Neural Networks (ECE 553)
Lucas Yeykelis
Professor Mingzhe Chen
University of Miami
November 2025

Project 1
Vehicle Positioning

Abstract

This project implements machine learning (ML) and neural network methods for vehicle positioning in an underground parking lot using Channel State Information (CSI) data. CSI data provides information about wireless signal propagation between transmitters and receivers. Using CSI to determine vehicle positioning is useful, one example being when applied to autonomous vehicles that have to perform vehicle navigation and tracking to avoid collisions and to communicate with other vehicles. Results show that for both tasks, neural networks outperform ML models in accuracy, but require more computation and time for training.

Data Preprocessing Methods

To begin, we need to convert the complex-valued CSI matrices to real-valued matrices because they cannot be used directly as input to a general neural network. The raw CSI data is a complex-valued matrix $H \in \mathbb{C}^{(4x1632)}$, where 4 represents the number of antennas of the remote radio unit (RRU) and 1632 represents the number of subcarriers. To solve this, take the modulus of the CSI matrices:

$$\text{real\_csi} = \text{abs(complex\_csi)} = \sqrt{(real^2 + imaginary^2)}$$

After doing this, we will have (1) 15000 training samples and (2) 5000 samples left over for testing where both (1) and (2) are real-valued matrices.

Next, we need to normalize this data because the different features in the CSI data may have different scales. Therefore, data normalization can improve the performance of machine learning models and neural networks by helping to keep inputs within a manageable range which stabilizes training. We applied min-max normalization to scale all CSI values between 0 and 1.

Min-max normalization was chosen because this prevents features with large value ranges from dominating the learning process and accelerates neural network convergence. An important distinction to note is that we computed the normalization parameters using only the training set statistics and applied them to both training and validation data, preventing data leakage. If we used (valid csi modulus - valid_min) / (valid_max - valid_min) for valid_CSI_norm, this would not accurately validate the model and let the model "cheat" which can be manifested in suspiciously high validation accuracy.

After normalization, the data must be formatted differently depending on the model type. For the ML models, the 4D CSI matrices [batch, 1, 4, 1632] are flattened into 2D arrays [batch, 6528] where 6528 = 1 × 4 × 1632. This is necessary because ML algorithms require 1D feature vectors as input.

Description of Neural Network and Machine Learning Models

LoS/NLoS Classification using Machine Learning (Logistic Regression)

Logistic regression is a machine learning algorithm that is commonly used for binary classification tasks, which fits the application of LoS/NLos classification. The activation function is a sigmoid function that takes in a linear combination of inputs and outputs a value between 0 and 1. This model output can be interpreted as the probability value indicating the likelihood that the data belongs to either the LoS or NLoS class.

Performance Metrics:

- Accuracy: 0.9438
- Precision: 0.9699
- Recall: 0.7222
- F1 score: 0.8279
- Training Time: 3.66 seconds
- Testing Time: 0.058 seconds

LoS/NLoS Classification using Neural Networks (CNN Classifier)

The CNN architecture is more complex and preserves the 2D spatial structure of CSI matrices (4 antennas × 1632 subcarriers), enabling the network to learn spatial patterns that logistic regression cannot detect.

Network Design:

- Input: Batches of 32 samples, shape [batch=32, 1, 4, 1632]
- Conv Layer 1: 1→32 filters (3×3 kernel) + Batch Normalization + ReLU + MaxPool (2×2)
- Conv Layer 2: 32→64 filters (3×3 kernel) + Batch Normalization + MaxPool (2×2)
- Flatten Layer: Reshapes the 4D tensor output [batch, 64, 1, 408] into a 2D tensor [batch, 26112] (26112 = 64 × 1 × 408) because fully connected layers require 1D feature vectors as input.
- Fully Connected Layers: Three dense layers (26112→256→128→2) with ReLU activations and 30% dropout
- Output: 2 nodes with softmax producing LoS/NLoS probabilities

Batch normalization after each convolution stabilizes training; dropout in hidden layers reduces overfitting risk. The loss function is Cross-entropy loss and the optimizer is Adam with a learning rate of 0.001. The model trained for 10 epochs and the model has a total of 6,737,090 parameters.

