National College of Ireland

Configuration Manual for Multilingual Sentiment Analysis Models Using Transfer Learning

MSc Research Project
MSc in Data Analytics

Rithish Kumar Yalla
Student ID: 23188910

School of Computing
National College of Ireland

Supervisor: Cristina Hava Muntean

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Rithish Kumar Yalla
Student ID:	x23188910
Programme:	MSc in Data Analytics **Year:** 2025
Module:	MSc Research Project
Supervisor:	Cristina Hava Muntean
Submission Due Date:	29/01/2025
Project Title:	Multilingual Sentiment Analysis Models Using Transfer Learnings
Word Count:	456 **Page Count:** 12

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Rithish Kumar Yalla

Date: 29/01/2025

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual for Multilingual Sentiment Analysis Models Using Transfer Learning

Rithish Kumar Yalla
x23188910

1. Introduction

This Configuration Manual provides detailed guidelines on setting up, running, and evaluating the sentiment analysis models used in this research. The models include traditional machine learning algorithms (Naive Bayes, Logistic Regression, and Support Vector Machine) as well as the advanced Hybrid Transformer Model based on XLM-RoBERTa with Meta Learning. The following sections will guide you through the necessary setup, dependencies, dataset preparation, and configuration of both the traditional machine learning models and the Hybrid Transformer Model.

2. System Requirements

Before starting, ensure the following system requirements are met:

- Operating System: Linux, macOS, or Windows (Linux recommended for better compatibility)
- Memory (RAM): 8 GB minimum (16 GB recommended)
- Processor: Intel Core i5 or higher (i7/i9 recommended for faster training)
- Disk Space: 10 GB minimum (more if using large datasets)
- Python Version: Python 3.8 or higher

3. Software Dependencies:

The following Python libraries are required for the execution of the sentiment analysis models:

General dependencies:

- numpy: For numerical operations
- pandas: For data manipulation
- scikit-learn: For machine learning algorithms and metrics
- matplotlib: For plotting results
- seaborn: For enhanced visualization
- scipy: For scientific calculations
- tqdm: For progress bars during training
- sklearn: For model evaluation metrics

For Hybrid Transformer Model:

- transformers: For leveraging pre-trained models like XLM-RoBERTa
- torch: PyTorch library for model training and deep learning functionalities
- tensorflow: TensorFlow is an alternative to PyTorch for certain tasks
- datasets: To handle datasets efficiently, especially when working with multilingual data
- accelerate: For optimizing model training using multiple GPUs (if available)

- sentencepiece: For tokenization with XLM-RoBERTa

4. Dataset

Dataset Source: The primary dataset for this research is sourced from Kaggle, specifically from the Multi-Task Learning dataset (available at Kaggle Link). This dataset contains tweets in multiple languages, annotated with sentiment labels, making it a rich resource for sentiment analysis across diverse linguistic contexts.

- Dataset Description: The dataset consists of 4,917 entries, each comprising a tweet, its language, and a corresponding sentiment rating. The sentiment ratings are classified into five categories: '1 star', '2 stars', '3 stars', '4 stars', and '5 stars'.

Sample Data:

	tweet	language	sentiment
0	Lionel Messi, que ha estado vinculado con un t...	es	3 stars
1	This is a guest post by The Joy of Truth. To r...	en	4 stars
2	Nous sommes tous conscients de la popularité d...	fr	5 stars
3	El baño en el sistema de metro de la ciudad de...	es	4 stars
4	"Ich habe dies seit über 20 Jahren getan und i...	de	5 stars

5. Execution of the Code Implementation

Import the necessar libraries for the task

```
#import the necessary libaries for the task
import pandas as pd   # For data manipulation and analysis
import numpy as np    # For numerical operations
import matplotlib.pyplot as plt # For basic data visualization
import seaborn as sns  # For enhanced data visualization
from wordcloud import WordCloud
import re, string #data processing
from sklearn.preprocessing import LabelEncoder #For label encoding categorical variable
from sklearn.model_selection import train_test_split  #For splitting the data into train and test

# machine learning models
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.naive_bayes import MultinomialNB
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC

#transformers
from transformers import BertTokenizerFast
import torch
from torch.utils.data import TensorDataset, DataLoader
from transformers import DistilBertTokenizer
from transformers import DistilBertForSequenceClassification, AdamW
from transformers import XLMRobertaTokenizer, XLMRobertaForSequenceClassification, AdamW
from tqdm import tqdm
```

