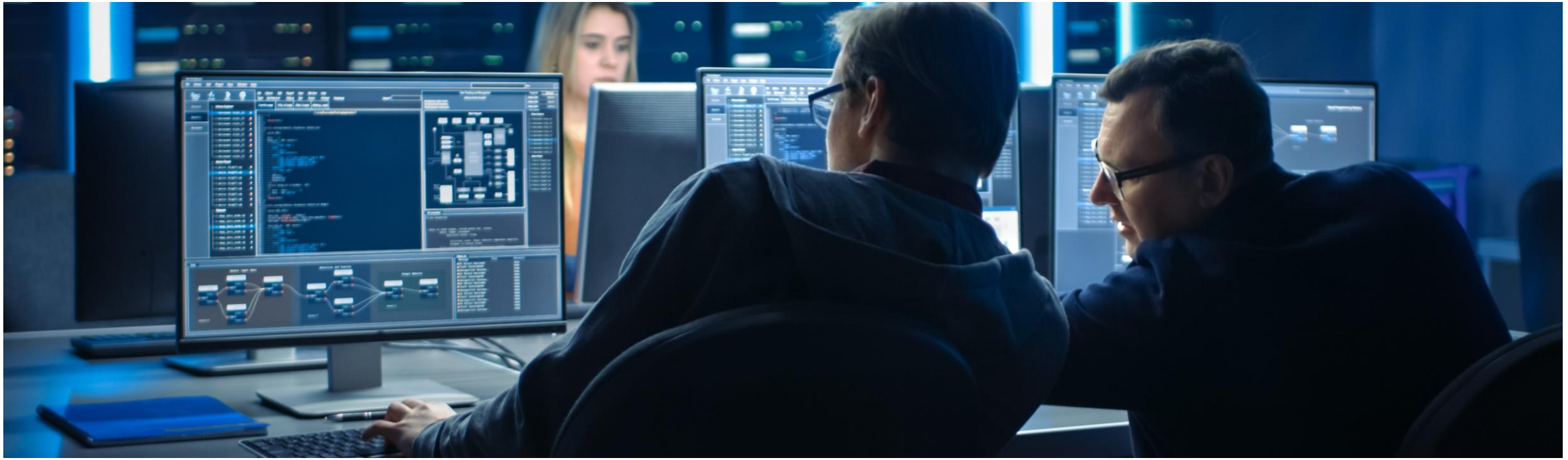
BOQ DATE DELIVERY PREDICTION (BOQ_DDP)
A RECURRENT NEURAL NETWORK FOR DELIVERY DATE PREDICTION



IN A NUTHSHELL

1. Objective
    a. BoQ Delivery Date Prediction (BoQ_DDP) neural network, developed as part of a Proof-of-Concept (POC) to predict delivery dates for equipment items based on historical shipment data. Useful for Presales/PM/PEO
2. Dataset
    a. Source: Shipment delivery report 2024 and Q12025
    b. Size: 76,000 records
    c. Features: 14 variables
    - Useful for training
        - Customer Request Date
        - Order In Take Date
        - Actual Ship Date
        - LeadTimeDays
        - IntakeLag
3. Model Goals
    a. Build an accurate and generalizable model for managing time sequences of events
    b. Analyze feature importance
    c. Visualize training behavior and performance metrics
4. Development
    a. Framework: Colab of Google, Tensorflow
    b. Language: Python 3.13 and 3.10
    c. Libraries: NumPy, Pandas
    d. Collaborator: ChatGPT 4o Plus
5. Bibliography and Inspiration:
    a) *Python and Machine Learning A. Bellini, A. Guidi, McGraw-Hill.* Study case: *Predict the price of a stock, in our case Microsoft, starting from datasets available on Kaggle (https://www.kaggle.com/dgawlik/nyse)*