Performance Metrics:

- Accuracy: 0.9968
- Precision: 1.0000
- Recall: 0.9829
- F1 score: 0.9914

- Training Time: 608.08 seconds
- Testing Time: 9.02 seconds

Vehicle Position Estimation using Machine Learning (Linear Regression)
In machine learning, linear regression is commonly used to predict a continuous numerical value by fitting a linear function to the corresponding data, which works for this application since the objective is to predict continuous (x,y) coordinates of the vehicle relative to the RRU.

Performance Metrics:

- Mean squared error (MSE): 22.99 m^2
- Root mean squared error (RMSE): 4.80 meters
- Training Time: 28.73 seconds
- Testing Time: 0.04 seconds

Vehicle Position Estimation using Neural Networks (CNN Regressor)
The architecture of the CNN Regressor is similar to the CNN classifier, with differences in the output layer and loss function such as that it outputs continuous values representing spatial coordinates. The output layer has no activation function, allowing the network to predict any real-valued coordinates.

Architecture:

- Input layer: Batch size 32, shape [batch_size=32, 1, 4, 1632]
- Convolutional layers: Same as CNN classifier (2 layers with 32 and 64 filters)
- Batch normalization after each convolutional layer
- Pooling layers: MaxPool2d with kernel size 2×2
- (Same flatten layer as before) → Fully connected layers: 3 layers (FC1: 26112→256, FC2: 256→128, FC3: 128→2)
- Output layer: 2 nodes (x coordinate, y coordinate)

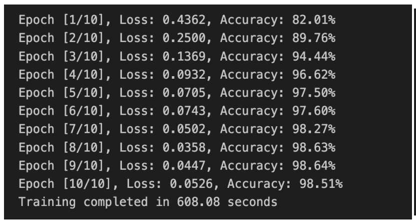
The activation function is ReLU, loss function is Mean Squared Error (MSE) loss, optimizer is Adam with learning rate = 0.001. The model trained for 10 epochs and has 6,737,090 total parameters.

Performance Metrics:

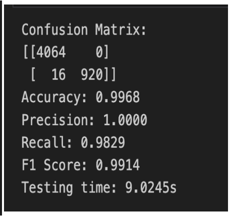
- MSE: 14.26 m^2
- RMSE: 3.78 meters
- Training Time: 578.51 seconds
- Testing Time: 9.87 seconds

Comparative Results Between Machine Learning and Artificial Neural Network Models

CNN Training for LoS/NLoS

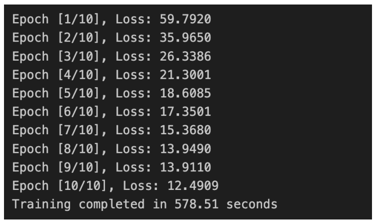


```
Epoch [1/10], Loss: 0.4362, Accuracy: 82.01%
Epoch [2/10], Loss: 0.2500, Accuracy: 89.76%
Epoch [3/10], Loss: 0.1369, Accuracy: 94.44%
Epoch [4/10], Loss: 0.0932, Accuracy: 96.62%
Epoch [5/10], Loss: 0.0705, Accuracy: 97.50%
Epoch [6/10], Loss: 0.0743, Accuracy: 97.60%
Epoch [7/10], Loss: 0.0502, Accuracy: 98.27%
Epoch [8/10], Loss: 0.0358, Accuracy: 98.63%
Epoch [9/10], Loss: 0.0447, Accuracy: 98.64%
Epoch [10/10], Loss: 0.0526, Accuracy: 98.51%
Training completed in 608.08 seconds
```

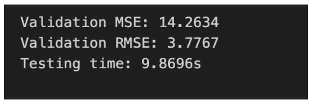


```
Confusion Matrix:
[[4064    0]
 [  16  920]]
Accuracy: 0.9968
Precision: 1.0000
Recall: 0.9829
F1 Score: 0.9914
Testing time: 9.0245s
```