Load the Dataset

```
# read & load our data
tweetsData = pd.read_csv("/kaggle/input/mulitlingual-sentiment-dataset/data.csv")
```

```
# display the first few rows of the dataset
tweetsData.head()
```

	tweet	language	sentiment
0	Lionel Messi, que ha estado vinculado con un t...	es	3 stars
1	This is a guest post by The Joy of Truth. To r...	en	4 stars
2	Nous sommes tous conscients de la popularité d...	fr	5 stars
3	El baño en el sistema de metro de la ciudad de...	es	4 stars
4	"Ich habe dies seit über 20 Jahren getan und i...	de	5 stars

```
# Display the last five rows of the dataset
tweetsData.tail()
```

	tweet	language	sentiment
4912	\nA former CIA officer and CIA director has pl...	en	1 star
4913	Karen M. Felt, Ph.D. La ricerca è stata condot...	it	4 stars
4914	Mit all der Aufmerksamkeit, die dem Thema Abtr...	de	2 stars
4915	L'élément le plus important dans le processus ...	fr	4 stars
4916	On voit souvent dans les films quelqu'un qui a...	fr	3 stars

Exploration of the Dataset

```
# check out some of our tweets
tweetsData['tweet'][0:5]
```

```
0    Lionel Messi, que ha estado vinculado con un t...
1    This is a guest post by The Joy of Truth. To r...
2    Nous sommes tous conscients de la popularité d...
3    El baño en el sistema de metro de la ciudad de...
4    "Ich habe dies seit über 20 Jahren getan und i...
Name: tweet, dtype: object
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4917 entries, 0 to 4916
Data columns (total 3 columns):
 #   Column     Non-Null Count  Dtype 
---  ------     --------------  ----- 
 0   tweet      4917 non-null   object
 1   language   4917 non-null   object
 2   sentiment  4917 non-null   object
dtypes: object(3)
memory usage: 115.4+ KB
None
```

Data Preprocessing

```python
# Function to translate the language codes values to Actual Name of Language in the dataframe
def translate_language_codes(df):
    # Define a mapping from language codes to their full names
    language_mapping = {
        'es': 'Spanish',
        'en': 'English',
        'fr': 'French',
        'de': 'German',
        'it': 'Italian'
    }

    # Replace the language codes in the 'language' column using the mapping
    df['language'] = df['language'].replace(language_mapping)

    return df


# Convert the language codes
translate_language_codes(tweetsData)
```

	tweet	language	sentiment
0	Lionel Messi, que ha estado vinculado con un t...	Spanish	3 stars
1	This is a guest post by The Joy of Truth. To r...	English	4 stars
2	Nous sommes tous conscients de la popularité d...	French	5 stars
3	El baño en el sistema de metro de la ciudad de...	Spanish	4 stars
4	"Ich habe dies seit über 20 Jahren getan und i...	German	5 stars
...	...	...	...
4912	\nA former CIA officer and CIA director has pl...	English	1 star
4913	Karen M. Felt, Ph.D. La ricerca è stata condot...	Italian	4 stars
4914	Mit all der Aufmerksamkeit, die dem Thema Abtr...	German	2 stars
4915	L'élément le plus important dans le processus ...	French	4 stars
4916	On voit souvent dans les films quelqu'un qui a...	French	3 stars

4917 rows × 3 columns

```
#checks the Unique all star values in a Sentiments Type column in the dataset
tweetsData['sentiment'].unique()
```

```
array(['3 stars', '4 stars', '5 stars', '2 stars', '1 star'], dtype=object)
```

Feature Engineering

```
# Calculate the average word length in each tweet
tweetsData['average_word_length'] = tweetsData['tweet'].apply(lambda text: sum(len(word) for word in text.
```

