Navigation of Wall Following Robot using Machine Learning

Aditya Joglekar
Carnegie Mellon University

Parth Malpathak
Carnegie Mellon University

Ketaki Tamhankar
Carnegie Mellon University

Viraj Ranade
Carnegie Mellon University

Abstract

Machine learning techniques are used in this paper for predicting the next step of a wall-following robot given its position. Results obtained from a SCITOS G5 mobile robot are used, which is sensor data collected after navigating the robot in a room by following walls in a clockwise direction. Classification models like Support Vector Machines, K Nearest Neighbors, Random Forest, Logistic Regression, Gaussian NB and Neural Networks are implemented and their results compared to find a machine learning algorithm that gives the best results. New data is generated for the wall-following task in an anti-clockwise direction. The accuracy is improved by reducing class imbalance. The most important sensors of the 24 used for the clockwise wall-following task as well as the combined clockwise and anticlockwise wall-following task are found.

1. Introduction

This paper aims to explore and add to the wall following navigation task of the SCITOS G5 mobile robot. Wall following tasks are extremely common and repetitive and they are largely being allocated to autonomous or semi-autonomous robots for monitoring and data collection. Previous work shows that wall following can be called a nonlinear problem, and neural classifiers have been used to model the same. This paper uses a number of different Machine Learning algorithms to model this task. Specifically, Random Forest, K Nearest Neighbors (KNN), Support Vector Machines (SVM), Logistic Regression, Gaussian NB and Neural Networks will be used for this problem. A comparative study of the different machine learning algorithms and their prediction accuracies for this dataset is carried out. The methods used involve data mirroring and a heuristic routine for creating a larger dataset, and a proposal to extend the task to anticlockwise wall following by the same robot. Moreover, experiments are carried out for reducing the number of sensors used for this task.

2. Related work

'Short-Term Memory Mechanisms in Neural Network Learning of Robot Navigation Tasks: A Case Study' by Ananda et al. [1] includes data acquisition from the SCITOS G5 robot, and exploration of the wall following task of the robot, in the clockwise direction. It investigates the use of STM (short term memory) mechanisms in conjunction with a variety of neural classifiers and checks if their performance in navigation and perception is influenced. Specifically, four neural classifiers are used: Logistic Perceptron, Multilayer Perceptron, Mixture of Local Experts (ME) and Elman Recurrent Network. The clockwise wall following task is conducted using these classifiers and then it is repeated with the use of short-term memory mechanisms. The MLP achieved best results, out of all the classifiers. Clear improvement in results was seen with the inclusion of STM mechanisms in the case of Logistic Perceptron. ME showed no improvements with the inclusion of time lagged values and MLP showed no improvements with STM inclusion. An alternative STM mechanism that uses recurrent neural connections was used to evaluate the Elman network. It was seen that the network gave the same performance as before but with 2 less neurons. This paper however does not explore other machine learning algorithms such as KNN and SVM on this task, and also does not explore anticlockwise direction wall-following direction classification.

Sarah Madi et al. in 'Classification Techniques for Wall-Following Robot Navigation: A Comparative Study' [2] used different machine learning algorithms for the SCITOS -G5 dataset. The paper focused on improving accuracy of KNN, by using an existing work related to genetic algorithms, local search, and Condensed Nearest Neighbors termed Memetic Controlled Local Search algorithm (MCLS). However, classification task for combined clockwise and anticlockwise wall-following directions has not been undertaken in this paper.

Generation of anticlockwise data, benchmarking of different machine learning algorithms for the

clockwise direction wall-following dataset, combined clockwise and anticlockwise dataset label classification, improving accuracy by reduction in class imbalance and finding the most optimum number and most important sensors for this dataset is the novel approach undertaken in the following sections.