CNN for Vehicle Position Estimation (VPE)

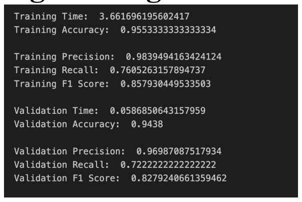


```
Epoch [1/10], Loss: 59.7920
Epoch [2/10], Loss: 35.9650
Epoch [3/10], Loss: 26.3386
Epoch [4/10], Loss: 21.3001
Epoch [5/10], Loss: 18.6085
Epoch [6/10], Loss: 17.3501
Epoch [7/10], Loss: 15.3680
Epoch [8/10], Loss: 13.9490
Epoch [9/10], Loss: 13.9110
Epoch [10/10], Loss: 12.4909
Training completed in 578.51 seconds
```



```
Validation MSE: 14.2634
Validation RMSE: 3.7767
Testing time: 9.8696s
```

Logistic Regression for LoS/NLoS



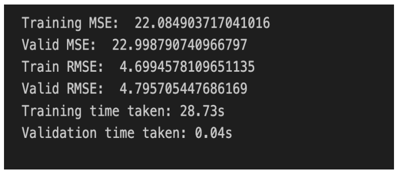
```
Training Time:  3.661696195602417
Training Accuracy:  0.9553333333333334

Training Precision:  0.9839494163424124
Training Recall:  0.7605263157894737
Training F1 Score:  0.857930449533503

Validation Time:  0.0586850643157959
Validation Accuracy:  0.9438

Validation Precision:  0.96987087517934
Validation Recall:  0.7222222222222222
Validation F1 Score:  0.8279240661359462
```

Linear Regression for VPE



```
Training MSE:  22.0849037170410l6
Valid MSE:  22.998790740966797
Train RMSE:  4.6994578109651135
Valid RMSE:  4.795705447686169
Training time taken: 28.73s
Validation time taken: 0.04s
```

As illustrated by the screenshots of training and validation runs, neural networks typically achieve better accuracy and performance in comparison to simple ML regression models, but this is a tradeoff between accuracy and efficiency. CNNs take significantly longer to train and are more compute intensive, but are typically more accurate for this task compared to ML.

Table 1. Model Efficiency-Accuracy Tradeoff and Comparative Study

Task/Objective	Model	Training time (s)	Testing time (s)	Accuracy
Los/NLoS Classification	Logistic Regression	3.66	0.058	94.4%
	CNN (Classifier)	606.08	9.02	99.7%
Vehicle Position Estimation	Linear Regression	28.73	0.04	MSE: 22.99m² RMSE: 4.80m
	CNN (Regressor)	578.51	9.87	MSE: 14.26m² RMSE: 3.78m

According to Table 1, neural networks had higher accuracy than the ML models in both tasks but at a cost of compute resources and time. If the application of a model demands more efficient methods, perhaps due to computing or time constraints, and the difference in accuracy between the neural networks and ML models are not significant, it would be advantageous to implement ML models instead of neural networks due to their simplicity. On the contrary, neural networks are more robust models that can learn more complex features and on average deliver better accuracy results if computational resources, like sufficient GPUs, are available. Neural networks, due to their complexity and high number of parameters, are less interpretable than regression models that have fewer parameters and use linear relationships. The CNN models implemented contained 6,737,090 trainable parameters vs. approximately 13,058 parameters for the linear regression model and 6,529 parameters for the logistic regression model, which shows the large difference in size between the models.

Conclusion

After evaluating neural network models and machine learning models on their performance with respect to the Los/NLoS classification and vehicle position estimation tasks, neural networks outperform machine learning models in terms of accuracy. However, machine learning models are significantly more efficient in terms of compute and time. The decision to choose a model depends on multiple factors like the importance of accuracy, compute and time restraints, and interpretability.