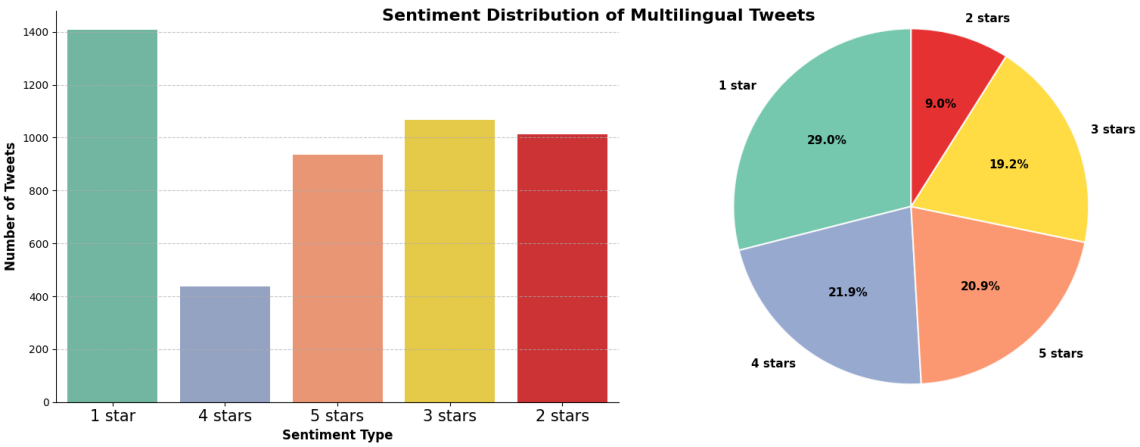
```
# Function to clean the content of tweets
def clean_tweet_content(text):
    text = re.sub(r"@[A-Za-z0-9]+", "", text)  # Remove mentions
    text = " ".join(text.split())  # Remove extra spaces
    text = re.sub(r"http\S+|www\S+|https\S+", "", text)  # Remove URLs
    text = text.replace("#", "")  # Remove hashtags
    return text

# Clean the tweet text and store it in a new column
tweetsData['cleaned_tweet'] = tweetsData['tweet'].apply(clean_tweet_content)
```

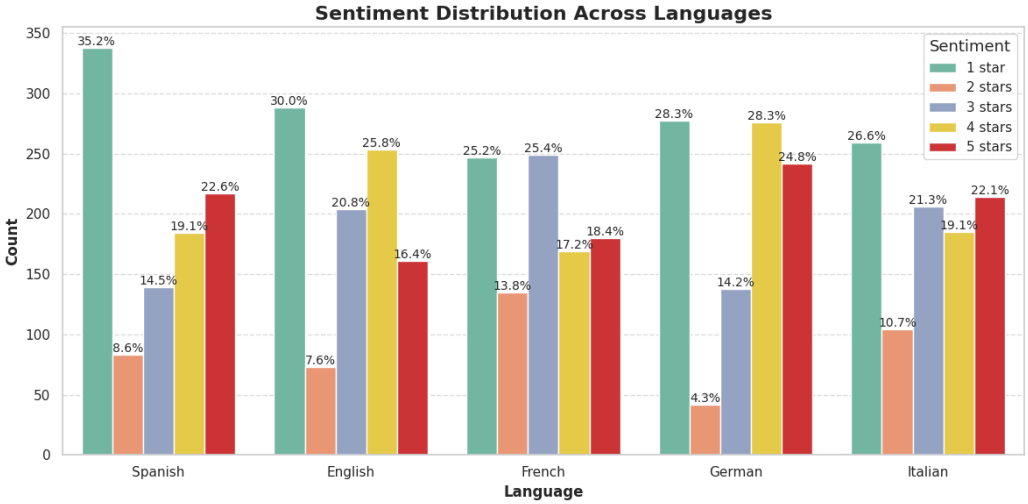
```
# Display a sample of the processed data
tweetsData.head()
```

