshooting         128
Subscription Queries    128
Technical Support       128
Security & Privacy      128
Product Information     128
Name: count, dtype: int64
```

Section - 2: Text Processing

```
def clean_text(text):
    text = text.lower()
    text = re.sub(r'[^a-zA-Z\s]', '', text)
    return text.strip()

train_df = pd.DataFrame({
    'text': train_data['Questions'].apply(clean_text),
    'labels': train_data['Labels']
})

val_df = pd.DataFrame({
    'text': val_data['Questions'].apply(clean_text),
```

```
    'labels': val_data['Labels']
})

print("\nSample Processed Data:")
print(train_df.head())
```

```
Sample Processed Data:
                                                  text              labels
137               where can i find my account history  Account Management
377                 where can i find proof of payment      Payment Issues
388         why cant i log in with my correct password     Troubleshooting
824         how do you secure remote access to systems  Security & Privacy
767  how do i resolve problems with ssl certificate...   Technical Support
```

Section - 3: Text Embedding using BERT and RoBERTa

```
bert_model = ClassificationModel('bert', 'bert-base-uncased', num_labels=2, use_cuda=False)

roberta_model = ClassificationModel('roberta', 'roberta-base', num_labels=2, use_cuda=False)
```

```
/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
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(
config.json: 100% 570/570 [00:00<00:00, 16.8kB/s]
model.safetensors: 100% 440M/440M [00:05<00:00, 99.1MB/s]
Some weights of BertForSequenceClassification were not initialized from the model checkpoint at bert-base-uncased and are newly ini
You should probably TRAIN this model on a down-stream task to be able to use it for predictions and inference.
tokenizer_config.json: 100% 48.0/48.0 [00:00<00:00, 670B/s]
vocab.txt: 100% 232k/232k [00:00<00:00, 4.24MB/s]
tokenizer.json: 100% 466k/466k [00:00<00:00, 7.42MB/s]
config.json: 100% 481/481 [00:00<00:00, 9.37kB/s]
model.safetensors: 100% 499M/499M [00:07<00:00, 45.5MB/s]
Some weights of RobertaForSequenceClassification were not initialized from the model checkpoint at roberta-base and are newly initi
You should probably TRAIN this model on a down-stream task to be able to use it for predictions and inference.
tokenizer_config.json: 100% 25.0/25.0 [00:00<00:00, 279B/s]
vocab.json: 100% 899k/899k [00:00<00:00, 7.01MB/s]
merges.txt: 100% 456k/456k [00:00<00:00, 6.99MB/s]
tokenizer.json: 100% 1.36M/1.36M [00:00<00:00, 18.0MB/s]
```

Section - 4: Model Training with BERT and RoBERTa

```
import shutil
shutil.rmtree("outputs", ignore_errors=True)
```

```
!rm -rf outputs/
```

```
label_encoder = LabelEncoder()
train_df['labels'] = label_encoder.fit_transform(train_df['labels'])
val_df['labels'] = label_encoder.transform(val_df['labels'])
num_labels = len(label_encoder.classes_)

bert_args = ClassificationArgs(
    overwrite_output_dir=True,
    output_dir="outputs_bert"
)

roberta_args = ClassificationArgs(
    overwrite_output_dir=True,
    output_dir="outputs_roberta"
)
```

```
bert_model = ClassificationModel('bert', 'bert-base-uncased', num_labels=num_labels, args=bert_args, use_cuda=False)
roberta_model = ClassificationModel('roberta', 'roberta-base', num_labels=num_labels, args=roberta_args, use_cuda=False)

bert_model.train_model(train_df)
roberta_model.train_model(train_df)
```

```
Some weights of BertForSequenceClassification were not initialized from the model checkpoint at bert-base-uncased and are newly ini
You should probably TRAIN this model on a down-stream task to be able to use it for predictions and inference.
Some weights of RobertaForSequenceClassification were not initialized from the model checkpoint at roberta-base and are newly initi
You should probably TRAIN this model on a down-stream task to be able to use it for predictions and inference.
2/? [00:01<00:00, 2.23it/s]
Epoch 1 of 1: 100% 1/1 [20:59<00:00, 1259.57s/it]
Epochs 1/1. Running Loss: 0.9609: 100% 103/103 [20:54<00:00, 10.13s/it]
2/? [00:00<00:00, 2.21it/s]
Epoch 1 of 1: 100% 1/1 [21:33<00:00, 1293.17s/it]
Epochs 1/1. Running Loss: 0.7850: 100% 103/103 [21:02<00:00, 9.99s/it]
(103, 1.5923569867911849)
```

