EMBRY-RIDDLE Aeronautical University

Graduate Research Project (GRP)

08-12-2022

Olfactory Odor Source Localization Robot Using Deep Neural Network

Mohammed Alajaji

Alajajm1@my.erau.edu

Acknowledgement

Firstly, I want to thank my parents Ali Alajaji and Latifah Alkhorayef for their uncounted support and unconditional love. Their help got me to what I was dreaming for a master's degree in electrical engineering from one of the well-known universities in United States, Embry-Riddle Aeronautical University.

Secondly, I would like to thank prof. Shuo Pang for supporting, encouraging, and advising me during working on this project. Prof. Pang has helped me to get better sight of the graduation research project (GRP) and the methods that I needed to utilize. Also, I want to thank Dr. Lingxiao Wang for his guidance, starting with explaining his dissertation, codes, and the collected data and ending with the real-world testing. Without his assistance, the project would be much more difficult.

Finally, I appreciate the Department of Electrical Engineering and Computer science faculty and staff for their continuous encouragement, inspiration, and support through my master's degree journey. Additionally, I feel deep grateful for my academic advisor Dr. Jianhua Liu for his supervision and unflinching guidance.

Contents

Acknowledgement i

List of Figures iii

Abstract iv

1. Introduction...1

2. Methodology...3

   2.1 Machine Learning - DNN Method..3

      2.1.1 Convolutional Neural Network (CNN)......................................5

      2.1.2 Recurrent Neural Network (RNN)..6

      2.1.3 Transfer Learning...7

   2.2 Data..8

   2.3 Ordo Robot Source Localization Robot 8

3. Project Development 10

   3.1 Data Preprocessing...11

   3.2 Model Training..13

4 Results & Conclusion 14

   4.1 Machine Leaning Models' Results...14

   4.2 Conclusion...19

References...20

List of Figures

FIGURE 1: ARTIFICIAL INTELLIGENCE, MACHINE LEARNING, AND DEEP LEARNING [2]. 3

FIGURE 2: RELATIONSHIP BETWEEN THE NETWORK, LAYERS, LOSS FUNCTION, AND OPTIMIZER [2].......... 4

FIGURE 3: THE STRUCTURE OF THE CNN MODEL [1]. 5

FIGURE 4: THE STRUCTURE OF RNN MODEL [1]. 7

FIGURE 5: TRANSFER LEARNING PROCESS [5]. 8

FIGURE 6: THE ODOR SOURCE LOCALIZATION MOBILE ROBOT. DESIGNED AND CONSTRUCTED BY DR. LINGXIAO WANG [1]. 9

FIGURE 7: SYSTEM CONFIGURATION, IT CONTAINS THREE MAIN COMPONENTS, INCLUDING MOBILE ROBOT, GROUND STATION, AND INDOOR LOCALIZATION. THE SOLID CONNECTION LINE INDICATES PHYSICAL CABLES, AND THE DOTTED CONNECTION LINE REPRESENTS WIRELESS LINK [1]. 9

FIGURE 8: COMPUTER SIMULATION DATA (COLLECTED USING MOTH-INSPIRED METHOD).......... 11

FIGURE 9: CNN MODEL LAYERS. 13

FIGURE 10: RNN MODEL LAYERS. 13

FIGURE 11: THE TRAINING AND VALIDATION METRICS AND LOSS FUNCTIONS OF IMPLEMENTED TRANSFER LEARNING THAT USED THE PRE-TRAINED RNN MODELS. (A) MOTH-INSPIRED METHOD DATA (B) BAYESIAN-BASED METHOD DATA (C) FUSION METHOD DATA. 15

FIGURE 12: THE TRAINING AND VALIDATION METRICS AND LOSS FUNCTIONS OF IMPLEMENTED CNN MODELS. (A) MOTH-INSPIRED METHOD DATA (B) BAYESIAN-BASED METHOD DATA (C) FUSION METHOD DATA. 16

FIGURE 13: THE TRAINING AND VALIDATION METRICS AND LOSS FUNCTIONS OF IMPLEMENTED TRANSFER LEARNING THAT USED THE PRE-TRAINED RNN MODELS. (A) MOTH-INSPIRED METHOD DATA (B) BAYESIAN-BASED METHOD DATA (C) FUSION METHOD DATA. 17

FIGURE 14: THE TRAINING AND VALIDATION METRICS AND LOSS FUNCTIONS OF IMPLEMENTED RNN MODELS. (A) MOTH-INSPIRED METHOD DATA (B) BAYESIAN-BASED METHOD DATA (C) FUSION METHOD DATA. 18

FIGURE 15: MATLAB ERROR MESSAGE. 19

Abstract

In recent years, robotic odor source localization has become a popular research field that enables a mobile robot or an autonomous vehicle to determine/localize an odor source in unknown environments. There are various algorithms that have been proposed to guide the robot to detect and trace a plume and localize the odor source effectively and efficiently. The pros and cons of these algorithms depend mainly on the environment (laminar or turbulent), computational cost, and the available data. In previous work, some of these algorithms have shown successful results in one environment and poor results in other environments. For instance, a moth-inspired algorithm runs accurately in a laminar environment, while it barely localizes the chemical source in a turbulent environment. In this project, we will apply the machine learning methods using deep neural networks, namely the convolutional neural network (CNN) and the recurrent neural network (RNN) methods, to improve the search efficiency with lower computational cost and greater accuracy. In this project, since the collected real-world data is inadequate for training the CNN and RNN models, transfer learning will be implemented to tackle this issue. CNN and RNN models will be trained on computer simulated data, and the knowledge gained will be transferred to a model with inputted the real-world data. Transfer learning technique results in much better performance than the trained CNN and RNN models on real data in terms of generalization.

1. Introduction

Recently, robotics has been studied intensely in many different sectors around the world. In most applications, robotics mimics human's or animals' behaviors that can be used in dangerous situations or that human or animal are not able to do. The main goal of designing robotics is to achieve certain capabilities that can help or assist human. To do that, robotics needs to perceive the world as human does, sight, smell, touch, taste, and hearing. Most of the human's senses have been implemented in robotics, and they showed great results. However, Olfaction has not been sufficiently exploited in mobile robotics yet as other senses. Moreover, it is not the most vital sense for humans, but it is a valuable source of information that might alarm humans about the situations around them. For instance, people will be worried if there is an odor that associated to gas in the house and starts looking for the causes of that odor that might save human lives.