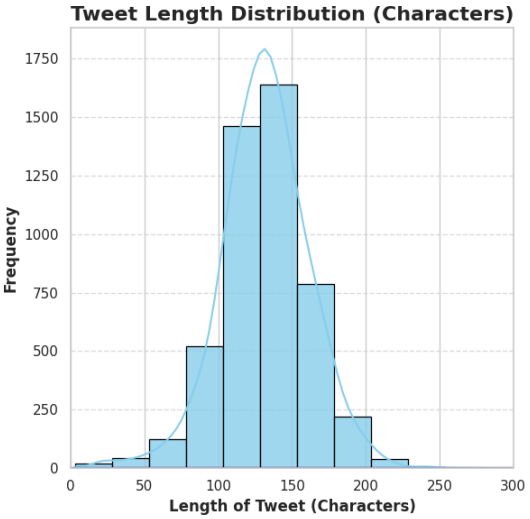
Python

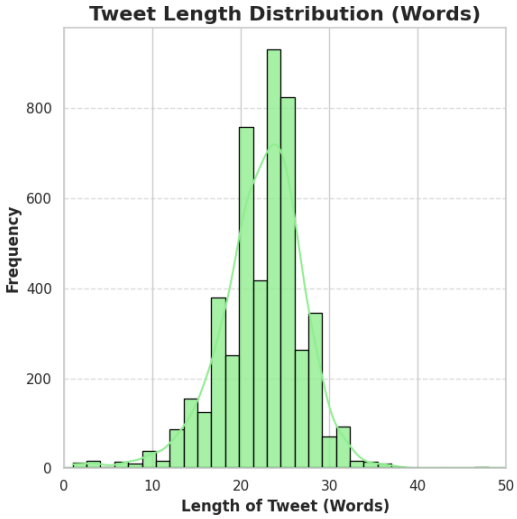
	tweet	language	sentiment	average_word_length	cleaned_tweet
0	Lionel Messi, que ha estado vinculado con un t...	Spanish	3 stars	4.833333	Lionel Messi, que ha estado vinculado con un t...
1	This is a guest post by The Joy of Truth. To r...	English	4 stars	3.423077	This is a guest post by The Joy of Truth. To r...
2	Nous sommes tous conscients de la popularité d...	French	5 stars	4.900000	Nous sommes tous conscients de la popularité d...
3	El baño en el sistema de metro de la ciudad de...	Spanish	4 stars	3.904762	El baño en el sistema de metro de la ciudad de...
4	Ich habe dies seit über 20 Jahren getan und ic...	German	5 stars	4.076923	Ich habe dies seit über 20 Jahren getan und ic...

