Max Liu

Prof. Hernandez

CISB 62

22 October 2023

Reuters Newswire Classification

Project Overview

The Reuters Newswire Classification project leverages deep learning techniques to categorize news articles into predefined topics. The project encompasses several phases, including data preprocessing, model architecture, training, and performance analysis. It also explores advanced Deep Learning techniques to enhance the classification accuracy.

1. Data Preparation

```
# Load the Reuters dataset
from tensorflow.keras.datasets import reuters

# Load the dataset and specify the number of top words to consider
max_words = 10000
(x_train, y_train), (x_test, y_test) =
reuters.load_data(num_words=max_words)

# Load the word index for mapping words to integers
word_index = reuters.get_word_index(path="reuters_word_index.json")

# Reverse the word index to map integers to words
reverse_word_index = dict([(value, key) for (key, value) in
word_index.items()])
```

In this section, the Reuters dataset is loaded using TensorFlow and Keras. The dataset is preprocessed for text classification, and the top words (maximum vocabulary size) are specified. Additionally, the word index is reversed to map integers back to words for text decoding.

2. Deep Learning Techniques

```
# Build a neural network model for text classification
model = keras.Sequential([
    keras.layers.Embedding(input_dim=max_words, output_dim=128,
input_length=max_sequence_length),
    keras.layers.LSTM(64, dropout=0.2, recurrent_dropout=0.2),
    keras.layers.Dense(64, activation='relu'),
    keras.layers.Dense(46, activation='softmax')
])
```

This section outlines the application of deep learning techniques. The model architecture is designed to include embedding layers, LSTM layers, and dense layers. Techniques such as dropout and activation functions like ReLU are employed to enhance the model's text classification capabilities.

3. Model Configuration

```
# Compile the model
model.compile(optimizer='adam',
              loss='categorical_crossentropy',
              metrics=['accuracy'])

# Define callbacks for TensorBoard and ModelCheckpoint
tensorboard_callback = keras.callbacks.TensorBoard(log_dir="logs",
histogram_freq=1)
model_checkpoint = keras.callbacks.ModelCheckpoint("best_model.h5",
save_best_only=True)
```

This section focuses on configuring the neural network model. The model is compiled with the Adam optimizer and categorical cross-entropy loss. Callbacks for TensorBoard and model checkpointing are defined to monitor and save the training progress.

4. Training the Model

```
# Train the model with callbacks
history = model.fit(x_train, y_train, epochs=10, batch_size=64,
                    validation_data=(x_val, y_val),
                    callbacks=[tensorboard_callback,
model_checkpoint])
```

The model is trained using the prepared dataset. The code demonstrates how training is performed with specified epochs, batch size, and callbacks for validation and model checkpointing.

5. Performance Analysis

```
# Generate a confusion matrix
y_pred = model.predict(x_test)
y_pred_classes = np.argmax(y_pred, axis=1)
y_test_classes = np.argmax(y_test, axis=1)
confusion = confusion_matrix(y_test_classes, y_pred_classes)
```

This section focuses on performance analysis. A confusion matrix is generated to assess the classification performance. Key metrics such as precision, recall, and F1 score are analyzed, and the confusion matrix is visualized using Seaborn.

References:

DATA FROM: *https://keras.io/api/datasets/reuters/#reuters-newswire-classification-dataset*

GITHUB Link: https://github.com/NMOD-Max/Reuters-Newswire-Classification