Transport and Communications Science Journal

SELF-SUPERVISED DEEP LEARNING FOR GNSS TIME SERIES IMPUTATION: A COMPARATIVE STUDY OF NEURAL NETWORK ARCHITECTURES

Le Khanh Giang*, Tran Duc Cong, Ho Thi Lan Huong

University of Transport and Communications, No 3 Cau Giay Street, Hanoi, Vietnam

ARTICLE INFO

TYPE: Research Article
Received: 01/12/2025
Revised: 07/01/2026
Accepted: 12/01/2026
Published online: 15/01/2026
https://doi.org/10.47869/tcsj.77.1.10

* *Corresponding author*
Email: gianglk@utc.edu.vn; Tel: +84983663031

Abstract. Global Navigation Satellite System (GNSS) time series are widely used for structural health monitoring (SHM) and deformation analysis, but real-world recordings frequently contain short and long contiguous gaps that degrade downstream interpretation. This study addresses the challenge of accurate imputation for GNSS displacement series by proposing a self-supervised learning framework that trains models directly on real, unlabelled data using contiguous-span masking. We evaluate four neural architectures (ANN, CNN, GRU, LSTM) under a unified pipeline comprising signal denoising (moving-average, Kalman smoothing, Haar wavelet), sliding-window segmentation, Z-score normalization, and middle-region masking. Experiments use a year-long 10-minute-sampled dataset from the Can Tho cable-stayed bridge (sensor can519501, x/y/z components) and assess reconstruction quality via R^2, MAE, and MSE on withheld masked segments. Results indicate that recurrent architectures, particularly LSTM, produce the most faithful reconstructions: LSTM attains the highest validation R^2 (≈0.948) and the lowest MAE (≈0.137) and MSE (≈0.052) among tested models, while GRU offers competitive performance and CNN/ANN show substantially weaker recovery. These findings demonstrate that masking-based self-supervision is an effective strategy for GNSS gap recovery and that LSTM-like sequence models are well suited to capture the long-range temporal dependencies in bridge displacement data. The proposed approach enhances the reliability and continuity of GNSS-derived time series for structural monitoring and can inform future multi-sensor fusion and uncertainty quantification work.

Keywords: self-supervised learning, GNSS time-series imputation, LSTM, structural health monitoring (SHM), sliding-window masking, cable-stayed bridge.

1. INTRODUCTION

Global Navigation Satellite System (GNSS) observations have become a fundamental data source for monitoring ground deformation, structural displacement, and infrastructure stability. Their ability to provide continuous, high-precision, and long-term positioning makes GNSS time series indispensable in applications such as structural-health monitoring of cable-stayed bridges, geohazard assessment, and real-time deformation tracking [1,2]. However, the reliability of these applications strongly depends on the completeness and continuity of the recorded time series. Missing data caused by signal obstruction, satellite-geometry variations, multipath effects, equipment malfunction, or transmission interruptions pose a major challenge. These gaps may appear as short random losses or long contiguous outages, and they can significantly reduce the accuracy of displacement estimation, noise modelling, and subsequent engineering interpretation [3]. Unlike conventional sensor data, GNSS time series are characterized by strong temporal autocorrelation, multipath-induced noise, and nonlinear drift components, which make the imputation of missing values substantially more challenging.

Traditional imputation methods, including linear interpolation, spline fitting, K-Nearest Neighbours (KNN), and Kalman filtering, offer reasonable performance for smooth temporal patterns or short-duration gaps [4,5]. Yet, they often struggle with the complex characteristics of GNSS time series, which include non-linear temporal evolution, intermittent jumps, multipath-induced fluctuations, long periods of missing data, and strong multi-scale dependencies [5]. Moreover, as modern GNSS monitoring networks produce higher-frequency and multi-station observations, classical statistical approaches become increasingly inadequate for ensuring high-fidelity reconstruction of missing segments.

To overcome these challenges, deep learning has emerged as a powerful alternative. Recurrent neural networks (RNN), particularly Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRU), have demonstrated capabilities in learning temporal dependencies, capturing dynamic trends in sequential data, and handling missing-value imputation tasks more effectively than many classical methods [6,7]. Convolutional neural networks (CNN) and attention-based models provide additional advantages in parallelization and in modeling long-range interactions. Despite these advances, supervised deep learning for imputation is inherently limited by the lack of ground-truth labels for missing GNSS data and the difficulty of obtaining extensive annotated datasets that represent diverse outage scenarios [8].

Self-supervised learning (SSL) addresses this bottleneck by enabling models to learn meaningful latent representations directly from raw, unlabeled data through pretext tasks such as masking, reconstruction, and contrastive learning [9]. These approaches reduce dependency on large labelled datasets and have shown significant promise in time-series domains [10]. Masking-based SSL is particularly well matched to GNSS applications: the artificial masks applied during training closely mimic real-world GNSS data outages, allowing models to implicitly learn temporal dependencies, long-range correlations, and noise characteristics. Recent SSL-based imputation frameworks such as Transformer-based architectures and diagonally masked self-attention mechanisms have shown promising results in multivariate sensor time series [11,12,13,14]. However, their potential for GNSS time-series imputation remains largely unexplored. Two key gaps persist in existing literature, including:

(1) No comparative study is conducted to assess how different deep neural architectures perform under a unified SSL framework specifically for GNSS time-series reconstruction. Existing GNSS studies predominantly rely on filtering, regression, and classical machine

learning, with limited integration of recent advances in SSL-driven models;
(2) The influence of different masking strategies such as random masking, span masking, and block-based masking on the robustness and generalization performance of SSL models has not been systematically evaluated for GNSS applications, despite their critical role in shaping model behavior.

To address these research gaps, this study proposes a self-supervised deep-learning framework tailored for GNSS time-series imputation. We systematically evaluate multiple neural architectures, including LSTM, GRU, CNN, and fully connected artificial neural networks (ANN), under a range of masking strategies that simulate realistic GNSS missing-data patterns. Our objectives are:

(1) to benchmark and compare the imputation performance of these architectures in terms of accuracy, robustness, and computational efficiency; and
(2) to investigate how various SSL masking strategies influence model learning and generalization.

Through extensive experiments and rigorous evaluation metrics, we demonstrate that SSL-based deep-learning models consistently outperform traditional interpolation and filtering methods, particularly for long, irregular, or noise-affected gaps. The significance of this research extends beyond GNSS monitoring: the findings provide deeper insights into the applicability of SSL for high-integrity geospatial time series, offering methodological guidance for next-generation deformation monitoring systems and sensor networks. Overall, this work contributes to advancing the reliability, continuity, and analytical value of GNSS-derived time series through an integrated self-supervised learning approach.

