DEEP LEARNING BEAM PREDICTION USING MUTI-MODAL SENSORS

Junnan Wang[1]
[1]University of Stuttgart

Junnan Wang, st165067@stud.uni-stuttgart.de

Abstract – *With the rapid development of the communications sector, the spectrum resources available for exploitation are decreasing. Millimeter wave, which has abundant spectrum resources, is beginning to receive widespread attention. In this paper, a machine learning based model is trained to predict the millimeter wave beam indices. In this paper, position data and camera data are fused as input to the neural network to train the model. The prediction accuracy of the trained model with Sensor fusion data is improved by approximately 2%. The best model found using NNI improved the prediction accuracy further by approximately 5%. The DBA-Score of the top 3 beams corresponding to the best model was close to 0.9. The final results also show that the best model trained in known scenarios can achieve prediction accuracy of around 52% and DBA-Score of 0.65 for top 3 beams in the unseen scenario. On the test dataset, the DBA-Score for the top 3 beams model trained with the overall data was 0.59. The DBA-Score for the top 3 beams model trained with each scenario was 0.63, an improvement of 0.04.*

Keywords – sensor-aided beam prediction, mmWave, ML model, NNI

1. INTRODUCTION

Nowadays millimeter waves are receiving more and more attention in the field of smart communications. This paper aims to fuse sensor data to predict millimeter wave (mmWave) beam indices.

Millimeter waves are electromagnetic waves in the frequency band between 30 and $300\,GHz$ with a wavelength of $1\,mm$ to $10\,mm$. With the rapid development of 4G and 5G for mobile communications, most of the frequency resources within $30\,GHz$ are being put to use. The number of free bands that can be exploited is decreasing. Millimeter wave, which has not been put into use, offers a vast resource space for communication networks.

According to the White Paper [1], 5G millimeter wave has six major technical advantages: abundant frequency resources, great bandwidth, easy integration with beamforming technology, very low latency, dense deployment, high accuracy positioning and high integration. Based on these advantages, 5G millimeter wave can be applied in a wider range of scenarios and areas.

2. EXPERIMENT STATEMENT

In recent years, research related to sensor-aided beam prediction has also yielded some results. In order to bring this solution closer to the real world, two issues are of importance to explore. Firstly can the sensor-aided beam prediction solution perform well on real-world data? How well they perform in the real world will determine whether these solutions can be applied and serve the real world. Secondly, whether a machine learning model for beam prediction trained in certain known scenarios is still applicable in new scenarios. If the model could be generalised, it would not only reduce overhead but would also be relevant.

The main objective of this paper is to develop a machine learning (ML) model for prediction of beam indices in a dataset of known scenes using multi-modal sensors data collected at different locations with different environmental characteristics as input and to test the performance of the model in scenes that have not been seen before.

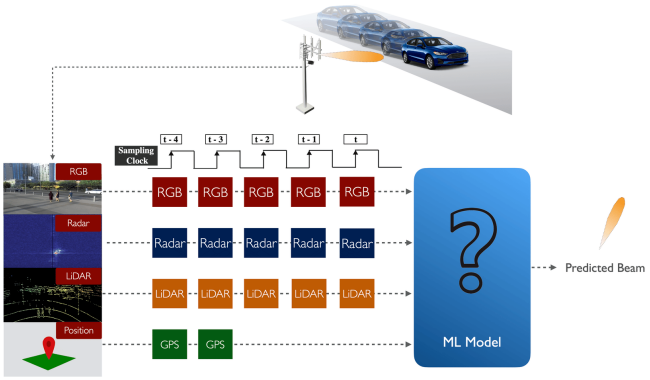


Fig. 1 – Schematic representation of the input data sequence [2]

In line with the above objectives, the main direction of this paper is multi-modal sensor-aided beam prediction. The sensors used in this paper are GPS, LiDAR sensor, radar sensor and camera. These sensors are mounted on the base station and on the mobile vehicle respectively. Each sensor is recording data all day and all night. Using the sequence of sensors data (i.e. LiDAR, radar and images) captured at the base station, the current optimal beam indice is predicted. As shown in the Fig. 1, at any time instant t, a sequence of 5 samples of the current and pre-

viously observed sensors data is provided, i.e. $[t-4, ..., t]$. LiDAR, radar and image data are all sequences consisting of 5 samples. Only the ground truth GPS positions of the first two samples in the sequence, i.e. time instant $t-4$ and $t-3$ at the mobile end, are known. Using this data sequence as input, a machine learning model is applied to predict the beam indice at time t.

