CPSC-5616: Assignment 4 Report

Group Number: 10
Group Member:

NAME	STUDENT#	EMAIL
Haoliang Sheng	0441916	hsheng@laurentian.ca
Songpu Cai	0441024	scai1@laurentian.ca

1. Introduction

Building on prior exploration of distance estimation, we focus on leveraging deep learning techniques to transition from traditional machine learning models to advanced neural network approaches. This study develops and evaluates neural network models that learn complex patterns from IMU sensor data to accurately estimate the distance traveled by a robot. By comparing the performance of conventional machine learning methods, such as Random Forest Regressor (RFR) and Support Vector Regressor (SVR), with deep learning models, including CNN-GRU hybrids, CNN-only, and GRU-only architectures, we aim to highlight the advantages and limitations of each approach. This investigation provides valuable insights into the application of deep learning techniques for time-series sensor data and their potential to improve motion estimation accuracy.

2. Data Preprocessing

Preprocessing

The raw IMU data, containing accelerometer and gyroscope readings, underwent several preprocessing steps to ensure consistency and usability for subsequent modeling tasks:.

- **Normalization**: The sensor data for each axis (x, y, z) was normalized to standardize the scale across features, ensuring that all features contribute equally to the model.
- **Noise Reduction**: A moving average filter with a window size of 5 was applied to smooth out noise in the signal.
- **Resampling**: The data was resampled to a uniform sampling rate of 25 Hz, which provides consistency across all segments while maintaining sufficient granularity for capturing motion dynamics.

Segmentation

To isolate meaningful motion patterns, the preprocessed data was segmented based on **activity density**:

- **Activity Density:**
  - Computed as the rolling standard deviation of the vertical acceleration (z-axis). High activity density indicates movement, while low activity density indicates stationary states.
- **Segmentation Parameters:**

- A threshold of 0.2 was used to identify active motion segments.
- Segments shorter than 75 time steps were discarded to ensure adequate temporal coverage for analysis.

Meaningful segments capturing distinct motions were successfully extracted, as shown in Fig. 1, which highlights activity density over time with segmented regions marked.

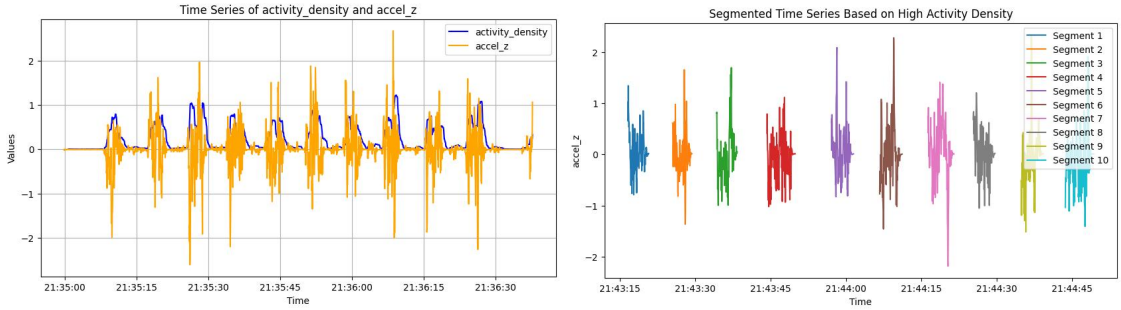

Fig. 1 Segmentation Result

This segmentation approach ensures that only relevant data is used for modeling, reducing noise and computational overhead.