2. METHODOLOGY

2.1. Dataset overview

This study uses a dataset collected from a dense network of GNSS sensors deployed for structural health monitoring (SHM) of Can Tho cable-stayed bridge in Vietnam. The bridge spans the Hau River in southern Vietnam, with a total length of 2,750 m, a main span of 550 m, and towers reaching 171 m in height. Its SHM system has been in operation since 2010, incorporating not only Global Positioning System (GPS) equipment but also various sensors, including temperature sensors, anemometers, and accelerometers.

The GPS subsystem comprises nine rover receivers (located at characteristic locations such as the tops of the two towers, on the main beam, at some pillars) and two base stations (located near the system management office (1km away) and at the top of the pile near the North tower pillar). This network provides continuous, high-precision displacement measurements and redundancy across multiple structural components. The GPS devices used are Leica GMX 902 units with the accuracy parameters indicated is (±10mm ± 1ppm) in terms of ground position and (±20mm ± 1ppm) in height. The frequency of GPS signal reception is 20Hz, and the monitoring data is stored as data averaging 1 minute, 10 minutes, 1 hour and 1 day.

For the analyses presented in this study, a representative GNSS sensor was selected: can519501, which includes three displacement components (x, y, z). This sensor was chosen because it offers a long, continuous, and high-quality time series suitable for evaluating the proposed self-supervised reconstruction framework.

Data were continuously recorded throughout 2017, from January 1st to December 30th, at a sampling interval of 10 minutes, resulting in 144 samples per day. The dataset provides a rich temporal resolution suitable for detailed time-series analysis and imputation tasks.

2.2. Data structure and format

The dataset was acquired in CSV format, spanning multiple columns representing measurements from various sensors and corresponding quality flags. The primary components of the dataset consisted of:

(1) Temporal Component: Timestamps formatted as YYYYMMDDHHMMSS integers, providing precise temporal resolution for each measurement.
(2) Measurement Data: Individual measurements from GPS sensors including coordinates (x, y, z); Each measurement was recorded as floating-point values.
(3) Quality Indicators: Flag values (0 or 1) associated with each measurement; Missing data indicated by -9999 sentinel values.

2.3. Preprocessing pipeline

The data preprocessing approach involved a meticulous validation and cleaning strategy to ensure high-quality input for self-supervised learning. We systematically address data integrity through a carefully designed workflow that identifies and mitigates potential measurement inconsistencies. The workflow includes signal denoising, sliding-window segmentation, window filtering, and normalization.

2.3.1. Signal denoising

The GNSS signal denoising pipeline consists of two stages designed to reduce noise while preserving the signal's dynamic structure (Fig. 1). The first stage applies a weighted moving-average filter with a 9-sample window to remove high-frequency noise and stabilize the signal. The second stage combines Kalman smoothing, using a constant-velocity model, with a three-level Haar wavelet denoising; the Kalman filter provides optimal state estimates in the temporal domain, while the wavelet step removes residual noise via soft-thresholding based on the median absolute deviation (MAD). This combination yields a smooth, low-noise signal that still preserves the physical characteristics required for subsequent feature extraction and modelling.

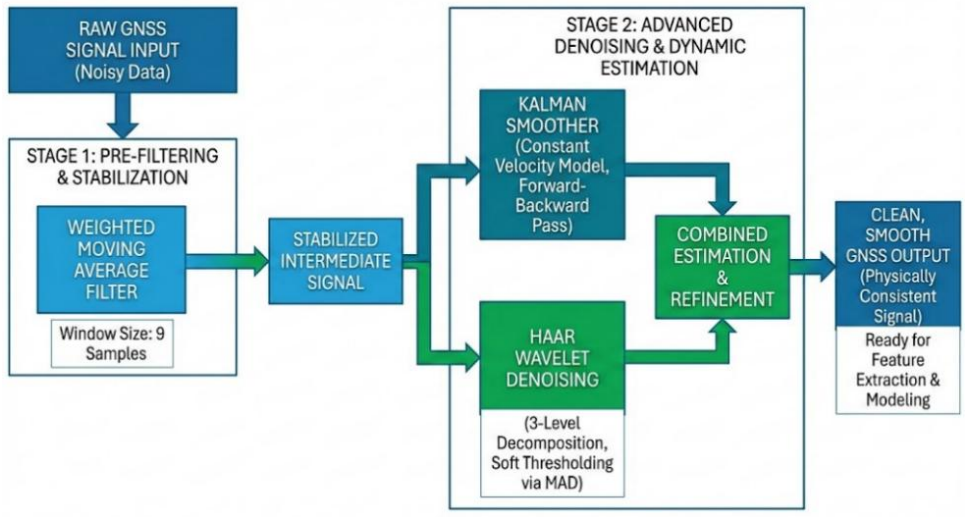

Figure 1. GNSS signal denoising pipeline.

2.3.2. Sliding window segmentation, window filtering, and normalization

During data preparation, the GNSS time series are segmented using a sliding-window mechanism, filtered by data completeness, and normalized before being fed to the self-supervised model (Fig. 2). This procedure ensures stable-quality inputs that capture the relevant dynamic features and meet the requirements of reconstruction-based learning.

First, the observation series are partitioned into consecutive segments via sliding windows of length $W = 100$ with stride $S = 2$. The number of windows is given by:

$$N = 1 + \frac{T-W}{S} \tag{1}$$

where T is the total number of time stamps. Each window maps to an index set $I_i = \{ iS, iS + 1, \dots, iS + W - 1 \}$, producing overlapping segments that preserve temporal continuity.

Next, each window is evaluated for its data completeness ratio using a validity mask $\text{mask} = \neg\text{NaN}(X)$. Let $x_{t,f}$ denote the value of feature f at time index t. The completeness ratio of window i is computed as:

$$r_i = \frac{1}{WF}\sum_{t=1}^{W}\sum_{f=1}^{F} 1\left(x_{t,f} \neq NaN\right) \tag{2}$$

with $F = 3$ features (x, y, z). Only windows satisfying $r_i \geq 0.95$ and containing no NaNs are retained.