Dr. Lingxiao Wang, a visiting assistant professor at Embry-Riddle Aeronautical University, designed an olfactory source localization robot [1] that finds and traces a plum, and localizes the source. He implemented various methods (traditional and engineering) and tested them in real-world applications in different environments. Some of the methods showed fine results in one environment and bad in another environments, and other methods did not show any good results in both environments. Two types of deep neural network (DNN) method, namely convolutional neural network (CNN) and FNN (feedforward neural network), were implemented and tested in [1]. CNN showed greater performance results than FNN which missed almost all trial tests. In Table 1, CNN successfully localized the odor source in almost all tests in 5 different environments while FNN failed in 12 tests out of 15. In fact, the FNN structure can hardly learn an effective plum tracing method which is clearly not suitable for this kind of application. Comparing CNN method with two expert methods, CNN outperformed the moth-inspired method in terms of the success rate (93% vs 67%), and it achieved a shorter averaged search time than the Bayesian-inference method (112.8 sec vs 127.3 sec). Moreover, the CNN method failed in one test out of 15 (the search time ran-out after 400 sec) while Bayesian-inference method succeeded to localize the odor source in all tests before the run-out time. Nevertheless, CNN is preferable than Bayesian-inference for two reasons: 1) the query time of the machine learning model is predictable and unaffected by the size of the search area; 2) it can learn other successful navigation methods from demonstrations with explicating the specific searching algorithms.

Table 1: Statistical Results of Repeated Tests and the comparison of Different Navigation Methods [1].

	Total Tests	Successful Tests	Success Rate	Averaged Search Time (s)
Moth-inspired Method	15	10	67%	111.3
Bayesian-inference Method	15	15	100%	127.3
The Proposed FNN	15	3	20%	106.0
The Proposed CNN	15	14	93%	112.8

This project is a continuation of the previous work of Dr. Wang to try to improve the performance of the odor source localization algorithm to enhance the search time and the accuracy of localizing the odor source. To that purpose, two different deep neural network methods is implemented, namely, CNN and RNN. CNN showed an excellent performance in [1], while RNN is powerful for modeling sequence data such as time series. The size of the dataset and precision of the collected data are essential for the success of deep neural network. only about 120 real data was collected, which is not enough to train a model. Therefore, a computer simulation was conducted using three expert methods moth-inspired, Bayesian-inference, and fusion methods, and collecting around 6000 odor source localization trails for each method. In this case, transfer learning is needed to transfer the gained knowledge using the computer simulation data to another model using real-world data. The model evaluations show that transfer learning improved the generalization of the DNN model, and it helps to avoid overfitting or underfitting problems.

2. Methodology

The research and development of the odor source localization robot in recent years have shown different methods that can be implemented to determine the odor (or chemical) source effectively and efficiently. The key to correctly determine an odor source is employing the right olfactory-based navigation algorithms in the right environment. There are many algorithms methods that have implemented and tested in previous works. In [1], it developed different method such as moth-inspired method, Bayesian-inference method, and machine learning - deep neural network methods. Lately, DNN shows great performance in many real-world applications like voice recognition, image classification, and robotics.

This section provides survey of two DNN methods (CNN and RNN) of machine learning, as well as transfer learning. Moreover, it presents the three methods that used to collect the training datasets of the OSL robot. Also, it shows the actual designed of the OSL robot that is used to implement the real-world experiments.

2.1 Machine Learning - DNN Method

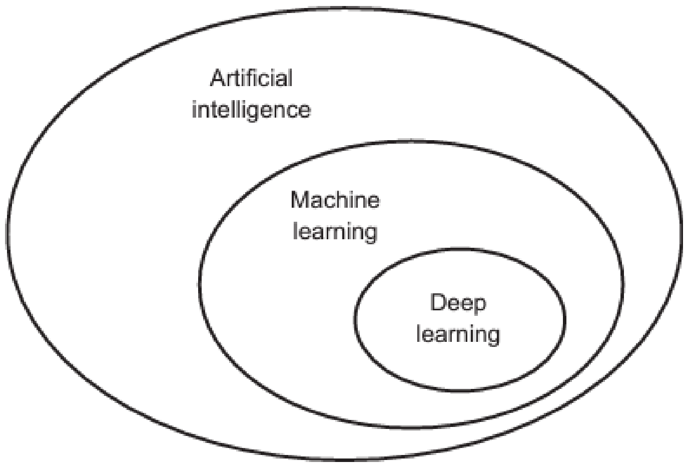


Figure 1: Artificial intelligence, machine learning, and deep learning [2].

Machine learning has become the most popular and successful subfield of artificial intelligence due to the advance and the faster hardware, and the massive available data. Basically, it is closely related to mathematical statistics, but machine learning is engineering oriented and deals with large, complex dataset. Machine learning has many approaches that have been useful to many applications, and they are different in the sense of their way to approach the targets. It divides into four common categories as they are defined in [2]: 1) supervised learning which consists of learning to map input data to known targets, given a set of examples (often annotated by humans), and it's the most common case; 2) unsupervised learning which consists of finding interesting transformations of the input data without the help of any target; 3) self-supervised learning which is supervised learning without human-annotated labels; 4) Reinforcement learning which is based on rewarding and/or punishing, it's an agent receives information about its environment and learns to choose actions that will maximize rewarding and minimize punishment.

Deep learning is the faster-growing and the most significant subfield of machine learning see Figure 1, and almost all deep learning applications belong to supervised learning category. Deep learning is "a mathematical framework for learning representations from data" [2]. In these days, it involves tens or even hundreds of successive layers to increasingly extract meaningful representations from some input data.

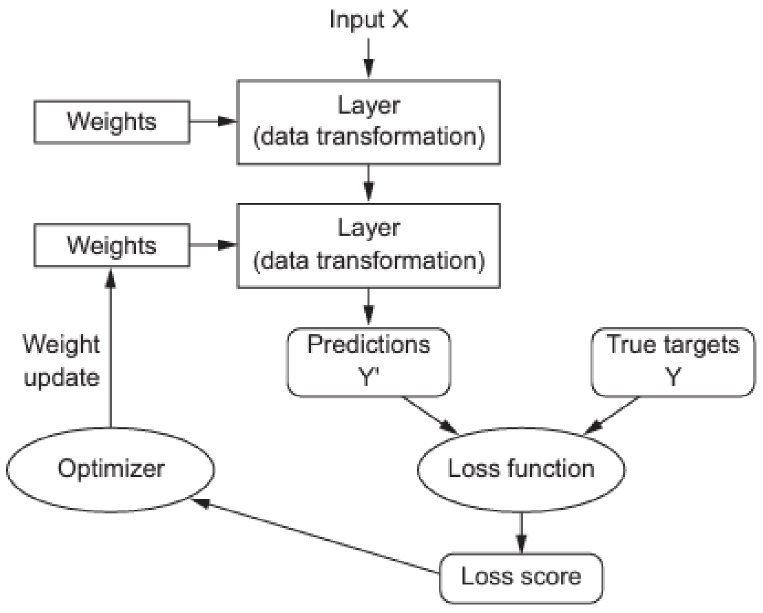


Figure 2: Relationship between the network, layers, loss function, and optimizer [2].

