Journal of Computer Science

Research Paper

Enhancing Fleet Management with GPS Data in Port Terminals: A Deep Learning Architecture for Truck Position Prediction

Mohammed El Karkraoui, and Hicham Attariuas

Department of Emerging Computer Technologies, The Faculty of Sciences, Tetouan, Morocco

Article history
Received: 4 March 2025
Revised: 24 May 2025
Accepted: 20 June 2025

*Corresponding Author:
Mohammed El Karkraoui,
Department of Emerging Computer Technologies, The Faculty of Sciences, Tetouan, Morocco
Email: mohamed.karkraoui@gmail.com

Abstract: Effective truck fleet management in port terminals is crucial for improving operational efficiency and reducing delays. However, GPS-based positioning systems often face limitations in port environments due to signal obstructions and network issues. This study explores the utilization of advanced methods for deep learning, particularly Recurrent Neural Networks (RNNs) for predicting truck positions when GPS signals are unavailable or unreliable. The study utilizes GPS data from an operation terminal, transformed into local coordinate systems, to train and evaluate the models. We propose a novel approach that integrates convolutional layers (CNN) to extract temporal features such as speed and acceleration, followed by RNN-based models to capture sequential dependencies, and finally, a Multi-layer Perceptron (MLP) layers to refine the position estimates. The model that is founded on Gated Recurrent Unit (GRU) performed the best with a minimum Mean Position Error (MPE) of 1.56 meters, outperforming LSTM (Long Short-Term Memory) and Simple RNN models in terms of both reliability and computational efficiency. The results highlight the model's capability to predict future truck positions, offering significant potential to improve operational efficiency in port terminals.

Keywords: Recurrent Neural Networks; Deep Learning; Truck Position Prediction; GPS; Port Terminal Operations; Fleet Management; Convolutional Neural Networks

Introduction

This paper addresses a practical industrial challenge faced by the port authority of Eurogate Tangier, which manages a fleet of trucks operating within the quay zone. In the complex environment of port terminals, two physical phenomena (multipath propagation and non-line-of-sight (NLOS) conditions) frequently deteriorate standard GPS positioning accuracy. These phenomena arise from the presence of numerous machines and structures that obstruct or reflect GPS signals (Fig. 1), leading to significant positioning errors.

To overcome these limitations, the fleet's positioning is maintained using a sophisticated system based on Differential GPS (DGPS). DGPS is a high-accuracy positioning solution commonly used in airports, port terminals, and other critical industrial applications. The system employs two or more GPS receivers, one of which is fixed at a known location with precisely determined coordinates. By calculating the difference between this known true position and the measured position of the fixed receiver, the DGPS system estimates the error affecting the GPS signals. This error estimate is then used to correct the measurements of other GPS devices within the quay zone, ensuring improved accuracy in the position information provided to the port authority.

Each truck in the fleet is equipped with an embedded GPS device running on Windows CE. Under normal operating conditions, these devices transmit real-time positioning data via a cellular network (UMTS) every

second, typically achieving an accuracy of approximately 1 meter. This level of precision supports effective fleet management and seamless synchronization with other equipment in the quay area. However, due to occasional network disruptions or software issues, the system sometimes fails to deliver timely or accurate position data. This results in scenarios where trucks may temporarily disappear from the tracking screen or appear in illogical positions (e.g., a truck seemingly located at sea or inside a container), which disrupts the real-time management of the fleet.

A previous team developed a web-based visualization application for tracking trucks within the quay zone using the DGPS system as its foundational layer. When DGPS encounters an issue, the application displays the raw GPS position. To address the limitations of this system, particularly when it relies solely on GPS data, we propose a machine learning framework to predict the true next position based on previous positions. By forecasting the subsequent location, the visualization system can display the predicted position when the raw data appears illogical or is missing entirely, thereby smoothing the tracking process and enhancing operational reliability.

(a)

(b)

Fig. 1. Examples of Constructions within Port Terminals, (a): Gantry Crane; (b): Straddle Carriers

This study was conducted using data collected from the truck fleet of the EuroGate container terminal at Tanger Med Port. The EuroGate terminal, a key hub in the Mediterranean, presents unique challenges for vehicle positioning due to its dense infrastructure and high volume of container traffic. These conditions often cause significant signal obstructions that impact the accuracy of traditional GPS-based positioning systems. The data collected from this environment provides a robust testbed for evaluating the utility of advanced machine learning models in real-world industrial settings.

We propose a novel hybrid algorithm that combines Recurrent Neural Networks (RNNs), Convolutional Neural Networks (CNNs), and Multi-Layer Perceptrons (MLPs) for truck position prediction. The algorithm begins with data acquisition, where GPS positional data is collected from the truck fleet operating within the port terminal. This raw data reflects the real-world conditions of truck movements in a dense and dynamic environment. The collected GPS data is then transformed from the WGS-84 global reference system into a local reference frame aligned with the terminal's operational layout. This transformation process involves converting geodetic coordinates into Cartesian coordinates using the Earth-Centered Earth-Fixed (ECEF) system and subsequently projecting them into the local frame.

Once transformed, the data is segmented into temporal sequences representing the historical movements of trucks. These sequences are processed through three stages: CNNs extract spatial-temporal features such as velocity and acceleration, RNNs model temporal dependencies across time steps, and MLPs refine the final position predictions. This work provides a comparative analysis of three RNN-based architectures: Gated Recurrent Unit (GRU), Simple RNN, and Long Short-Term Memory (LSTM). Our objective is to identify the most efficient model by evaluating both predictive accuracy and computational efficiency.

This manuscript is ordered like this: in Materials and Methods we describe how GPS data was collected and transformed into a local reference frame, the model architecture along with the steps taken to train the models. We will compare the performance of the proposed models and discussing their strengths and weaknesses in Results and Discussion. Eventually, the concluding section provides the conclusion and outlines directions for coming research.

