National Tsing Hua University

11220IEEM 513600

Deep Learning and Industrial Applications

Homework 4

Name: 劉奕　　Student ID: 111003817

In this homework, the target was normalized (Z-score) to facilitate weight optimization during model training. When calculating the model performance (MSE), the output results were inversely transformed to obtain the original scale result. Normalization parameters are only fitted on the training set and then applied to the other datasets.

1. Experiment with different window sizes and steps.

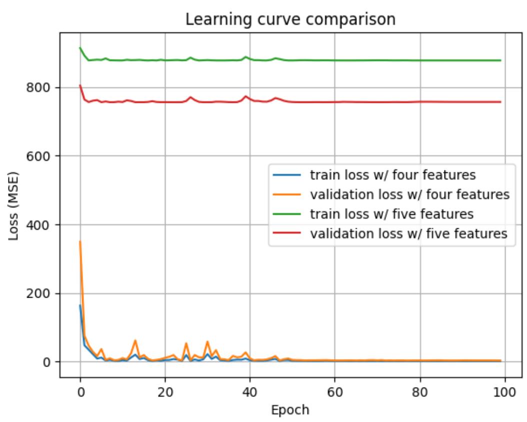
window size	step	Train MSE	Validation MSE	Test MSE
10	30	11.104976	2.191004	26.329634
10	10	1.956300	0.514049	10.465737
30	30	1.573753	2.329780	16.950134

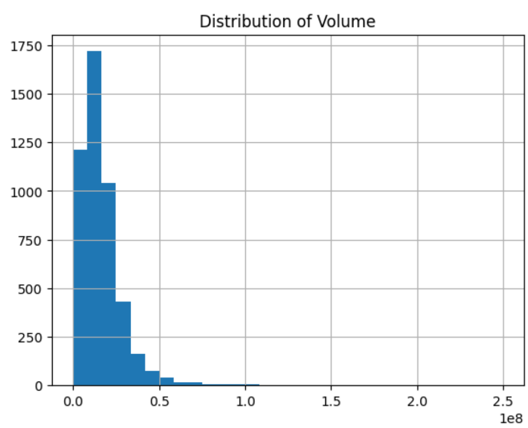
When the value of the step increases, the sample size decreases, leading to worse training and testing performance. When increasing the window size, the information contained in one sample increases, leading to better training and testing performance. So, choosing a suitable combination of window size and step is important in data preparation in time-series modeling.

2. Question about the input combinations.

   i. Include 'Volume' as an additional input feature in your model and discuss the impact of incorporating 'Volume' on the model's performance.

   The trained model with 'Volume' as an additional feature performs worse than the original one. The reason may be that the value range of 'Volume' is too large, making the model difficult to optimize. Normalization tricks can improve the issue.





ii. Explore and report on the best combination of input features that yields the best MSE. Briefly describe the reasons for your attempts and analyze the final, optimal input combination.

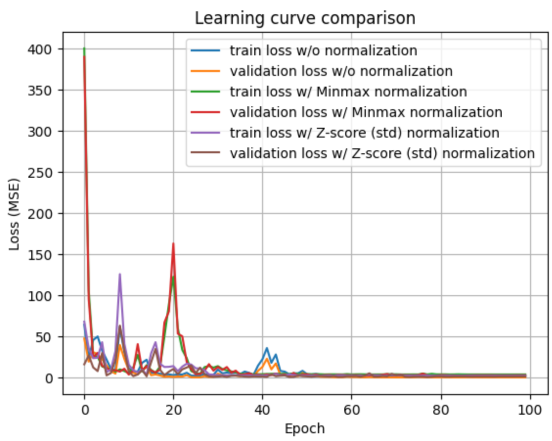
Based on the experimental findings, the best input strategy is to use the `high` feature exclusively for model construction. In this experiment, a simple LSTM model for regression tasks might need help extracting detailed information from raw data (e.g., integrating the convolutional layer). Therefore, utilizing the most straightforward feature combination, focusing on one feature alone, is appropriate for this scenario.

Open	High	Low	Close	Volume	Train MSE	Valid MSE	Test MSE
1	1	1	1	0	1.957	2.366	13.35
1	0	0	0	0	2.551	1.819	8.063
0	1	0	0	0	2.580	0.530	4.950
0	0	1	0	0	1.183	6.882	10.401
0	0	0	1	0	15.96	3.207	40.12
0	0	0	0	1	913.04	624.66	1346.06

3. Analyze the performance of the model with and without normalized inputs in Lab 4.

According to the experimental results, models trained with Z-score normalization typically perform better than those using min-max or no normalization. This superior performance is likely due to Z-score normalization transforming the data into a symmetric distribution, which improves the model's optimization process. The findings suggest that employing normalization techniques that promote symmetry in data distribution can significantly enhance model optimization and overall performance.

Normalization	Test MSE
None	11.06
Min-max	11.74
Z-score	8.80



4. Why should the window size be less than the step size in Lab 4?

   The window size less than the step size leads to the overlap between the processed samples, which may cause the overfitting issue but could be useful in other applications such as data augmentation in source separation and high-frequency signal modeling.

5. Describe one method for data augmentation specifically applicable to time-series data. Cite references to support your findings. (Approximately 100 words.)

   One of the augmentation techniques for time-series data is the short-time Fourier transform (STFT). By converting time series into the time-frequency domain, variations can be introduced through operations such as local averaging or shuffling of the time-frequency features. This method is beneficial for augmenting data in applications like human activity classification using deep LSTM networks, as it creates diverse training examples that can improve model performance.

   (Ref: Time Series Data Augmentation for Deep Learning: A Survey)

6. Discuss how to handle window size during inference in different model architectures (approximately 150 words):

   i. Convolution-based models

      Convolutional layers can adapt to different window sizes, but when followed by flatten and fully-connected layers, the input size must be fixed. This is because the flattening operation requires a consistent output size from the convolutional layers to match the expected input size of the fully-connected layers.

   ii. Recurrent-based models

      These models can handle different window sizes more flexibly since they process data one step at a time. However, using a window size similar to training is generally better to ensure the model performs well.

   iii. Transformer-based model

      While these models can also manage variable window sizes due to their self-attention mechanism, using a window size consistent with training helps maintain accuracy. Padding or truncating might be necessary to match the training setup.