It is important to note that in real life, location data may not be available at every time of day and there may be additional delays. Thus only the first two sampled GPS data of the sequence are set here to be known. This also indicates that the beam prediction model studied in this paper does not completely depend on GPS position information.

3. INTRODUCTION TO THE DATASET

3.1 Data soure

The data in this paper is derived from DeepSense 6G [3]. DeepSense 6G is a real-world multi-modal dataset that includes co-existing multi-modal sensing and communication data for different scenarios. On the one hand, the data sources are realistic and valid, and on the other hand, the different scenarios and multi-modal sensing data make for a rich and comprehensive data composition.

In this paper, the data from scenarios 31-34 for model training and beam indices prediction are applied. Using realistic datasets as a basis for beam indices prediction makes the experimental study more realistic.

3.2 Dataset in the experiment

Scenarios 31-34 were all collected in an outdoor wireless environment representing a two-way city street. The sensor datasets collected include position data, radar data, LiDAR data and camera data.

The majority of the training datasets consisted of datasets from 32-34 three scenarios. The composition of each data sample was as follows.

- Sequence of 5 data samples (image, LiDAR and radar)
- Ground truth GPS positions of the transmitters for the first two instances in the sequence
- 64×1 power vector corresponding to the 5-th sample in the sequence
- Optimal beam indice corresponding to the 5-th sample in the sequence

For the composition of the test set, 50% of the test dataset is from scenarios 32-34 and the remaining 50% is from scenario 31, where the data in scenario 31 contains only a sequence of 5 data samples (image, LiDAR and radar) and the ground truth position data of the first two samples in the sequence. Scenario 31 is at a completely different location and street from the other three scenarios. Thus, the prediction effectiveness of the training model in unseen scenes can be checked.

3.3 Processing of position data

The position data contains the latitude and longitude of the base station and the experimental vehicle. Map projections require the establishment of a one-to-one correspondence between points on the Earth's surface and points in the projection plane. The Universal Transverse Mercator Projection (UTM) is an internationally standardised map projection. With UTM, the latitude and longitude coordinates can be transferred to a Cartesian coordinate system in the plane. To show the experimental vehicle trajectory more clearly, the $x-y$ plane is shown here.

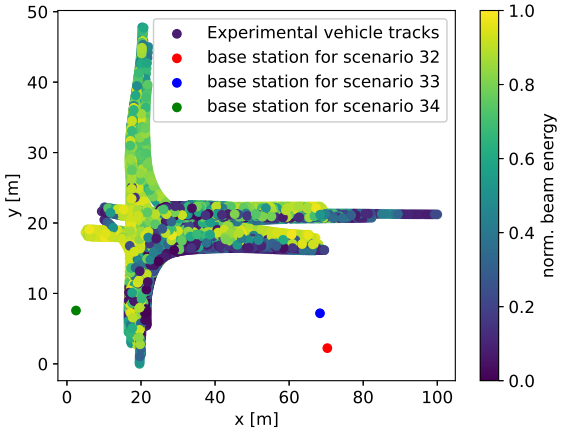
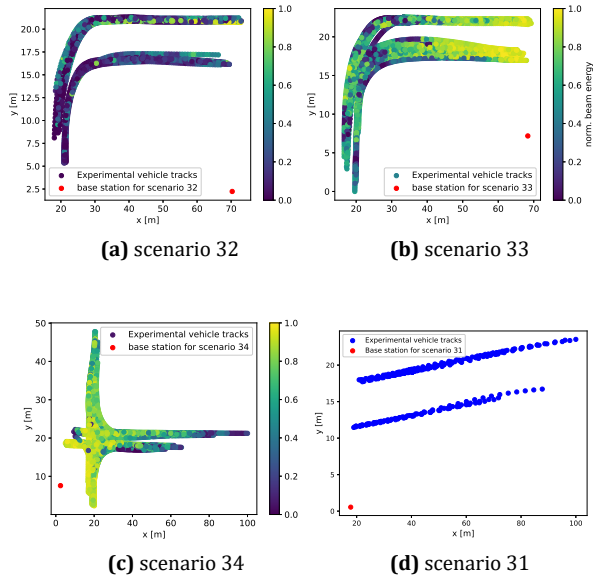


Fig. 2 – Top view of experimental vehicle track in Scenario 32-34

The top view of the converted base station and mobile is shown in Fig. 2. The Fig. 2 shows the trajectory of the experimental vehicle in scenarios 32-34. The colourbars represent the mmWave beam energy corresponding to the experimental vehicles at different locations. It ranges from 0 to 1. The locations of the base stations in scenario 32, 33 and 34 are marked in red, blue and green respectively in the Fig. 2. As can be seen, the experimental vehicles travel on two roads that are perpendicular to each other. And each road contains two lanes. The base stations in scenarios 32 and 33 are very close together, both on the side of the road. The base station in scenario 34 is close to the crossroads. The normalised beam energy shows no obvious rule.