```
bert_args_hp = ClassificationArgs(
    num_train_epochs=3,
    train_batch_size=8,
    eval_batch_size=8,
    learning_rate=3e-5,
    max_seq_length=128,
    weight_decay=0.01,
    warmup_steps=0,
    logging_steps=50,
    save_steps=200,
    overwrite_output_dir=True,
    output_dir="outputs_bert_hp"
)

roberta_args_hp = ClassificationArgs(
    num_train_epochs=3,
    train_batch_size=8,
    eval_batch_size=8,
    learning_rate=3e-5,
    max_seq_length=128,
    weight_decay=0.01,
    warmup_steps=0,
    logging_steps=50,
    save_steps=200,
    overwrite_output_dir=True,
    output_dir="outputs_roberta_hp"
)
bert_model_hp = ClassificationModel('bert', 'bert-base-uncased', num_labels=num_labels, args=bert_args_hp, use_cuda=False)
roberta_model_hp = ClassificationModel('roberta', 'roberta-base', num_labels=num_labels, args=roberta_args_hp, use_cuda=False)

bert_model_hp.train_model(train_df)
roberta_model_hp.train_model(train_df)
```

```
Some weights of BertForSequenceClassification were not initialized from the model checkpoint at bert-base-uncased and are newly ini
You should probably TRAIN this model on a down-stream task to be able to use it for predictions and inference.
Some weights of RobertaForSequenceClassification were not initialized from the model checkpoint at roberta-base and are newly initi
You should probably TRAIN this model on a down-stream task to be able to use it for predictions and inference.
2/? [00:00<00:00,  2.56it/s]
Epoch 3 of 3: 100% 3/3 [1:03:44<00:00, 1271.72s/it]
Epochs 1/3. Running Loss:  1.1840: 100% 103/103 [20:29<00:00,  9.70s/it]
Epochs 2/3. Running Loss:  1.2159: 100% 103/103 [21:06<00:00, 11.53s/it]
Epochs 3/3. Running Loss:  0.6069: 100% 103/103 [20:30<00:00,  9.92s/it]
2/? [00:00<00:00,  2.62it/s]
Epoch 3 of 3: 100% 3/3 [1:06:26<00:00, 1326.40s/it]
Epochs 1/3. Running Loss:  1.3408: 100% 103/103 [21:14<00:00, 10.72s/it]
Epochs 2/3. Running Loss:  0.3266: 100% 103/103 [21:46<00:00, 11.98s/it]
Epochs 3/3. Running Loss:  0.4982: 100% 103/103 [21:04<00:00, 11.40s/it]
(309, 0.8439472924592426)
```

Section - 5: Evaluation on Validation Set

```
result_bert, _, _ = bert_model.eval_model(val_df)
print("\nBERT Evaluation Results (Basic):", result_bert)

result_roberta, _, _ = roberta_model.eval_model(val_df)
print("\nRoBERTa Evaluation Results (Basic):", result_roberta)

result_bert_hp, _, _ = bert_model_hp.eval_model(val_df)
print("\nBERT Evaluation Results (Fine-Tuned):", result_bert_hp)

result_roberta_hp, _, _ = roberta_model_hp.eval_model(val_df)
print("\nRoBERTa Evaluation Results (Fine-Tuned):", result_roberta_hp)
```

```
1/0 [00:00<00:00,  2.76it/s]
Running Evaluation: 100% 3/3 [01:40<00:00, 27.89s/it]