3. Data

The SCITOS G5 is a mobile robot that is commonly used for perception and navigation tasks. The dataset used in this paper has been taken from the following experiment previously done in [1]. 24 ultrasound sensors were attached circularly along the waist of the robot. The sensors were numbered clockwise from 1 to 24 and each sensor spanned a 15-degree arc. Each sensor collected 9 readings per second. The readings have a maximum distance of 5m. The data collected was for 4 rounds of the clockwise wall following task. The dataset being used provides three sets of sensor readings:

1. 24 readings data: This data consists of all reading obtained from all 24 sensors and this will largely be used in the experiments conducted in this paper.

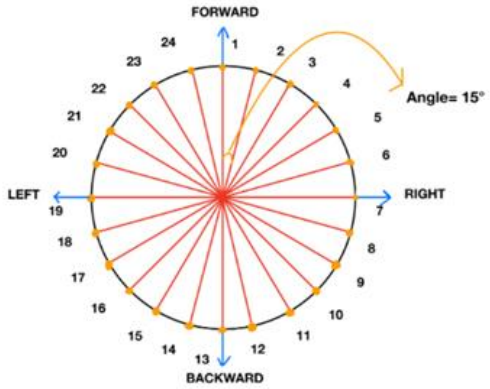

Figure 1: Sensor positions of the SCITOS-G5 robot

2. 4 readings data: This set considers four sets of sensors at the front, back, left, and right. Each set spans 60 degrees and the minimum reading of each set is taken as the 'simplified reading' for that set. This paper uses these readings for label generation and creation of new labels.
3. 2 reading data: This is the set of 'Front' and 'Left' simplified distances from the 4 readings dataset. This set will not be used in the experiments presented in this paper.

A heuristic software routine was used to generate the decisions taken by the robot in order to complete the wall following task. The navigation algorithm made use of IF-THEN rules based on 'Front' and 'Left' distance thresholds [1]. The four labels generated were:

1. Move forward: 2205 samples (40.41%)
2. Slight Right turn: 826 samples (15.13%)
3. Sharp Right turn: 2097 samples (38.43%)
4. Slight Left turn: 328 samples (6.01%)

The data did not have any missing attributes.

The scope of the project was improved by a data generation process which enabled the data collection if the robot followed the wall in anti-clockwise direction. A heuristic approach, like the one used for generation of the original clockwise direction wall-following task was used to generate the new dataset. In addition to the original data with 5456 data points, additional 5456 data points consisting of sensor positions for anti-clockwise fashion wall-following were generated. The pseudo code for generating the new data was as follows:

if *Right Distance > 0.9*
    then
    if *Front Distance <= 0.9*
    then *Stop and turn to the left*
    else *Slow down and turn to the right*
else
    if *Front Distance<=0.9*
    then *Stop and turn to the left*
    else if *Right Distance < 0.495*
    then *Slow down and turn to the left*
    else *Move Forward*

4. Methods

4.1 Benchmarking

Benchmarking was carried out for different machine learning algorithms to find the best algorithm which can be used to predict the direction for the given

dataset. The following machine learning algorithms were used in the project for benchmarking and comparisons:

1. K Nearest Neighbors:

K Nearest Neighbors (KNN) is a supervised machine learning algorithm, in which classification labels are computed from a simple majority vote of the K nearest neighbors of each point. KNN constantly evolves and this allows the algorithm to respond quickly to changes in the input during real-time use. When the number of features increases, KNN requires more data and hence KNN performs better with a lower number of features than a large number of features. This helped reducing the number of sensors used as shown in section 5.

2. Logistic Regression:

In classification using logistic regression, the probabilities describing the possible outcomes of a single trial are modelled using a logistic function. Training a model with this algorithm does not require high computational power. In a low dimensional dataset having enough training samples, logistic regression is less prone to overfitting. Due to its simple probabilistic interpretation, the training time of a logistic regression algorithm is far less than other complex algorithms. The drawback about Logistic Regression is that it does not give non-linear decision boundaries and assumes all predictors are independent of each other, which resulted in low prediction accuracies as shown in section 5.