To ensure statistical stability, all valid windows are normalized using Z-score normalization. The mean and standard deviation are computed across the entire set of windows as:

$$\mu_f = \text{E}\left[X_{:,:,f}\right], \qquad \sigma_f = \sqrt{\text{Var}[X_{:,:,f}]} \tag{3}$$

and each value is transformed according to:

$$X'_{t,f} = \frac{X_{t,f}-\mu_f}{\sigma_f} \tag{4}$$

The windows are then randomly shuffled and split into training and validation sets with a fixed random seed at an 85%–15% ratio to ensure a stable data distribution independent of temporal order.

Finally, a contiguous-span masking mechanism is applied to support self-supervised learning. This strategy simulates continuous data loss commonly observed in GNSS networks, where sensor failures or transmission interruptions often produce consecutive missing intervals. For each window, up to two masked spans may be generated. Each span is instantiated with probability 0.6, with its length sampled uniformly at random from 4 to 10 time steps; the start position is chosen randomly within the central region of the window so that contextual data at both ends remain available. All values within a span are set to 0 and the mask is represented in binary form as $\text{mask}[s{:}\, e] = 1$. The model input is formed as:

$$X_{\text{masked}} = X * (1 - \text{mask}), \qquad \tilde{X} = [\, X_{\text{masked}}, \text{mask}\,] \tag{5}$$

where * denotes element-wise multiplication.

The choice of the span masking probability $p = 0.6$ is motivated by the principle of task-difficulty balancing in masked self-supervised learning. A masking probability that is too low results in an insufficient number of masked points, causing the model to learn only local dependencies, whereas an excessively high masking probability substantially reduces the remaining contextual information, thereby hindering the reconstruction process. Recent studies have shown that masking ratios in the range of approximately 0.5–0.6 represent a common and stable choice for masked modeling of time series, including SimMTM [15], TI-MAE [16], NCDE-based framework [17], and TimeXer [18]. A comparative summary of masking probabilities and span configurations used in related studies and in the proposed method is provided in Table 1.

Regarding the span length, the interval $l \in [4, 10]$ is selected to avoid overly short spans that would lead to a trivial imputation task, while simultaneously preventing excessively long spans that would disrupt temporal contextual continuity. In the context of GNSS data with a 10-minute sampling interval, this range corresponds to 40–100 minutes, which is consistent with the duration of data gaps commonly observed in practice.

Moreover, given a typical window size of $window\_size = 100$, a span length of 4–10 time steps accounts for approximately 4–10% of the window length, creating a masking level that is sufficiently challenging for the model while still preserving adequate contextual information for accurate inference.

If no span is randomly generated for a window, a short default span is added to guarantee that every window contains a reconstruction task. This design balances task difficulty and model stability during training and forces the model to learn the important temporal and spatial dependencies required to restore GNSS signals.

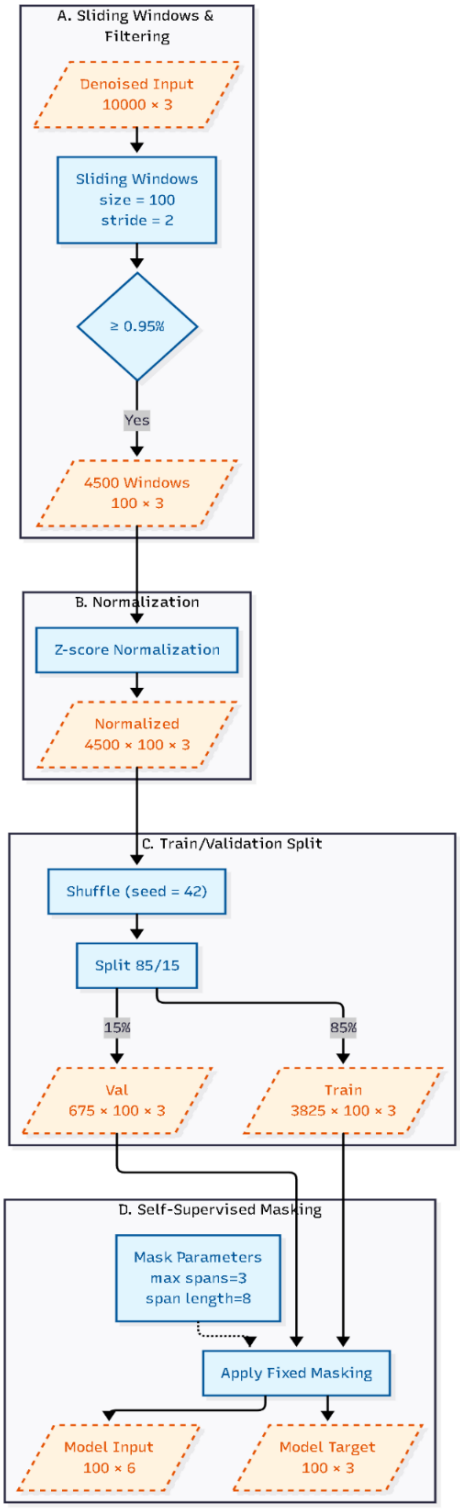

Figure 2. Sliding window segmentation, window filtering, and normalization, and contiguous-span masking

Table 1. Comparison of masking parameters between the proposed method and related studies.

Study	Mask ratio / probability	Span length
SimMTM [15]	≈ 0.5	≈ 5–10
TI-MAE [16]	≈ 0.5	≈ 10–15
NCDE-based Framework [17]	0.5	5 (contiguous)
TimeXer [18]	0–99% (evaluation)	–
Proposed method	**0.6**	**4–10**

2.4. Neural network architectures

To evaluate the capability of modeling and imputing missing values in GNSS time series, four representative neural network architectures were implemented: ANN, GRU, LSTM, and CNN. These models embody different perspectives on sequence representation learning, ranging from feedforward networks without temporal memory to recurrent networks with gating mechanisms and convolutional architectures designed for temporal feature extraction. All models employ the same input and output format: (batch_size, 100, 2F) for the input and (batch_size,100, F) for the output, where F denotes the number of features.

2.4.1. Artificial Neural Network

The ANN is utilized as a baseline model and implemented in a standard feedforward configuration. In this architecture, each time step $t$is processed as an independent sample; consequently, the network does not leverage temporal correlations across the sequence. Let $\mathbf{x}_t$denote the input vector at time step t. The ANN predicts the corresponding output through a nonlinear mapping expressed as [19]:

$$\widehat{y_t} = \sigma(W_2\, \phi(W_1 x_t + b_1) + b_2) \tag{6}$$

where $\phi(\cdot)$ is the ReLU activation function, and $\{W_1, W_2, b_1, b_2\}$ are the trainable parameters of the network. Owing to its simplicity, the ANN offers rapid training and low computational cost. However, the lack of temporal modeling capacity generally leads to inferior performance in sequence-based imputation problems, where continuity and time-dependent patterns are essential for accurate reconstruction.