The Fig. 3 shows the trajectory of the experimental vehicle in the different scenarios. It can be seen that the experimental vehicles travel similar trajectories in scenarios 32 and 33. The base stations in scenarios 32 and 33 are also close to each other, with scenario 32 collecting sensor data mainly during the daytime. Scenario 33, on the other hand, collects sensor data mainly at night. Scenarios 32 and 33 both measure the route of the experimental vehicle from a straight line through an intersec-

tion followed by a turn onto another road. The trajectory of the experimental vehicle is more singular. The base station 34 is located at an intersection, the trajectory of the experimental vehicle is richer and more complex in scenario 34. It contains almost all possible routes.



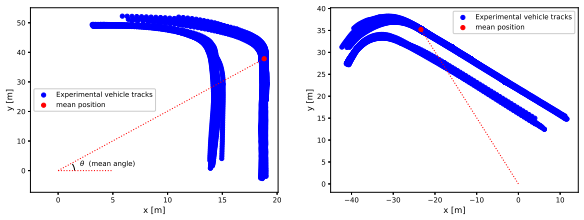
(a) scenario 32 **(b)** scenario 33

(c) scenario 34 **(d)** scenario 31

Fig. 3 – Top view of the tracks of the experimental vehicle in each scenario

Scenario 31 is in a completely different street to scenarios 32 to 34. The trajectory of the experimental vehicle in scenario 31 is shown in Fig. 3. It is clear from the figure that the experimental vehicle travels a relatively single trajectory in scenario 31, i.e. two straight lines. Compared to scenarios 32 to 34, in scenario 31 the vehicles only go straight. The base station in scenario 31 is located at one end of the road.

Based on the base station coordinates and the trajectory of the experimental vehicle in different scenarios, the distance range and angle range from the vehicle to the base station can be calculated. The formulae are shown in Eq. *(1)*a.



(a) Original coordinates **(b)** Calibrated coordinates

Fig. 4 – Calibration of experimental vehicle coordinates

In order to make the angle calculations closer to the field of view (FoV), the trajectory coordinates of the experimental vehicle were rotated around the expected angle of the radar. When calculating the angle for each scenario, the mean position in each scenario was chosen to calculate the mean angle and was calibrated with that angle. The coordinates were calibrated using the formula in Eq. *(2)*. The results of the calibration for scenario 32 are shown in Fig. 4. The left figure shows the original trajectory of the vehicle and the right figure shows the calibrated trajectory of the vehicle. In this paper, the angular range of the experimental vehicle will be calculated using the calibrated trajectory coordinates.

$$d = \sqrt{(x - x_0)^2 + (y - y_0)^2} \qquad (1)\text{a}$$

$$a = \arctan \frac{x_{new}}{y_{new}} \qquad (1)\text{b}$$

$$\begin{aligned} x_{new} &= x \cdot \cos\theta - y \cdot \sin\theta \\ y_{new} &= x \cdot \sin\theta + y \cdot \cos\theta \end{aligned} \qquad (2)$$

The range of distances and angle ranges for the experimental vehicles in each scenario were calculated and are shown in Table 1. The range of distances and angles varies in each scenario. Overall, the range of distance is from $9.8\,m$ to $98.5\,m$, and the range of angles is from $-56.4\,^\circ$ to $63.7\,^\circ$.

Table 1 – Range of distances and angles for experimental vehicle trajectories

Scenario	Distance range[m]	Angle range[$^\circ$]
scenario 32	13.9 ∼ 53.1	-56.4 ∼ 38.2
scenario 33	9.8 ∼ 51.8	-38.4 ∼ 63.7
scenario 34	11.0 ∼ 98.5	-54.5 ∼ 39.2
scenario 31	10.9 ∼ 85.3	-39.2 ∼ 32.0
overall	9.8 ∼ 98.5	-56.4 ∼ 63.7

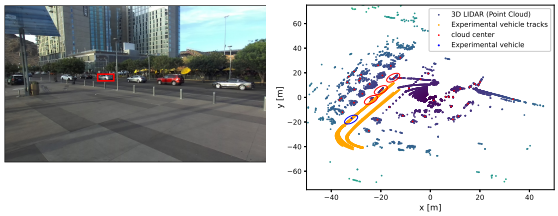
As the GPS data is more accurate and contains only base station and target vehicle information, in the later data processing, this paper applies GPS data to filter other sensor data to extract the corresponding target vehicle information.