Literature Review

Researchers have long valued the ability of ML (machine learning) and DL (deep learning) to increase productivity in port terminals. Burger et al. (2022) and Carvalho et al. (2019), for example, showed how ML methods with the application of tools like Support Vector

Machines and neural networks can analyse port machinery utilizing sensor data to anticipate failures in advance then effectively minimizing downtime.

Work by Izquierdo et al. (2017) and Xiao et al. (2018) has shown that by examining historical data, ML models such as SVMs and ANNs can accurately forecast future positions even in challenging conditions like urban areas or indoors. Similarly, when it comes to forecasting trajectories, early studies by Liu and Shoji (2020) and Liu et al. (2016) revealed that recurrent neural networks (RNNs) excel at capturing temporal patterns despite issues like vanishing gradients. Building on this, later research by Violos et al. (2020), Tian et al. (2019), and Yang et al. (2019), Altché and de La Fortelle (2017) has confirmed that LSTMs, with their ability to remember long-term dependencies, are especially well-suited for predicting complex, long-term trajectories.

Deep learning also plays a key role in optimizing operations within port terminals. According to Ferretti et al. (2022), container movement forecasting can be enhanced by models that combine CNNs and LSTMs, stacking strategies, and even crane scheduling. At the same time, Noh et al. (2020) and Bouktif et al. (2020) have integrated DL into supply chain management, leading to better planning and inventory control.

Traffic management is another area where ML is effective. Studies by Kim and Lee (2018) and Li et al. (2023) used time series analysis and clustering to forecast traffic patterns and optimize flow, which in turn helps allocate resources more efficiently and reduce congestion. More recent work has introduced GRUs as a simpler, faster alternative to LSTMs. Wang et al. (2020), Han et al. (2019), and Suo et al. (2020) suggests that GRUs, that merge certain gating mechanisms, can achieve similar results in real-time trajectory forecasting, such as in autonomous driving or maritime navigation.

In addition, several groups have explored hybrid models that join RNNs (including LSTMs and GRUs) with CNNs to catch both spatial and temporal dependencies. For example, Jiang et al. (2019), Ma and Tian (2020), Xie et al. (2020), Wang et al. (2024), and Lu et al. (2021) developed such models to improve performance in video prediction and multi-step forecasting tasks.

Attention mechanisms have become popular because it helps models center on the most important parts of an input sequence. Lin et al. (2022), Yan et al. (2020), Casabianca et al. (2021), Yu et al. (2021), and Nawaz et al. (2007) have all shown that this approach can lead to better predictions for complex tasks like forecasting aircraft paths or pedestrian movements.

Transfer learning, has been applied successfully, with Akabane and Kato (2021) and Šerić et al. (2021) demonstrating that pre-trained models can improve performance when data is rare in fields like robotic navigation and sports analytics. Finally, reinforcement learning has been used to optimize trajectory forecasting over time, as evidenced by the job of Huang et al. (2022) and Yang et al. (2024).

Materials and Methods

One Dimension Convolutional Neural Networks

CNNs (Convolutional Neural Networks) belong to a class of DL models largely employed for analyzing structured grid-like data like images. In image processing, by applying convolutional filters to particular regions on the image, CNNs are able to automatically learn hierarchical features, capturing spatial patterns such as edges, textures, and object parts. Traditionally, CNNs operate in two dimensions (2D) to handle data like images, where both height and width are spatial dimensions. However, CNNs can also be adapted for 1D problems, where the data is sequential, such as time series or position data.

In our case, the 1D CNN is applied to sequential input data of truck positions (x, y) at different time steps ("Fig. 2"), to predict the future positions x_{n+1} and y_{n+1}. The input consists of a time series sequence $\{(x_0,y_0)$, (x_1,y_1) , … , $(x_n,y_n)\}$, where each position corresponds to the truck's coordinates at a specific time step. The CNN processes each dimension x and y separately through two or more convolutional layers to capture temporal dependencies and spatial patterns.

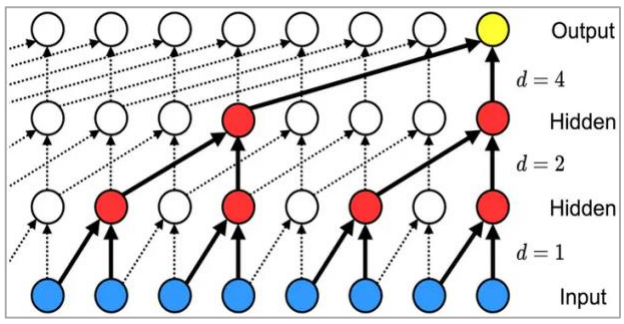

Fig. 2. 1D Convolutional Neural Network Architecture

For the x-dimension, the first convolutional layer applies a kernel of size K1 and stride d1 to the sequence $\{x_0$, x_1 , … , $x_n\}$, and the convolution operation is described as:

$$z_t^x = \sum_{k=0}^{K1} \omega_k^x . x_{t-k.d1} \quad (1)$$

where z_t^x is the result of the first convolutional layer for the x-dimension at once time step (t), $x_{t-k.d1}$ stand for the x-coordinate at the foregoing time step, and ω_k^x are filter weights of the first convolutional kernel. The result is then passed through a RELU (Rectified Linear Unit) activation function:

$$a_t^x = ReLU(z_t^x) \quad (2)$$

Similarly, for the y-dimension, the first convolutional layer applies the same operation to the sequence $\{y_0, y_1, \ldots, y_n\}$:

$$z_t^y = \sum_{k=0}^{K1} \omega_k^y . y_{t-k.d1} \quad (3)$$

After this convolution, the output for the y-dimension is passed through a ReLU activation function:

$$a_t^y = ReLU(z_t^y) \quad (4)$$

The second convolutional layer for each dimension applies another kernel with size K2 and stride d2. For the x-dimension, the second layer operates on the output of the first layer, a_t^x, as follows:

