Predictive Maintenance using Deep Learning

Author(s)

Zainab Fayyaz 2020-EE-03
Zuha Nadeem 2020-EE-04
Asmara Rauf 2020-EE-102
Eman Fatima 2020-EE-105

Abstract

Presented is a project which introduces integration of Gated Recurrent Units (GRU) and Convolutional Neural Networks (CNN) into a Predictive Maintenance Model. The reason behind utilizing GRU is to grasp successional patterns in time series data, on the other hand CNN mines spatially observed features from sensor readings. The development model, therefore, aims at predicting equipment breakdowns and detecting maintenance requirements by using these structures. Based on actual data, the assessment of this prototype has shown that it can improve maintenance effectiveness and reduce operational expenses at all levels.

1 Steps involved in Model Development

Following are the steps involved in the model development for predictive maintenance:

1.1 Importing Necessary Libraries

In order to develop the model, we need to bring in a good number of packages such as pandas to manipulate our data, numpy for mathematical operations and matplotlib for graphing purposes. Similarly, we have also imported Sequential for building a sequential model, Dense for adding a fully connected layer, Dropout which helps during the training process and GRU in case we wish to add an extra layer. Likewise, we have Activation used to specify activation functions, Input creating input layers, Conv1D as an addition of 1D convolutional layer, MaxPooling1D that can add 1D max pooling layer, Flatten is meant for reshaping the inputs whereas concatenate is responsible for joining layers.

1.2 Loading the training and test dataset

The datasets for both validation and training are transformed into dataframes using the following code:

```
train_df = pd.read_csv('train_data.txt', sep=" ", header=None)
train_df.columns = ['id','cycle','setting1', 'setting2', 'setting3', 's1', 's2', 's3',
                    's4', 's5', 's6', 's7', 's8', 's9', 's10', 's11', 's12', 's13', 's14',
                    's15', 's16', 's17', 's18', 's19', 's20', 's21','RUL', 'label1', 'label2', 'cycle_norm']
train_df
```

```
test_df = pd.read_csv('test_data.txt', sep=" ", header=None)
test_df.columns =  ['id','cycle','setting1', 'setting2', 'setting3', 's1', 's2', 's3',
                    's4', 's5', 's6', 's7', 's8', 's9', 's10', 's11', 's12', 's13', 's14',
                    's15', 's16', 's17', 's18', 's19', 's20', 's21','RUL', 'label1', 'label2', 'cycle_norm']
test_df
```

1.3 Training and Evaluation

The model was trained using the training data with appropriate hyperparameters such as batch size, number of epochs, and validation split. Training metrics such as loss curves and validation accuracy were monitored to assess model convergence and generalization.

Evaluation was carried out on the test data to assess the model's performance in a real-world scenario. Metrics such as accuracy were calculated to measure the model's performance in binary classification. The results were compared with previous models to evaluate the effectiveness of the GRU and CNN architectures in predictive maintenance applications.

```
Epoch 1/10
70/70 ━━━━━━━━━━━━━━━━━━━━ 86s 1s/step - accuracy: 0.8602 - loss: 0.3325 - val_accuracy: 0.9372 - val_loss: 0.1722
Epoch 2/10
70/70 ━━━━━━━━━━━━━━━━━━━━ 81s 1s/step - accuracy: 0.9528 - loss: 0.1226 - val_accuracy: 0.9495 - val_loss: 0.1181
Epoch 3/10
70/70 ━━━━━━━━━━━━━━━━━━━━ 82s 1s/step - accuracy: 0.9751 - loss: 0.0685 - val_accuracy: 0.9658 - val_loss: 0.0716
Epoch 4/10
70/70 ━━━━━━━━━━━━━━━━━━━━ 85s 1s/step - accuracy: 0.9773 - loss: 0.0565 - val_accuracy: 0.9754 - val_loss: 0.0646
Epoch 5/10
70/70 ━━━━━━━━━━━━━━━━━━━━ 84s 1s/step - accuracy: 0.9783 - loss: 0.0523 - val_accuracy: 0.9686 - val_loss: 0.0662
Epoch 6/10
70/70 ━━━━━━━━━━━━━━━━━━━━ 84s 1s/step - accuracy: 0.9827 - loss: 0.0452 - val_accuracy: 0.9686 - val_loss: 0.0704
Epoch 7/10
70/70 ━━━━━━━━━━━━━━━━━━━━ 91s 1s/step - accuracy: 0.9818 - loss: 0.0437 - val_accuracy: 0.9699 - val_loss: 0.0594
Epoch 8/10
70/70 ━━━━━━━━━━━━━━━━━━━━ 83s 1s/step - accuracy: 0.9779 - loss: 0.0474 - val_accuracy: 0.9727 - val_loss: 0.0584
Epoch 9/10
70/70 ━━━━━━━━━━━━━━━━━━━━ 84s 1s/step - accuracy: 0.9786 - loss: 0.0498 - val_accuracy: 0.9781 - val_loss: 0.0698
Epoch 10/10
70/70 ━━━━━━━━━━━━━━━━━━━━ 85s 1s/step - accuracy: 0.9820 - loss: 0.0430 - val_accuracy: 0.9768 - val_loss: 0.0589

<keras.src.callbacks.history.History at 0x1b0067df950>
```

2 Results

2.1 Training Phase Results

During the training phase, the GRU and CNN-based models demonstrated strong learning capabilities, as evidenced by their convergence and improvement in training metrics such as loss reduction and validation accuracy. The models effectively learned from the training data, capturing temporal and spatial patterns in the sensor data.

2.2 Testing Phase Results

In the testing phase, the GRU and CNN models exhibited robust performance on unseen test data. The models maintained high accuracy showcasing their ability to generalize well and make accurate predictions in real-world scenarios. These results indicate the models' effectiveness in capturing complex features and detecting patterns related to equipment failures.

3 Conclusion

The results underscore the significance of leveraging advanced deep learning techniques like GRU and CNN for predictive maintenance applications. These models excel in automatically learning relevant features from complex sensor data, reducing the reliance on manual feature engineering. The fair performance on test data reflects their potential to enhance predictive maintenance strategies by accurately predicting equipment failures and optimizing maintenance schedules.