2.4.2. Gated Recurrent Unit

The GRU is a modified form of the Recurrent Neural Network (RNN) designed to address the vanishing gradient problem while maintaining the ability to capture long-term temporal dependencies. It incorporates two gating mechanisms, the reset gate and the update gate, that regulate how information is forgotten or preserved across time steps [20].

At each time step t, the GRU computes the reset gate r_t and the update gate z_t through a sigmoid activation applied to a linear transformation of the current input x_t and the previous hidden state h_{t-1}. The candidate hidden state $\tilde{\mathrm{h}}_t$ is then obtained by blending the current input with a gated version of the past hidden state, followed by a hyperbolic tangent activation. Finally, the new hidden state h_t is formed as a convex combination of the previous hidden state and the candidate state, where the update gate controls the contribution of each component. These operations are summarized as [20]:

$$\mathrm{r}_t = \sigma(\mathrm{W}_r \mathrm{x}_t + \mathrm{U}_r \mathrm{h}_{t-1}) \tag{7}$$
$$\mathrm{z}_t = \sigma(\mathrm{W}_z \mathrm{x}_t + \mathrm{U}_z \mathrm{h}_{t-1}) \tag{8}$$
$$\tilde{\mathrm{h}}_t = \tanh\big(\mathrm{W}_h \mathrm{x}_t + \mathrm{U}_h(\mathrm{r}_t * \mathrm{h}_{t-1})\big) \tag{9}$$
$$\mathrm{h}_t = (1 - \mathrm{z}_t) * \mathrm{h}_{t-1} + \mathrm{z}_t * \tilde{\mathrm{h}}_t. \tag{10}$$

Through this gating structure, the GRU can selectively retain informative temporal patterns while discarding irrelevant past states, leading to more stable training behavior and reduced computational cost compared with the LSTM architecture.

2.4.3. Long Short-Term Memory

The LSTM network extends the GRU by introducing a dedicated cell state c_t, which enables more stable preservation of long-term information. At each time step t, the LSTM uses a forget gate f_t to determine which past information to discard, an input gate i_t to regulate how new information is stored, and an output gate o_t to control how much of the updated cell state contributes to the hidden state. These operations are expressed as [21]:

$$f_t = \sigma(W_f x_t + U_f h_{t-1}), i_t = \sigma(W_i x_t + U_i h_{t-1}) \quad (11)$$

$$\tilde{c}_t = \tanh(W_c x_t + U_c h_{t-1}), c_t = f_t * c_{t-1} + i_t * \tilde{c}_t \quad (12)$$

$$o_t = \sigma(W_o x_t + U_o h_{t-1}), h_t = o_t * \tanh(c_t) \quad (13)$$

By explicitly separating the cell state from the hidden state, the LSTM captures long-term dependencies more effectively than the GRU, though at the cost of higher computational complexity and a larger number of parameters. This makes it well suited for modeling GNSS time series with strong nonlinear and long-range temporal behavior.

2.4.4. Convolutional Neural Network

Unlike recurrent architectures, CNN processes time-series data using one-dimensional convolutions, to extract local temporal patterns with high computational parallelism. For an input sequence x, the convolutional output at time step t is computed as [22]:

$$y_t = \sum_{i=0}^{K-1} w_i \, x_{t-i}, \quad (14)$$

where K is the kernel size and w_i denotes the convolutional weights. Because CNNs operate on the entire sequence simultaneously, they enable substantially faster training compared with recurrent models. However, their receptive field, the temporal span of context each output can access, is fundamentally limited by the kernel size, which restricts their ability to capture long-term dependencies.

2.4.5. Model Configuration Summary

All four architectures share common training configurations to ensure fair comparison. Table 2 summarizes the key architectural hyperparameters for each model.

ANN Architecture: The feedforward network uses three hidden layers with sizes [64, 128, 64], forming a bottleneck structure that compresses information before reconstructing the output. This architecture processes each timestep independently, making it computationally efficient but unable to capture temporal dependencies.

Recurrent architectures (GRU and LSTM): Both recurrent models employ 2 stacked layers with 128 hidden units each. The GRU uses update and reset gates for efficient sequence modeling, while LSTM includes additional cell state and forget gate mechanisms. These architectures process the entire sequence of 100 timesteps, capturing long-range temporal dependencies crucial for GNSS time series.

CNN architecture: The convolutional network uses 32 channels with kernel size 3, enabling local pattern extraction across the temporal dimension. The small kernel size allows fine-grained feature detection while maintaining computational efficiency.

Dropout regularization with rate 0.1 is consistently applied across all models to prevent overfitting. The input dimension is 6 (3 features × 2, including mask indicator channels), and output dimension is 3 (recovered feature values).

Table 2. Model architecture hyperparameters.

Model	Type	Layers	Hidden/Channel Sizes	Kernel	Dropout
ANN	Feedforward	3 FC	[64, 128, 64]	-	0.1
GRU	Recurrent	2 GRU + FC	[128, 128] → 3	-	0.1
LSTM	Recurrent	2 LSTM + FC	[128, 128] → 3	-	0.1
CNN	Convolutional	Conv1d stack	32 channels	3	0.1

All models are trained with the same training strategy for fair comparison. The shared training hyperparameters are summarized in Table 3.

Table 3. Training hyperparameters.

Parameter	Value	Description
Epochs	200	Maximum training iterations
Batch size	128	Samples per gradient update
Learning rate	1×10−3	Initial learning rate
Weight decay	1×10−4	L2 regularization strength
Optimizer	AdamW	Decoupled weight decay optimizer
LR schedule	Cosine + Warmup	20 epoch warmup, then cosine decay
Early stopping	Enabled	Patience: 15 epochs, min_delta: 1×10−5

2.5. GNSS Self-Supervised Training Pipeline

Fig. 3 shows GNSS SSL pipeline, including preprocessing, sliding-window generation, masking, model training, and evaluation. The workflow integrates denoising, window construction, and middle-region masking to enable reconstruction of missing segments in a self-supervised manner. The figure provides an overview of how raw GNSS data is transformed into training-ready sequences and evaluated through synthesized and real gap recovery.