BERT Evaluation Results (Basic): {'mcc': np.float64(0.54425802359509), 'eval_loss': 1.2428195873896282}
1/0 [00:00<00:00,  1.63it/s]
Running Evaluation: 100% 3/3 [01:37<00:00, 27.16s/it]

RoBERTa Evaluation Results (Basic): {'mcc': np.float64(0.7507922673309897), 'eval_loss': 0.8442981441815695}
1/0 [00:00<00:00,  4.77it/s]
Running Evaluation: 100% 26/26 [01:30<00:00,  3.19s/it]

BERT Evaluation Results (Fine-Tuned): {'mcc': np.float64(0.7376736041944651), 'eval_loss': 0.7109730398425689}
1/0 [00:00<00:00,  1.28it/s]
Running Evaluation: 100% 26/26 [01:30<00:00,  3.30s/it]

RoBERTa Evaluation Results (Fine-Tuned): {'mcc': np.float64(0.8169562965145866), 'eval_loss': 0.5591316813459764}
```

Section - 6: Saving the models

```
bert_model.save_model('bert_best_model')

roberta_model.save_model('roberta_best_model')

bert_model_hp.save_model('bert_best_model_hp')

roberta_model_hp.save_model('roberta_best_model_hp')
```

```
!ls -lhR outputs_bert/
!ls -lhR outputs_roberta/
!ls -lhR outputs_bert_hp/
!ls -lhR outputs_roberta_hp/
```

Show hidden output

Section - 7: Prediction on Real-World Input

```
from simpletransformers.classification import ClassificationModel
import os

bert_basic_path = "outputs_bert"
roberta_basic_path = "outputs_roberta"
bert_finetuned_path = "outputs_bert_hp"
roberta_finetuned_path = "outputs_roberta_hp"


for model_path in [bert_basic_path, roberta_basic_path, bert_finetuned_path,
roberta_finetuned_path]:
    if not os.path.exists(f"{model_path}/model.safetensors"):
        raise FileNotFoundError(f"Model missing in {model_path}! Train and save
        it first.")

bert_model_loaded = ClassificationModel("bert", bert_basic_path,
use_safetensors=True, use_cuda=False)
roberta_model_loaded = ClassificationModel("roberta", roberta_basic_path,
use_safetensors=True, use_cuda=False)

bert_model_loaded_hp = ClassificationModel("bert", bert_finetuned_path,
use_safetensors=True, use_cuda=False)
roberta_model_loaded_hp = ClassificationModel("roberta",
roberta_finetuned_path, use_safetensors=True, use_cuda=False)

real_world_text = ["How to reset password? I forgot it", "Can I know the
support hours"]

predictions_bert, _ = bert_model_loaded.predict(real_world_text)
print(f"\nBERT Predictions (Basic): {predictions_bert}")

predictions_roberta, _ = roberta_model_loaded.predict(real_world_text)
print(f"\nRoBERTa Predictions (Basic): {predictions_roberta}")

predictions_bert_hp, _ = bert_model_loaded_hp.predict(real_world_text)
print(f"\nBERT Predictions (Fine-Tuned): {predictions_bert_hp}")

predictions_roberta_hp, _ = roberta_model_loaded_hp.predict(real_world_text)
print(f"\nRoBERTa Predictions (Fine-Tuned): {predictions_roberta_hp}")
```

```
1/0 [00:00<00:00,  4.98it/s]
100% 1/1 [00:03<00:00,  3.72s/it]

BERT Predictions (Basic): [7, 2]
1/0 [00:00<00:00,  1.46it/s]
100% 1/1 [00:03<00:00,  3.68s/it]

RoBERTa Predictions (Basic): [0, 1]
1/0 [00:00<00:00,  5.56it/s]
100% 1/1 [00:03<00:00,  3.47s/it]

BERT Predictions (Fine-Tuned): [7, 1]
1/0 [00:00<00:00,  2.04it/s]
100% 1/1 [00:01<00:00,  1.06s/it]

RoBERTa Predictions (Fine-Tuned): [6, 1]
```