INPUT DATA AND PROCESSING

1. Dataset
    a. Source: Shipment delivery report 2024 and Q12025
    b. Size: 76,000 records
    c. Features: 14 variables
2. Useful for training
    a. Customer Request Date
    b. Order In Take Date
    c. Actual Ship Date
    d. LeadTimeDays (as Actual Ship Date - Customer Request Date)
    e. IntakeLag (as Order In Take Date - Customer Request Date)
3. Preprocessing
    a. Convert the Item_ID column (which contains product identification strings) into a numeric representation that the neural network can use as input
    b. Sorting the dataset by Cust_Req_Ship_Date in ascending order within each group
    c. Normalization. MinMax normalization scales the feature LeadTimeDays in the range [0, 1]
    d. Convert the DataFrame into a time sequence dataset for each Item_ID, useful for training sequential models such as LSTM or GRU (sliding windows)
    e. Splits the dataset into a training set (70%) and a validation set (30%), based on a specified percentage. Training set is used for training phase; validation set is used for validation phase. Validation is used to verify that the training carried out with the training set does not generate overfitting.

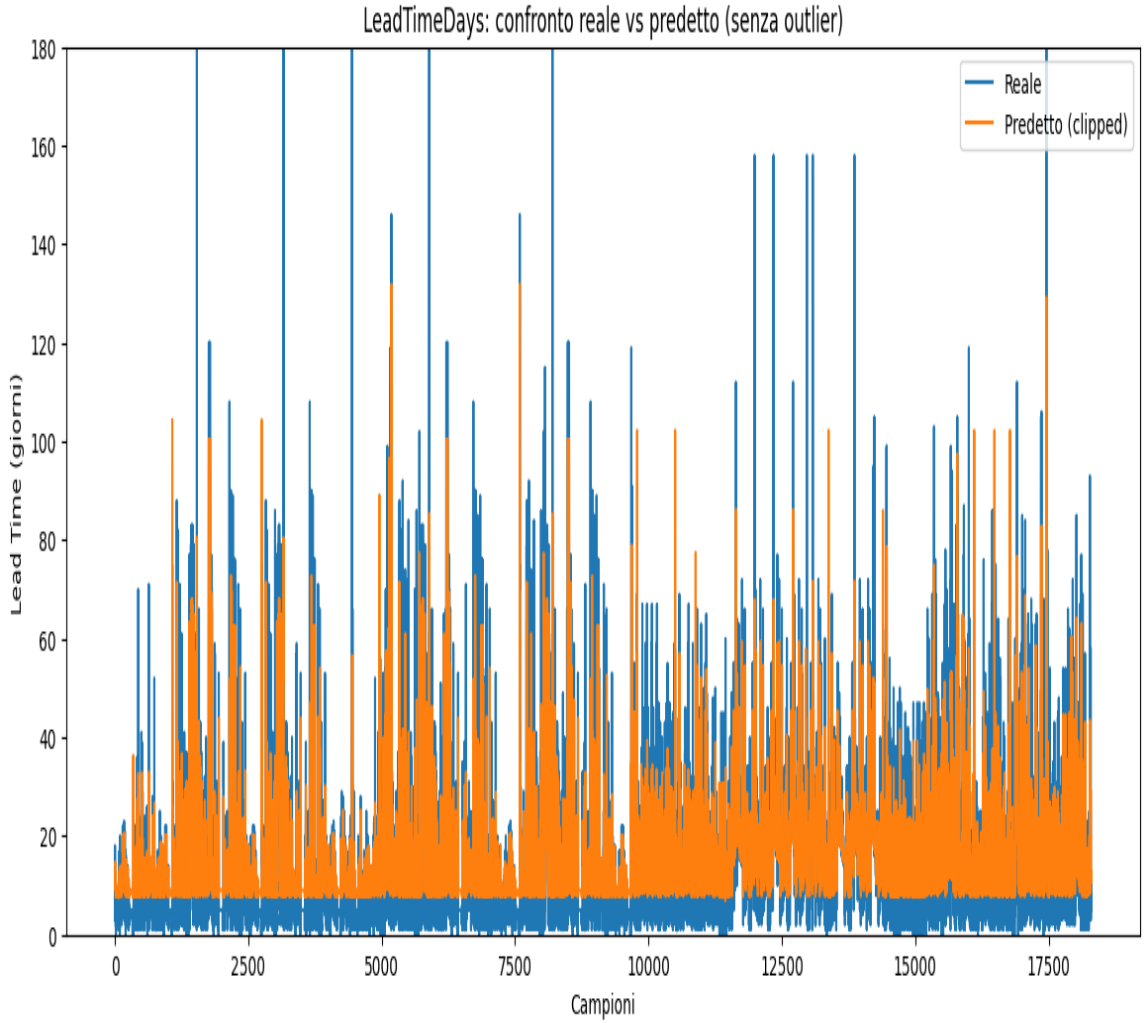
ARCHITECTURE OF RNN (GRU+LSTM)