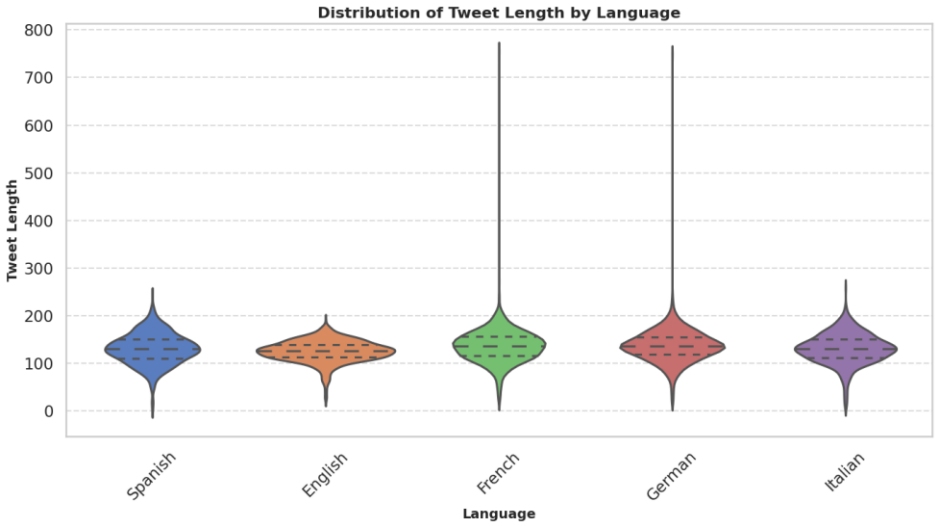
Visualize the Dataset

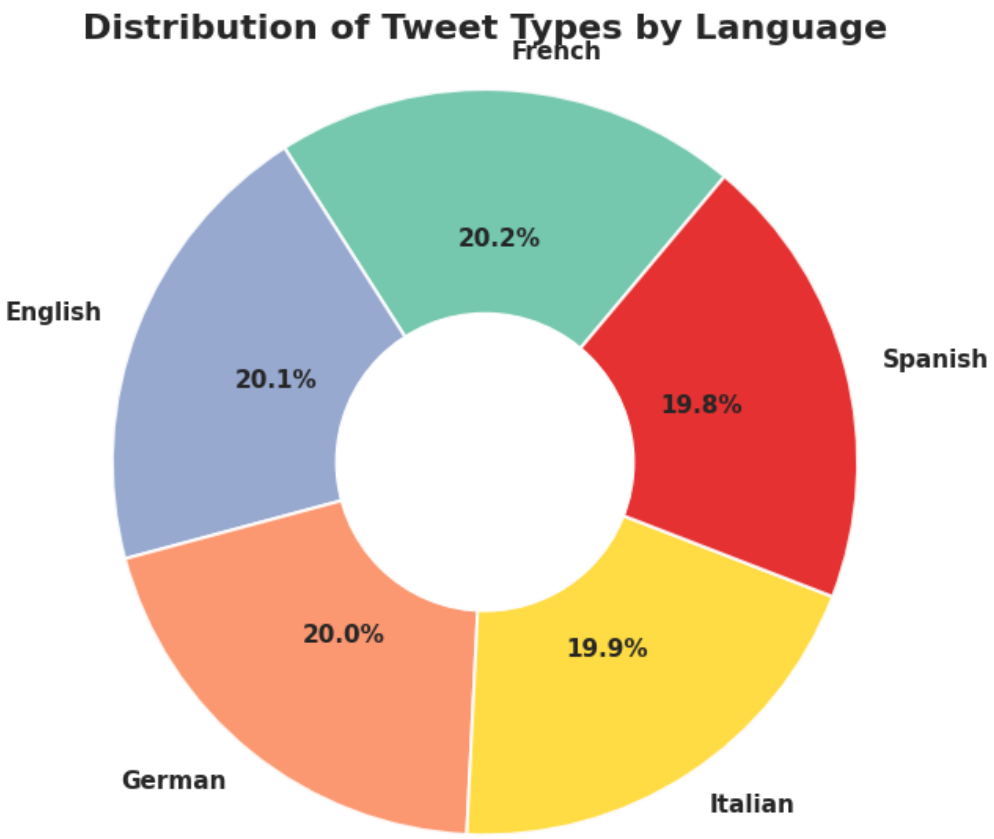



Sentiment Distribution Across Languages
Sentiment
1 star
2 stars
3 stars
4 stars
5 stars
Count
Language
Spanish
English
French
German
Italian
35.2%
8.6%
14.5%
19.1%
22.6%
30.0%
7.6%
20.8%
25.8%
16.4%
25.2%
13.8%
25.4%
17.2%
18.4%
28.3%
4.3%
14.2%
28.3%
24.8%
26.6%
10.7%
21.3%
19.1%
22.1%


Tweet Length Distribution (Characters)
Frequency
Length of Tweet (Characters)


Tweet Length Distribution (Words)
Frequency
Length of Tweet (Words)


Distribution of Tweet Length by Language
Tweet Length
Language
Spanish
English
French
German
Italian



Model Feature Engineering & Splitting of the Dataset

```
# Encoding sentiments to numerical labels
label_encoder = LabelEncoder()
tweetsData['sentiment_encoded'] = label_encoder.fit_transform(tweetsData['sentiment'])
```

- **Split the data into training (70%) and testing (30%) sets**

```
# Split the data into training (70%) and testing (30%) sets
train_data_70, test_data_70 = train_test_split(tweetsData, test_size=0.3, random_state=42,

# Check the shape of the training and test sets
train_data_70.shape, test_data_70.shape
```

```
((3404, 9), (1459, 9))
```

- **Split the data into training (75%) and testing (25%) sets**

```
# Split the data into training (75%) and testing (25%) sets
train_data_75, test_data_75 = train_test_split(tweetsData, test_size=0.25, random_state=42,

# Check the shape of the training and test sets
train_data_75.shape, test_data_75.shape
```

```
((3647, 9), (1216, 9))
```