3.4 Processing of radar data

MmWave radar can be used to measure the distance, azimuth and speed data between the experimental vehicle and the base station. In this paper, the two-dimensional range-angle and range-velocity data can be obtained through the three-dimensional FFT transform. In the experiment, it is difficult to distinguish the data corresponding to the experimental vehicle when the two objects are very close together or when there is occlusion. When applying the distance and angle information obtained in the previous section to filter the radar data, the experimental vehicles were often undetectable or had large errors. As can be seen in Table 1, the maximum distance from position data in the scenarios is $98.5\,m$, while the effective detection distance of the radar currently used is only $45\,m$. Thus, this paper did not choose to use the radar data to predict the beam indices.

3.5 Processing of LiDAR data

LiDAR sensors are commonly used to generate 3D point cloud data to show details of the surroundings and objects. Take Scenario 32 as an example. The 3D point cloud data detected by the Scenario 32 base station is shown in Fig. 5, right. The red points are the centre points of the individual point cloud data, the blue points are the experimental vehicles and the orange points are the trajectories of the experimental vehicles in scenario 32. By comparison to the RGB image on the left it can be seen that there are multiple vehicles including the experimental vehicle driving on the road in the actual environment. However, only the three closest vehicles are detected in the 3D point cloud map. The experimental vehicles and other more distant vehicles cannot be detected. The three red ellipses in the figure circle the three detected vehicles and the blue ellipses circle the experimental vehicles. The comparison also shows that most of the track locations in the actual trajectory are not detected. Therefore, the LiDAR data is not chosen to predict the beam indices in this paper.



(a) RGB image **(b)** 3D LiDAR

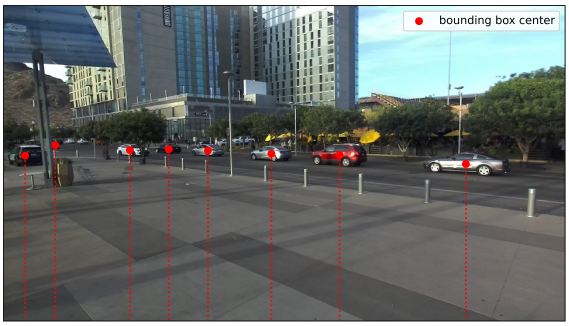
Fig. 5 – RGB image and corresponding 3D LiDAR (Point Cloud)

3.6 Processing of camera data

In the camera image, the environment and vehicle details in the scenario can be clearly seen. The target vehicle can be visually located from the image. But in addition to containing information about the target vehicle the image also contains information about the environment, the road, pedestrians and other non-target vehicles. This paper uses a combination of the popular target detection algorithm YOLOv7 and the multi-target tracking algorithm DeepSORT [4] to detect target vehicles from images and extract valid data. In short using YOLOv7 as the detector in DeepSORT for target detection.

Initially, five RGB image samples from each sequence are stitched together in moment order to form a small video, which is used as input to the YOLOv7+DeepSORT algorithm. The angles already calculated in the previous sections were used to filter out the experimental vehicle detection bounding boxes. However, sometimes the experimental vehicles are far away from the base station and the calculated angle deviates from the FoV, so sometimes the correct experimental vehicle is not tracked. In order to improve the detection accuracy, the images in each scenario are stitched together into one long video in moment order. For target detection, this paper sets the target detection number to 2, i.e. only the car in the image is detected. The final detection effect is shown in Fig. 6. Fig. 6 shows the detection effect of a frame in the video. As can be seen from the figure, each car has a corresponding detection bounding box. The car id, car and confidence are marked on the top of the detection bounding box. Each frame will be labelled with a different id for each detected car. The detected vehicles are continuously tracked in successive frames, i.e. ideally the id of the same vehicle will not change. As long as the id of the target vehicle can be determined, the experimental vehicle can be tracked by its car id and the corresponding detection bounding box data can be obtained.

