Sentiment Analysis for Financial News using Deep Learning

Megha Raj

1 Project Overview

The project aims to develop a deep learning model to analyze and classify the sentiment of financial news related to Apple Inc. (AAPL). By using natural language processing (NLP) and deep learning techniques, the project seeks to derive actionable insights from news data, which can assist in making informed investment decisions.

2 Sub-objectives and Methodology

2.1 Data Collection and Preprocessing

Data Collection:

```
import requests
import pandas as pd

ticker = "AAPL"
url = f'https://www.alphavantage.co/query?function=NEWS_SENTIMENT&
    tickers={ticker}&sort=LATEST&limit=1000&apikey=YOUR_API_KEY'
r = requests.get(url)
data = r.json()
summaries_list = [item['summary'] for item in data["feed"]]
summaries_df = pd.DataFrame(summaries_list, columns=['Summary'])
```

Description: News summaries related to AAPL are retrieved using the Alpha Vantage API. These summaries are collected into a list and then converted into a DataFrame for further analysis. This approach allows aggregation of relevant news data for sentiment analysis.

Text Cleaning:

```
import re
import nltk
from nltk.corpus import stopwords
from nltk.stem.wordnet import WordNetLemmatizer

nltk.download('stopwords')
nltk.download('wordnet')
stop_words = set(stopwords.words('english'))
lmtzr = WordNetLemmatizer()

def pre_process(text):
    text = text.lower()
```

```
    text = re.sub(r'\w+:\/{2}[\d\w-]+(\.[\d\w-]+)*(?:(?:\/[^\s/]*))
        *', '', text)
    text = re.sub("&lt;/?.*?&gt;", " &lt;&gt; ", text)
    text = re.sub("(\\d|\\W)+", " ", text)
    text = [word for word in text.split() if word not in stop_words
        and len(word) >= 3]
    text = [lmtzr.lemmatize(word) for word in text]
    return ' '.join(text)

summaries_df['Summary'] = summaries_df['Summary'].apply(pre_process)
```

Description: Text preprocessing is essential for cleaning and standardizing the news summaries. The text is converted to lowercase, URLs and HTML tags are removed, and irrelevant characters are filtered out. Stopwords are eliminated, and lemmatization is applied to reduce words to their base forms. This ensures the text data is in a suitable format for modeling.

2.2 Model Building and Training

Data Preparation:

```
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from keras.preprocessing.text import Tokenizer
from keras.preprocessing.sequence import pad_sequences

# Concatenating datasets
source = pd.concat([source_1_df, source_2_df], ignore_index=True)

# Splitting data
X_train, X_test, y_train, y_test = train_test_split(source['text'],
    source['label'], test_size=0.3, random_state=18)

# TF-IDF Vectorization
vectorizer = TfidfVectorizer(max_features=5000)
X_train_vec = vectorizer.fit_transform(X_train)
X_test_vec = vectorizer.transform(X_test)

# Tokenization and Padding
tokenizer = Tokenizer(num_words=5000)
tokenizer.fit_on_texts(source['text'])
X_train_seq = tokenizer.texts_to_sequences(X_train)
X_test_seq = tokenizer.texts_to_sequences(X_test)
X_train_pad = pad_sequences(X_train_seq, maxlen=100)
X_test_pad = pad_sequences(X_test_seq, maxlen=100)
```

Description: The data is prepared for the model by combining datasets, splitting into training and testing sets, and converting text data into numerical features using TF-IDF and tokenization. Padding ensures that all text sequences are of uniform length, which is crucial for training deep learning models.

Model Building:

```
from keras.models import Sequential
from keras.layers import Dense, Embedding, LSTM

model = Sequential()
model.add(Embedding(5000, 128, input_length=100))
```

```
model.add(LSTM(128, dropout=0.2, recurrent_dropout=0.2))
model.add(Dense(3, activation='softmax'))
model.compile(loss='sparse_categorical_crossentropy', optimizer='
    adam', metrics=['accuracy'])
history = model.fit(X_train_pad, y_train, epochs=10, batch_size=64,
    validation_data=(X_test_pad, y_test))
```

Description: A Sequential neural network model is built with an embedding layer to convert word indices into dense vectors, an LSTM layer to capture sequential dependencies in the text, and a dense output layer with softmax activation to classify sentiments into three categories. The model is trained using the prepared data, with metrics monitored to evaluate performance.