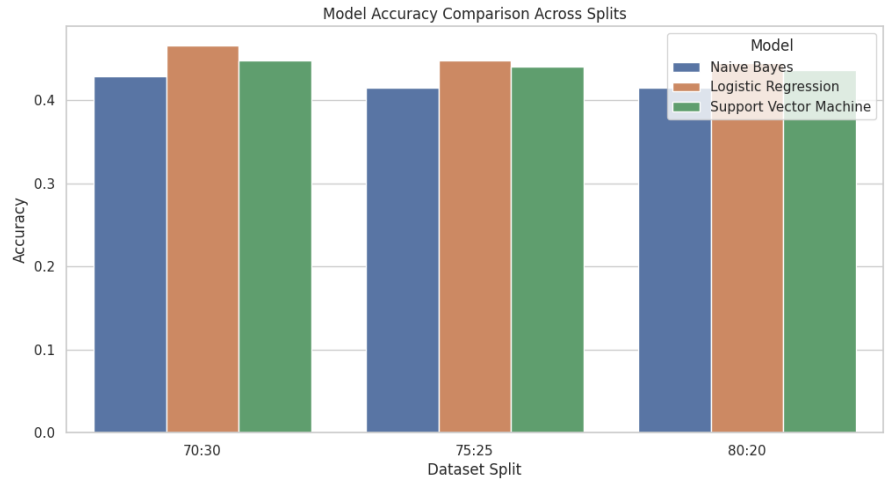
- **Split the data into training (80%) and testing (20%) sets**

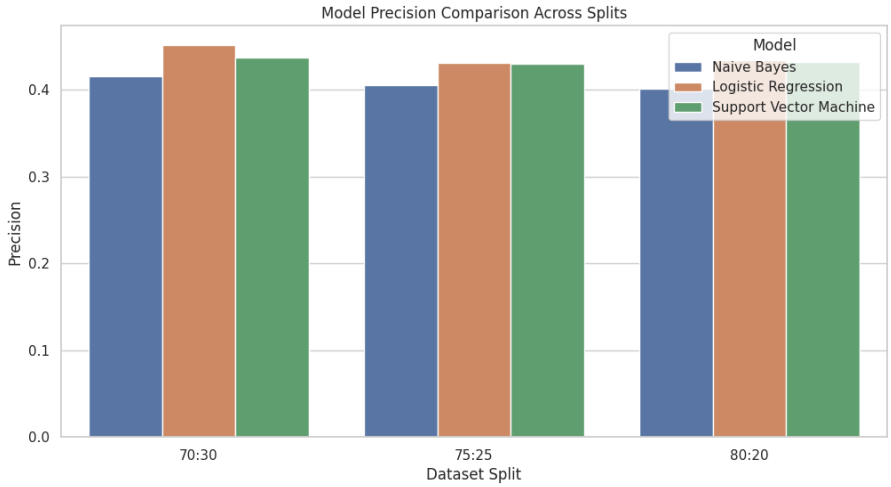
```
# Split the data into training (80%) and testing (20%) sets
train_data_80, test_data_80 = train_test_split(tweetsData, test_size=0.2, random_state=42,

# Check the shape of the training and test sets
train_data_80.shape, test_data_80.shape
```

```
((3890, 9), (973, 9))
```

Model Training & Evaluation of Baseline Models

```
# Train and evaluate each baseline model
for model_name, model in baseline_models.items():
    model.fit(X_train_70, train_data_70['sentiment_encoded'])
    predictions_70 = model.predict(X_test_70)
    accuracy, precision, recall, f1 = collect_metrics(true_labels_70, predictions_70)
    metrics_summary["Split"].append("70:30")
    metrics_summary["Model"].append(model_name)
    metrics_summary["Accuracy"].append(accuracy)
    metrics_summary["Precision"].append(precision)
    metrics_summary["Recall"].append(recall)
    metrics_summary["F1-Score"].append(f1)

    # Classification report
    print(f"Classification Report for {model_name} (70:30 split):")
    print(classification_report(true_labels_70, predictions_70))

    # Confusion Matrix
    conf_matrix_70 = confusion_matrix(true_labels_70, predictions_70)
    print(f"Confusion Matrix for {model_name} (70:30 split):")
    print(conf_matrix_70)

    # Plot Confusion Matrix
    plt.figure(figsize=(10, 7))
    sns.heatmap(conf_matrix_70, annot=True, fmt='d', cmap='Blues', xticklabels=label_encoder.classes_,
    plt.ylabel('Actual')
    plt.xlabel('Predicted')
    plt.title(f'Confusion Matrix for {model_name} (70:30 split)')
    plt.show()
```