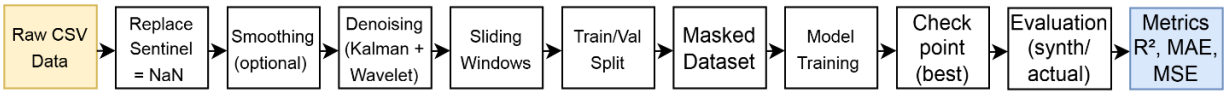

Figure 3. GNSS self-supervised learning pipeline.

Before model evaluation, the GNSS time-series data are preprocessed through smoothing, Kalman filtering, and wavelet denoising, as illustrated in Fig. 4. The cleaned signals are then temporally split into Training and Validation sets, where contiguous masked segments are created to form “marked samples” (Fig. 5), enabling ground-truth-based assessment. Using the Masked-Region Protocol, ANN, GRU, LSTM, and CNN models predict only the intentionally hidden intervals, and their outputs are directly compared against the withheld ground truth. This evaluation design allows reconstruction performance to be assessed entirely on real GNSS data while maintaining fairness and reproducibility across all model architectures.

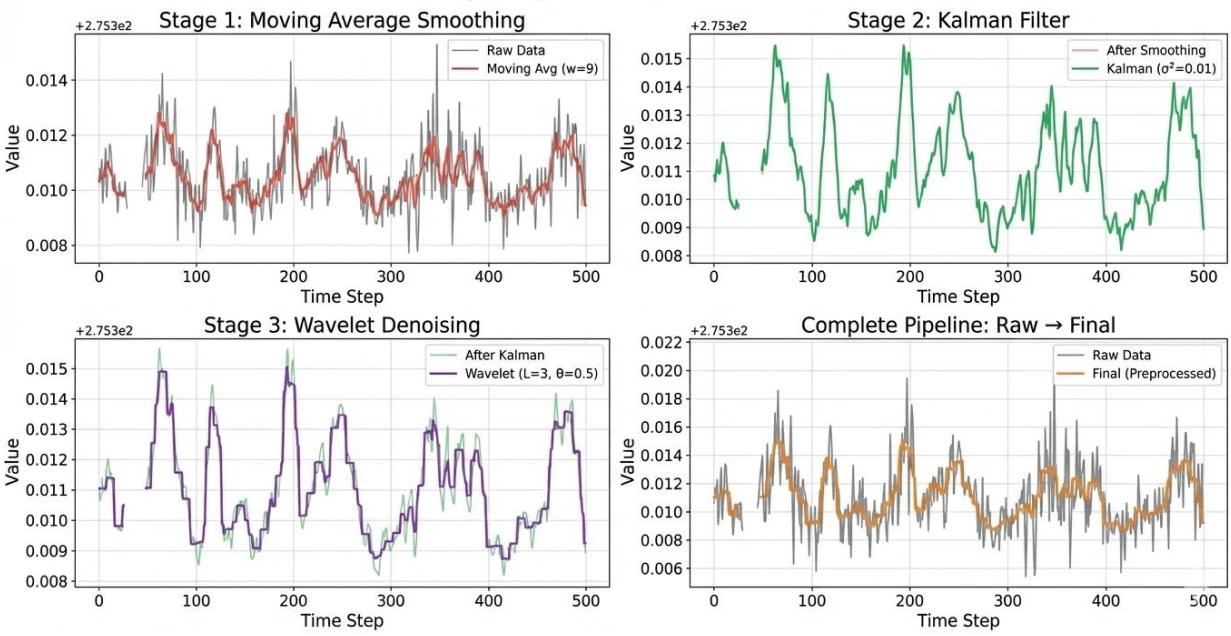

Figure 4. GNSS preprocessing pipeline for sample can519501x, including moving-average smoothing, Kalman filtering, wavelet denoising, and the final combined output.

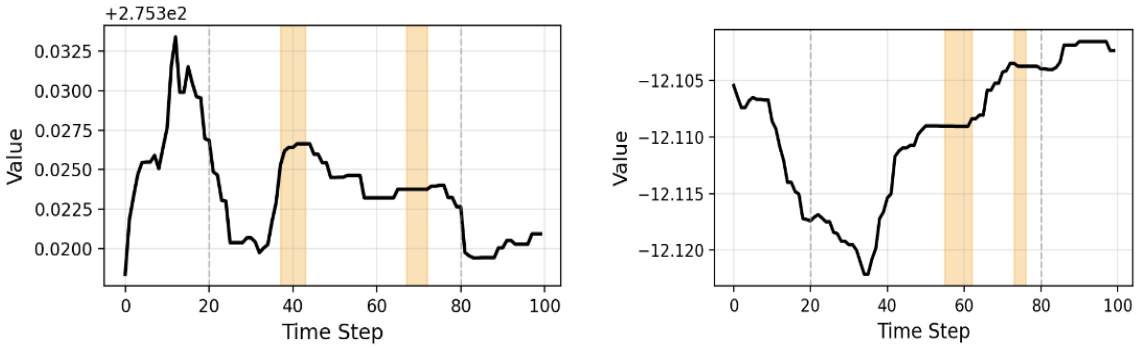

Figure 5. Example of a marked validation sample showing contiguous masked regions used for the Masked-Region Evaluation Protocol.

3. EVALUATION AND RESULTS

3.1. Evaluation Metrics

3.1.1. Coefficient of Determination (R^2)

$$R^2 = 1 - \frac{\mathrm{SS}_{res}}{\mathrm{SS}_{tot}} \tag{15}$$

This metric measures how much of the variance in the true signal is explained by the reconstructed signal. Higher values indicate better reconstruction quality [23].

3.1.2. Mean Absolute Error (MAE)

$$MAE = \frac{1}{N}\sum_{i=1}^{N}\left|y_i^{pred} - y_i^{true}\right| \tag{16}$$

MAE represents the average deviation between predicted and true values, expressed in the same physical units as the GNSS displacement [23].

3.1.3. Mean Squared Error (MSE)

$$MSE = \frac{1}{N}\left(y_i^{pred} - y_i^{true}\right)^2 \quad (17)$$

MSE penalizes larger errors more strongly and is therefore sensitive to severe reconstruction failures or outliers [23].

3.2. Experimental Results

3.2.1. Training Performance

This subsection presents the comparative evaluation of the four architectures including LSTM, GRU, CNN, and ANN based on reconstruction plots, loss-curve behavior, and quantitative error metrics. Across the experiments, the LSTM model provides the most accurate and stable reconstructions, as reflected in both the visual and numerical assessments.