3. Random Forest:

Random Forest uses Ensemble Learning technique. It creates many trees on the subset or sub-samples of data and combines the output of all the trees. The sub-sample size is always the same as the original input sample size, but the samples are drawn with replacement. This way, it reduces the overfitting problem of decision trees and reduces the variance. It also increases the accuracy. Random forest works well with categorical and continuous variables. One drawback of random forest is that it is slow in real time prediction and difficult to implement.

4. Support Vector Machines:

Support Vector Machines or SVM is a supervised machine learning algorithm which works very well in classifications of higher dimensional data into classes. Though SVM may seem slow when it comes to training large datasets, it is very fast when it comes to class prediction. Section 5 compares effectiveness of this algorithm with others for the dataset.

5. Gaussian Naïve Bayes:

Gaussian Naïve Bayes is a classification algorithm which is very efficient as it requires very less computational power. Naïve Bayes algorithms train very fast and are suitable for large datasets as compared to SVM and Neural Networks. Gaussian Naïve Bayes was not successful in giving satisfactory results as explained in section 5 because it fundamentally considers every feature to be independent and assumes that every feature makes equal contribution towards the target class.

6. Neural Networks:

Neural Networks is a mathematical model with approximation function which can be used for classification problems due to its flexibility. Neural Networks have shown strong results for non-linear data and are used in this project for the same reason. Neural Networks make fast predictions once trained.

4.2 Feature Importance Analysis

Experiments were undertaken to observe whether all the 24 sensors are in fact necessary for good prediction accuracy. Reducing the number of sensors will reduce the cost involved for navigation of the robot. The feature importance was found using the ANOVA-F method, which is generally used for numerical input variables and categorical output variables [4], as is the case here. As can be seen in Figure 2, some sensors contribute a lot towards prediction of the label, while some are not important at all and can be excluded. For the clockwise and anticlockwise data combined, it can be seen in Figure 3, that unlike the clockwise data, now a greater number of sensors have a significant contribution. The right as well as left sensors are important for the task of clockwise and anticlockwise navigation. Section 5 analyses whether a similar test accuracy as with 24 sensors can be achieved with a lower number of sensors.

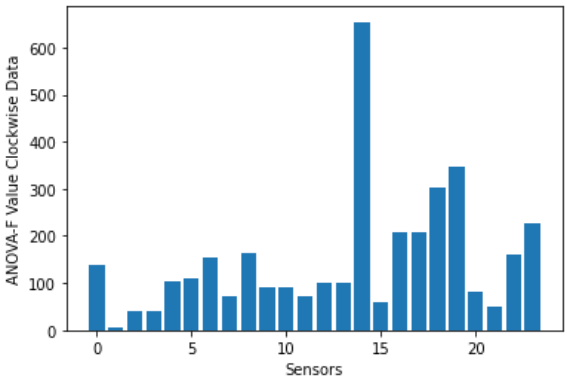

Figure 2: Feature importance for clockwise wall-following dataset

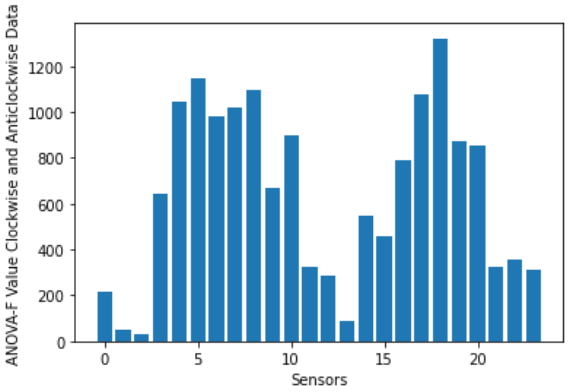

Figure 3: Feature importance for clockwise and anticlockwise wall-following combined dataset

