災難推文辨識

小組成員和負責工作：

109590041 范遠皓 撰寫程式和測試

109590043 柯瑞霖 撰寫程式和報告

環境

使用的語言：Python

所需套件：

requirements.txt 檔案內

pytorch 安裝

安裝辦法：

Zip 檔內有 requirements.txt

終端機輸入以下指令安裝

pip3 install -r requirements.txt

資料讀取：

```python
data = pd.read_csv('nlp-getting-started/train.csv')
test_data = pd.read_csv('nlp-getting-started/test.csv')
```

資料預處理：

將 keyword 標籤內有包含 '%20' 的替換掉

移除網址 、 @someone

```python
def keyword_preprocess(text):
    """Clean keyword by removing '%20'"""
    if pd.notnull(text):
        text = text.replace("%20", " ")
    else:
        text = ''
    return text

def remove_url(text):
    url_pattern = re.compile(r'https?://t\.co/[^\s]*')
    new_text = url_pattern.sub('', text)
    return new_text

def remove_at(text):
    at_pattern = re.compile(r'@[^\s]*')
    new_text = at_pattern.sub('', text)
    return new_text

def text_preprocess(text):
    """Clean text by removing url and @someone"""
    text = remove_url(text)
    text = remove_at(text)
    return text

# remove url and @ from text
data['text'] = data['text'].apply(text_preprocess)
test_data['text'] = test_data['text'].apply(text_preprocess)

# remove %20 from keyword
data['keyword'] = data['keyword'].apply(keyword_preprocess)
test_data['keyword'] = test_data['keyword'].apply(keyword_preprocess)
```

將 keyword 加入 text(tweet)

```python
# combine keyword and text
data['keyword_text'] = data.apply(lambda row: row['keyword'] + ' ' + row['text'], axis=1)
test_data['keyword_text'] = test_data.apply(lambda row: row['keyword'] + ' ' + row['text'], axis=1)
```

將資料轉成 dataset

```python
train_data_dict = {
    "text": data["keyword_text"].tolist(),
    "label": data["target"].tolist()
}

test_data_dict = {
    "text": test_data["keyword_text"].tolist()
}

train_dataset = Dataset.from_dict(train_data_dict)
test_dataset = Dataset.from_dict(test_data_dict)
```

載入 BERT 模型：

初始化一個 BERT tokenizer ，並將文本轉換成 BERT 模型所需的輸入格式，載入預訓練模型 BERT，用於訓練一個基於 BERT 模型的序列分類器

```python
checkpoint = "bert-base-uncased"
tokenizer = AutoTokenizer.from_pretrained(checkpoint)

def tokenize_function(example):
    return tokenizer(example["text"], truncation=True)

tokenized_train_dataset = train_dataset.map(tokenize_function, batched=True)
tokenized_test_dataset = test_dataset.map(tokenize_function, batched=True)

# use dynamic padding
data_collator = DataCollatorWithPadding(tokenizer=tokenizer)

model = AutoModelForSequenceClassification.from_pretrained(checkpoint, num_labels=2)
```

調整訓練參數並輸出結果：

```
training_args = TrainingArguments(
    "test-trainer",
    report_to='none',
    num_train_epochs=2,
    save_strategy = "epoch"
)

trainer = Trainer(
    model,
    training_args,
    train_dataset=tokenized_train_dataset,
    data_collator=data_collator,
    tokenizer=tokenizer,
)

trainer.train()

predictions = trainer.predict(tokenized_test_dataset)
preds = np.argmax(predictions.predictions, axis=-1)

submission = pd.DataFrame({'id':test_data['id'],'target':preds})
submission.to_csv('nlp-getting-started/mySubmission.csv', index=False)
```

執行結果：

有預處理

submission1.csv Complete · 19h ago	0.83634

沒有預處理

submission_test.csv Complete · now	0.84063

找出受災地區：

使用了 spaCy 套件的 en_core_web_sm 模型，檢測每條貼文中的地點

```
import en_core_web_sm
nlp = en_core_web_sm.load()
def location_detect(text):
    doc = nlp(text)
    data = [(X.text, X.label_) for X in doc.ents]
    for word, pos in data:
        if pos == 'GPE':
            return(word)
test_data_location['location'] = test_data_location['text'].apply(location_detect)
# print(test_data_location['location'])
result = pd.DataFrame({'text':test_data_location['text'],'target':preds,'location':test_data_location['location']})
result = result[result['target'] == 1]
result = result[result['location'].notnull()]
print(result)
```

```
                                                  text  target  \
4          Typhoon Soudelor kills 28 in China and Taiwan       1
15     Birmingham Wholesale Market is ablaze BBC News...       1
34       Accident on A27 near Lewes is it Kingston Ro...       1
36     For Legal and Medical Referral Service  Call u...       1
52     'We are still living in the aftershock of Hiro...       1
...                                                  ...     ...
3238   Wreckage 'Conclusively Confirmed' as From MH37...       1
3239   Wreckage 'Conclusively Confirmed' as From MH37...       1
3254   Officials: Alabama home quarantined over possi...       1
3257   The death toll in a #IS-suicide car bombing on...       1
3260                   Green Line derailment in Chicago        1

                   location
4                     China
15               Birmingham
34                    Lewes
36                    Legal
52                 Hiroshima
...                      ...
3238               Wreckage
3239               Wreckage
3254                Alabama
3257  the Village of Rajman
3260                Chicago

[441 rows x 3 columns]
```