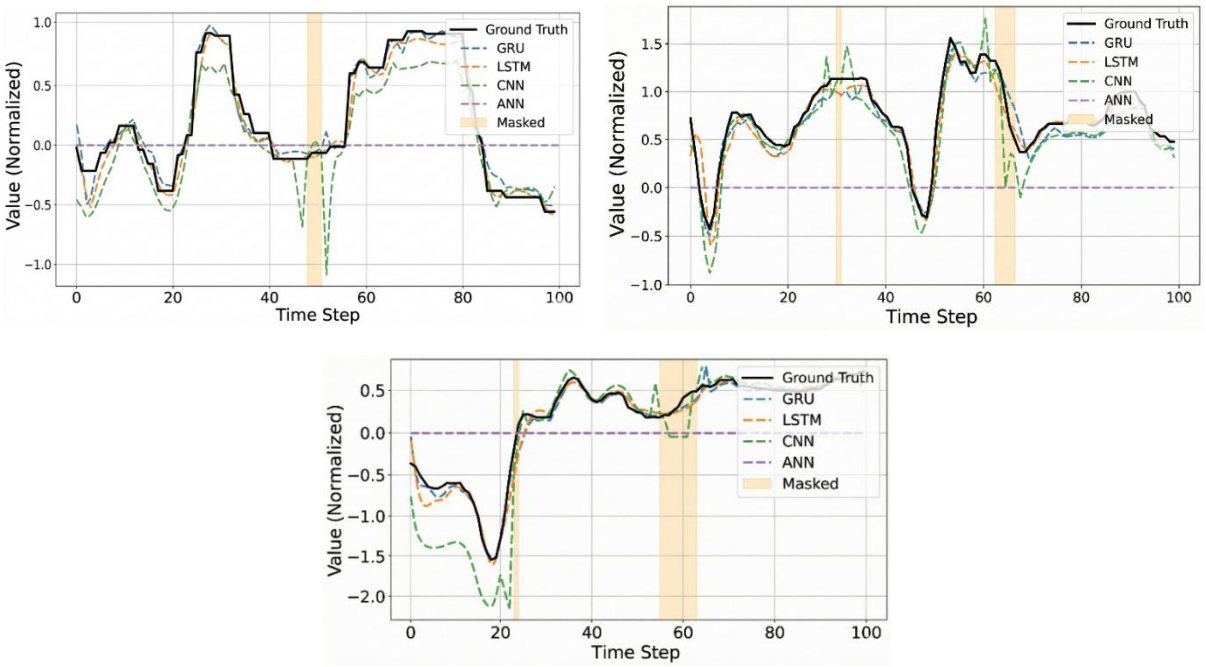

Figure 6. Time-series reconstruction comparison of models, including masked regions.

Fig. 6 illustrates the reconstruction performance on three representative GNSS segments, including masked intervals where the models must infer missing values using only temporal context. In all cases, LSTM produces trajectories that are most closely aligned with the Ground Truth, capturing both the global trend and fine-scale variations. Within masked regions, LSTM maintains consistent phase alignment and amplitude behavior relative to the surrounding context, resulting in smooth and physically coherent reconstructions.

The GRU model also follows the Ground Truth reasonably well, although occasional phase shifts and reduced fidelity appear in segments with high-gradient changes. The CNN model shows less stable behavior, particularly around the boundaries of masked regions, where it tends to overshoot and produce oscillatory artifacts. The ANN baseline yields oversmoothed, nearly flat responses that fail to reflect the temporal dynamics of the original signal, indicating its

limited capacity for sequence modeling. Overall, the reconstruction plots suggest that recurrent models, especially LSTM are better suited for capturing the temporal dependencies present in GNSS displacement data.

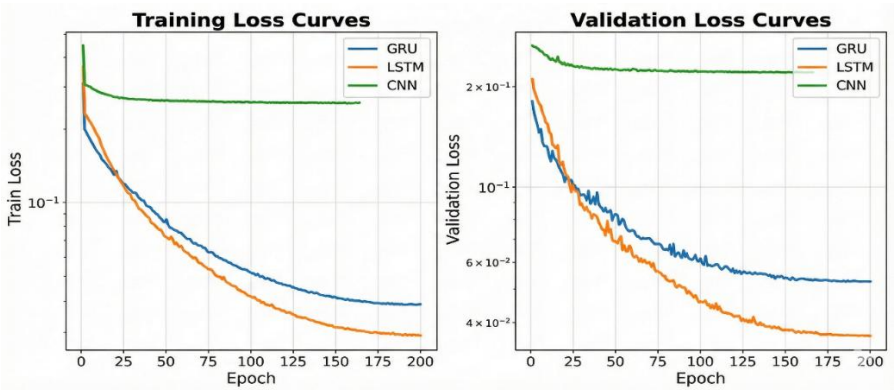

Figure 7. Training and validation loss curves of LSTM, GRU, and CNN over 200 epochs.

The comparison of training and validation loss curves (Fig. 7) further supports these observations. The LSTM achieves the lowest training and validation losses and exhibits smooth, monotonic convergence throughout the full 200-epoch training period. GRU shows comparable but consistently higher losses and larger fluctuations, particularly in the early epochs. By contrast, the CNN model shows slow and shallow improvement, with its loss plateauing early, indicating limited ability to learn long-range temporal relationships. The ANN baseline converges quickly to a high-loss regime, consistent with its flatline reconstruction behavior. Together, these trends highlight the advantages of the LSTM architecture under the tested settings, particularly for reconstructing masked or partially missing GNSS time-series data.

The quantitative results in Table 4 provide statistical evidence of LSTM's superiority. LSTM achieves the highest R^2 scores on both training (0.9431) and validation sets (0.9476), reflecting the strongest correspondence between predicted and true signals. Correspondingly, LSTM also yields the lowest MAE (≈0.137) and MSE (≈0.052) across all experiments.

The GRU model ranks second, with R^2 scores around 0.93 and moderately higher MAE/MSE values, confirming its solid yet inferior performance compared to LSTM. The CNN model shows markedly weaker predictive capability ($R^2 \approx 0.76$ and MAE/MSE nearly five times larger), while ANN displays negative R^2 values, indicating poor learning and total inability to model temporal patterns.

Table 4. Quantitative performance metrics of models on training and validation sets.