Fig. 6 – Yolov7+DeepSORT's target detection and tracking results



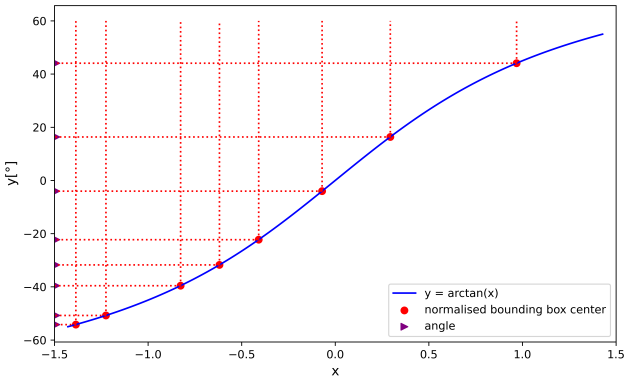


Fig. 7 – Calculation of the angle corresponding to the detection bounding box in the image

In this paper the angular data is used as a basis for filter-

ing the experimental vehicles in the RGB images. Take an RGB image from scenario 32 as an example. The process of calculating the angle of the target vehicle in the RGB image is shown in Fig. 7. According to the description in the dataset, the FoV of the camera in this paper ranges from -55° to 55°. The formula for calculating the field of view shows that the inverse trigonometric function (arctan) is used. The range of the field of view allows the range of the corresponding tangent to be determined. Based on the range of angles and the range of tangents, a figure can be drawn, i.e. Fig. 7 bottom. The coordinates of the center of the bounding box of the detected vehicle are then calculated from the coordinates of the bounding box, and the center of each detected vehicle is marked with a red dot in the RGB image. Here the x-axis coordinates are chosen to calculate the corresponding angle. The x coordinate of the center point is normalised to the specified range and the inverse trigonometric function is calculated to obtain the angle of the detected vehicle. To avoid the effect of vehicle size on the angle, the angle corresponding to the left and right sides of the detection bounding box can be calculated, and the vehicle closest to the detection bounding box is the experimental vehicle.

As only the position data of the first two samples in a sequence are known, accordingly only the ids and bounding boxes of the experimental cars in the RGB images of the first two samples can be filtered based on the position data. The ids and bounding boxes of the experimental vehicles in the rest of the RGB images will be obtained by tracking.

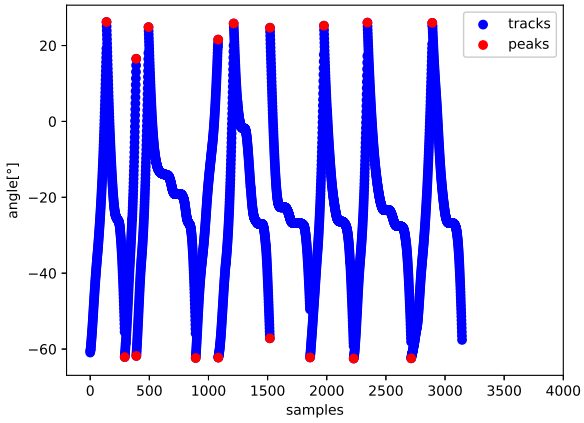


Fig. 8 – All tracks in scenario 32

All trajectories are plotted in moment order in each scenario. Take scenario 32 as an example. As shown in Fig. 8, the x-axis represents the sample index,and the y-axis represents the angle corresponding to the coordinates of each sample. A track is divided by the change of angle. The blue dots represent the variation of the experimental vehicle trajectory with angle. The red points represent the end of the previous trajectory and sometimes the beginning of the current trajectory. These starting or ending points are extracted and recorded. The interval between two adjacent red points is a trajectory. Each track is considered as an interval within which the id_x of the car filtered by the calculated angle is recorded within a range of $35\,m$. $35\,m$ is chosen because sometimes the distance is too far to capture the experimental vehicle with a large error, and too close with too few valid values (only some of the RGB images correspond to known GPS data). The id_x with the highest number of occurrences is recorded as id_r. In each interval, all id_x are reset to id_r. Then find the corresponding detection bounding boxes based on id_r. Finally, iterate through the array of detection bounding boxes, resetting the array to the previous non-zero array for values of 0.

At this point, the detection bounding box of the experimental vehicle corresponding to each RGB image is obtained. By filling in the detection bounding box data according to the original sequence, a detection bounding box sequence is obtained. These detection bounding box sequences are subsequently used as part of the input to the neural network.

4. MODEL

In the previous section the valid input data was extracted by fusing the individual sensor data with the help of the position data of the first two samples of the sequence. That is, the position data sequence and the detection bounding box sequence. The output data is the mmWave beam indice corresponding to the last sample of the sequence. The next step is to train model for beam prediction. Normal neural network is used in this paper.

