# Deep Learning HW4

113034507 龔良軒

**Q1.**

| Exp | Window | Step | Train Shape | Val Shape | Test Shape | Train MSE | Val MSE | Test MSE |
|---|---|---|---|---|---|---|---|---|
| baseline | 10 | 15 | (227, 10, 4) | (57, 10, 4) | (31, 10, 4) | 2.84 | 0.45 | 17.68 |
| 1 | 10 | 3 | (1133, 10, 4) | (284, 10, 4) | (157, 10, 4) | 2.93 | 1.49 | 4.82 |
| 2 | 10 | 2 | (1700, 10, 4) | (425, 10, 4) | (236, 10, 4) | 1.95 | 3.99 | 2.30 |
| 3 | 15 | 2 | (1699, 15, 4) | (425, 15, 4) | (235, 15, 4) | 1.30 | 2.83 | 3.43 |

*** The row in orange background is baseline experiment**

*** The row in blue background is the experiment with the best performance**

I tested three combinations of window size and step to assess their impact on prediction performance. The model with a window size of 10 and step 2 achieved the lowest test MSE (2.30), suggesting a good trade-off between temporal context and data volume. The smallest step values generally yielded better generalization, as they increased the amount of training data. Increasing the window size to 15 slightly improved training accuracy but led to a higher test MSE, likely due to overfitting. It would achieve better results if window size > Step.

**Q2. (i).**

| Exp | Window | Step | Train Shape | Val Shape | Test Shape | Train MSE | Val MSE | Test MSE |
|---|---|---|---|---|---|---|---|---|
| baseline | 10 | 15 | (227, 10, 5) | (57, 10, 5) | (31, 10, 5) | 861.57 | 827.92 | 1333.41 |
| 1 | 10 | 3 | (1133, 10, 5) | (284, 10, 5) | (157, 10, 5) | 976.00 | 685.42 | 1055.90 |
| 2 | 10 | 2 | (1700, 10, 5) | (425, 10, 5) | (236, 10, 5) | 935.48 | 898.42 | 1001.01 |
| 3 | 15 | 2 | (1699, 15, 5) | (425, 15, 5) | (235, 15, 5) | 873.81 | 1172.72 | 1029.53 |

After including "Volume" as one of the input features, the model's performance dropped significantly, and it began outputting nearly identical values for all test samples. This suggests that the model failed to learn meaningful patterns during training. A likely cause is that the distribution or scale of "Volume" differs greatly from the other features. If "Volume" has a

much larger range or is heavily skewed, it could dominate the training process, leading the model to rely disproportionately on it while ignoring other informative inputs. Without proper normalization or transformation, this imbalance can severely hinder the model's ability to generalize.

**Q2. (ii).**

| Exp | Window | Step | Feature combination | Train MSE | Val MSE | Test MSE |
|---|---|---|---|---|---|---|
| best | 10 | 2 | High, Open, Low | 1.76 | 1.68 | 1.64 |

I start by removing one feature at a time and training a model using the remaining features. I repeat this process until each feature has been dropped once. After evaluating the performance of each model, I drop the feature whose removal results in the best performance. I then repeat this process iteratively with the remaining features until no further improvement is observed.

This feature combination yielded the lowest test MSE of **1.64**, indicating strong predictive accuracy on unseen data. Removing the "Close" feature likely helped reduce redundancy or noise, improving generalization.

**Q3.**

| Exp | Window | Step | Feature combination | Train MSE | Val MSE | Test MSE |
|---|---|---|---|---|---|---|
| Normalization | 10 | 2 | High, Open, Low | 1.60 | 2.62 | 1.27 |

After applying normalization to scale inputs and labels between 0 and 1, the model showed better performance. The test MSE became lower compared to the result without normalization, which means the model made more accurate predictions. Normalization helps the model learn more effectively by keeping all input values within a similar range. This makes training more stable and usually leads to faster and better results.

**Q4.**

The step size should not exceed the window size to avoid missing crucial temporal information. When the window < step, the model might fail to capture valuable overlapping time segments, which can negatively impact prediction performance. Therefore, the window size should be equal to or greater than the step size to ensure the integrity of each training sample and the continuity of the time series.

Reference: *Franssens, D., et al. (2020). Impact of Window Size and Step Size in Time Series Forecasting Using LSTM. arXiv preprint arXiv:2005.10052.*

**Q5.**

A simple and effective method for augmenting time-series data is time warping. This technique works by stretching or compressing the time axis of the data, essentially changing the speed at which the sequence progresses. It creates new data that is similar to the original, but with slight variations. Time warping helps the model become more flexible by adding diversity to the data while keeping the key patterns intact.

Reference: *Zhu, Y., Zhang, Y., & Zhang, C. (2017). Time series data augmentation for deep learning: A survey.*

**Q6.**

(i) **Convolution-based models:** The window size is crucial for both training and inference. During inference, if the input sequence is longer than the training window, a sliding window approach is used. For shorter sequences, zero-padding or truncation may be needed.

(ii) **Recurrent-based models:** In LSTMs, the window size determines how many past timesteps the model considers. During inference, the window size is either the same as during training or sliding windows are used, with the model's state updated across windows for long sequences.

(iii) **Transformer-based models:** Transformers use fixed-length windows during training, and during inference, the same window size is used. For long sequences, the input can be processed in multiple windows, with attention mechanisms handling context across them. Sliding windows can also be applied for long-term dependencies.