The structure of deep learning is a bunch of layers of representation that stacked on top of each other which are learned via models called neural networks. As in Figure 2, dataset "Input X" is fed into the deep learning model layers; each layer is kind of data transformation that learns by exposure to examples. A layer is parameterized by its weight, which is adjusted in each iteration until finding a set of values of all layers in a network that correctly map example inputs to their associated targets. It is randomly assigned values initially. *Loss function* is the measurement that measures the accuracy of the network output by computing the distance score of the predictions of the network and the true targets. The score uses as a feedback signal to tune the weights values through the *Optimizer* in a direction that will lower the *loss score*. Deep learning has different models that serve various applications in different sectors nowadays. For the purpose of this project, we chose two types of deep leaning the CNN and RNN.

2.1.1 Convolutional Neural Network (CNN)

CNN has shown astonishing outcomes in many applications such as medical research, image processing, and robotics. CNN is a model for processing data that has a grid pattern, and it is a mathematical construct that includes convolutional layers and other layers that depend on the purpose of the model and the nature of the dataset. CNN is constructed to learn spatial hierarchies of features automatically and adaptively through a backpropagation algorithm. In this project, CNN is implemented to produce the robot commands based on time series data. Figure 3 shows the structure of the CNN to train the model using computer simulation dataset.

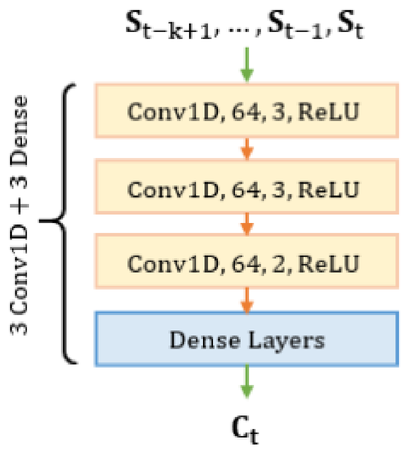


Figure 3: The Structure of the CNN model [1].

The first three layers are convolutional neural network layers ‘Conv1D’, which all have the same specifications, filter size = 64, kernel size = 3, and activation function = ReLU. The last block is ‘Dense Layers’ which contains 3 consecutive dense layers with 128, 128, and 2 filters’ sizes, respectively. The CNN training model is considered as complete if the training epoch reaches the limit (20 epochs) or the validation error does not improve in 2 consecutive epochs. The mathematical representation of the CNN model can be presented by [1]:

$$C_t = \mathcal{F}_{\theta_{CNN}}\left(S_{(t-k)\sim t}, C_{(t-k)\sim(t-1)}\right)$$

Where $\mathcal{F}_{\theta_{CNN}}$ represents the parameter vector of the CNN model, t is the time, k is the length of recording window, $S_{(t-k)\sim t}$ ($S_{(t-k)\sim t} = (S_{t-k}, S_{t-k+1}, \dots, S_t)$) is a vector of sensor data from time $t-k$ to t, and $C_{(t-k)\sim(t-1)}$ ($C_{(t-k)\sim(t-1)} = (C_{t-k}, C_{t-k+1}, \dots, C_{t-1})$ is a vector of historical robot commands from time $t-k$ to $t-1$ [1].

2.1.2 Recurrent Neural Network (RNN)

RNN is a class of neural networks that is powerful for modeling sequence data such as time series or natural language [3]. It is more capable of learning long-term dependencies (especially in sequence prediction problems) when a special kind of RNN long-short term memory (LSTM) is applied. LSTM, basically, is an extension of RNN that attempt to replicate the way the human brain functions and find the underlying relationships in the given sequential data. LSTM assigns weight to support RNN to figure out what data needs to be forgotten, and what it needs to be saved for later reference. LSTM method has been applied in many real-world applications for its great performance. Similar to CNN, RNN (or LSTM) is employed to generate robot commands to trace a plume and localize the chemical source. As presented in Figure 4, two LSTM layers on the top, and 3 consecutive dense layers in the bottom. Each LSTM layers is specified as units =256 (a dimensionality of the output space [4]), and return_sequences = true (which determines whether to return the last output, in the output sequence, or the full sequence [4]). The filter sizes in the dense layers are 64, 64, and 2, respectively.

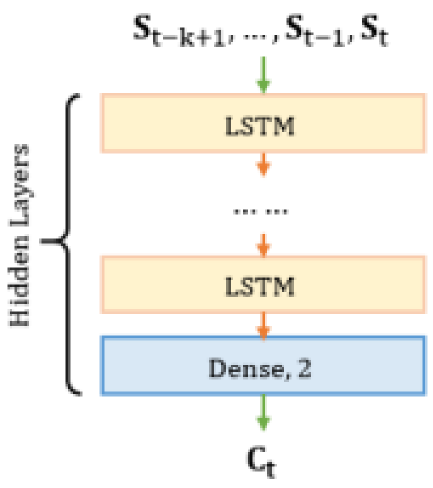


Figure 4: The structure of RNN model [1].

2.1.3 Transfer Learning

Machine learning, especially DNN, needs specific and very large amount of data to get a better predictive performance of a trained model, and to avoid overfitting and underfitting problems. As most real-world situations, the available data is not enough to train a DNN model, especially the uncommon applications as the case of the OSL robot. The OSL robot has only 120 real data, which is not enough to train DNN model. On the other hand, it has 6000 data that was collected using computer simulation. Therefore, transfer learning, which is a reuse of knowledge gained from a pre-trained model and transfer it to a new task [5] see Figure 5, is a solution to solve the lack of real data in OSL robot application. In this project, the transfer learning is used to transfer the learned knowledge from the DNN (CNN or RNN) model that is supplied with computer simulation data to another model that fed with real data. With transfer learning, we basically save time and cost of not conducting real-world experiments to collect enough data that covers versatile searching situations to train DNN model.

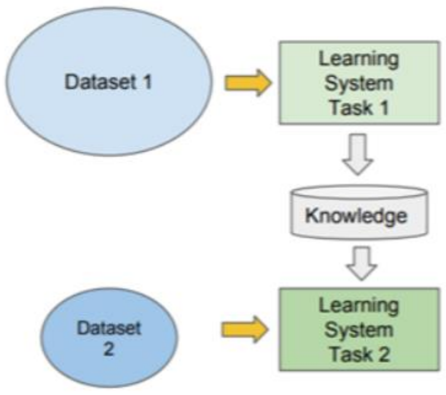


Figure 5: Transfer learning process [6].

2.2 Data

The data is the main part of the success of any machine learning model, it needs to be collected and preprocessed precisely and correctly. Collecting massive data can lower the estimation variance which leads to better predictive performance. Therefore, more data increases the probability that it contains useful information that helps the model prediction. Aforementioned, the available datasets for OSL robot are 3 real datasets with size 120×12 each and 3 computer simulation datasets with size 6000×14 (all datasets were collected by Dr. Wang). The real datasets were collected by conducting real-world experiments using three different algorithms, moth-inspired, Bayesian-inference, and fusion methods. While the computer simulation datasets were used a computer to simulate the actions of the OSL robot to collect as much information as possible in different search situations and using the same three algorithms that used in collecting real-world datasets. Moth-inspired method is commanding the robot to mimic successful odor finding, tracing and localization behaviors of the moth animals. it’s inspired by mate-seeking behavior of male moths. The behavior is tracking pheromone plumes which are emitted by female moths over a long distance and through many obstacles such as trees, and it ends with locating the female. On the other hand, Bayesian-inference is an engineering-based method, which it uses measured wind data and a Gaussian plume dispersion model to determine the position of the odor source in reverse. The fusion method is a combination of the two methods moth-inspired and Bayesian-inference.