$$z_t^{x(2)} = \sum_{k=0}^{K2} \omega_k^{x(2)} . a_{t-k.d2}^x \quad (5)$$

Similarly, for the y-dimension, the second layer applies:

$$z_t^{y(2)} = \sum_{k=0}^{K2} \omega_k^{y(2)} . a_{t-k.d2}^y \quad (6)$$

After the second convolution for both x- and y-dimensions, the results are passed through another ReLU activation function:

$$a_t^{x(2)} = ReLU(z_t^{x(2)}) \quad (7)$$

$$a_t^{y(2)} = ReLU(z_t^{y(2)}) \quad (8)$$

At this point, the output from both dimensions, $a_t^{x(2)}$ and $a_t^{y(2)}$, represents the learned features from the truck's positions over time. These features are then fed into the other predictive model to forecast the future positions x_{n+1} and y_{n+1}.

Recurrent Neural Networks

An RNN is made to deal with sequential data, such time-series or text, by incorporating memory of previous inputs into their computations. Unlike traditional feedforward networks, RNNs process input sequences step by step, with each step influenced by prior information, enabling them to model temporal dependencies effectively. This enables RNNs well-suited for prediction functions where understanding the progression of past data is critical, such as forecasting future values in time-series datasets or predicting trajectories in dynamic environments. LSTM and GRU are two versions of RNN that are more sophisticated, designed to improve the prediction capabilities of RNNs in both short-term and long-term contexts, which is essential in our case of trajectory prediction.

Data Acquisition

The study utilizes National Marine Electronics Association (NMEA) files collected from a fleet of 31 trucks operating within a port terminal. The NMEA standard makes it easier to obtain positioning and navigational data and is mostly utilized in marine contexts from various electronic devices. In this application, each truck is equipped with a GPS receiver transmitting data in NMEA format. This data comprises essential navigational sentences, such as Global Positioning System Fix Data (GGA), which provides latitude, longitude, and altitude, and The Recommended Minimum Specific GNSS Data (RMC) message adds data like timestamp, position, speed over ground, and course over ground, both crucial for the subsequent geospatial transformations. Fig. 3 illustrates the processing pipeline designed to transform GPS data into local coordinate systems.

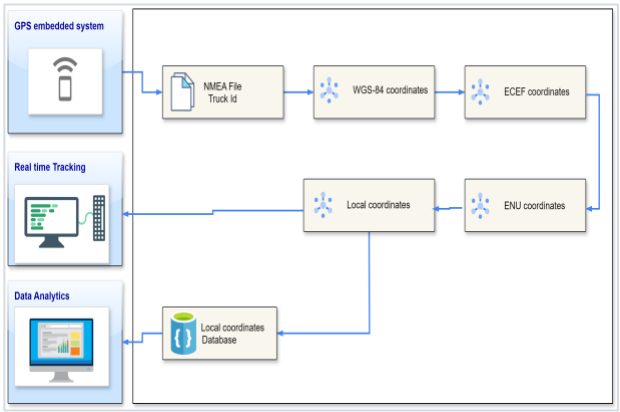

Fig. 3. Data Transformation from GPS to Local Coordinate Systems

The GPS system of our maritime operator adopts a custom NMEA file format (Fig. 4) to track the positions and movements of trucks. This custom format includes fields such as "ip"," Latitude", "Longitude", "Altitude", "PDOP", "VDOP", "HDOP", "SatCount", "Speed", "datetimeCE", and "datetimebdd", The data fields in this custom NMEA format are designed to provide

comprehensive geospatial and satellite quality information, essential for accurate tracking and analysis.

While the format is custom, the system still relies on standard NMEA sentences for processing GPS data, specifically the GGA and RMC messages.

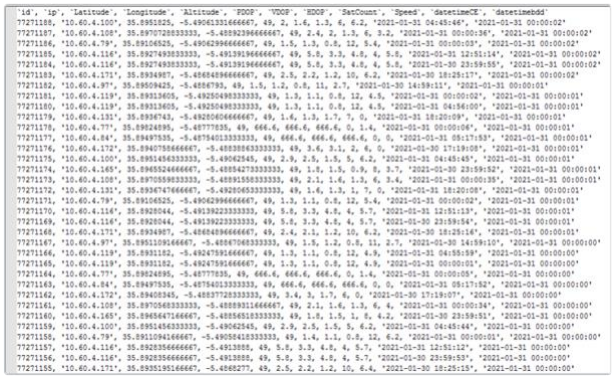

Fig. 4. Example of Costume NMEA File from the Dataset

Below are the fields included in the custom NMEA file:

Ip: Represents the unique truck identifier.
Latitude: The geographical coordinate that defines the north-south position on the Earth's surface, measured in degrees. Ranges from -90° to +90°.
Longitude: The geographical coordinate that indicates the east-west position on the Earth's surface, measured in degrees. Ranges from -180° to +180°.
Altitude: The height above a reference point, typically above sea level, measured in meters.
(PDOP) Position Dilution of Precision: A metric of precision of GPS position in 3D (latitude, longitude, and altitude). Lower PDOP value indicates higher accuracy.
(VDOP) Vertical Dilution of Precision: A indicator of the precision of the GPS position specifically in the vertical (altitude) direction. Like PDOP, a lower value indicates better accuracy.
(HDOP) Horizontal Dilution of Precision: A measure of the precision of the GPS position in the horizontal (latitude and longitude) directions. Again, a lower value indicates better accuracy.
SatCount: The number of satellites being used by the GPS device to calculate the position. More satellites generally lead to higher accuracy.
Speed: The rate at which an object is shifting, generally stated in meters per second (m/s) or kilometers per hour (km/h) in our case.
datetimeCE (date-time Current Epoch): Time is given in UTC (Universal Time Coordinated), typically as "hhmmss" (hours, minutes, seconds). This represents the time when the GPS data was captured.
datetimebdd: A timestamp representing the local time and date at the moment the GPS data was captured.

