Introduction

Social media platforms generate vast amounts of text data daily, providing valuable insights into public opinion, sentiment trends, and user behavior. Understanding and classifying sentiment in tweets can help businesses, policymakers, and researchers analyze public reactions to events, products, or policies in real-time. Prior studies have demonstrated the efficacy of Natural Language Processing (NLP) techniques in sentiment classification, with models such as BERT and LSTMs achieving high accuracy in Twitter sentiment analysis (Camacho-Collados et al., 2020; Mohammad, 2018). This project aims to utilize the TweetEval dataset from Hugging Face, which provides a benchmark for various tweet classification tasks, including sentiment analysis. The dataset is well-suited for this analysis, as it is preprocessed and labeled, allowing for efficient implementation of supervised machine learning models. Our goal is to explore different NLP approaches, such as TF-IDF + Logistic Regression, Transformer-based models (e.g., BERT), and Neural Networks, to classify the sentiment of tweets into predefined categories (positive, negative, neutral).

Research Question

To guide our analysis, we pose the following research question:

"How effectively can different machine learning models classify the sentiment of tweets using the TweetEval dataset, and what features contribute most to classification accuracy?"

2. Method

2.1 Corpus

The dataset used in this study is the **TweetEval dataset**, a benchmark dataset for Twitter-related Natural Language Processing (NLP) tasks (Camacho-Collados et al., 2020). This dataset consists of multiple sub-tasks, with **sentiment analysis** being the focus of this research.

- **Dataset Statistics**:
  - The dataset contains thousands of tweets, each labeled with one of three sentiment categories:
    - **Positive (1)**
    - **Negative (-1)**
    - **Neutral (0)**
  - The dataset is **balanced**, ensuring that each sentiment class has a reasonable representation.
- **Data Source & Preprocessing**:
  - The data is sourced from **Twitter posts** and preprocessed for NLP tasks.
  - Each sample consists of a **tweet text and its sentiment label**.
  - The dataset has been **cleaned and tokenized**, making it suitable for direct implementation in sentiment classification models.

This dataset serves as a **rich and diverse** resource for training and evaluating NLP models for sentiment classification.

2.2 TF-IDF Features

Term Frequency-Inverse Document Frequency (TF-IDF) is a well-established technique in NLP for text vectorization. It converts textual data into numerical representations based on word frequency and importance within a document.

- **TF (Term Frequency):** Measures how often a word appears in a document, normalized by the total number of words in that document.
- **IDF (Inverse Document Frequency):** Weighs words based on their rarity across the entire corpus. Less frequent words receive higher importance.
- **Implementation:**
  - We apply **sklearn's TfidfVectorizer** with **n-gram features (unigram, bigram)** for capturing meaningful word dependencies.
  - **Stopword removal** is performed to eliminate common words that do not contribute to sentiment classification.
  - **Normalization techniques**, such as **L2 norm**, are applied to improve feature representation.

The TF-IDF features provide a **strong baseline representation** for traditional machine learning models like **Logistic Regression, Naïve Bayes, and Random Forest**.

2.3 Word Embeddings

To capture **contextual meaning and semantic relationships**, we incorporate **word embeddings** such as **Word2Vec and Transformer-based embeddings (BERT)**.

- **BERT Embeddings:**

- We fine-tune the **BERT-based model (bert-base-uncased)** from the Hugging Face library.
- The model extracts **context-aware vector representations**, improving classification accuracy.

- **Word2Vec & FastText:**
  - Pre-trained embeddings trained on large Twitter datasets are used for feature extraction.
  - These embeddings capture **word relationships and sentiment tendencies** effectively.

2.4 Model Training & Evaluation

To assess sentiment classification performance, we train and evaluate multiple models:

2.4.1 Traditional Machine Learning Models

- **Logistic Regression:** Serves as a baseline classifier using **TF-IDF features**.
- **Naïve Bayes & Random Forest:** Evaluated for their efficiency in text classification tasks.

2.4.2 Deep Learning Models

- **LSTM (Long Short-Term Memory):** Utilized for sequential processing of tweet text.
- **BERT-based Transformer Model:** Fine-tuned for sentiment classification using transfer learning.

Each model is trained using **cross-validation**, and hyperparameter tuning is conducted via **GridSearchCV** for optimal performance.

3.Data Prepocessing

The data preprocessing pipeline is designed to clean and prepare the TweetEval dataset for sentiment analysis. The preprocessing steps include removing noise, tokenizing text, and extracting relevant features to enhance the performance of machine learning models.