2.3 Ordo Robot Source Localization Robot

In this project, the designed mobile robot (by Dr. Wang) as shown in Figure 6 will be used to conduct real-world tests. The robot is equipped with two sensors, a chemical sensor (MQ-3, Waveshare), and an anemometer sensor (SindSonic, Gill Instruments), which are both connected to one of the microcontrollers (Arudino Mega, Arduino) for fetching sensor measurements. The other microcontroller controls the two robot's wheels motors via a motor driver (Sabertooth, Dimension Engineering).

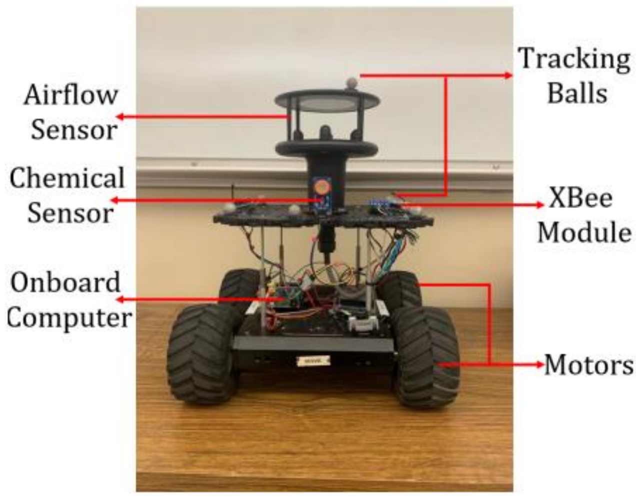


Figure 6: The odor source localization mobile robot. designed and constructed by Dr. Lingxiao Wang [1].

The configuration system includes ground station and indoor localization system beside the mobile robot, which is clearly demonstrated in Figure 7.

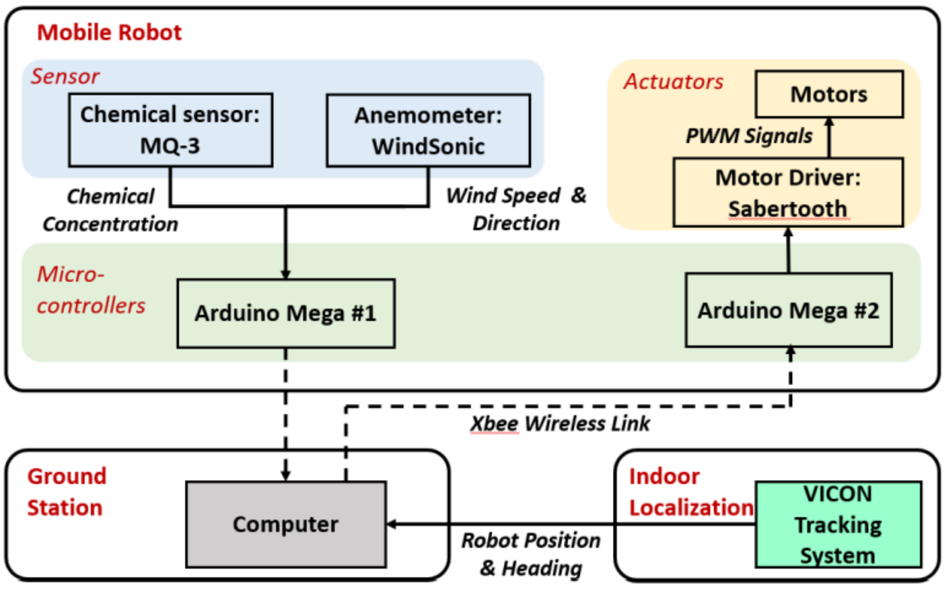


Figure 7: system Configuration, it contains three main components, including mobile robot, ground station, and indoor localization. The solid connection line indicates physical cables, and the dotted connection line represents wireless link [1].

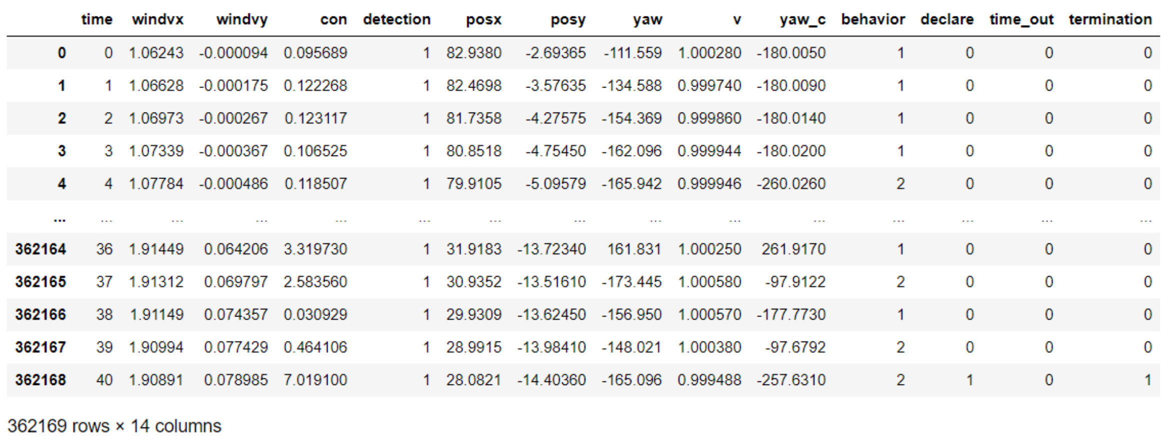
The two mounted microcontrollers on the mobile robot can communicate with the ground station via wireless communication network (Xbee, Digital international) to send sensors measurements and receive the robot heading commands. The robot heading commands are the output of the navigation algorithm that is in the ground station (a PC station). The indoor localization has the Vicon tracking system (Vicon Inc.) to determine indoor position, send robot positions, and orientation to the ground station via an Ethernet cable (for more details, see [1]).

3. Project Development

The core idea of this project is to train intelligent ML models to provide accurate commands to the OSL robot to locate the chemical source correctly in less than 400 seconds. There are many ML algorithms that can be implemented to the plume tracing robot, but we selected two deep neural networks, including convolutional neural network (CNN) and recurrent neural network (RNN). The procedures of training CNN and RNN models are almost the same, while the structure of the models is a little bit different (it is described in the previous section). So, in this section, we will start with the data preprocessing processes followed by model training.