Data Transformation

Once the geodetic WGS-84 coordinates are extracted from the NMEA files, they are converted to Earth-Centered, ECEF (Earth-Fixed) coordinates (X, Y, Z) (Fig. 5) with the following formulas:

$$X = (N(\phi) + h).cos(\phi).cos(\lambda) \quad (9)$$

$$Y = (N(\phi) + h).cos(\phi).sin(\lambda) \quad (10)$$

$$Z = (\frac{a^2}{b^2}N(\phi) + h).sin(\phi) \quad (11)$$

where N(ϕ) is the prime vertical radius of curvature, calculated as:

$$N(\phi) = \frac{a^2}{\sqrt{a^2.\cos^2(\phi)+b^2.\sin^2(\phi)}} \quad (12)$$

and a = 6378137.0 m and b = 6356752.3142 m represent the WGS-84 ellipsoid constants.

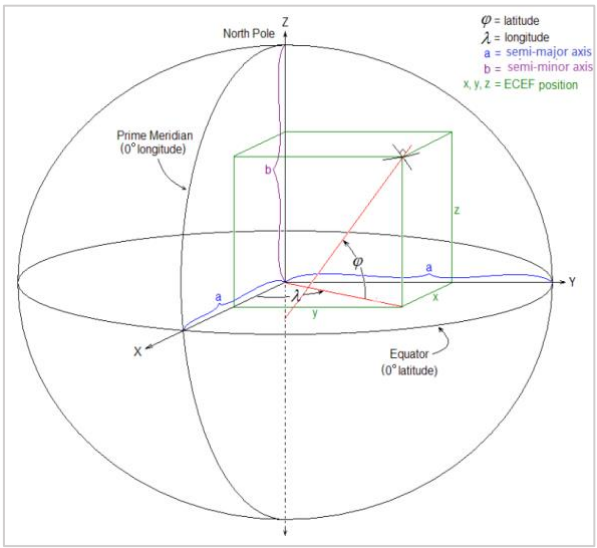

Fig. 5. Geodetic (WGS-84) and ECEF Referentials

To localize the ECEF coordinates to the specific context of the port terminal, a transformation matrix T is utilized at the predefined local origin (ϕ0, λ0), expressed as:

$$T = \begin{bmatrix} -\sin(\lambda 0) & \cos(\lambda 0) & 0 \\ -\sin(\phi 0).\cos(\lambda 0) & -\sin(\phi 0)\sin(\lambda 0) & \cos(\phi 0) \\ \cos(\phi 0).\cos(\lambda 0) & \cos(\phi 0).\sin(\lambda 0) & \sin(\phi 0) \end{bmatrix} \quad (13)$$

Applying this matrix to the global ECEF

coordinates relative to the local origin yields the East, North, Up (ENU) coordinates:

$$\begin{bmatrix} E \\ N \\ U \end{bmatrix} = T. \begin{bmatrix} X - X0 \\ Y - Y0 \\ Z - Z0 \end{bmatrix} \quad (14)$$

Transform the ENU coordinates to the locale referential XYZ which is rotated by an angle α = -46.23 degree, this referential is normal to the port terminal structure ("Fig. 6"), by using rotation equation:

$$\begin{cases} X = E\cos\alpha - N\sin\alpha \\ Y = E\sin\alpha + N\cos\alpha \\ Z = E \end{cases} \quad (15)$$

Fig. 6. Shema of Local Referential Within the port Terminal

Considering that the value of α has an error margin smaller than 0.01 degree and given that E and N are both smaller than 1000 m, the resulting error in the X and Y coordinates is less than 0.175 m. This level of error is insignificant in our context.

For compose a dataset ready to training for the RNN based Models we added some features for each data point as following ("Table 1"):

Time: Timestamp in seconds.

Truck ID: Unique identifier for each truck.

Position (X, Y): Represents the truck's location within the terminal in meters relative to the local reference frame, the Z coordinate is not included for this data because is a fixed value.

Table 1. Example from the time series form.

Time (s)	Truck ID	X (m)	Y (m)
17146	10.60.4.79	459.013811	156.642092
17146	10.60.4.112	439.975594	42.728514
17146	10.60.4.108	97.3924317	91.656369
17147	10.60.4.15	97.356268	91.514155
17147	10.60.4.58	441.059455	43.558941
17147	10.60.4.42	451.328317	156.845233

Data Size: The dataset comprised a total of 56,760 observations.

Sequence Creation: We created sequences of historical positions for each truck. Each sequence contained the latest 10 positions data points (among others the present position), serving as the input for the prediction model.

It is important to note that the dataset used in this study spans approximately one hour of fleet activity within the EuroGate terminal. Over such a limited period, the port infrastructure is reasonably assumed to be partially stationary, which aligns with typical operations in container terminals. Therefore, our model is designed for scenarios where the layout remains mostly unchanged during operation. Nonetheless, the model may still accommodate small or localized changes in the port environment. A more generalizable solution would require a larger dataset ideally covering several days or weeks which was not available at the time of this research. We identify this as a promising direction for future extensions.

Model Architecture and Training

Model Architecture

The aim of the proposed model is to predict the next position (X_{t+10}, Y_{t+10}) of a moving truck based on a sequence of past positions $(X_{t+i}, Y_{t+i})_{0 \leq i < 10}$. To achieve this, we propose three different architectures based on Simple RNN (SRNN), Long Short-Term Memory (LSTM), and Gated Recurrent Unit (GRU) (Fig. 7). Each architecture integrates feature extraction, sequential modeling, and fully connected layers to capture both temporal dependencies and spatial relationships in the data. The key components of the architectures are as follows:

Input Layer: The input to the model is a series of 2D positional coordinates (x, y), provided as a time-series tensor with shape (10, 2), representing 10 consecutive position measurements.