Data Augmentation

To enhance the robustness and generalization of the model, the dataset was augmented using the following techniques:

- **Window Overlap:**
  - Overlapping windows of motion segments were generated to increase the dataset size while preserving temporal consistency.
- **Synthetic Variability:**
  - **Jittering**: Small Gaussian noise was added to the data to simulate sensor noise.
  - **Scaling**: Data values were scaled by random factors to simulate changes in sensor sensitivity.
  - **Time-Warping**: The time axis of the data was stretched or compressed to simulate variations in movement speed.
- **Segment Combination**:
  - To capture a diverse range of motion scenarios, segments of varying lengths were synthesized. Specifically, segments ranging from **1 meter to 50 meters**, increasing incrementally by 1 meter, were generated. For each segment length L (e.g., 1 meter, 2 meters, ... up to 50 meters), overlapping windows were combined sequentially from the dataset. This process ensures that the model is exposed to a wide variety of motion sequences across different distances

These augmentation strategies effectively doubled the training dataset, as illustrated in Fig. 2 (augmented vs. original dataset distribution).

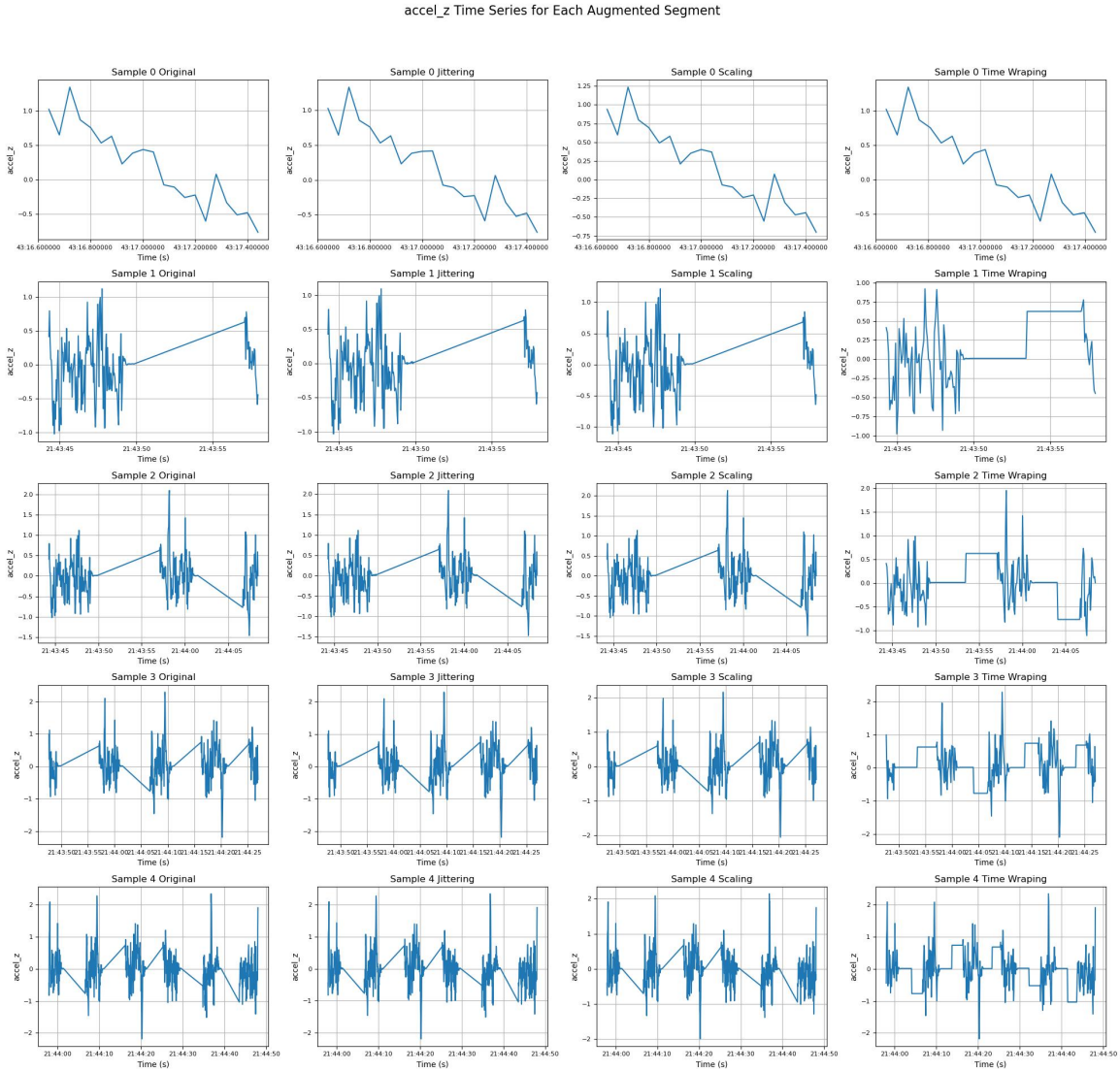

Fig. 2 Data Augmentation

Feature Extraction

Feature extraction was tailored to the requirements of both traditional machine learning models (RFR, SVR) and deep learning models (CNN, GRU, CNN-GRU), with adjustments for computational efficiency and model-specific needs.

- **Padding and Normalization:**
  - Each sample was padded to a fixed length of 3000 rows (equivalent to 120 seconds at a sampling rate of 25Hz) using zeros.
  - This ensured uniform input size across all samples.
- **Window Size Calculation:**
  - The window size for calculating statistical features was determined as 3000 / sequence_length:
    - For machine learning (sequence length = 10): window size = 3000 / 10 = 300.
    - For deep learning (sequence length = 300): window size = 3000 / 300 = 10.

- **Feature Calculation:**
  - Within each window, the following statistical features were computed for acceleration and gyroscope data: mean, standard deviation, max, min, median, and integral.
  - These features captured key properties of the motion data for each axis (x, y, z) and their magnitudes.

For Traditional Machine Learning:

- Extracted features were flattened into a single feature vector per sequence, resulting in 480 features (10 windows × 48 features) per segment.
- This structure was suitable for models like RFR and SVR, which operate on tabular data but lack temporal awareness.

For Deep Learning Models:

- The data retained its 2D structure (sequence × features), preserving the temporal relationships across 300 windows.
- This approach allowed CNN and GRU-based models to learn both spatial and temporal patterns directly from the data.

By leveraging padding and appropriate window sizes, this approach balanced computational efficiency with model-specific requirements, enabling both types of models to perform effectively.

3. Model Architecture

CNN-GRU Model

The CNN-GRU model was designed to combine the strengths of Convolutional Neural Networks (CNN) and Gated Recurrent Units (GRU):

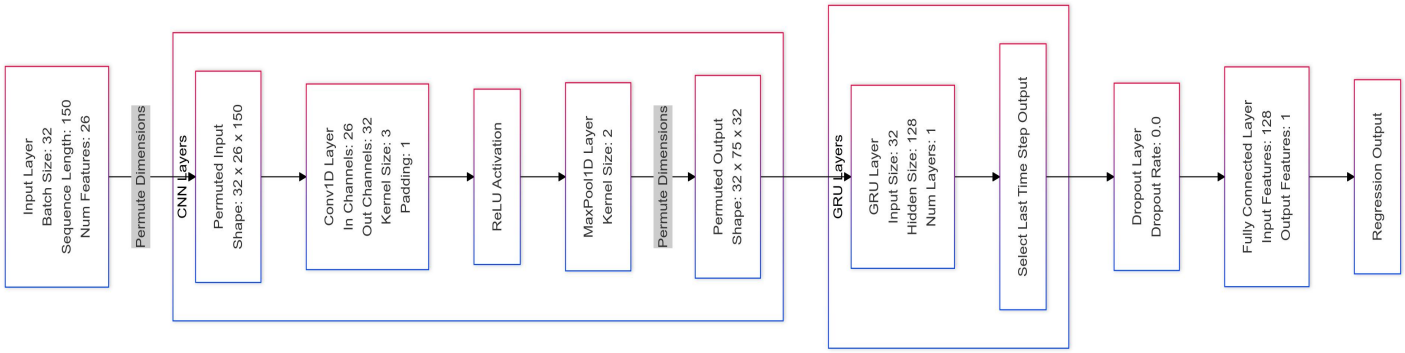

Fig. 3 CNN-GRU Architecture

- **CNN Layers:**
  - Extract spatial patterns and local dependencies from the time-series data.
  - Utilize convolutional operations followed by pooling to reduce dimensionality and focus on significant features.
- **GRU Layers:**
  - Capture temporal dependencies in the sequential data.
  - Process the output from the CNN layers to understand motion trends over time.
- **Fully Connected Layers:**
  - Integrate features from the GRU's output to produce a final regression value, estimating the distance traveled.