3.1 Data Preprocessing

Data preprocessing is a data mining technique used to turn the raw data into a format that is practical and effective. In ML, data preprocessing is very important part to convert the raw data into a well-readable format. it helps to achieve the purposes of training the ML model and enhance the prediction performance. Data preprocessing consists of dropping missing, inconsistent, and noisy value, combining two or more features, reordering the data, normalization, and so on. In this project, we have six different data that were collected in real-world experiments and in computer simulations using three methods, namely, moth-inspired, engineering-based, and fusion methods. Each data of the collecting data has slightly different than others such as the letters case (majuscule and minuscule), different features' names, or different ordering. Since the 6 datasets have slight difference between each other; therefore, I will walk you through the computer simulation data – (moth-inspired method).



	time	windvx	windvy	con	detection	posx	posy	yaw	v	yaw_c	behavior	declare	time_out	termination
0	0	1.06243	-0.000094	0.095689	1	82.9380	-2.69365	-111.559	1.000280	-180.0050	1	0	0	0
1	1	1.06628	-0.000175	0.122268	1	82.4698	-3.57635	-134.588	0.999740	-180.0090	1	0	0	0
2	2	1.06973	-0.000267	0.123117	1	81.7358	-4.27575	-154.369	0.999860	-180.0140	1	0	0	0
3	3	1.07339	-0.000367	0.106525	1	80.8518	-4.75450	-162.096	0.999944	-180.0200	1	0	0	0
4	4	1.07784	-0.000486	0.118507	1	79.9105	-5.09579	-165.942	0.999946	-260.0260	2	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
362164	36	1.91449	0.064206	3.319730	1	31.9183	-13.72340	161.831	1.000250	261.9170	1	0	0	0
362165	37	1.91312	0.069797	2.583560	1	30.9352	-13.51610	-173.445	1.000580	-97.9122	2	0	0	0
362166	38	1.91149	0.074357	0.030929	1	29.9309	-13.62450	-156.950	1.000570	-177.7730	1	0	0	0
362167	39	1.90994	0.077429	0.464106	1	28.9915	-13.98410	-148.021	1.000380	-97.6792	2	0	0	0
362168	40	1.90891	0.078985	7.019100	1	28.0821	-14.40360	-165.096	0.999488	-257.6310	2	1	0	1

362169 rows × 14 columns

Figure 8: Computer simulation data (collected using moth-inspired method).

The raw dataset of the computer simulation- moth-inspired data as show in the Figure 8 has 6000 of simulated OSL robot trials. It contains every movement that the robot occurs in every second. The trial ends when the robot declare the odor source, or the time exceeds 400 seconds (which is the maximum for each trial). The data has 13 features that are recorded based on sensors' reading. The features meaning is described in the Table 2. The data preprocessing steps are described as follows:

Table 2: Data's features description.

time (s)	It records the time in seconds from trail starts 0s until it ends 400s ≤.
windvx (m/s)	Wind velocity in x-axis direction.
windvy (m/s)	Wind velocity in y-axis direction.

con (mmpv)	Chemical concentration in the robot position
detection (1 or 0)	1 when detecting a plume and 0 otherwise.
posx (m)	Robot position in x-axis
posy (m)	Robot position in y-axis
yaw (deg)	Robot heading angle
v (m/s)	Robot speed
yaw_c (deg)	Robot heading command
behavior	The robot's behaviors (finding, tracing, tracking, …, etc.)
declare (1 or 0)	If the robot reaches the odor source, then declare = 1 and 0 otherwise.
time_out (1 or 0)	1 when the time reaches 400 s
termination (1 or 0)	Terminate the robot in two cases declare the odor source or the trials time = 400 s

1- Drop all the empty cells to avoid any biases during training the ML model.
2- Drop ineffective features which are 'time', 'posx', 'posy', 'yaw', 'v', 'behavior', 'declare', 'time_out', and 'termination'.
3- Convert the robot heading commands 'yaw_c' negative values into positive by adding 360° to each element.
4- Convert the robot heading commands 'yaw_c' from degree to radians using the below equation:

$$yaw_c = yaw_c \times \left(\frac{\pi}{180}\right)$$

5- Drop 'yaw_c' column and add new two features 'Cx' and 'Cy', which determine the robot heading commands in x-axis and y- axis, respectively.

$$Cx = \cos(yaw_c)$$
$$Cy = \sin(yaw_c)$$

6- Normalize the whole data using the mean and standard deviation (std) as in:

$$data = \frac{data - mean}{std}$$

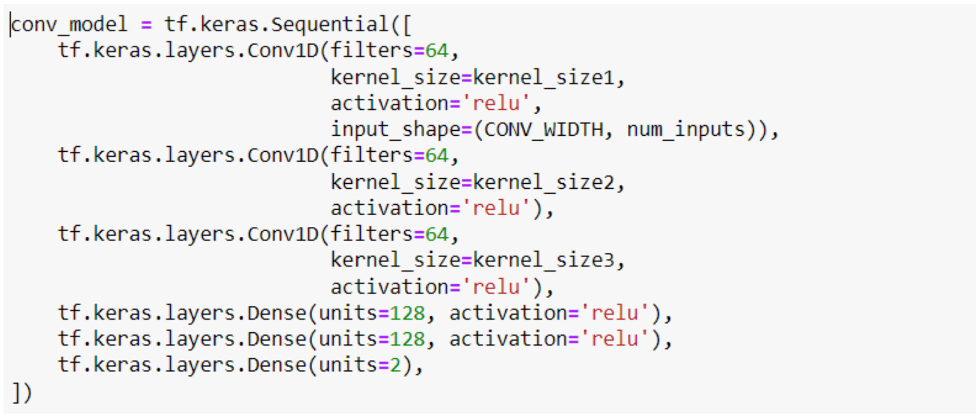
3.2 Model Training

Now, the dataset is ready to be fed into DNN model. Before doing that, the dataset needs to be split into three sets, training, validation, and test sets. The training data set is the samples that are used to create the model, whereas the validation and test data set are used to fine tune and evaluate the model. In this project, 80% of the data is a training data set, while the validation and test sets are 10% each of the data. The followings are an explanation of the model training processes:

1- Define a time series window generator that can handle the indexes and offsets and split windows of features into (feature, labels) pairs [7].