Convolutional Layers (Feature Extraction): Two 1D convolutional layers are employed to extract temporal features such as velocity (differences between consecutive positions), acceleration (rate of change of velocity), and other spatial-temporal dynamics from the sequential positional data.

- The first convolutional layer contains 10 filters with a kernel size of 3 and operates without a nonlinear

activation function to preserve the numerical relationships in the position data.

- The second convolutional layer contains 10 filters with a kernel size of 2 and uses the ReLU (Rectified Linear Unit) activation function to introduce nonlinearity.

The extracted features are then passed to the subsequent layers for temporal modeling.

Temporal Modeling Layers: Three variants of recurrent layers are proposed to model sequential dependencies in the data:

- **LSTM-based Model:** An LSTM layer with 128 units captures long-term dependencies in the sequence. The ReLU activation function is used to improve gradient flow and accelerate convergence.
- **GRU-based Model:** A GRU layer with 128 units provides lightweight temporal modeling with computational efficiency. The ReLU activation function is applied.
- **Simple RNN-based Model:** A Simple RNN layer with 128 units performs basic sequential modeling. The ReLU activation function is employed.

Regularization Layer: A dropout layer with a rate of 0.2 is applied after each recurrent layer to reduce overfitting by randomly deactivating 20% of the neurons during training.

Multi-Layer Perceptron (MLP): After the temporal layers, two fully connected layers with 128 units each are added to integrate the learned features. A final dense layer with 2 output units provides the predicted 2D coordinates (Xt+10, Yt+10).

The complete architecture follows a pipeline structure: Input → 1D CNN (feature extraction) → RNN variant (temporal modeling) → Dropout (regularization) → MLP (prediction refinement) → Output (predicted coordinates). This hybrid approach leverages the strengths of convolutional layers for local feature extraction and recurrent layers for capturing temporal dependencies, resulting in robust position prediction for truck trajectories in the port environment.

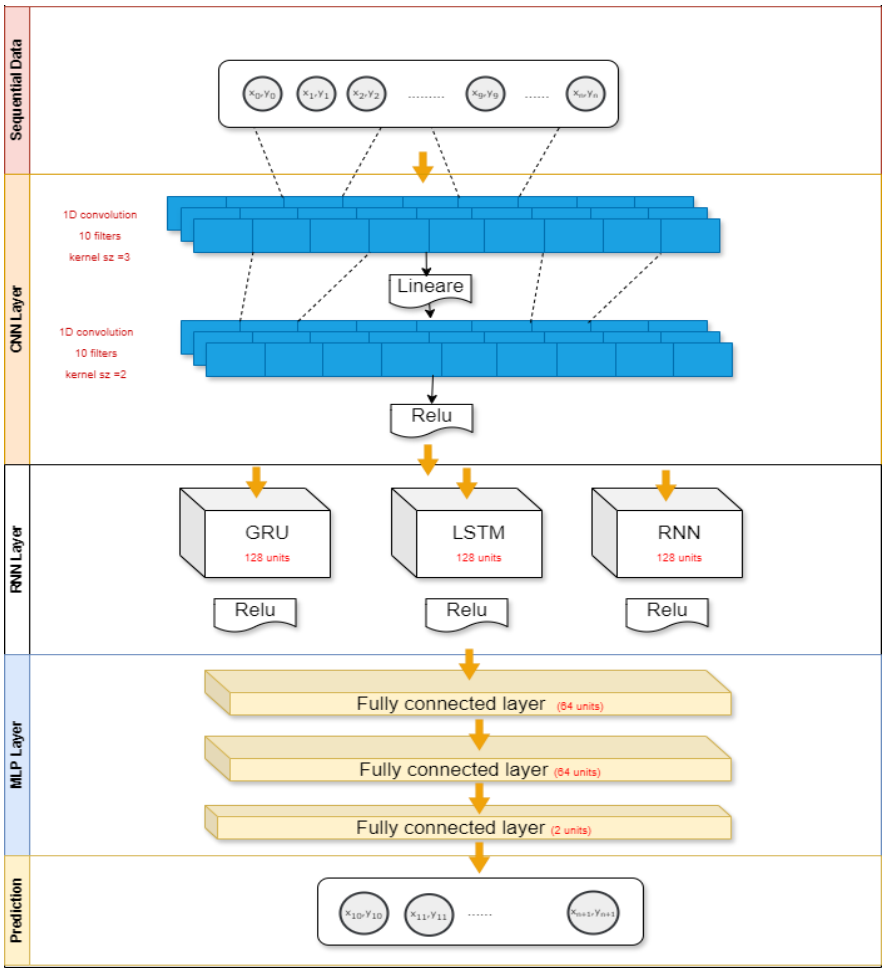

Fig. 7. Adopted Deep Learning Architecture

Model Training

Data Split: In order to train and assess the model, we separated the dataset into 80% training and 20% validation sets.

We trained all models using the following parameters:

Optimizer: The Adamax (Kingma et Ba, 2017)

optimizer, for its stability in handling sparse gradients, and varying feature scales. Its adaptive learning rate ensures smooth convergence and improved generalization.

Loss Function: Euclidean Error (EE)

Metrics: Mean Position Error (MPE)

The training process was executed in 300 epochs with a batch size of 1000, ensuring that the model had ample opportunity to learn from the data. In addition, we used random search to systematically explore various hyperparameter settings including adjustments in learning rate, batch size, and network architecture to identify the configuration which produced the best results on the validation set. This approach allowed us to optimize convergence behavior and enhance overall model performance.