2.3 Model Evaluation and Application

Evaluation:

```
from sklearn.metrics import classification_report, confusion_matrix
import matplotlib.pyplot as plt

loss, accuracy = model.evaluate(X_test_pad, y_test)
print(f'Test Loss: {loss}')
print(f'Test Accuracy: {accuracy}')

y_pred = model.predict(X_test_pad)
y_pred_classes = y_pred.argmax(axis=1)
print(classification_report(y_test, y_pred_classes))
print(confusion_matrix(y_test, y_pred_classes))

plt.figure(figsize=(12, 4))
plt.subplot(1, 2, 1)
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('Model accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.legend(['Train', 'Validation'], loc='upper left')

plt.subplot(1, 2, 2)
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('Model loss')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.legend(['Train', 'Validation'], loc='upper left')

plt.show()
```

Description: The model's performance is evaluated on the test data using accuracy and loss metrics. Classification reports and confusion matrices provide detailed insights into model performance, including precision, recall, and F1-score. Training and validation accuracy/loss are plotted to visualize the learning process.

Sentiment Prediction:

```
new_texts = summaries_df['Summary'].tolist()
new_texts_seq = tokenizer.texts_to_sequences(new_texts)
new_texts_pad = pad_sequences(new_texts_seq, maxlen=100)
```

```
predictions = model.predict(new_texts_pad)
predicted_classes = predictions.argmax(axis=1)

summaries_df['Predicted_Label'] = predicted_classes
sentiments = {0: "Bearish", 1: "Bullish", 2: "Neutral"}
summaries_df['Sentiment'] = summaries_df['Predicted_Label'].map(
    sentiments)
```

Description: The trained model is applied to new news summaries to predict sentiments. Predictions are converted into human-readable labels ("Bearish", "Bullish", "Neutral") and added to the DataFrame. This helps in assessing the overall sentiment towards AAPL stock.

3 Challenges and Roadblocks

3.1 Twitter Data Access

Issue: Limited access to Twitter data due to subscription constraints and API limitations.

Impact: Unable to incorporate recent tweets, affecting the richness and timeliness of the dataset.

3.2 Web Scraping Limitations

Issue: Full articles could not be retrieved due to web scraping restrictions and potential IP blocking.

Impact: Limited availability of complete articles reduced the dataset's completeness and depth.

3.3 NewsAPI Limitations

Issue: NewsAPI could not be utilized effectively as it did not interpret ticker symbols accurately, and using generic terms like "apple" returned unrelated articles.

Impact: The relevance of collected news articles was compromised, affecting the quality of sentiment analysis.

4 Future Work

4.1 Deep Learning Enhancements

Objective: Improve the performance of the deep learning model through advanced fine-tuning and hyperparameter tuning.

Approach: Experiment with various hyperparameters and optimization techniques to enhance model accuracy and robustness.

4.2 CNN for Sentiment Weight Prediction

Objective: Explore the use of Convolutional Neural Networks (CNNs) to predict weights used in sentiment score calculation.

Approach: Develop and test CNN models to capture local patterns and features in text data, potentially improving sentiment classification. Compare CNN performance with LSTM-based models to determine their relative effectiveness.

4.3 Data Expansion

Objective: Increase the diversity and volume of data by adding more sources such as financial reports, additional news sources, and social media platforms.

Approach: Integrate data from various sources and ensure that the data aggregation process adheres to legal and ethical guidelines. Utilize APIs and web scraping techniques, where permissible, to gather a broader range of information.

5 Conclusion

This project successfully developed and implemented a deep learning-based sentiment analysis model for financial news related to AAPL. By leveraging natural language processing and deep learning techniques, the project provided valuable insights into the sentiment surrounding the stock. Despite challenges related to data access and integration, the project achieved its primary goal of classifying news sentiment. The outlined future work aims to further enhance the model's performance, expand data sources, and improve interpretability and robustness, potentially leading to more informed investment decisions and advanced financial analysis tools.