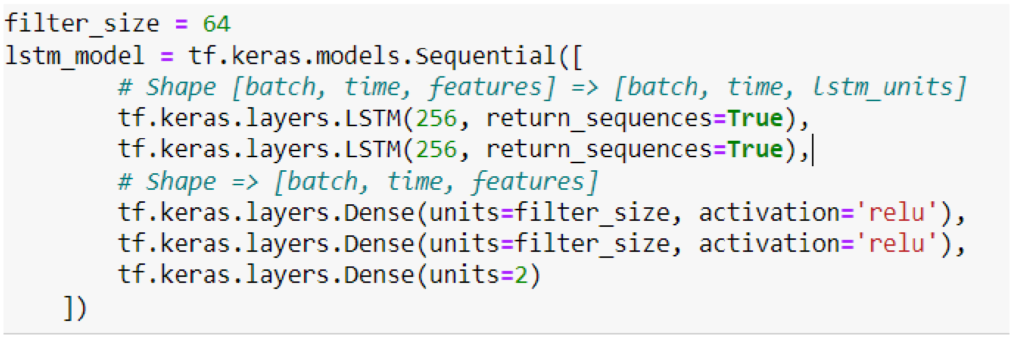
2- Define a compile, which defines the loss function, optimizer, and the metrics. In all models, we use the mean square error as a loss function, Adam optimizer as an optimizer, and mean absolute error as metrics.

3- Define the model. CNN model has three convolutional neural network layers, two dense layers as shown in Figure 9, while RNN model has two LSTM layers, and three dense layers as seen in Figure 10.



```
conv_model = tf.keras.Sequential([
    tf.keras.layers.Conv1D(filters=64,
                           kernel_size=kernel_size1,
                           activation='relu',
                           input_shape=(CONV_WIDTH, num_inputs)),
    tf.keras.layers.Conv1D(filters=64,
                           kernel_size=kernel_size2,
                           activation='relu'),
    tf.keras.layers.Conv1D(filters=64,
                           kernel_size=kernel_size3,
                           activation='relu'),
    tf.keras.layers.Dense(units=128, activation='relu'),
    tf.keras.layers.Dense(units=128, activation='relu'),
    tf.keras.layers.Dense(units=2),
])
```

Figure 9: CNN model layers.



```
filter_size = 64
lstm_model = tf.keras.models.Sequential([
        # Shape [batch, time, features] => [batch, time, lstm_units]
        tf.keras.layers.LSTM(256, return_sequences=True),
        tf.keras.layers.LSTM(256, return_sequences=True),
        # Shape => [batch, time, features]
        tf.keras.layers.Dense(units=filter_size, activation='relu'),
        tf.keras.layers.Dense(units=filter_size, activation='relu'),
        tf.keras.layers.Dense(units=2)
    ])
```

Figure 10: RNN model layers.

4- Now, the model is ready to be trained with 20 epochs (it's determined based on previous Dr. Wang's experiments). The number of epochs determines how many times the learning algorithm will run over the whole training data set.
5- Implement transfer learning:
    a. Freeze the trained CNN and RNN models (used computer simulation data) layers.
    b. Add new trainable layers (2 dense layers).
    c. Train the new layers using real data.
    d. Fine-tune the model.

4 Results & Conclusion

4.1 Machine Leaning Models' Results

Transfer learning is a very effective for training models that have a small amount of data using the knowledge of the pre-trained model. The results of using transfer learning technique in OSL robot show much better model generalization comparing with the CNN and RNN models' results. Generalization in machine learning refers to the model's ability to adapt properly to new, previously, and unseen data. The model is overfitting or overfitting the training data if it is not generalized well. Aforementioned, the transferring learning reuses the knowledge of the trained CNN and RNN models that use the computer simulation data. On the other hand, another CNN and RNN models are trained on the real-world data, and they show very less generalization.

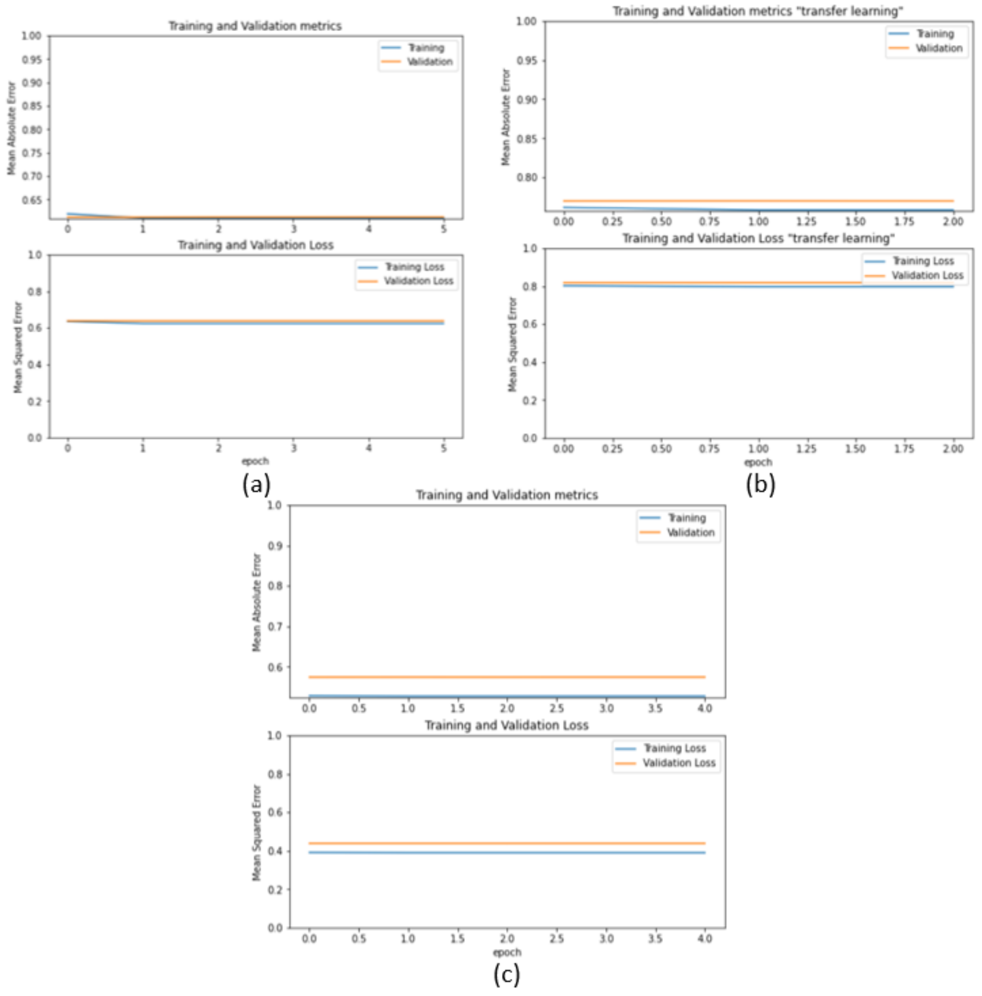


Figure 11: The training and validation metrics and loss functions of implemented transfer learning that used the pre-trained RNN models. (a) moth-inspired method data (b) Bayesian-based method data (c) fusion method data.

Figure 11 and Figure 13 show the metrics - mean absolute errors (MAEs) and the loss function – mean square errors (MSEs) of implementing transfer learning technique with using three different datasets. Generally, the training and validations metrics and loss functions are almost topical to each other, which means that the models are generalized well. In contrast, Figure 12 and Figure 14 show the metrics and loss functions of CNNs and RNNs models on the real-world datasets, and it indicates that the models are not generalized to applied in all OSL robot environments.