4.3 Class Imbalance Reduction

For the clockwise and anticlockwise generated dataset, the 'Slight Left' labels are 11.55% and 'Slight Right' labels are 11.49% of the entire dataset, while the 'Move Forward' labels are 39.64% of the dataset. This class imbalance leads to lesser test accuracy and hence needs to be removed. Synthetic Minority Over-Sampling Technique (SMOTE) is used for reducing this class imbalance. It does not just create the same data points again for the class with lower labels. It uses the k nearest neighbors algorithm to find samples among the minority class that are similar, and creates synthetic data points at a random interval between a given data point and its nearest neighbor. This is advantageous over simply duplicating the minority data as some variation is intruduced in the resampled data to reduce overfitting. In Section 5, analysis of the improvement in the prediction accuracy of the machine learning algorithms because of reduction in class imbalance is undertaken.

5. Experiments

5.1 Clockwise Direction Wall-Following Data

Firstly, benchmarking of various machine learning models was performed for the clockwise direction dataset with 24 sensors (Refer Table 1). The results suggest that the label prediction task is non-linear in nature since logistic regression gives poor accuracy. This has been concluded in [1] as well. Furthermore, the results show that Gaussian NB does not give good accuracy on this dataset. This can be because of the dependence of the features, which is ignored by Gaussian NB. Hence, Logistic Regression and Gaussian NB are not included in further analysis. It is also observed that Random Forest gives very high train and test accuracy. This is because of the heuristic nature of the task.

Analysis of change in test accuracy with number of sensors was done after the benchmarking. The best 'n' features were selected and the train and test accuracy were found for these best 'n' sensors. Graphs were plotted of the best 'n' sensors, 'n' varying from 1 to 24, against the train and test accuracy, for different machine learning algorithms (refer Figure 4, Figure 5 and Figure 6). For Random Forest, a test accuracy of above 97% was obtained with just 5 sensors. The KNN graph shows that the best number of features is 4 for KNN, achieving the highest accuracy of 93.58%. As the number of features increases, more data is required for KNN to give good results, and hence the test accuracy is lower for 24 features. For SVM, the 'rbf' kernel was used. A linear kernel could not classify the data with good accuracy, with a maximum test accuracy of just 72.14%, clearly indicating the non-linearity of the data. Analysis of the graph with Neural Networks as the classifier shows that the test accuracy for 9 sensors is in fact a little greater than that for 24 sensors. The reason for this is that with 24 sensors, the model overfits due to high variance, and the training accuracy is high. This decreases the test accuracy. With lesser features, overfitting is avoided and though train accuracy reduces, test accuracy improves.

Table 2 shows the train and test accuracy results for 5, 7 and 9 best sensors for Random Forest, KNN, SVM and Neural Network.

Accuracy	Random Forest	KNN	Logistic Regression	Gaussian NB	SVM	Neural Network
Train	100.00	90.63	70.65	53.76	90.55	97.91
Test	99.45	84.79	67.68	52.35	89.13	92.36

Table 1: Clockwise direction wall-following dataset with 24 sensors

Number of Sensors	Accuracy	Random Forest	KNN	SVM	Neural Network
9	Train	100.00	93.30	84.84	95.10
9	Test	98.59	89.61	83.50	92.85
7	Train	100.00	94.68	84.89	94.60
7	Test	98.23	91.20	83.84	92.18
5	Train	100.00	95.36	83.50	92.56
5	Test	97.49	93.34	82.89	91.20

Table 2: Clockwise direction wall-following dataset accuracy comparisons for best ‘n’ sensors

	With Class Imbalance				Without Class Imbalance			
Accuracy	Random Forest	KNN	SVM	Neural Network	Random Forest	KNN	SVM	Neural Network
Train	100.00	89.30	88.03	97.41	100.00	94.33	91.96	98.41
Test	99.38	84.36	85.61	90.41	99.49	91.65	91.48	95.78

