1.

Experiment	Window Size	Step Size	Epochs	Best Val MSE	Test MSE	Notes
Exp 1	10	15	100	188.7909	281.5446	Baseline
Exp 2	20	10	100	164.4628	274.1987	More overlap
Exp 3	30	10	100	144.8411	246.5731	High overlap
Exp 4	20	20	100	514.5502	603.0838	Balanced overlap

From the results, Exp 3 (window size 30, step size 10) exhibits the lowest Mean Squared Error (MSE), which suggests that a larger window size, coupled with a smaller step size, allows the model to capture more comprehensive contextual information, thereby improving its generalization ability. Exp 2 demonstrates a marked improvement over Exp 1, reinforcing the hypothesis that increased overlap facilitates better data coverage, which in turn enhances model performance by reducing the risk of underfitting. In contrast, Exp 4, which features a balanced overlap configuration, results in significantly higher MSE values. This suggests that while balance is important, overly constrained overlap might restrict the model's capacity to adapt to complex data patterns, potentially leading to over-regularization. These findings underscore the importance of optimizing the balance between window size and step size—larger windows with appropriately smaller steps are critical for minimizing MSE and maximizing predictive accuracy.

2.

(i) Incorporating 'Volume' as an additional input feature can enhance model performance by providing crucial information about market activity and investor sentiment. Volume reflects the intensity of trading and liquidity, often serving as a precursor to price movements or confirming trends. For instance, high volume during price increases may indicate strong bullish sentiment, while high volume during price decreases could signal bearish sentiment. In predictive models, including 'Volume' allows for a more comprehensive understanding of market dynamics, thereby improving accuracy in tasks such as stock price forecasting or risk analysis. However, the inclusion of 'Volume' necessitates careful consideration of feature correlation. If highly correlated with price or other indicators, it may introduce multicollinearity, requiring regularization or feature selection to prevent overfitting. Thus, its impact on model performance hinges on appropriate integration and data preprocessing techniques.

(ii)

The optimal input feature combination for price prediction is Close, High, Low, and Volume, achieving the lowest Test MSE of 69.0354 on the original price scale. This outperforms both the full feature set and all other subsets. Notably, Volume consistently enhances model accuracy, indicating its critical role in capturing market activity and sentiment. The inclusion of High and Low reflects intraday price dynamics, while Close represents consensus market valuation. Conversely, the Open feature adds redundancy or noise, as its exclusion improves performance. This combination achieves a superior balance between predictive accuracy and model simplicity, reducing dimensionality without sacrificing performance. The results align with financial theory, emphasizing the importance of price range and trading volume in forecasting. Therefore, the most effective and parsimonious input set is Close, High, Low, Volume, providing a professional, data-driven basis for robust price prediction modeling.

Feature Combination	Test MSE (Original Price Scale)
Open, High, Low, Close, Volume	69.2439
Close, High, Low, Volume	69.0354 (Best)
Close, Volume	72.7786
Open, Close, Volume	113.5097
High, Low, Close	106.8861
Open, Close	123.2502
Open, High, Low, Close	281.5446

3.

Normalizing input features is essential for improving both the speed and quality of deep learning model training. In Lab 4, our experiments showed that models with normalized inputs learned faster, had smoother loss curves, and achieved higher final accuracy compared to models trained without normalization. This happens because normalization reduces problems like unstable gradients and sharp curvature in the optimization landscape. Without normalization, training was slower and more inconsistent. These results are consistent with well-established findings in the field, where normalization techniques are known to make neural networks easier to optimize.

Reference: Ioffe, S., & Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. Proceedings of the 32nd International Conference on Machine Learning (ICML), 448–456.

4. In Lab 4, using a window size smaller than the step size reduces overlap between samples, ensuring greater variability and minimizing data redundancy. While this

improves computational efficiency, it may limit the model's ability to capture long-term dependencies. Thus, its correctness depends on the specific task and data characteristics (Fawaz et al., 2019).

Reference:Fawaz, H. I., Forestier, G., Weber, J., Idoumghar, L., & Muller, P.-A. (2019). Deep learning for time series classification: a review. Data Mining and Knowledge Discovery, 33(4), 917–963.

5. Time warping is a technique that modifies the time intervals between data points, effectively stretching or compressing segments of a time series. This method preserves the overall temporal structure while introducing variability in the speed of the observed patterns. Time warping is especially useful for applications like activity recognition and medical diagnostics, where events may occur at different rates. By learning from temporally distorted sequences, models can become more robust to real-world variation. Studies have shown that time warping improves generalization in deep learning models for time-series tasks (Forestier et al., 2017; Fawaz et al., 2019).

References:Forestier, G., Petitjean, F., Senin, P., & Webb, G. I. "Generating synthetic time series to augment sparse datasets." ECML PKDD (2017).
Fawaz, H. I. et al. "Deep learning for time series classification: a review." Data Mining and Knowledge Discovery, 33(4), 917–963 (2019).

6.

(i) Convolution-based models:

In convolutional architectures, the window size during inference determines the receptive field that affects prediction quality. Since convolutions operate on local patterns, the model expects input sequences of a fixed length. During inference, the same window size used in training should be preserved to ensure consistent spatial hierarchies. Padding or truncation may be applied if input lengths differ.

(ii) Recurrent-based models:

Recurrent models, such as LSTMs or GRUs, naturally handle variable-length sequences. However, during inference, using the same window size as training ensures alignment in temporal dependencies. Longer sequences can be processed by truncating or using a sliding window, but too long a window may introduce noise or increase computation.

(iii) Transformer-based models:

Transformers process sequences in parallel and require positional encodings. Window size affects memory and computation due to the quadratic complexity of self-attention. During inference, it is crucial to match the maximum sequence length seen during training, or apply techniques like chunking or sparse attention to extend beyond the training window size.