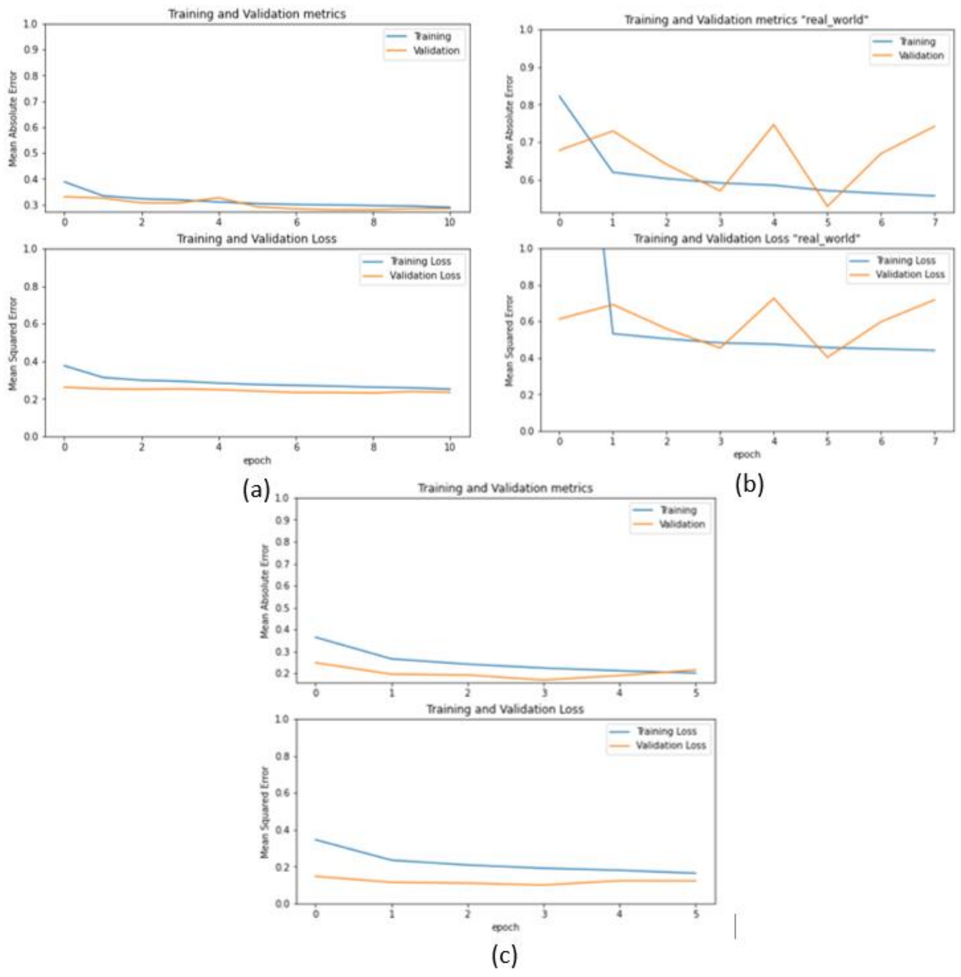


Figure 12: The training and validation metrics and loss functions of implemented CNN models. (a) moth-inspired method data (b) Bayesian-based method data (c) fusion method data.

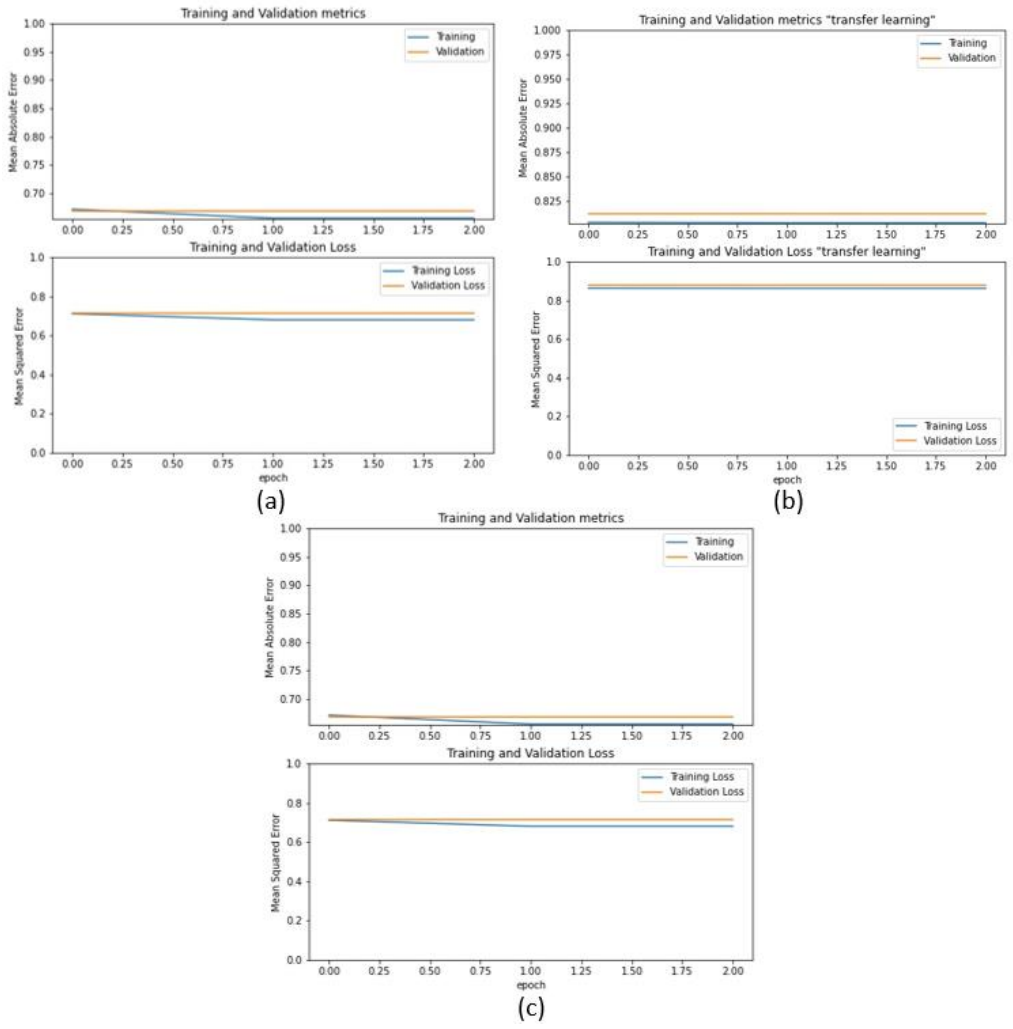


Figure 13: The training and validation metrics and loss functions of implemented transfer learning that used the pre-trained RNN models. (a) moth-inspired method data (b) Bayesian-based method data (c) fusion method data.