Results and Discussion

Evaluation metrics

The following metrics were used to evaluate models on the validation dataset:

Euclidean Error (EE): The Euclidean Error calculates the cumulative Euclidean distance error among predicted and true coordinates:

$$EE = \sum_{i=1}^{n} \sqrt{(Xi\_pred\ -\ Xi\_)^2 + (Yi\_pred\ -\ Yi\_)^2} \quad (16)$$

Mean Position Error (MPE): Represents the mean Euclidean distance among predicted and actual positions:

$$MPE = \frac{\sum_{i=1}^{n} \sqrt{(Xi\_pred - Xi\_)^2 + (Yi\_pred - Yi\_)^2}}{n} \quad (17)$$

where:

n: Total number of data points in the dataset
Xi_pred : Predicted x-coordinate at sequence i
Yi_pred: Predicted y-coordinate at sequence i
Xi_: True x-coordinate at sequence i
Yi_: True y-coordinate at sequence i

Results

This section presents the predictive capability analysis of three proposed architectures: Conv-LSTM, Conv-RNN, and Conv-GRU.

Fig. 8 presents the results for the LSTM-based model, which demonstrates good generalization with consistent training and validation Mean Position Error (MPE). The minimum validation MPE of 1.79 m indicates a high level of accuracy, though it is slightly outperformed by the GRU and RNN models in this case. The LSTM's robustness to fluctuations and smooth convergence suggest that it effectively handles long-term dependencies in the positional data.

The LSTM model is especially useful for jobs with complex temporal relationships. However, in this specific dataset, the additional computational complexity of the LSTM architecture may not provide a significant advantage over simpler models like GRUs or RNNs as demonstrated in the following sections.

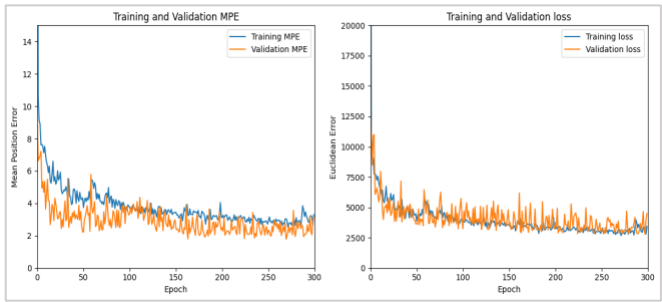

Fig. 8. MPE and Loss curve for the Conv-LSTM model.

Fig. 9 presents the results for the RNN-based model, which achieves a minimum MPE of 1.68 m, which is better than the LSTM model but not as low as the GRU model. The training and validation curves show more fluctuations compared to the LSTM and GRU models, indicating less stability during training.

While the RNN captures temporal dependencies adequately for this dataset, it may struggle with stability and handling long-term dependencies. The slightly better MPE compared to the LSTM could be due to its simplicity, which avoids overfitting in this particular task.

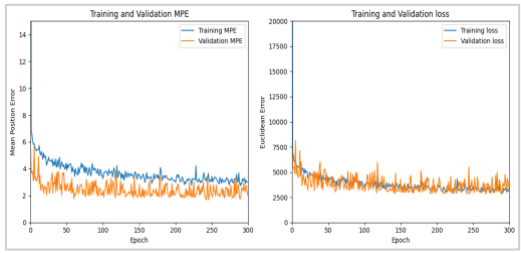

Fig. 9. MPE and Loss curve for the Conv-RNN model.

Fig. 10 presents the results for the GRU-based model, which achieves the lowest minimum MPE of 1.56 m, outperforming both the LSTM and RNN models. The GRU's training and validation curves are smooth and aligned, showing good generalization and stability.

The GRU's capacity to model temporal relationships and computing efficiency are balanced making it the best-performing model for this dataset. Its architecture appears well-suited for the task, efficiently capturing both short-term and medium-term dependencies.

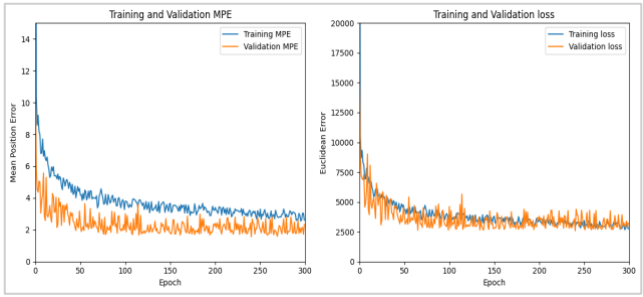

Fig. 10. MPE and Loss curve for the Conv-GRU model.

In summary, the Conv-GRU model achieves the best performance with a minimum MPE of 1.56 m, outperforming both the Conv-LSTM and Conv-RNN models. “Table 2” summarizes the accuracy metrics of the proposed RNN-based models, comparing their accuracy, convergence speed, stability, overfitting tendencies, and generalization capabilities.

Table 2. Comparison of Performance of RNN-Based Models.

Model Based	MPE (m)	Convergence Speed	Stability	Generalization
LSTM	1.79	Moderate	Low	Good
RNN	1.68	Fast	Moderate	Good
GRU	1.56	Fast	Good	Good

Discussion

This study compared three CNN-RNN based architectures: Conv-RNN, Conv-LSTM, and Conv-GRU for predicting trucks localization within port terminal operations. The complex environment of the EuroGate terminal at Tanger Med Port, characterized by dense infrastructure and frequent signal obstructions, presented a challenging testbed for evaluating the effectiveness of machine learning models in this task.

The experimental results indicate that Conv-GRU model achieved the lowest MPE of 1.56 m, outperforming both the Conv-LSTM and Conv-RNN models. Conv-GRU model demonstrated a balance between computational efficiency and capturing both short-term and medium-term relationship. Its lower complexity compared to Conv-LSTM allowed for faster training while maintaining predictive accuracy. Conv-LSTM model, although effective at capturing long-term dependencies, did not provide a significant improvement in accuracy for this dataset, likely due to the added computational overhead. The Simple Conv-RNN model showed moderate performance, but its lack of specialized mechanisms for handling long-term dependencies limited its accuracy compared to the Conv-GRU and Conv-LSTM.