Table 3: Clockwise and Anticlockwise combined dataset with 24 sensors

		With Class Imbalance				Without Class Imbalance			
Number of Sensors	Accuracy	Random Forest	KNN	SVM	Neural Network	Random Forest	KNN	SVM	Neural Network
18	Train	100.00	88.92	87.05	94.80	100.00	94.46	90.19	97.40
18	Test	99.32	83.81	84.76	90.71	99.52	91.49	89.64	95.84
15	Train	100.00	89.34	85.74	94.87	100.00	93.60	85.22	94.65
15	Test	98.99	84.6	83.75	91.91	98.06	89.76	84.28	92.69
12	Train	100.00	88.93	79.61	89.75	100.00	93.70	83.65	92.88
12	Test	96.79	83.29	77.46	85.77	97.99	90.26	83.37	91.43
9	Train	100.00	90.31	77.17	88.58	100.00	94.07	81.37	91.90
9	Test	96.09	84.48	75.11	87.01	97.76	91.15	80.95	90.72

Table 4: Clockwise and Anticlockwise combined dataset accuracy comparison for best ‘n’ sensors

Training Report					Test Report				
	Precision	Recall	F1-score	Support		Precision	Recall	F1-score	Support
Slight Left	94.27	97.54	95.88	894	Slight Left	81.28	86.38	83.75	367
Sharp Right	99.19	98.59	98.89	1487	Sharp Right	93.65	94.26	93.95	610
Slight Right	96.25	94.59	95.41	868	Slight Right	87.80	85.75	86.76	386
Move Forward	97.03	96.71	96.87	3042	Move Forward	89.44	89.02	89.23	1284
Sharp Left	99.19	99.41	99.30	1347	Sharp Left	96.59	94.74	95.65	627
Accuracy			97.41	7638	Accuracy			90.41	3274
Macro Avg	97.18	97.37	97.27	7638	Macro Avg	89.75	90.03	89.87	3274
Weighted Avg	97.42	97.41	97.41	7638	Weighted Avg	90.48	90.41	90.43	3274

Table 5: Neural network clockwise and anticlockwise combined dataset classification report