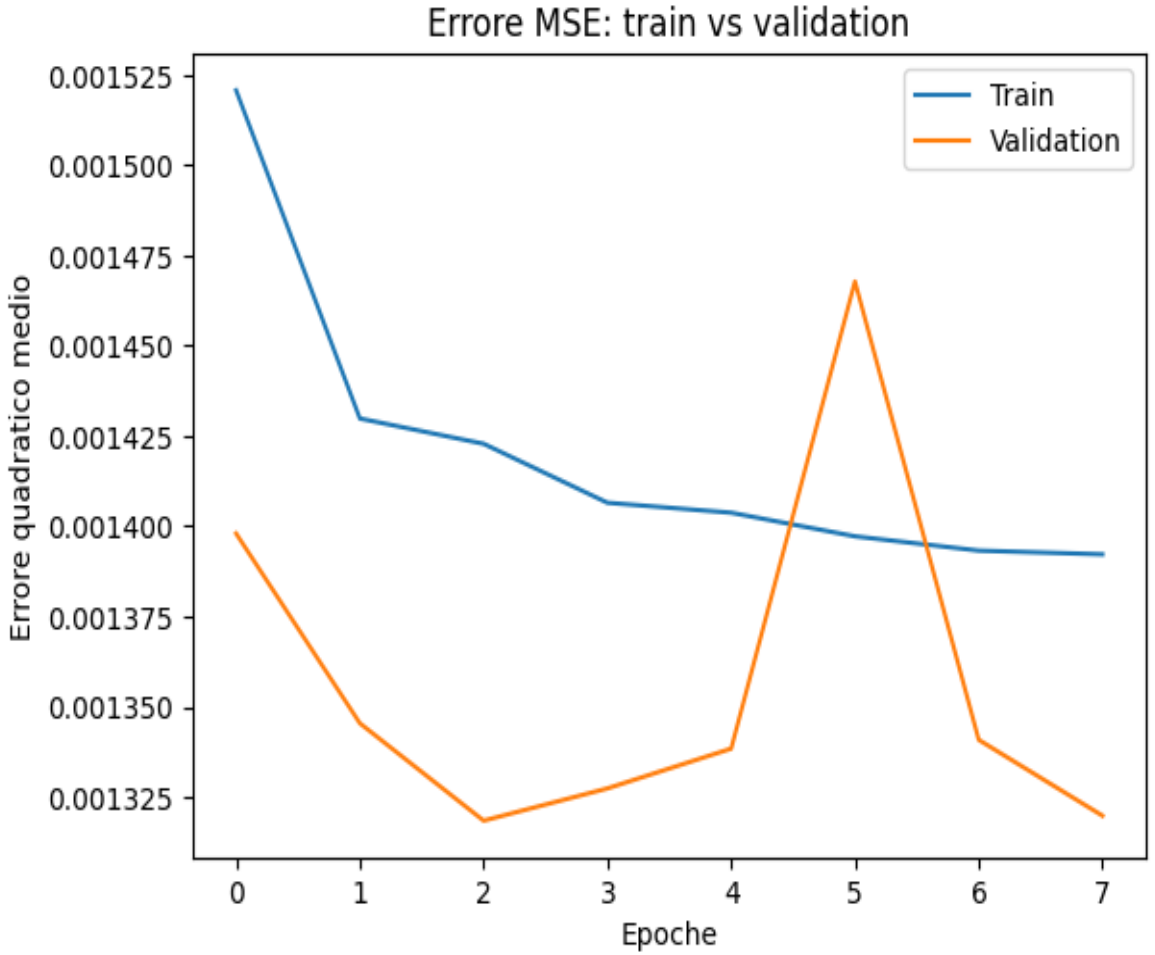
1. GRU+LSTM
    a. The problem is sequential/temporal in nature: lead time prediction relies on past chronological data per item (e.g., IntakeLag, historical LeadTimeDays).
    b. GRU and LSTM architectures enhance memory handling and temporal dependency learning.
2. In our model:
    a. GRU processes the raw input and returns a sequence of representations.
    b. LSTM receives the sequence and learns a compact representation.
    c. Dense layers produce the final prediction.
3. Conclusion
    a. The GRU+LSTM combination is a balanced solution for time-series prediction problems.
    b. It allows:
        I. Noise reduction in early-stage data (GRU)
        II. Retention of useful sequence memory (LSTM)
        III. Inference from historical component-based patterns.
    c. Results indicate a good trade-off between accuracy and computational complexity.