Model	Dataset	R2	MAE	MSE
GRU	Train	0.937148	0.143798	0.061454
	Validation	0.934828	0.144287	0.061995
LSTM	Train	0.943125	0.139102	0.054305
	Validation	0.947622	0.13676	0.052019
CNN	Train	0.768843	0.248892	0.230254
	Validation	0.763774	0.252543	0.245707
ANN	Train	-0.01409	0.769897	0.98244
	Validation	-0.013036	0.77841	1.047807

The LSTM is the most effective and reliable model for self-supervised GNSS time-series reconstruction. It consistently achieves the lowest prediction errors, highest explanatory power,

best stability across masked regions, and the most favorable convergence properties. GRU remains a competitive alternative, while CNN and ANN are not suitable for this class of tasks. The results solidify LSTM as the recommended architecture for downstream applications involving GNSS-based displacement reconstruction, SHM, and digital twin modeling.

3.2.2. Actual Gap Recovery Performance

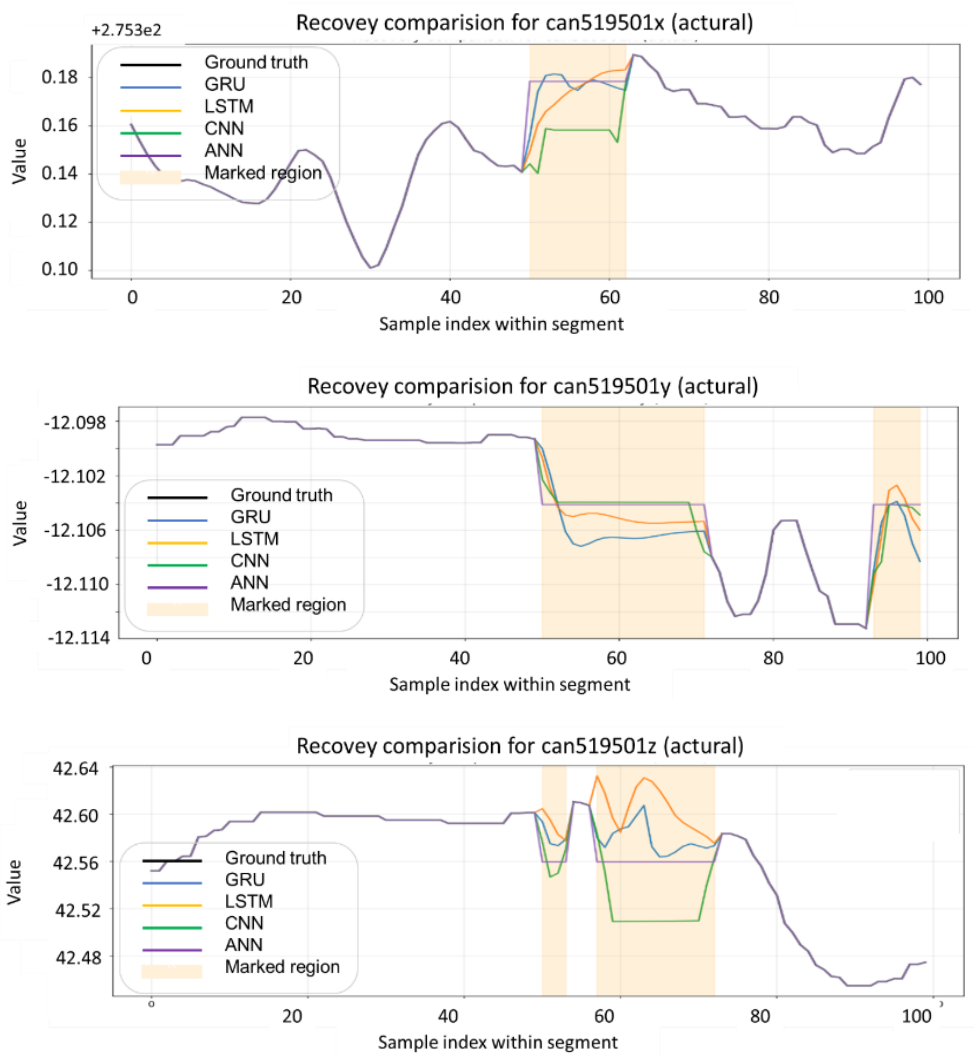

Figure 8. Actual gap recovery results for can519501x, can519501y, and can519501z components.

The recovery results for the actual missing-data segments in the can519501x, can519501y, and can519501z components are shown in Fig. 8. Because these gaps correspond to real data losses, no Ground Truth is available within the masked intervals. Thus, the evaluation focuses on the continuity of the reconstructed signal and whether each model produces estimates that remain consistent with the observable trends immediately before and after the gap.

The LSTM model typically generates smooth and continuous reconstructions, preserving the local trend and avoiding abrupt fluctuations within the missing region. The GRU model exhibits generally similar behavior, although minor deviations can occasionally be observed near the boundaries of the gap. The CNN model tends to produce plateau-like or step-shaped

reconstructions, indicating difficulty in extrapolating temporal dynamics when long-range context is unavailable. Meanwhile, the ANN model frequently yields nearly constant or weakly varying outputs, reflecting its limited capability to infer temporal patterns in the absence of direct observations.

Overall, the models demonstrate distinct reconstruction characteristics under real missing data conditions: LSTM tends to maintain smooth transitions across the gap, GRU follows the surrounding trend with slight boundary inconsistencies, CNN often produces discontinuities or overly simplified representations, and ANN exhibit minimal temporal adaptation. These behaviors provide insight into how each architecture responds when required to infer unobserved portions of the GNSS time series based solely on contextual information.

4. CONCLUSION

GNSS displacement measurements using deep neural networks. The proposed pipeline integrates smoothing, Kalman filtering, wavelet denoising, sliding-window segmentation, and middle-region masking, enabling model training directly on real GNSS time series without requiring synthetic gap generation. Evaluation using the Masked-Region Protocol demonstrated that the framework yields consistent and reproducible assessments, allowing models to be compared objectively by predicting intentionally hidden segments with ground-truth availability. The experimental results revealed distinct reconstruction behaviors across ANN, GRU, LSTM, and CNN architectures, underscoring the effectiveness of the self-supervised strategy for real-world GNSS gap recovery and its potential applicability within SHM and digital twin systems.

