Final Project NLP

DATASET

Enlace de kaggle de la data:

https://www.kaggle.com/datasets/aadyasingh55/twitter-emotion-classification-dataset/code

Preprocesamiento de Datos de Twitter para Clasificación de Emociones

Este documento describe el proceso de preprocesamiento de datos de Twitter para un modelo de clasificación de emociones. Se proporciona un script en Python que realiza la conversión de los datos a un formato adecuado para ser utilizado por un modelo de lenguaje. A continuación, se explica el flujo detallado de código y los pasos necesarios para llevar a cabo esta tarea.

Descripción del Script

El script tiene dos funciones principales:

1. **Dividir el conjunto de datos en entrenamiento y validación**
2. **Preprocesar los datos para formatearlos adecuadamente para la clasificación de emociones**

El código divide los datos proporcionados en un conjunto de entrenamiento y uno de validación. Luego, realiza una conversión de etiquetas a emociones, así como un formato de salida adecuado que puede ser utilizado por un modelo de lenguaje.

Estructura del Código

Importación de Bibliotecas

```
import pandas as pd
```

Se utiliza la biblioteca `pandas` para manejar y procesar los datos.

Definición de la Función `preprocess_twitter_data`

```
def preprocess_twitter_data(input_parquet, output_file):
    label_to_emotion = {
        0: "sadness",
        1: "joy",
        2: "love",
        3: "anger",
        4: "fear",
        5: "surprise"
```

```
    }

    df = pd.read_csv(input_parquet)
    df['emotion'] = df['label'].map(label_to_emotion)
    df['formatted'] = df['emotion'].apply(lambda x: f"<{x}>") + " " +
df['text']

    with open(output_file, 'w', encoding='utf-8') as f:
        f.write("\n".join(df['formatted']))

    print(f"Datos preprocesados guardados en {output_file}")
```

La función `preprocess_twitter_data` realiza las siguientes tareas:

- Mapea los valores de las etiquetas (`label`) a emociones (“joy”, “sadness”, etc.) usando el diccionario `label_to_emotion`.
- Formatea los datos agregando una etiqueta textual de la emoción antes del texto original, por ejemplo: `<joy> texto_del_tweet`.
- Escribe los datos preprocesados en un archivo de texto.

Ejecución Principal

```
if __name__ == "__main__":
    input_parquet = "corpus/train-00000-of-00001.parquet"
    output_file_train = "data/train.txt"
    output_file_valid = "data/valid.txt"

    df = pd.read_parquet(input_parquet)
    train_df = df.sample(frac=0.8, random_state=42)
    valid_df = df.drop(train_df.index)

    train_df.to_csv("data/train_data.csv", index=False)
    valid_df.to_csv("data/valid_data.csv", index=False)

    preprocess_twitter_data("data/train_data.csv", output_file_train)
    preprocess_twitter_data("data/valid_data.csv", output_file_valid)
```

En la sección principal del código se lleva a cabo lo siguiente:

1. **Cargar el archivo de entrada**: Se carga un archivo `parquet` que contiene los datos originales.
2. **Dividir los datos**: Los datos se dividen en un conjunto de entrenamiento (80%) y un conjunto de validación (20%).
3. **Guardar conjuntos de entrenamiento y validación**: Se exportan los conjuntos resultantes a archivos CSV.
4. **Preprocesar y guardar los datos formateados**: Los conjuntos de entrenamiento y validación se formatean y se guardan en archivos de texto (`train.txt` y `valid.txt`).

Archivos Resultantes

- **data/train_data.csv**: Contiene los datos de entrenamiento en formato CSV.
- **data/valid_data.csv**: Contiene los datos de validación en formato CSV.
- **data/train.txt**: Conjunto de entrenamiento preprocesado en un formato adecuado para el modelo.
- **data/valid.txt**: Conjunto de validación preprocesado en un formato adecuado para el modelo.