4.1 Normal neural network

Neural networks are self-learning and self-adaptive. Neural networks generally consist of an input layer, an hidden layer and an output layer in terms of their structure. Each layer of a neural network is composed of a different number of neurons, and the number of neurons in each layer varies with the number of problem solving neurons. The output layer is usually followed by an activation function. The activation function is the mathematical equation that determines the data of a neural network. The output of each layer of the neural network is a linear function of the input of the upper layer. The reference to the activation function adds a non-linear element to the output of the neural network, allowing the neural network to approximate any non-linear function. Commonly used activation functions are Sigmod, tanh, ReLU, LeakyReLU, Softmax and Softplus, where ReLU and LeakyReLU are commonly used in regression tasks, Sigmoid and tanh are commonly used in binary classification tasks, and Softmax is commonly used in multi-classification tasks.

In the experiment, the input data is a 24-dimensional feature vector containing the filtered and valid sensor data. In detail the input data contains the po-

sition data of the first two samples in the sequence, i.e. $[distance, angle]$, and also contains a sequence of bounding boxes for the five samples in the sequence, i.e. $[x_{center}, y_{center}, width, height]$. The number of hidden layers in neural network is 5. In the initial experiments, the activation function ReLU was used in the hidden layer. The choice of hyperparameters, weights and activation functions in neural networks is very important and affects the effectiveness of the training. Specific parameter optimisation and selection is described in later sections.

4.2 Loss function

After the machine learning model is determined, the next step is the selection of the loss function. The commonly used loss functions in machine learning algorithms are Mean Squared Error (MSE), Binary Cross Entropy (BCE), Cross Entropy (CE), etc.

$$L = \frac{1}{N}\sum_i L_i = -\frac{1}{N}\sum_i [y_i \cdot \log(p_i) + (1-y_i) \cdot (1-\log(1-p_i))] \quad (3)$$

The target values for this experiment are beam indices in the range of 0 to 63. One-hot coding is used to set the output data as a 64-dimensional vector based on the beam indices. As the vector contains only 0 and 1, it can also be regarded as a special binary classification task. The BCE is thus chosen as the loss function to train the neural network. BCE is calculated as shown in the Eq. *(3)*.

4.3 Neural Network Intelligence

In the process of training a model, one often has to manually adjust the parameters and structure of the model, train it over and over again and compare the results to get the best model in order to get a better output and model. This approach is undoubtedly inefficient and time-consuming. To solve this problem, an automated machine learning tool is applied in this paper.

This paper uses the open source automated learning tool neural network intelligence developed by Microsoft [5]. Neural Network Intelligence (NNI) is a lightweight yet powerful tool. It contains four main functions: hyperparameter optimization, neural architecture search (NAS), model compression and feature engineering.

An overview of how NNI is used and what it does is shown in the Fig. 9. Fill in the python script with the statements relevant to the function you want to implement. Configure the config file and search space file for each function. Experiments can be run and managed using the NNICTL command line. The results of the experiments can be displayed on the web through a dedicded port. The web interface shows the progress of the experiment and allows you to set the number of experiments and the maximum run time. The web interface can also display the results of the top k run and the corresponding hyperparameters.

The visualisation of the experiment results allows to see dynamically how the experiment is running and to obtain the hyperparameters and network structure corresponding to the best results.

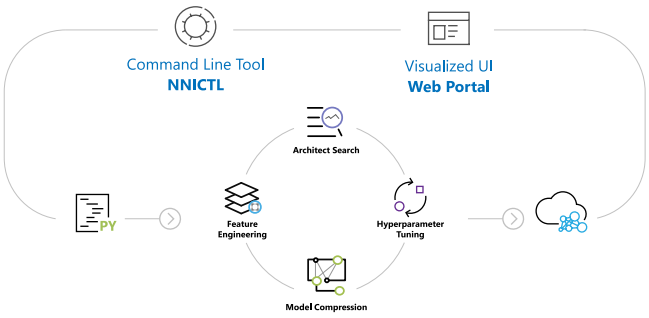


Fig. 9 – Overview of NNI functions and operational processes [5]]

This paper mainly applies neural architecture search function. The number of hidden layers, the number of neurons in each hidden layer and the choice of activation function in a neural network all need to be adjusted and optimised. For this reason, this paper uses the NAS framework Retiarii [6] from NNI. Retiarii is the first deep learning framework to support exploratory training on whole machines, and the first deep learning framework to support exploratory training on neural network models [6].