Future research will explore advanced sequence models such as Transformers, Temporal Convolutional Networks (TCN), and diffusion-based generative frameworks to further enhance long-range temporal modeling. Incorporating multi-sensor fusion combining GNSS with accelerometers, IMUs, or vision-based displacement estimation may improve robustness under complex environmental and operational conditions. Additionally, adaptive and context-aware masking strategies, together with uncertainty quantification, will be investigated to better reflect realistic missing-data patterns and provide reliability measures for reconstructed outputs. Finally, the proposed method will be validated on long-term field datasets from operational bridge monitoring systems, aiming to assess scalability, environmental resilience, and readiness for deployment in next-generation SHM and digital twin platforms.

ACKNOWLEDGEMENT

This research is funded by University of Transport and Communications (UTC) under grant number T2025-CT-007TĐ.

REFERENCES

[1]. J. M. Dow, R. E. Neilan, C. Rizos, The International GNSS Service in a changing landscape of global navigation satellite systems, Journal of Geodesy, 83 (2009) 191–198. https://doi.org/10.1007/s00190-008-0300-3

[2]. D. C. Tran, T. L. H. Ho, K. G. Le, V. H. Le, Application of unsupervised clustering algorithms in GNSS-RTK data analysis for cable-stayed bridge monitoring, Transport and Communications Science Journal, 76 (2025) 1138–1150 (in Vietnamese). https://doi.org/10.47869/tcsj.76.8.8

[3]. H. Liu, L. Li, Missing data imputation in GNSS monitoring time series using temporal and spatial Hankel matrix factorization, Remote Sensing, 14 (2022) 1500. https://doi.org/10.3390/rs14061500.

[4]. M. Lepot, J.-B. Aubin, F. H. Clemens, Interpolation in time series: An introductive overview of

existing methods, their performance criteria and uncertainty assessment, Water, 9 (2017) 796. https://doi.org/10.3390/w9100796
[5]. J. Wang, W. Du, Y. Yang, L. Qian, W. Cao, K. Zhang, W Wang, Y Liang, Q Wen, Deep learning for multivariate time series imputation: A survey, arXiv preprint, arXiv: 2402.04059, 2024.
[6]. T. Kim, J. Kim, W. Yang, H. Lee, J. Choo, Missing value imputation of time-series air-quality data via deep neural networks, International Journal of Environmental Research and Public Health, 18 (2021) 12213. https://doi.org/10.3390/ijerph182212213
[7]. F. M. Shiri, T. Perumal, N. Mustapha, R. Mohamed, A comprehensive overview and comparative analysis on deep learning models: CNN, RNN, LSTM, GRU, arXiv preprint, arXiv:2305.17473, 2023.
[8]. S. F. Ahmed, M. S. B. Alam, M. Hassan, M. R. Rozbu, T. Ishtiak, N. Rafa, M. Mofijur, A. B. M. S. Ali, A. H. Gandomi, Deep learning modelling techniques: Current progress, applications, advantages, and challenges, Artificial Intelligence Review, 56 (2023) 13521–13617. https://doi.org/10.1007/s10462-023-10466-8
[9]. L. Ericsson, H. Gouk, C. C. Loy, T. M. Hospedales, Self-supervised representation learning: Introduction, advances, and challenges, IEEE Signal Processing Magazine, 39 (2022) 42–62. https://arxiv.org/pdf/2110.09327
[10]. Z. Liu, A. Alavi, M. Li, X. Zhang, Self-supervised contrastive learning for medical time series: A systematic review, Sensors, 23 (2023) 4221. https://doi.org/10.3390/s23094221
[11]. Y. Wang, H. Ding, H. Li, Multivariate time-series missing data imputation with convolutional transformer model, Symmetry, 17 (2025) 686. https://doi.org/10.3390/sym17050686
[12]. M. Casella, N. Milano, P. Dolce, D. Marocco, Transformers deep learning models for missing data imputation: An application of the ReMasker model on a psychometric scale, Frontiers in Psychology, 15 (2024) 1449272. https://doi.org/10.3389/fpsyg.2024.1449272
[13]. W. Du, D. Côté, Y. Liu, SAITS: Self-attention-based imputation for time series, Expert Systems with Applications, 219 (2023) 119619. https://doi.org/10.1016/j.eswa.2023.119619
[14]. O. I. Abiodun, A. Jantan, A. E. Omolara, K. V. Dada, N. A. Mohamed, H. Arshad, State-of-the-art in artificial neural network applications: A survey, Heliyon, 4 (2018) e00998. https://doi.org/10.1016/j.heliyon.2018.e00938
[15]. J. Dong, H. Wu, H. Zhang, L. Zhang, J. Wang, M. Long, SimMTM: A Simple Pre-Training Framework for Masked Time-Series Modeling, Advances in Neural Information Processing Systems, 36 (2023) 29996-30025. https://doi.org/10.48550/arXiv.2302.00861
[16]. Z. Li, Z. Rao, L. Pan, P. Wang, Z. Xu, Ti-MAE: Self-Supervised Masked Time Series Autoencoders, (2023). https://doi.org/10.48550/arXiv.2301.08871
[17]. Z. Liu, B. Du, J. Ye, X. Wen, L. Sun, An NCDE-based Framework for Universal Representation Learning of Time Series, in Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence (2024) 4623–4633. https://doi.org/10.24963/ijcai.2024/511
[18]. Y. Wang, H. Wu, J. Dong, G. Qin, H. Zhang, Y. Liu, Y. Qiu, J. Wang, M. Long, TimeXer: Empowering Transformers for Time Series Forecasting with Exogenous Variables, Advances in Neural Information Processing Systems, 37 (2024) 469-498. https://doi.org/10.48550/arXiv.2402.19072
[19]. R. Dey, F. M. Salem, Gate-variants of gated recurrent unit (GRU) neural networks, in: Proceedings of the IEEE 60th International Midwest Symposium on Circuits and Systems, (2017) 1597–1600. https://doi.org/10.1109/MWSCAS.2017.8053243
[20]. S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Computation, 9 (1997) 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735
[21]. J. Gu, Z. Wang, J. Kuen, L. Ma, B. Shahroudy, B. Shuai, et al., Recent advances in convolutional neural networks, Pattern Recognition, 77 (2018) 354–377. https://doi.org/10.1016/j.patcog.2017.10.013
[22]. D. C. Montgomery, E. A. Peck, G. G. Vining, Introduction to Linear Regression Analysis, sixth ed., John Wiley & Sons, 2021.
[23]. Y. Bengio, I. Goodfellow, A. Courville, Deep Learning, MIT Press, Cambridge, MA, 2017.