Model Training and Evaluation of the Hybrid Model

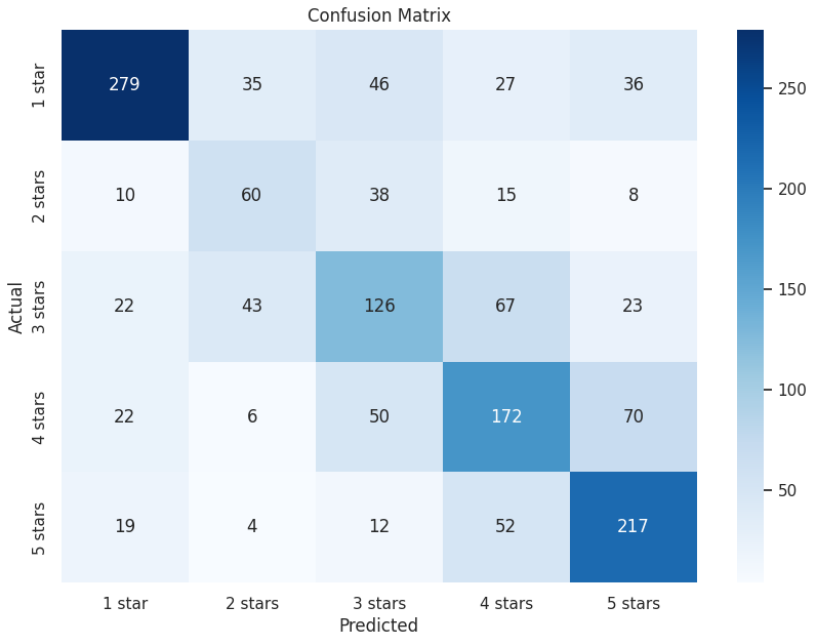
```python
# Initialization & Load  the pre-trained xlm-roberta of the Transformer model
model = XLMRobertaForSequenceClassification.from_pretrained('xlm-roberta-base', num_labels=len(label_encod

# Set up the optimizer and learning rate scheduler
optimizer = AdamW(model.parameters(), lr=5e-5)
```

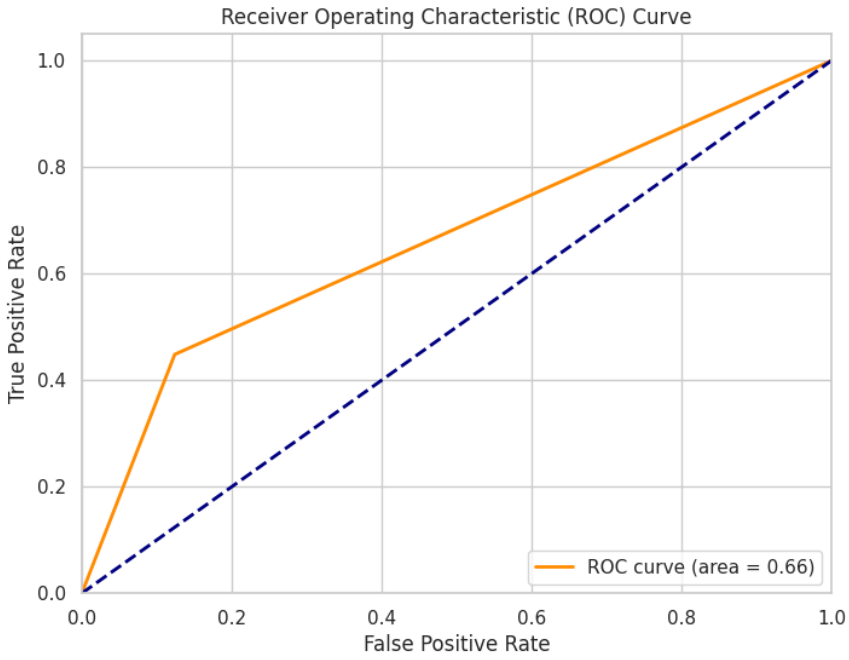
Python

```python
# Training the model with a meta-learning concept
def train_model_with_meta_learning(model, train_loader, optimizer, device, epochs=10):
    model.to(device)
    model.train()
    for epoch in range(epochs):
        total_loss = 0
        for batch in tqdm(train_loader):
            optimizer.zero_grad()
            input_ids, attention_mask, labels = [b.to(device) for b in batch]
            # Ensure labels are of type Long
            labels = labels.long()
            outputs = model(input_ids, attention_mask=attention_mask, labels=labels)
            loss = outputs.loss
            total_loss += loss.item()
            loss.backward()
            optimizer.step()
        print(f"Epoch {epoch + 1}/{epochs} - Loss: {total_loss / len(train_loader)}")
```