This paper uses $classnni.retiarii.nn.pytorch$ from Retiarii. Use $ValueChoice$ to set the range of parameters, for example in a neural network, to set the number of neurons in different hidden layers to any of [128, 256, 512, 1024, 2048]. $LayerChoice$ was used to select the best activation function for each hidden layer. Training objectives can be set during training, such as minimising loss or maximising accuracy. By using NNI for training, the model will autonomously choose one of the set parameters for each experiment, and finally come up with the best parameters and network structure that will give the best output. This automated machine learning greatly reduces the time required to find the best parameters and network results, and also greatly improves training efficiency. The webUI allows people to view the experimental process in real time. One can also view the results of the top k experiments and the corresponding parameters and model structure. The final training results will be shown and analysed in detail in the next section.

5. ANALYSIS OF RESULTS

This section focuses on the presentation and analysis of prediction results for neural networks trained with sensor data fusion as input. This paper is judged on two criteria, prediction accuracy and DBA-Score. Accuracy is more common and is shown in Eq. *(4)*. The Distance-based Accuracy Score (DBA-Score) is defined as shown in Eq. *(5)*a, where Y_k is defined as Eq. *(5)*b. y_n and $\hat{y}_{n,k}$ are ground truth beam index and the k-th predicted beam indice respectively. $\triangle$ is a normalization factor. Here, $\triangle$ is set to

5.

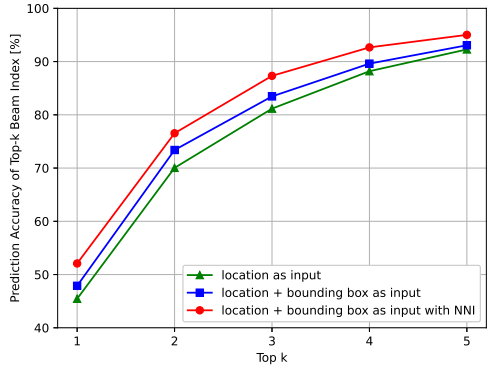
$$accuracy = \frac{Correct\ numbers}{Total\ numbers} \tag{4}$$

$$DBA - Score = \frac{1}{3}(Y_1 + Y_2 + Y_3) \tag{5}a$$

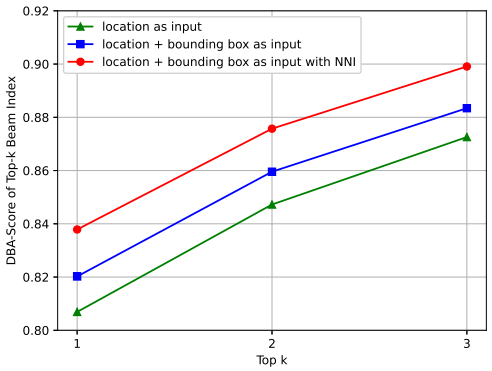
$$Y_k = 1 - \frac{1}{N}\sum_{n=1}^{N} \min_{1 \le k \le K} \min\left(\frac{|\hat{y}_{n,k} - y_n|}{\triangle}, 1\right) \tag{5}b$$

The input data of the model consists of following two components:

- Location data: contains the distance and angle data, only the position data of the first two samples in the sequence is known.
- Detection bounding box: contains the coordinates of the center point, the length and width of the bounding box.



(a) Accuracy



(b) DBA-Score

Fig. 10 – Comparison of prediction results for top k beams

The prediction results of training dataset corresponding to the different inputs are plotted in Fig. 10. As can be seen from the Fig. 10a, when only the position data is used as input, the prediction accuracy of top 1 beam is around 45% and the prediction accuracy of top 3 beams is around 80%. It is clear from the Fig. 10a that the prediction accuracy of the model is being optimised. When the position data was fused with the experimental vehicle detection bounding box data extracted from the RGB images, the prediction accuracy of the model increased by approximately 2%. It shows that multi-sensor data fusion provides more comprehensive information and effectively improves prediction accuracy. The first two are based on the basic neural network for training. The number of hidden neurons in the neural network is set empirically. The activation function of each hidden layer is set to ReLU. When NNI's neural architecture search was applied to find the best model, the prediction accuracy of the model improved by another approximately 5%. This shows that the neural architecture search helped the model to find the best hyperparameters, weights and activation functions. Here, the training objective set is to maximise the DBA-Score of top 3 beams.

As can be seen from the Fig. 10b, although the DBA-Score did not change as much as the accuracy scores, the DBA-Scores for top 3 beams were also optimised. Sensors data fusion improved DBA-Scores by approximately 0.01. NNI improved DBA-Scores by another approximately 0.02. The DBA-Score of the top 3 beams corresponding to the best model was close to 0.9.

