DMLab2 Homework report

110034007 謝欣妤

Prepare the Data

I loaded and combined tweet data from JSON and CSV files using pandas. The datasets were merged on tweet_id to include text, emotion, hashtags, and identification columns. Then, I split the data into training and test sets based on the identification column, keeping only the necessary fields: tweet_id, text, and emotion.

Pre-Processing

The preprocessing stage ensured that the data was clean and ready for modeling. First, I confirmed that there were no missing values in the dataset. Then, a custom text-cleaning function was applied to normalize the text by converting it to lowercase, removing URLs, mentions, hashtags, special characters, and excessive whitespace. This step reduced noise and preserved meaningful content.

Stopwords, which are common words that do not contribute to the meaning of a sentence, were removed using the NLTK library to focus on significant terms. This helped reduce dimensionality and improved the efficiency of subsequent analysis.

Emotion labels were encoded into numerical values using LabelEncoder to facilitate compatibility with machine learning algorithms. Finally, the dataset was split into training and validation sets, ensuring a clear division between data for learning and evaluation. These preprocessing steps collectively improved the dataset's quality and made it suitable for model training.

Model Training

For model training, we used a hybrid architecture combining embedding, convolutional, and recurrent layers to capture both local and sequential patterns in the text data.

I first tokenized the cleaned text data, converting words into integer sequences, and padded them to ensure consistent input lengths. This allowed the model to process text inputs of varying lengths effectively.

- **Model Architecture:**

Model: "sequential_1"

Layer (type)	Output Shape	Param #
embedding_1 (Embedding)	(None, 300, 128)	63,791,104
spatial_dropout1d_1 (SpatialDropout1D)	(None, 300, 128)	0
conv1d_1 (Conv1D)	(None, 296, 128)	82,048
batch_normalization_1 (BatchNormalization)	(None, 296, 128)	512
bidirectional_1 (Bidirectional)	(None, 296, 200)	183,200
global_max_pooling1d_1 (GlobalMaxPooling1D)	(None, 200)	0
dense_2 (Dense)	(None, 64)	12,864
dropout_1 (Dropout)	(None, 64)	0
dense_3 (Dense)	(None, 8)	520

Total params: 64,070,248 (244.41 MB)
Trainable params: 64,069,992 (244.41 MB)
Non-trainable params: 256 (1.00 KB)

  - **Embedding Layer:** Encoded words into dense vectors, preserving semantic relationships.
  - **SpatialDropout1D:** Regularized embeddings to prevent overfitting.
  - **Conv1D & BatchNormalization:** Captured local patterns in text (e.g., key phrases) and normalized activations for stable training.
  - **Bidirectional LSTM:** Leveraged context from both directions in the sequence, enhancing the model's ability to understand dependencies between words.
  - **GlobalMaxPooling:** Compressed LSTM outputs into a fixed-size representation.
  - **Dense Layers:** Learned non-linear combinations of features, with dropout added for regularization.
- **Callbacks:**
  - **ReduceLROnPlateau:** Adaptively reduced the learning rate to stabilize training.
  - **EarlyStopping:** Stopped training when validation accuracy stopped improving to save time and prevent overfitting.

```
9098/9098 ━━━━━━━━━━━━━━━━━━━━ 85s 9ms/step
              precision    recall  f1-score   support

           0       0.76      0.17      0.28      8062
           1       0.60      0.58      0.59     50129
           2       0.46      0.35      0.40     27789
           3       0.72      0.34      0.46     12679
           4       0.53      0.83      0.65    102986
           5       0.50      0.41      0.45     38793
           6       0.86      0.18      0.30      9790
           7       0.63      0.29      0.39     40885

    accuracy                           0.55    291113
   macro avg       0.63      0.39      0.44    291113
weighted avg       0.57      0.55      0.52    291113
```

Prediction

The trained model was used to predict emotions for the test dataset. Text data was tokenized and padded similarly to the training process. Predictions were converted back to their original labels using the label encoder. Finally, results were merged with the sample submission format and saved as csv file for submission.

Notes

I also experimented with Random Forest, but it was too slow and required significantly reducing the number of trees and max terms in the TF-IDF vectorizer to run within a reasonable time. However, reducing these parameters negatively impacted the model's performance, making it unsuitable for this task.