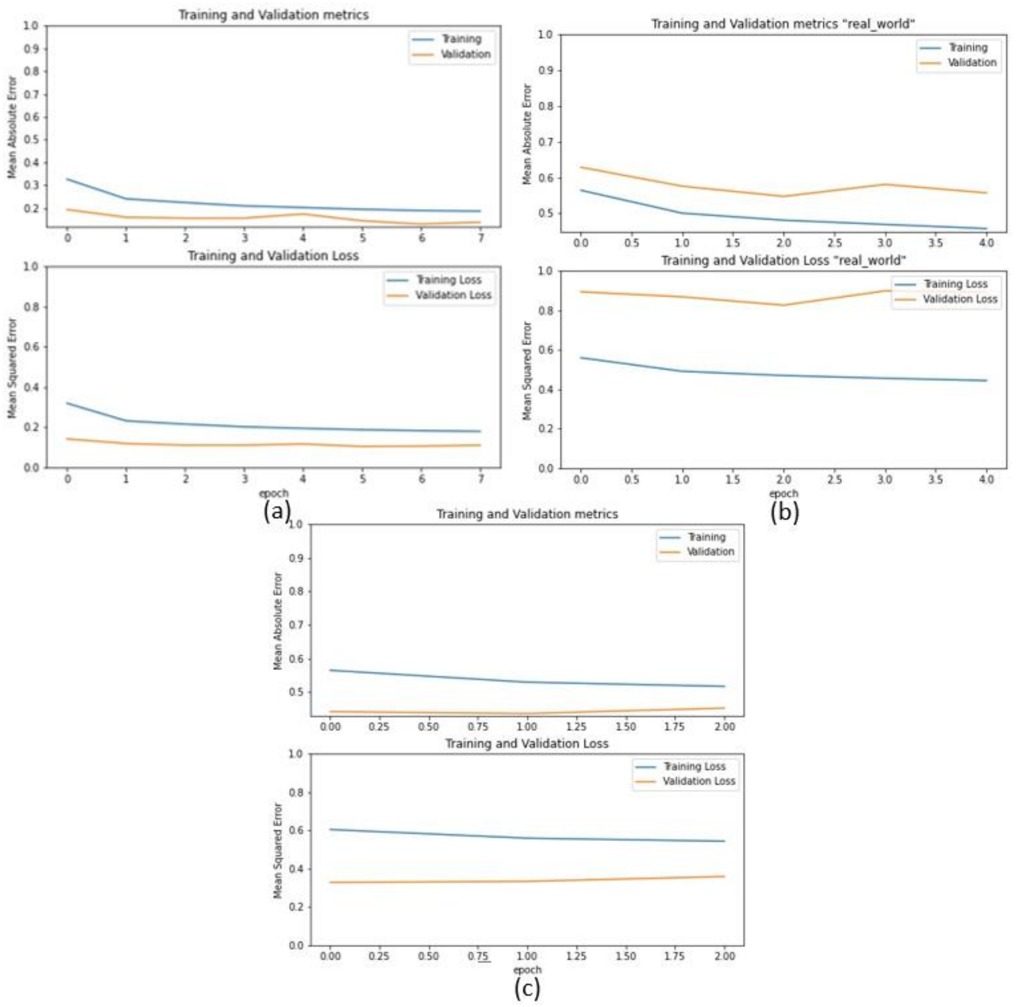
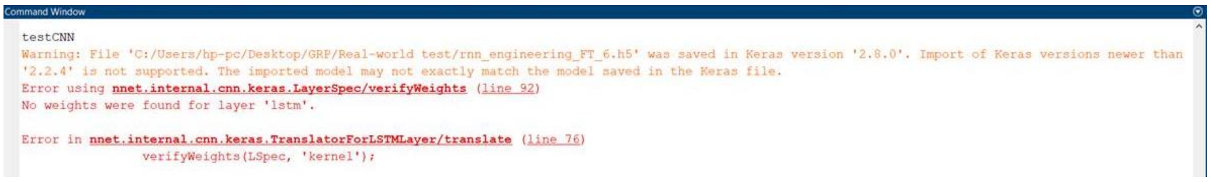


Figure 14: The training and validation metrics and loss functions of implemented RNN models. (a) moth-inspired method data (b) Bayesian-based method data (c) fusion method data.

4.2 Conclusion

The OSL robot is a very challenging problem that needs to take many variables into account such as the environment around the robot, and the search algorithms. The implementation of the CNN and RNN to produce the robot commands to detect and trace the plum, and find the source shows a very weak generalization since the available real-world data has very small samples, only 120. Therefore, Transfer learning was a solution to train a machine learning model using the knowledge of a pre-trained model that used similar data to the real-world data. It can be observed that transfer learning achieves better performance in terms of model generalization that supports the OSL robot to accomplish its goal of finding the chemical source.

One of the deliverables that I was planning to deliver in this project was the real-world experiments. I have tried many times with the help of Dr. Wang to test the algorithms in real-world experiments, but we had an issue of calling the DNN model (.h5 file) in the OSL robot code in MATLAB. The issue is the MATLAB was not able to find the weight for "Conv1D" and "LSTM" layers as shown in the Figure 15. Therefore, I ran out of the time to try other possibilities to solve this problem. However, the transfer learning output shows much better performance than other algorithms.



```
testCNN
Warning: File 'C:/Users/hp-pc/Desktop/GRP/Real-world test/rnn_engineering_FT_6.h5' was saved in Keras version '2.8.0'. Import of Keras versions newer than
'2.2.4' is not supported. The imported model may not exactly match the model saved in the Keras file.
Error using nnet.internal.cnn.keras.LayerSpec/verifyWeights (line 92)
No weights were found for layer 'lstm'.

Error in nnet.internal.cnn.keras.TranslatorForLSTMLayer/translate (line 76)
            verifyWeights(LSpec, 'kernel');
```

Figure 15: MATLAB error message.

References

[1] L. Wang, "Robotic Olfactory-Based Navigation with Mobile Robots," 2021.

[2] F. Chollet, Deep Learning with Python, Shelter Island, NY: Manning Publications Co., 2018.

[3] "TensorFlow," Google, 10 01 2022. [Online]. Available: https://www.tensorflow.org/guide/keras/rnn.

[4] "TensorFlow," Google, 28 06 2022. [Online]. Available: https://www.tensorflow.org/api_docs/python/tf/keras/layers/LSTM.

[5] P. Sharma, "Understanding Transfer Learning For Deep Learning," *Analytics Vidhya,* 2021.

[6] D. Sarkar, "Towards Data Science," 15 10 2018. [Online]. Available: https://towardsdatascience.com/a-comprehensive-hands-on-guide-to-transfer-learning-with-real-world-applications-in-deep-learning-212bf3b2f27a.

[7] "TensorFlow," Google, 16 06 2022. [Online]. Available: https://www.tensorflow.org/tutorials/structured_data/time_series.