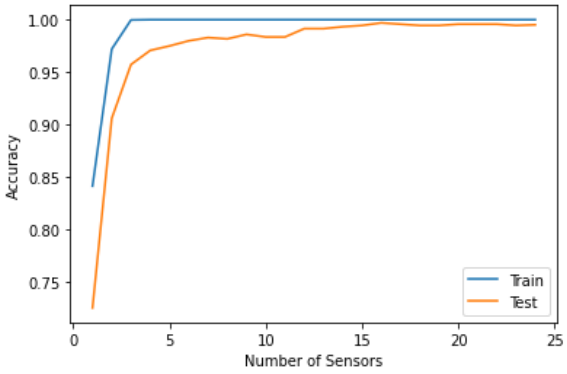

Figure 4: Random Forest clockwise dataset

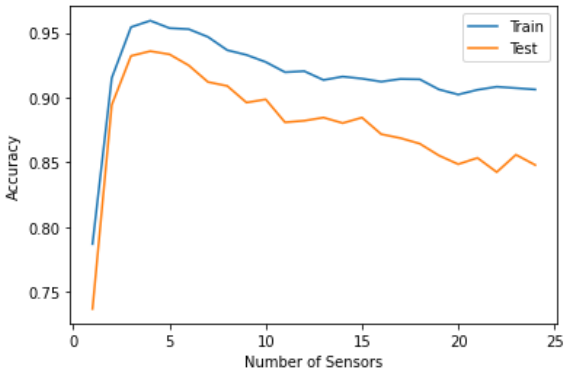

Figure 5: KNN clockwise dataset

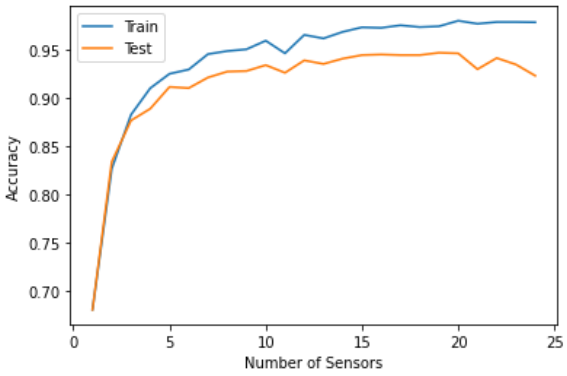

Figure 6: Neural Network clockwise dataset

5.2 Clockwise and Anticlockwise Direction Wall-Following Data

The label prediction results in clockwise and anticlockwise combined dataset are shown in Table 3. With class imbalance, compared to just clockwise data, the test accuracy does decrease for the all the models due to the added complexity of a new label and 100% more data, however it is not significant for KNN and Random Forest. Table 5 shows the training and test reports using neural network (multi-layer perceptron) for 24 sensors for the anticlockwise and clockwise combined dataset. It can be inferred that the test set F1-score of 'Slight Left' and 'Slight Right' labels is lower due to lesser number of samples available. We can see in Table 6 the confusion matrix for the dataset with imbalance, and in Table 7, the confusion matrix for the dataset after SMOTE is applied. The test F1-scores for 'Slight Left' and 'Slight Right' improved after balancing the number of samples in all classes, and the overall test accuracy also improved by 5.9% (refer Table 8).

Random Forest, KNN and SVM were also implemented for the balanced and unbalanced dataset. Table 3 shows the difference in accuracy for balanced and unbalanced dataset for each algorithm. It is inferred that test accuracy improves significantly for all algorithms and hence removing the class imbalance is a necessary step for this dataset.

For analysis of change in test accuracy with number of sensors, graphs are plotted similar to the clockwise data, with the best 'n' sensors varying from 1 to 24, against the train and test accuracy (refer Figure 7, Figure 8 and Figure 9). For the best 18 sensors, the test accuracy is within 2% to the test accuracy for 24 sensors for all the four machine learning algorithms used: Random Forest, KNN, SVM and Multi-Layer Perceptron (Neural Network). Table 4 shows the train and test accuracy results for 9, 12, 15 and 18 best sensors.

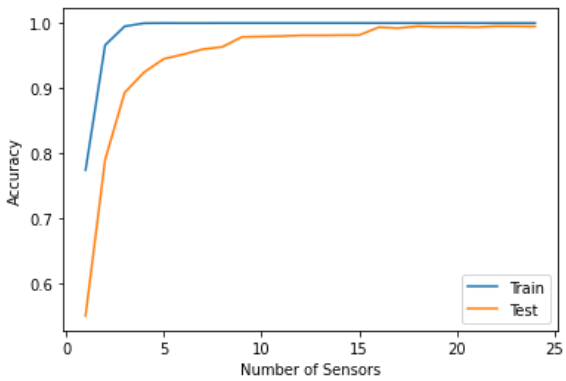

Figure 7: Random Forest clockwise and anticlockwise combined dataset

	Slight Left	Sharp Right	Slight Right	Move Forward	Sharp Rght
Slight Left	317	4	1	40	5
Sharp Right	6	575	2	24	3
Slight Right	2	5	331	47	1
Move Forward	60	29	40	1143	12
Sharp Left	5	1	3	24	594

Table 6: Confusion matrix with class imbalance for neural network

	Slight Left	Sharp Right	Slight Right	Move Forward	Sharp Rght
Slight Left	1316	1	0	16	1
Sharp Right	7	1241	2	21	1
Slight Right	1	0	1249	41	0
Move Forward	66	27	51	1154	12
Sharp Left	9	1	3	14	1255

Table 7: Confusion matrix without class imbalance for neural network

Label	Accuracy With Class Imbalance	Accuracy Without Class Imbalance
Slight Left	83.75	96.30
Sharp Right	93.95	97.64
Slight Right	86.76	96.22
Move Forward	89.23	90.30
Sharp Left	95.65	98.39
Total Accuracy	90.41	95.78

Table 8: Comparison of results with and without class imbalance for neural network