Why CNN-GRU?

The hybrid approach leverages:

- **CNN's strength**: Detecting local spatial features in the time-series data.
- **GRU's strength**: Capturing long-term dependencies and handling sequential data efficiently. This combination ensures the model is capable of learning both short-term variations and global motion trends, which are critical for estimating distances from IMU data.

4. Evaluation

Hyperparameter Tuning

A grid search was conducted to optimize the CNN-GRU model's key hyperparameters, including learning rate, batch size, dropout rate, number of filters, and hidden size. The results are summarized in Fig. 4, showing the validation loss for each hyperparameter configuration.

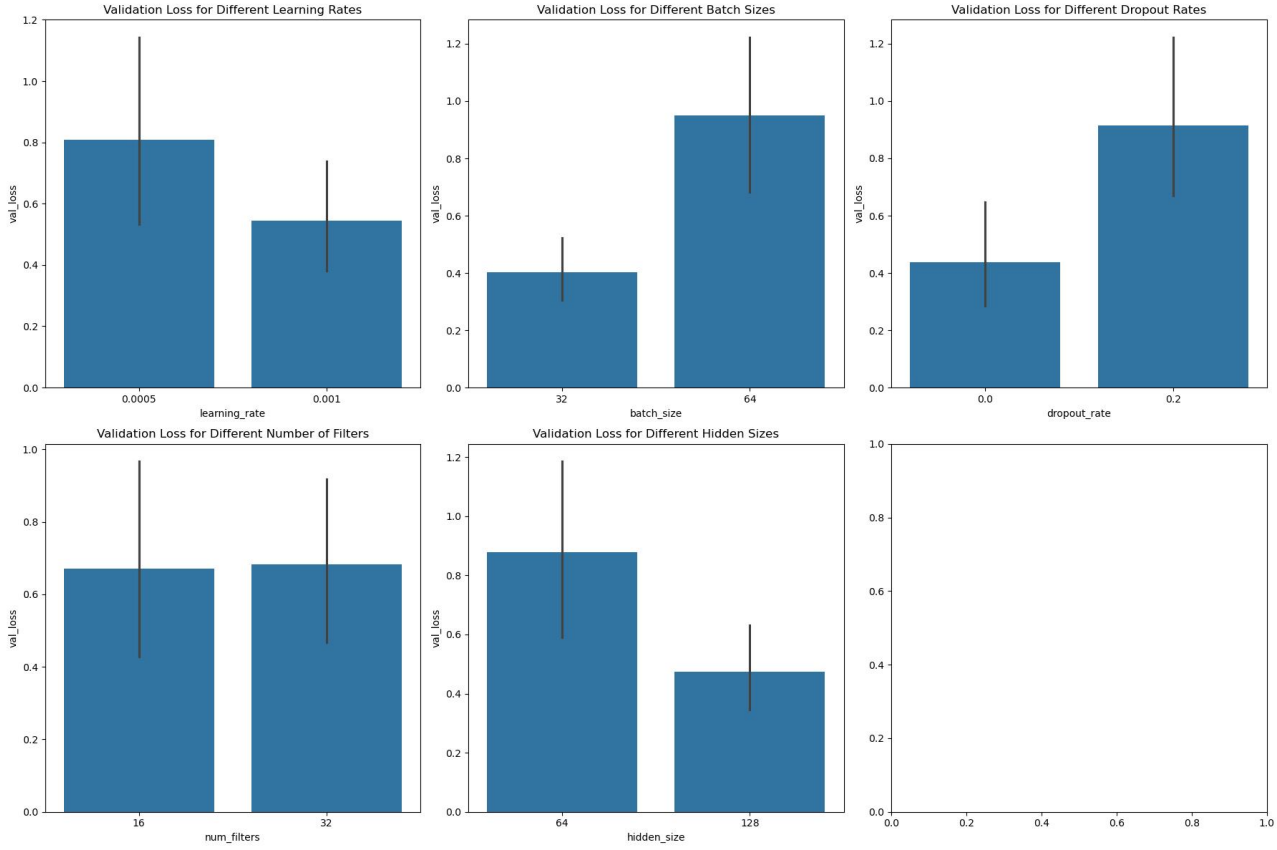

Fig. 4 Hyperparameter Tuning

The best-performing configuration was:

- **Learning Rate**: 0.001
- **Batch Size**: 32
- **Dropout Rate**: 0.0
- **Number of Filters**: 32
- **Hidden Size**: 128

This setup achieved the lowest validation loss and served as the final configuration for further evaluation.

Cross-Validation

To ensure the robustness of the CNN-GRU model, a 5-fold cross-validation was conducted. This process provides a comprehensive evaluation of the model's performance across different splits of the dataset, reducing the risk of overfitting and offering more reliable performance metrics.
The results of the cross-validation are summarized as follows:

Table 1 Cross-Validation

Metric	Average Value	Standard Deviation
MAE	0.25	± 0.02
RMSE	0.36	± 0.03
R^2 Score	1.00	± 0.00

These results demonstrate the consistent and superior performance of the CNN-GRU model, with minimal variation in metrics across the validation folds. The near-perfect R^2 score highlights the model's ability to capture the underlying patterns in the data effectively.