The inclusion of a convolutional layer as the first layer carries an important role in extracting temporal features like velocity and acceleration from the position data. This preprocessing enhances the performance of all three models by simplifying the temporal patterns they need to learn. The results of this investigation highlight the significance of selecting appropriate machine learning architectures based on the operational context. In environments with frequent disruptions and a need for efficient real-time processing, Conv-GRU models provide a compelling solution.

Conclusion

In the present study, we present a full process for enhancing the GPS positioning system used in a truck fleet of a terminal port. The process begins by pre-processing NMEA files to obtain geodetic data, which is further transformed into a local coordinate system that is more effective for subsequent analytics and visualizations. We introduce a novel deep learning model that fuses convolutional neural networks (CNN) to extract spatial features with three types of recurrent neural networks (RNN) simple RNN, GRU, and LSTM to predict truck position.

All three models demonstrate promising performance, with the GRU-based model showing particularly superior accuracy and computational efficiency. The Process results demonstrate that the proposed Conv-GRU model has the highest predictive accuracy, with a minimum Mean Position Error of 1.56 m, outperforming both the Conv-LSTM (1.79 m) and Conv-RNN (1.68m) models.

While the current model is trained to predict the immediate next position, it can be used recursively for short-term multi-step forecasting. However, the performance may degrade over longer horizons due to error accumulation. Future work will involve training sequence-to-sequence models capable of directly predicting multiple future positions in a single forward pass, which would improve robustness for real-time fleet tracking.

Additional future directions include the integration of attention mechanisms to better capture salient patterns in sequential data, as well as the real-time deployment of the system in live operational environments. If a larger dataset becomes available, we also aim to train a more powerful model capable of adapting to dynamic port infrastructures. Furthermore, fusing GPS data with other sensor data, like inertial or visual inputs, represents a promising approach for enhancing positioning accuracy in complex and signal-challenged environments.

Author Contributions

M. El Karkraoui: Conceptualization, Data Curation, Formal Analysis, Funding Acquisition, Methodology, Software, Validation, Visualization, Writing – Original Draft, Writing – Review & Editing.

H. Attariuas: Conceptualization, Data Curation, Formal Analysis, Funding Acquisition, Investigation, Methodology, Project Administration, Resources, Writing – Review & Editing.

Funding Information

Thereby study received no external funding.

Acknowledgments

The authors ought to consider to acknowledge the EuroGate Tanger team for providing the dataset used in this study and for their technical support throughout the manuscript framework.

Conflicts of Interest

The authors declare no conflicts of interest.

Ethics

The authors affirm that the manuscript is an original work and hasn't been previously published elsewhere.

References

Burger, M., Boer, C. A., Straub, E., & Saanen, Y. A. (2022). Predictive maintenance powered by machine learning and simulation. 2022 Winter Simulation Conference (WSC), 2807 - 2818. https://doi.org/10.1109/WSC57314.2022.10015520

Carvalho, T. P., Soares, F. A. A. M. N., Vita, R., Francisco, R. da P., Basto, J. P., & Alcalá, S. G. S. (2019). A systematic literature review of machine learning methods applied to predictive maintenance. Computers & Industrial Engineering, 137, 106024. https://doi.org/10.1016/j.cie.2019.106024

Izquierdo, R., Parra, I., Muñoz-Bulnes, J., Fernández-Llorca, D., & Sotelo, M. A. (2017). Vehicle trajectory and lane change prediction using ANN and SVM classifiers. 2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC), 1-6. https://doi.org/10.1109/ITSC.2017.8317838

Xiao, Z., Li, P., Havyarimana, V., Hassana, G. M., Wang, D., & Li, K. (2018). Goi : A novel design for vehicle positioning and trajectory prediction under urban environments. IEEE Sensors Journal, 18(13), 5586-5594. https://doi.org/10.1109/JSEN.2018.2826000

Liu, W., & Shoji, Y. (2020). Deepvm : Rnn-based vehicle mobility prediction to support intelligent vehicle applications. IEEE Transactions on Industrial Informatics, 16(6), 3997 - 4006. https://doi.org/10.1109/TII.2019.2936507

Liu, Q., Wu, S., Wang, L., & Tan, T. (2016). Predicting the next location: A recurrent model with spatial and temporal contexts. Proceedings of the AAAI Conference on Artificial Intelligence, 30(1). https://doi.org/10.1609/aaai.v30i1.9971

Violos, J., Tsanakas, S., Androutsopoulou, M., Palaiokrassas, G., & Varvarigou, T. (2020). Next position prediction using lstm neural networks. 11th Hellenic Conference on Artificial Intelligence, 232-240. https://doi.org/10.1145/3411408.3411426

Altché, F., & de La Fortelle, A. (2017). An LSTM network for highway trajectory prediction. 2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC), 353 - 359. https://doi.org/10.1109/ITSC.2017.8317913

Tian, S., Zhang, X., Zhang, Y., Cao, Z., & Cao, W. (2019). Spatio-temporal position prediction model for mobile users based on lstm. 2019 IEEE 25th International Conference on Parallel and Distributed Systems (ICPADS), 967 - 970. https://doi.org/10.1109/ICPADS47876.2019.00146

Yang, K., Bi, M., Liu, Y., & Zhang, Y. (2019). Lstm-based deep learning model for civil aircraft position and attitude prediction approach. 2019 Chinese Control Conference (CCC), 8689 - 8694. https://doi.org/10.23919/ChiCC.2019.8865874

Ferretti, M., Fiore, U., Perla, F., Risitano, M., & Scognamiglio, S. (2022). Deep learning forecasting for supporting terminal operators in port business development. Future Internet, 14(8), 221. https://doi.org/10.3390/fi14080221

Noh, J., Park, H.-J., Kim, J. S., & Hwang, S.-J. (2020). Gated recurrent unit with genetic algorithm for product demand forecasting in supply chain management. Mathematics, 8(4), 565. https://doi.org/10.3390/math8040565

