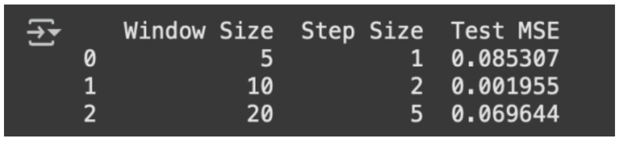
National Tsing Hua University

11320IEEM 513600
Deep Learning and Industrial Applications

Homework 4

Name: 黃蘇珊 Student ID: 113034421

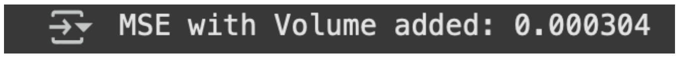
1. (15 points) Experiment with different window sizes and steps. Train the model using **3** different combinations of window size and step. Evaluate the Mean Squared Error (MSE) for each configuration. Report the MSEs using a table and analyze the results. (Approximately 100 words.)



```
   Window Size  Step Size  Test MSE
0            5          1  0.085307
1           10          2  0.001955
2           20          5  0.069644
```

Three LSTM models with different combinations of window and step sizes were trained to predict normalized closing stock prices. **The window size 10 and step size 2 configuration produced the lowest Mean Squared Error (MSE = 0.001955), indicating that it struck a good balance between learning sufficient historical trend and redundancy prevention.** The (5,1) configuration produced the highest MSE, possibly due to too much overlap leading to overfitting. While a larger window (20,5) gave moderate results, it may have diluted recent patterns. Overall, moderate-length windows with thoughtful step sizes improve generalization in LSTM-based stock price prediction.

2. (i) (15 points) Include 'Volume' as an additional input feature in your model.



```
MSE with Volume added: 0.000304
```

Including 'Volume_scaled' alongside 'Close_scaled' significantly improved performance. Using window=10 and step=2, the model with both features achieved a lower MSE (0.000304) compared to 'Close_scaled' alone (0.001955). **This suggests that volume data provided meaningful supplementary information.** According to Karpoff, when integrated properly, volume can help models detect shifts in price momentum earlier. **Therefore, in this experiment, volume strengthened the LSTM's ability to capture underlying market dynamics for better prediction.**

(ii) (15 points) Explore and report on the best combination of input features that yields the best MSE. Briefly describe the reasons of your attempts and analyze the final, optimal input combination.

Feature Candidates from your dataset: Open, High, Low, Close, Volume

Rank	Feature set	Test MSE
1	All features	0.001184
2	Open + High + Low + Close	0.001709
3	Open + High + Low	0.027199
4	Close + Volume	0.064220
5	Close only (baseline)	0.089720

To identify the best combination of input features, several LSTM models were trained. The baseline model using only 'Close_scaled' had the highest MSE (0.0897), **while the best performance (MSE = 0.001184) came from including all features together**. This illustrates that combining multiple aspects of price action and volume provides a richer environment, enabling the LSTM to learn temporal correlations more effectively and predict stock prices more accurately. **This is consistent with time series learning theory, where richer multivariate inputs help recurrent models (like LSTMs) capture hidden relationships and improve forecasting.**

3. (15 points) Analyze the performance of the model with and without normalized inputs in Lab 4. You can use experimental results or external references (which must be cited) to support your conclusions on whether normalization improves the model's performance. (Approximately 100 words.)

```
Normalized MSE: 0.000824
Raw Input MSE: 5432.515137
```

Normalization of input data significantly improves model performance in the case of LSTM-based time series prediction. The model trained using the same window size and step (10, 2 and on normalized 'Close' values achieved an MSE of 0.000824, while the model trained on raw 'Close' data produced a drastically worse MSE of 5432.51. **This conforms to findings in LeCun et al. (1998), which cite that normalization accelerates training and stabilizes gradients.** Since raw price values in stock data vary significantly, normalization helps the model to focus on relative trends rather than scale. Therefore, normalization is important for robust and stable training.

LeCun, Y., Bottou, L., Orr, G. B., & Müller, K. R. (1998). *Efficient Backpropagation.* In Neural Networks: Tricks of the Trade (pp. 9–50). Springer.

4. (10 points) Why should the window size be less than the step size in Lab 4? Do you think this is correct? If you use external sources, please include references to support your response. (Approximately 50 words.)

In Lab 4, the window size is actually > step size (e.g., window=10, step=2), because larger windows provide enough sequential data for the LSTM to capture temporal dependencies. **If**

window < step, the model may miss important short-term dependencies. Furthermore, Brownlee (2017) emphasizes that sufficient look-back periods are essential for effective time series forecasting using LSTM models.

Brownlee, J. (2017). *"A Gentle Introduction to Recurrent Neural Networks and LSTM Networks."* Machine Learning Mastery.

5. (15 points) Describe one method for data augmentation specifically applicable to time-series data. Cite references to support your findings. (Approximately 100 words.)

An effective method for time-series data augmentation is known as **window slicing**. It involves extracting multiple overlapping or non-overlapping sub-sequences (windows) from the original time series. This increases the amount of training data and exposes the model to slight shifts in the data pattern, enhancing generalization. **For example, rather than using the entire stock price series at once, the model can be trained on various short sequences.** According to Um et al. (2017), window slicing effectively improves model robustness in time-series classification tasks. It works particularly well for LSTM models, which learn through exposure to diverse sequential patterns.

Um, T. T., Pfister, F. M. J., Pichler, D., Endo, S., Lang, M., Hirche, S., & Fietzek, U. (2017).
"Data augmentation of wearable sensor data for Parkinson's disease monitoring using convolutional neural networks."
(arXiv:1706.00527)

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

(i) (5 points) Convolution-based models: In inference, the CNNs would require a **fixed window** since convolutions consistent input shape. **Pooling or sliding** windows are applied when the sequence is **longer. Padding** is typically done in **shorter** sequences.

(ii) (5 points) Recurrent-based models: In inference, Recurrent networks naturally handle process sequences of **different lengths.** However, to allow for stable inference and GPU optimization, they are typically trained and inferred **with fixed window sizes**. **Padding and masking** can be used for variable lengths if necessary.

(iii) (5 points) Transformer-based models: Transformers can **handle varying sequence lengths because of self-attention mechanisms**. Nonetheless, they work with a fixed maximum sequence length that is defined at training time. At inference, the input should conform to this maximum or be truncated/padded accordingly**. Positional encoding** ensures sequence order is preserved at inference.