3.1 Data Cleaning Steps

The following preprocessing steps are applied to ensure high-quality text data:

- Removing Missing or Corrupt Data:Tweets with missing sentiment labels are dropped to ensure proper supervised learning. Non-English tweets are removed to maintain dataset consistency. Filtering Non-Relevant Text Elements:
- Removing URLs, User Mentions, and Hashtags: These elements do not contribute to sentiment classification. Eliminating Special Characters & Emojis: Standardizing textual content for processing.
- Tokenization & Text Normalization: Splitting each tweet into individual words using spaCy's English tokenizer. Lowercasing all words to maintain uniformity. Removing punctuation, numbers, and non-alphabetic characters to focus on textual sentiment.

3.2 NLP analysis

- Feature Extraction: TF-IDF: Converting text into numerical vectors based on word frequency. Word Embeddings: Using pre-trained embeddings (e.g., BERT, Word2Vec) to capture contextual meaning.
- Model Selection & Training: Baseline Model: Logistic Regression with TF-IDF features. Deep Learning Model: Fine-tuned BERT-based sentiment classifier. Hybrid Model: Neural network trained on word embeddings.
- Model Evaluation Accuracy, Precision, Recall, and F1-score for each model. Confusion Matrices to visualize classification performance. Error Analysis: Examining misclassified tweets. First I install datasets to load the huggingface datasets. Then let's see what the dataset looks like.

```
! pip install datasets
```

```
from datasets import load_dataset

ds = load_dataset("cardiffnlp/tweet_eval", "sentiment")
```

```
# Data Manipulation Libraries
import pandas as pd
import numpy as np
import string
from itertools import chain, cycle

# Visualization Libraries
import matplotlib.pyplot as plt
import seaborn as sns
from IPython.display import display_html
```

```
# Machine Learning & NLP Libraries
from sklearn.feature_extraction.text import TfidfVectorizer, CountVectorizer
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.naive_bayes import MultinomialNB
from sklearn.svm import SVC

# NLP Libraries
import spacy
from spacy.lang.en.stop_words import STOP_WORDS
```

Here is the example of this dataset.

```
for i in range(5):
    print(ds["train"][i])
```

```
:ext': '"QT @user In the original draft of the 7th book, Remus Lupin survived the Battle of Hogwarts. #HappyBirthdayRemusLu
:ext': '"Ben Smith / Smith (concussion) remains out of the lineup Thursday, Curtis #NHL #SJ"', 'label': 1}
:ext': 'Sorry bout the stream last night I crashed out but will be on tonight for sure. Then back to Minecraft in pc tomorr
:ext': "Chase Headley's RBI double in the 8th inning off David Price snapped a Yankees streak of 33 consecutive scoreless i
:ext': '@user Alciato: Bee will invest 150 million in January, another 200 in the Summer and plans to bring Messi by 2017"'
```

```
# 3.3 Setting Up NLP Pipeline
# The NLP pipeline is configured using spaCy to tokenize, clean, and preprocess tweets.
# Load English tokenizer, disable unnecessary components
nlp = spacy.load("en_core_web_sm", exclude=["parser", "ner"])

# Define stopwords (adding extra domain-specific stopwords)
STOP_WORDS.add("rt")  # Common Twitter-specific stopword
stop_words = STOP_WORDS

# Function to preprocess text
def tokenize_docs(doc):
    tokens = []
    for tok in doc:
        if (not tok.is_punct
            and not tok.is_space
            and not tok.like_num
            and tok.is_alpha
            and len(tok.text) > 1
            and tok not in stop_words):
                tokens.append(tok.lemma_.lower())
    return tokens
```

```
/usr/local/lib/python3.11/dist-packages/spacy/util.py:1740: UserWarning: [W111] Jupyter notebook detected: if using `prefe
  warnings.warn(Warnings.W111)
```

4.Conclusion

This study applies various machine learning models to sentiment analysis on Twitter data. It aims to determine which approach yields the highest accuracy while maintaining interpretability. The results will inform future applications of NLP in social media monitoring, brand sentiment tracking, and opinion mining.

5. Limitations & Future Directions

Limitations: Data Bias: The dataset is limited to tweets, which may not generalize to other text sources. Contextual Challenges: Sarcasm and irony are difficult to detect in sentiment classification. Computational Cost: Transformer-based models require significant computing power. Future Directions: Expanding to Multilingual Sentiment Analysis: Applying models to non-English tweets. Exploring Multimodal Approaches: Combining text with images/videos for richer sentiment analysis. Fine-tuning for Industry-Specific Sentiment Analysis: Adapting models for sectors like finance or healthcare.

References

Camacho-Collados, J., et al. (2020). TweetEval: A Unified Benchmark for Classification of Tweets. Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing.

Mohammad, S. (2018). Sentiment Analysis: Detecting Valence, Emotions, and Other Affectual States from Text. Emotion Measurement.