TRAINING PROCESS, RESULTS AND ANALYSIS

1. Model compiled with the following configuration:
    a. Loss function: Mean Squared Error (MSE), suitable for a regression problem.
    b. Optimizer: Adam (learning rate = 0.001), with an initial learning rate of 0.001. A balanced value between learning speed and stability
    c. Metric: Mean Squared Error, again MSE is used for monitoring during training/validation
2. To prevent overfitting and unnecessary training (EarlyStopping Callback):
    a. Monitor: val_mean_squared_error, check the mean squared error on the validation data.
    b. Patience: 5 epochs, if there is no improvement for 5 consecutive epochs, training stops.
    c. Automatically restores best weights, upon completion, it automatically restores the epoch weights with the best val_mse value.
3. Final Model Performance Metrics
    a. MAE: **8.12 days** → Mean Absolute Error.
    b. MAE (Mean Absolute Error):
        I. *Answers to question*: "On average, by how many days does the model miss the prediction?"
4. For other metrics MSE (Mean Squared Error), RMSE (Root Mean Squared Error) see the detailed presentation

TRAINING PROCESS, RESULTS AND ANALYSIS

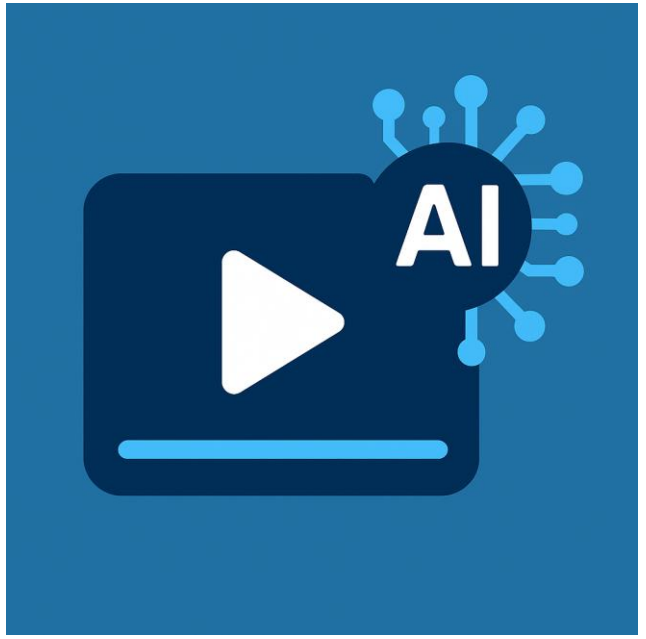




BOQ DATE DELIVERY PREDICTION WEB APPLICATION

- A web application developed with Streamlit for predicting the delivery date of a BoQ using a pre-trained LSTM-GRU neural network.
- Key functionalities:
  - Upload of an Excel file with Item_ID entries to be evaluated
  - Input of two dates: Customer Request Date (CRD) and Order Intake Date (OID)
  - Calculation of IntakeLag and sequence generation for each item
  - Prediction of Lead Time (in days) using the neural model
  - Automatic calculation of the predicted delivery date (excluding weekends)
  - Tabular display of predicted dates with confidence level
  - Informative messages about model performance and options to reset or exit the app

Demo



THANK YOU

- Corrado Vaccaro
- Senior Customer Program Manager, PMP®
- corrado.vaccaro@nokia.com