# **Assignment 4 (10%):**

**Distance Estimation of Robot Using IMU Data (Part 2/2): Deep Learning Approach**

DUE DATE: Friday, Dec 6th, 2024, 11:59 PM ET

**Assignment Guidelines:**

* All students must adhere to the [Student Academic Integrity](https://intranet.laurentian.ca/policies/2017.09.19%20-%20Policy%20and%20Procedures%20on%20Academic%20Integrity%20-%20EN.pdf).
* Assignments must follow the programming standards outlined in the document on the course website via D2L. Marks will be deducted if these standards are not followed.
* **Submit a single** .py **or** .ipynb **file and an associated PDF report per group**. Name the file according to your group "ID" and assignment number, per this example for Assignment 3 and Group 1: CPSC\_5616EL\_A4\_G1.py. Apply the same naming convention for the PDF: CPSC\_5616EL\_A4\_G1.pdf.
* **Do NOT zip the files** when submitting.
* Multiple submissions are allowed, but only the most recent version will be marked.
* A late penalty of 2% per hour (or part thereof) will be applied after the deadline. After 49 hours, the penalty will be 100%, and submissions will no longer be accepted.
* These assignments are key learning tools for the exams. Code your assignments independently within your group. Plagiarism detection software may be used to compare all submissions, and academic dishonesty will be pursued vigorously.
* **Generative AI tools (e.g. ChatGPT, Gemini, Autopilot, etc.) are permitted,** provided their use is properly cited. Ensure that any text or code generated through these tools is acknowledged in both your code comments and report. **Failure to cite generative AI usage will be considered a violation of academic integrity.**

1. **Introduction**

Building on Part 1/2, this assignment (Part 2/2) explores distance estimation using deep learning techniques, transitioning from traditional machine learning models to a deep neural network approach. You’ll develop and test a neural network model capable of learning complex patterns from IMU sensor data to estimate the distance travelled by the robot. This will allow you to observe the performance differences between conventional machine learning methods and deep learning techniques for time-series sensor data.

1. **Data Preparation (10 marks)**

* **Data Preprocessing**: Utilize the same preprocessing methods from Part 1 (normalization, noise reduction, resampling). If any adjustments are necessary for compatibility with the deep learning model, describe and justify them in your report.
* **Feature Extraction**: Extract features as input for the model. You may continue using statistical features or apply raw IMU sensor data directly.
* **Data Augmentation**: Implement data augmentation to further expand the training dataset, considering:
  + **Window Overlap**: Increase the number of samples by overlapping windows in the IMU data.
  + **Synthetic Variability**: Apply jittering, scaling, and time-warping techniques to add variability and enhance the model’s robustness.
  + **Segmentation Strategy**: Break down trajectories into smaller segments for finer-grained distance estimation.

1. **Hybrid Models (CNN-GRU) Design and Implementation - PyTorch (30 marks)**

* Model Architecture: Create a hybrid model that **combines CNN and GRU layers** to utilize the strengths of both architectures.
* Training: Train the hybrid model on the dataset.

**Hybrid Model Detailed Architecture**:

* **Input Layer**: Accept the preprocessed IMU data or feature set. Start with an input layer that matches the shape of the preprocessed dataset (time steps, features).
* **Convolutional Layers**: Add **one or more 1D convolutional layers** (Conv1D) to extract features from the time-series data. Follow each Conv1D layer with an activation function like ReLU, and include pooling layers (MaxPooling1D) to reduce the feature maps' dimensionality.
* **Flattening or Global Pooling**: After the convolutional layers, use a global pooling layer to reduce the output to a 1D vector, which can help decrease the number of parameters and prevent overfitting.
* **Recurrent Layers**: Feed the flattened output into GRU layers to process the data sequentially and capture time dependencies. The number of GRU layers can vary based on the complexity of the task.
* **Output Layer**: Provide a single output that estimates the total distance. Conclude the model with a dense layer using an activation function appropriate for the number of classes (e.g., softmax for multi-class classification).
* **Model Compilation**: Compile the model using an optimizer like Adam, a loss function such as categorical\_crossentropy for multi-class classification, and metrics like accuracy.

**Training and Validation**:

* Split the dataset into training, and validation sets.
* Use appropriate loss functions and optimizers. (Mean Squared Error (MSE) or Mean Absolute Error (MAE) for regression tasks is recommended).
* Experiment with various hyperparameters such as learning rate, batch size, and epochs.

1. **Report and Analysis (20 marks)**

**Evaluation:**

* **Evaluate the model using metrics from Part 1** (MAE, RMSE, R²) and discuss its performance compared to RFR and SVR.
* **Hyperparameter Tuning**: Tune hyperparameters like learning rate, batch size, dropout rate, and network depth to optimize model performance.
* **Cross-Validation**: Consider cross-validation to verify the model’s robustness.

**Report and Analysis:**

* **Data Processing**: Briefly describe the steps taken for data preprocessing and feature extraction.
* **Model Architecture and Justification**: Explain the architecture, layers, and parameters of your deep learning model. Discuss why you chose this specific architecture.
* Comparison with ML Models: Analyze the improvements or drawbacks of deep learning models compared to **traditional machine learning models implemented in Part 1 (RFR and SVR)**.

**Future Work and Improvements:**

* Suggest any future work or improvement such as other models and techniques

**Note on Toolset Usage:**

You are encouraged to make full use of all the machine learning libraries and tools we've discussed in class. This includes but is not limited to, libraries like **pytorch, scikit-learn, Numpy, pandas, and matplotlib**. The objective is to familiarize yourself with real-world application scenarios and to simplify certain processes where needed. However, always ensure you understand the underlying mechanics of the tools you're using. Proper documentation and justification of your chosen methods and tools are essential for a comprehensive assessment.

**Submission Guidelines:**

1. Submit your **code** as a Jupyter Notebook (.ipynb) - **preferable** or Python script (.py).
   1. If submission in a Python file, make sure to attach a readme in your submission for additional setup or packages needed to run your Python script.
2. Submit your **dataset (post-process)** along with the code and **report**.
3. You may use a comprehensive Jupyter Notebook as your report. In this case, save the notebook as a PDF and ensure it includes all required sections such as dataset justification, results discussion, and recommendations.
4. If submitting a separate report, it must be in PDF format. Ensure all plots, figures, and tables are correctly labelled.
5. Your submission should be self-contained, meaning a person should be able to understand your process and results just by reading your report and going through your code.

**Evaluation Criteria:**

1. **Preprocessing and Augmentation**: Effectiveness of data preparation, windowing, and augmentation.
2. **Model Implementation**: Correct implementation of the deep learning model with meaningful architecture.
3. **Performance Analysis**: Depth of comparative evaluation with prior models.
4. **Report Clarity**: Clarity of explanations, insights, and discussion on limitations and improvements.

**Group Contributions and Grading:**

If any group member feels another member’s contribution is not proportional, this can be addressed. Include an additional section in the report detailing the situation and the proposed grade adjustment, with the agreement of all group members. Ensure all group members provide consent to any grade changes.

Open communication and collaboration are always encouraged. However, if discrepancies in contributions are significant, this mechanism ensures fairness in grading.

**Best of luck! Remember, the learning process is just as important as the final result.**