Python

```
100%|██████████| 213/213 [00:46<00:00,  4.60it/s]
Epoch 1/10 - Loss: 1.452662427100777
100%|██████████| 213/213 [00:47<00:00,  4.51it/s]
Epoch 2/10 - Loss: 1.1145792304070343
100%|██████████| 213/213 [00:49<00:00,  4.32it/s]
Epoch 3/10 - Loss: 0.9244181732056846
100%|██████████| 213/213 [00:49<00:00,  4.28it/s]
Epoch 4/10 - Loss: 0.6977067842310023
100%|██████████| 213/213 [00:49<00:00,  4.29it/s]
Epoch 5/10 - Loss: 0.5343602915674868
100%|██████████| 213/213 [00:49<00:00,  4.29it/s]
Epoch 6/10 - Loss: 0.36227346418049416
100%|██████████| 213/213 [00:49<00:00,  4.29it/s]
Epoch 7/10 - Loss: 0.30436824678395275
100%|██████████| 213/213 [00:49<00:00,  4.29it/s]
Epoch 8/10 - Loss: 0.24144973416983243
100%|██████████| 213/213 [00:49<00:00,  4.28it/s]
Epoch 9/10 - Loss: 0.24951044324033417
100%|██████████| 213/213 [00:49<00:00,  4.29it/s]
Epoch 10/10 - Loss: 0.1916498545067551
```






```python
# function to predict the sentiment
def predict_sentiment(text):
    # Tokenize and prepare inputs
    inputs = tokenizer(text, truncation=True, padding=True, return_tensors='pt')
    inputs.to(device='cuda' if torch.cuda.is_available() else 'cpu')

    # Get predictions without computing gradients
    with torch.no_grad():
        outputs = model(**inputs)
        logits = outputs.logits

        # Get the predicted sentiment class (e.g., 0, 1, 2, etc.)
        predicted_class = torch.argmax(logits, dim=1).item()

        # Map the predicted class to the corresponding star rating
        star_rating = map_sentiment_to_stars(predicted_class)

        return star_rating

def map_sentiment_to_stars(predicted_class):
    # Map the predicted class to star rating
    sentiment_map = {
        0: '1 star',  #  0 maps to 1 star
        1: '2 stars', #  1 maps to 2 stars
        2: '3 stars', #  2 maps to 3 stars
        3: '4 stars', #  3 maps to 4 stars
        4: '5 stars'  #  4 maps to 5 stars
    }

    # Handle the case where the sentiment prediction is outside expected bounds
    return sentiment_map.get(predicted_class, 'Unknown sentiment')

# Predict the sentiment for the first tweet
new_tweet = "This is a guest post by The Joy of Truth. To read more of her essays on the topic, check out
predicted_sentiment = predict_sentiment(new_tweet)
print(f'Multilingual Predicted sentiment: {predicted_sentiment}')  # 1 for 1 star, 2 for 2 star, 3 for 3
```

Conclusion

This manual guides you through the setup and training of traditional machine learning models and a Hybrid Transformer Model for sentiment analysis. By following the configuration steps, you will be able to run sentiment analysis experiments and evaluate model performance effectively.

References

Python: https://www.python.org
Dataset Source: Kaggle Link