Bouktif, S., Fiaz, A., Ouni, A., & Serhani, M. A. (2020). Multi-sequence lstm-rnn deep learning and metaheuristics for electric load forecasting. Energies, 13(2), 391. https://doi.org/10.3390/en13020391

Kim, K.-I., & Lee, K. M. (2018). Deep learning-based caution area traffic prediction with automatic identification system sensor data. Sensors, 18(9), 3172. https://doi.org/10.3390/s18093172

Li, Y., Li, Z., Mei, Q., Wang, P., Hu, W., Wang, Z., Xie, W., Yang, Y., & Chen, Y. (2023). Research on multi-port ship traffic prediction method based on spatiotemporal graph neural networks. Journal of Marine Science and Engineering, 11(7), 1379. https://doi.org/10.3390/jmse11071379

Wang, S., Zhao, J., Shao, C., Dong, C., & Yin, C. (2020). Truck traffic flow prediction based on lstm and gru methods with sampled gps data. IEEE Access, 8, 208158 - 208169. https://doi.org/10.1109/ACCESS.2020.3038788

Han, P., Wang, W., Shi, Q., & Yang, J. (2019). Real-time short- term trajectory prediction based on gru neural network. 2019 IEEE/AIAA 38th Digital Avionics Systems Conference (DASC), 1 - 8. https://doi.org/10.1109/DASC43569.2019.9081618

Suo, Y., Chen, W., Claramunt, C., & Yang, S. (2020). A ship trajectory prediction framework based on a recurrent neural network. Sensors, 20(18), 5133. https://doi.org/10.3390/s20185133

Jiang, H., Chang, L., Li, Q., & Chen, D. (2019). Trajectory prediction of vehicles based on deep learning. 2019 4th International Conference on Intelligent Transportation Engineering (ICITE), 190 -195. https://doi.org/10.1109/ICITE.2019.8880168

Ma, L., & Tian, S. (2020). A hybrid cnn-lstm model for aircraft 4d trajectory prediction. IEEE Access, 8, 134668 - 134680. https://doi.org/10.1109/ACCESS.2020.3010963

Xie, G., Shangguan, A., Fei, R., Ji, W., Ma, W., & Hei, X. (2020). Motion trajectory prediction based on a CNN-LSTM sequential model. Science China Information Sciences, 63(11), 212207. https://doi.org/10.1007/s11432-019-2761-y

Wang, J., Liu, K., & Gong, Y. (2024). Vehicle position prediction using particle filtering based on 3d cnn-lstm model. IEEE Transactions on Mobile Computing, 23(4), 2992 - 3004. https://doi.org/10.1109/TMC.2023.3267853

Lu, B., Lin, R., & Zou, H. (2021). A novel cnn-lstm method for ship trajectory prediction. 2021 IEEE 23rd Int Conf on High Performance Computing & Communications; 7th Int Conf on Data Science & Systems; 19th Int Conf on Smart City; 7th Int Conf on Dependability in Sensor, Cloud & Big Data Systems & Application (HPCC/DSS/SmartCity/DependSys), 2431 - 2436. https://doi.org/10.1109/HPCC-DSS-SmartCity-DependSys53884.2021.00366

Lin, L., Li, W., Bi, H., & Qin, L. (2022). Vehicle trajectory prediction using lstms with spatial–temporal attention mechanisms. IEEE Intelligent Transportation Systems Magazine, 14(2), 197 - 208. https://doi.org/10.1109/MITS.2021.3049404

Yan, J., Peng, Z., Yin, H., Wang, J., Wang, X., Shen, Y., Stechele, W., & Cremers, D. (2020). Trajectory prediction for intelligent vehicles using spatial - attention mechanism. IET Intelligent Transport Systems, 14(13), 1855 - 1863. https://doi.org/10.1049/iet-its.2020.0274

Casabianca, P., Zhang, Y., Martínez-García, M., & Wan, J. (2021). Vehicle destination prediction using bidirectional lstm with attention mechanism. Sensors, 21(24), 8443. https://doi.org/10.3390/s21248443

Yu, J., Zhou, M., Wang, X., Pu, G., Cheng, C., & Chen, B. (2021). A dynamic and static context-aware attention network for trajectory prediction. ISPRS International Journal of Geo-Information, 10(5), 336. https://doi.org/10.3390/ijgi10050336

Nawaz, A., Huang, Z., Wang, S., Akbar, A., AlSalman, H., & Gumaei, A. (2020). Gps trajectory completion using end-to-end bidirectional convolutional recurrent encoder-decoder architecture with attention mechanism. Sensors, 20(18), 5143. https://doi.org/10.3390/s20185143

Akabane, R., & Kato, Y. (2021). Pedestrian trajectory prediction based on transfer learning for human-following mobile robots. IEEE Access, 9, 126172 - 126185. https://doi.org/10.1109/ACCESS.2021.3111917

Šerić, L., Pinjušić, T., Topić, K., & Blažević, T. (2021). Lost person search area prediction based on regression and transfer learning models. ISPRS International Journal of Geo-Information, 10(2), 80. https://doi.org/10.3390/ijgi10020080

Huang, Y., Du, J., Yang, Z., Zhou, Z., Zhang, L., & Chen, H. (2022). A survey on trajectory-prediction methods for autonomous driving. IEEE Transactions on Intelligent Vehicles, 7(3), 652 - 674. https://doi.org/10.1109/TIV.2022.3167103

Yang, Z., Wu, Z., Wang, Y., & Wu, H. (2024). Deep reinforcement learning lane-changing decision algorithm for intelligent vehicles combining lstm trajectory prediction. World Electric Vehicle Journal, 15(4), 173. https://doi.org/10.3390/wevj15040173

Kingma, D. P., & Ba, J. (2017). Adam : A method for stochastic optimization (arXiv:1412.6980). arXiv. https://doi.org/10.48550/arXiv.1412.6980