5. Results

The comparison between traditional machine learning models (RFR, SVR) and deep learning models (CNN-GRU Hybrid, CNN-only, GRU-only) reveals the following insights (Table 2):

Table 2 Comparison between Models

Model	Validation MAE	Validation RMSE	Validation R^2 Score	Test Data Prediction
RFR	0.43	0.71	1.00	43.145
SVR	0.64	0.94	0.99	35.24477002
CNN-GRU Hybrid	0.23	0.34	1.00	46.719585
CNN-only	0.49	0.71	1.00	39.293808
GRU-only	0.41	0.54	1.00	25.76682

- **Validation Metrics:**
  - Deep learning models generally outperform traditional machine learning models on validation metrics.
  - Among deep learning models, the **CNN-GRU Hybrid** demonstrates the best validation performance with the lowest MAE (0.23) and RMSE (0.34), alongside an R^2 Score of 1.00.
- **Test Data Predictions:**

- On the test data, **GRU-only** provides a prediction (25.77 meters) closest to the true distance (27 meters), outperforming the CNN-GRU Hybrid model (46.72 meters).
- The discrepancy between validation performance and test prediction accuracy for CNN-GRU Hybrid could indicate:
  - **Overfitting**: The model might have adapted too well to the training/validation data.
  - **Distribution Difference**: The training/validation data and test data may not share the same distribution.

- **Loss Curves:**
  - As we can see in Fig. 5:
    - The **GRU-only** model shows the fastest convergence, highlighting its efficiency in learning temporal dependencies.
    - The **CNN-only** model exhibits slower convergence, potentially due to its limited ability to capture sequential dependencies.
    - The **CNN-GRU Hybrid** balances both convergence speed and accuracy, leveraging the feature extraction capability of CNNs and the sequential modeling strength of GRUs.