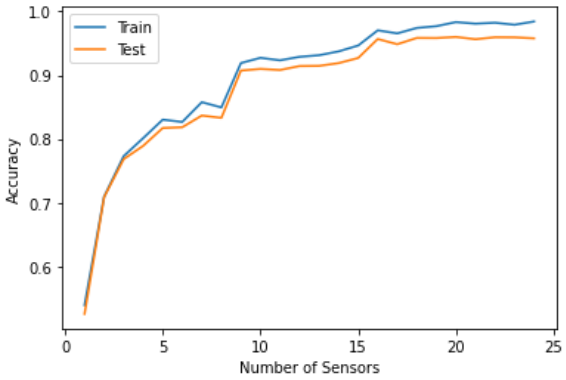

Figure 8: Neural Network clockwise and anticlockwise combined dataset

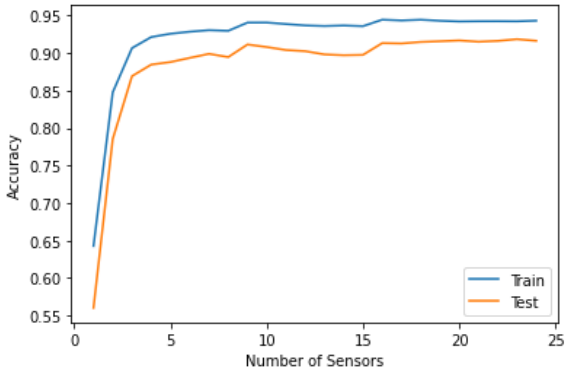

Figure 9: KNN clockwise and anticlockwise combined dataset

6. Conclusion

The available dataset for prediction of direction to be taken by the robot for clockwise wall-following given its current position was extended to an anticlockwise dataset. Random Forest, KNN, SVM, Logistic Regression, Gaussian NB and Neural Network algorithms were used and results were compared. Random Forest gave the highest accuracy while logistic regression and Gaussian NB failed to give good accuracy on this dataset. Class imbalance was reduced using SMOTE. The test accuracy improved by approximately 5% for each of KNN, SVM and Neural Networks because of reducing the class imbalance. The most important features (sensors) were found for the clockwise as well as the combined clockwise and anticlockwise dataset. For the clockwise dataset, sensors 15, 20, 19, 24, 18, 17, 9, 23, 7 were found to be the best 9 sensors (refer Figure 1). Using just these 9 sensors gave comparable test accuracy to the 24 sensors for all the algorithms. For the clockwise and anticlockwise combined dataset, using 18 sensors gave comparable test accuracy to using 24 sensors. Removal of the sensors not contributing to direction prediction will reduce the cost involved in further experiments done for similar tasks. Future work includes using machine learning for a more robust direction prediction for multiple tasks, given any set of obstacles and drastically varying proximity to the walls.

7. References

[1] Ananda L. Freire, Guilherme A. Barreto, Marcus Veloso and Antonio T. Varela, Short-Term Memory Mechanisms in Neural Network Learning of Robot Navigation Tasks: A Case Study, 6th Latin American Robotics Symposium, LARS 2009

[2] Classification Techniques for Wall-Following Robot Navigation: A Comparative Study, January 2019, Proceedings of the International Conference on Advanced Intelligent Systems and Informatics 2018 (pp.98-107)

[3] Wall following navigation task with mobile robot SCITOS-G5, https://www.kaggle.com/thehapyone/multiple-classifier-robot-movement-classification/data

[4] Jason Brownlee, How to Perform Feature Selection with Numerical Input Data, Machine Learning Mastery, https://machinelearningmastery.com/feature-selection-with-numerical-input-data/

8. Contributions

Aditya Joglekar: Feature importance, anticlockwise data generation, writing python code, class imbalance, final report: experiments and conclusion

Parth Malpathak: Feature importance, sensor positions and label generation for anticlockwise data, writing python code, final report: methods

Ketaki Tamhankar: Feature Importance, writing python code, anticlockwise data generation, final report: introduction, related work and data.