Compared with [2], the prediction accuracy of the models trained in this paper has also been improved significantly. As can be seen from Table 2, the prediction accuracy of top 1 beam has improved by about 20% and the prediction accuracy of top 3 beams has also improved by about 20%. This indicates that the sensor data fusion and neural architecture search from NNI effectively improved the prediction results of the models.

Table 2 – Comparison of the prediction accuracy of the original model and the optimised model

Models	**Accuracy**		
	top 1 [%]	**top 3 [%]**	**top 5 [%]**
Original model	31.87	63.64	77.24
Optimised model	52.08	87.32	95.02

The results shows that both sensor data fusion and NNI optimise the models to varying degrees and improve their prediction accuracies and DBA-Scores.

Table 3 – Prediction accuracy and DBA-Score in scenario 31

	Optimised model
Accuracy of top 1 [%]	20.0
Accuracy of top 3 [%]	52.0
DBA-Score of top 3	0.65

Above are the results of the tests for known scenarios 32 to 34. The results for the unseen scenario 31 are shown in Table 3. It can be seen that although the overall prediction results for scenario 31 are not as good as for known scenarios 32 to 34. The Prediction accuracy for top 3 beams can be achieved at around 52%. The DBA-Score for top

3 beams can be achieved at 0.65. The optimised model can achieve more than half prediction accuracy for top 3 beams. To a certain extent, this makes sense.

The performance of the model on the test set is shown in Table 4. Initially, the model obtained by training using all data from the three scenarios had an overall DBA-Score of 0.59 for performance on the test set. With the model performing well in scenarios 32 to 34. For further optimisation, the best model was trained for each scenario using the same training approach, based on the dataset for each scenario. Scenario 31 was trained based on the data from scenario 34, as the data from scenario 34 is more extensive. For each scenario in the test set, the DBA-Scores were improved to varying degrees. The overall performance score was 0.63, an improvement of 0.04.

Table 4 – DBA-Score for top 3 beams in test dataset

	DBA-Score for top 3 [%]	
	model for all	**model for each scenario**
scenario 32	79.01	80.12
scenario 33	82.90	88.57
scenario 34	81.63	88.65
scenario 31	35.16	37.0

6. CONCLUSION

As the use of mmWaves becomes more widespread, the prediction of mmWave beam indices becomes more and more important. In this paper, a machine learning model for predicting the mmWave beam indice is trained using a neural network with data fused from position data and bounding box data as input. In order to get better prediction results, the neural architecture search from NNI was used to find the best model. Based on the results, it can be seen that the fused data improved the prediction accuracy of the model by approximately 2% and the NNI improved the prediction accuracy of the model by another approximately 5%. The final best DBA-Score obtained for top 3 beams was close to 0.9. In the unknown scenario 31, top 3 beams achieved a prediction accuracy of around 52% and a DBA-Score of 0.65.

On the test dataset, the DBA-Score for the top 3 beams model trained with the overall data was 0.59. The DBA-Score for the top 3 beams model trained with each scenario was 0.63, an improvement of 0.04.

REFERENCES

[1] GSMA. *5g-mmwave-technology-white-paper-gsma*. Sept. 2020. URL: https://https://www.gsma.com/greater-china/wp-content/uploads/2020/09/5g-mmwave-technology-white-paper-gsma-a4.pdf.

[2] G. Charan, U. Demirhan, J. Morais, A. Behboodi, H. Pezeshki, and A. Alkhateeb. "Multi-Modal Beam Prediction Challenge 2022: Towards Generalization". In: *arXiv preprint arXiv:2209.07519* (2022).

[3] A. Alkhateeb, T. Charan G. and Osman, A. Hredzak, and N. Srinivas. "DeepSense 6G: Large-Scale Real-World Multi-Modal Sensing and Communication Datasets". In: *to be available on arXiv* (2022). URL: https://www.DeepSense6G.net.

[4] Feng Yang, Xingle Zhang, and Bo Liu. "Video object tracking based on YOLOv7 and DeepSORT". In: *arXiv preprint arXiv:2207.12202* (2022).

[5] Microsoft. *Neural Network Intelligence*. Computer Software. Version 2.9. Jan. 2021. URL: https://github.com/microsoft/nni.

[6] Quanlu Zhang, Zhenhua Han, Fan Yang, Yuge Zhang, Zhe Liu, Mao Yang, and Lidong Zhou. "Retiarii: A Deep Learning Exploratory-Training Framework". In: *14th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 20)*. 2020, pp. 919–936.

AUTHORS

Junnan Wang Bachelor's degree at Shandong University in 2018 and Master's degree at University of Stuttgart in 2022.