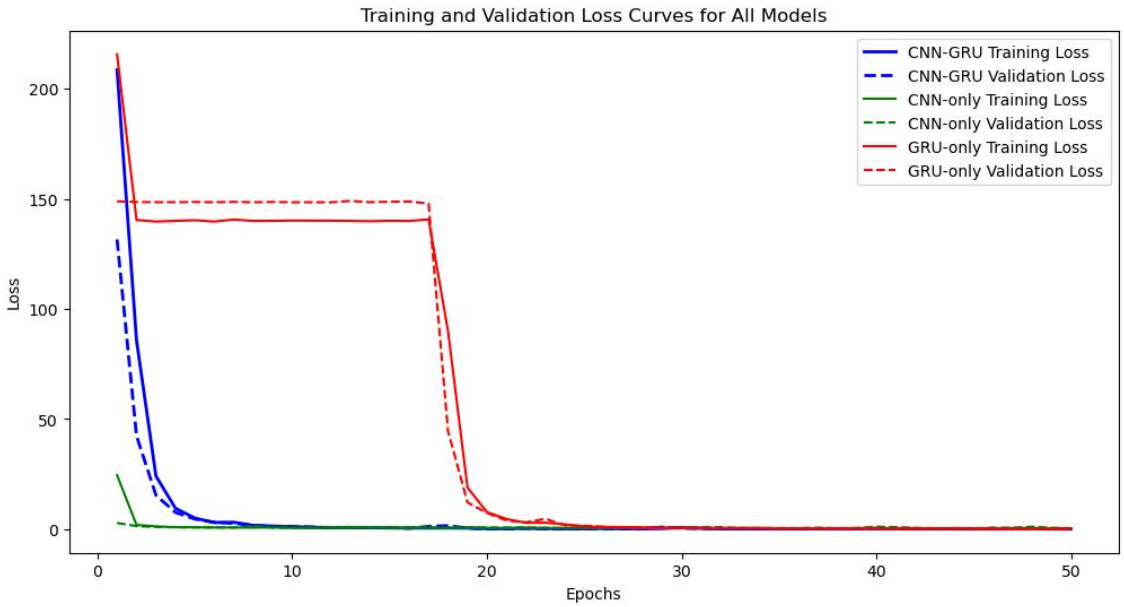

Fig. 5 Loss Curves

The results suggest that while CNN-GRU Hybrid excels in validation metrics, GRU-only provides more reliable predictions on unseen test data. This highlights the importance of analyzing both validation and test performance to ensure model generalization. Further investigation into the distribution of the training, validation, and test datasets could provide additional insights into model behavior.

6. Future Work and Improvements

- **Explore Advanced Models**

- **Transformers and Attention Mechanisms**: Transformers, with their self-attention mechanisms, can capture long-range dependencies more effectively than GRUs or LSTMs. Applying transformer-based architectures, such as the Time-Series Transformer, could further improve the model's ability to understand complex temporal patterns in IMU data. Additionally, attention layers could be added to highlight the most relevant time steps or features in the input sequence.
- **Hybrid Architectures**: Expanding the model design to include architectures like LSTM-CNN or transformer-CNN hybrids could leverage the sequential modeling power of recurrent networks or transformers and the spatial feature extraction strengths of CNNs. These hybrid architectures may provide more robust and flexible solutions for handling diverse motion data scenarios.

- **Enhance Data Diversity**
  - **Data Collection Across Diverse Environments**: To improve generalization, future work could involve collecting IMU data in varied environments, such as outdoor terrains, different weather conditions, and varied motion patterns. This would expose the model to a broader range of scenarios and reduce overfitting to specific training conditions.
  - **Domain Adaptation**: Techniques like transfer learning or adversarial domain adaptation can help align the training and test data distributions, addressing challenges posed by differences between these datasets. For example, models can be pre-trained on a larger, synthetic dataset and fine-tuned on a smaller real-world dataset for improved performance.
- **Improve Feature Engineering**
  - **Frequency-Domain Features**: Beyond the time-domain features currently used, frequency-domain features like power spectral density or wavelet transforms could be incorporated. These features can capture periodic patterns or oscillations in IMU data, providing additional insights into motion dynamics.
  - **Automated Feature Selection**: Techniques like recursive feature elimination (RFE), principal component analysis (PCA), or neural network-based autoencoders could be explored to identify the most informative features automatically. This would reduce redundancy and potentially enhance the model's efficiency and accuracy.

By focusing on these areas, future iterations of the model can address existing limitations, adapt to broader real-world applications, and achieve higher accuracy and robustness in motion prediction tasks.