Ship Navigation and Fuel Profiling based on Noon Report using Neural Network Generative Modeling

J Hadi[1], Z Y Tay[1], D Konovessis[2]

[1] Singapore Institute of Technology, 10 Dover Drive, Singapore 138683
[2] Strathclyde University, 16 Richmond Street, Glasgow, G11XQ, UK

Abstract. Harbor craft historical routes contain valuable information on how the experienced crews navigate around the known waters while performing jobs. The noon report logs each job timeframe which can be used to segregate the time-series positional data as routes. Other information from the noon report such as fuel consumption could be associated with a particular job as well. This paper offers a solution to encompass crew navigational experience into neural network models. The variational autoencoder, which is a generative model, can capture the routes into a knowledge base model. The same variational autoencoder is also able to train other neural networks to make predictions of route and fuel consumption based on job metadata (I.e., job duration, activity area, and route classification). The predicted routes could be used as a cost map for pathfinding algorithms such as A* or Dijkstra.

1. Introduction

One of the earliest pieces of evidence of marine activity is from 709,000 years ago when the ancestors of today's modern humans, made the sea crossings to and around what is now the Southeast Asian archipelagos[1]. Fast forward to the age of exploration, the marine navigation had been relying heavily on experienced crews or accounts from experienced crews for at least many centuries. Not only is the reliance on experienced crews due to the adverse and extreme conditions offshore, but also to the lack of assisting technology. It was not until the last century or so, that much of the technology and digitalization efforts have been put into marine applications. The application has gained even more traction into the 21st century[2]. One of the benefits of digitalization in the context of the current global climate crisis is the opportunity for achieving ship energy optimization and efficiency for the purpose of decarbonization[3]. In addition to achieving ship energy optimization and efficiency, decarbonization also mitigates a negative health effect on individuals[4].

The historical vessel routes from maritime navigation are a valuable source of information on how experienced crews navigate around known waters. It is especially true for harbor crafts that usually move about the area it serves frequently. This paper aims to capture a gold standard which is the crew's experience as a knowledge base of the historical routes, in the form of a trained neural network[5]. The knowledge base could be consumed by pathfinding algorithms to generate a suggested route[6]. The algorithm for pathfinding is essential in an autonomous or semi-autonomous system[7]. The most popular pathfinding algorithm such as A* or Dijkstra requires an area insight in form of a map[8,9]. The

knowledge base generated from the solution presented in this paper could become a cost map. Not only for the said pathfinding algorithms, but also potentially for other pathfinding algorithms[10].

Nonetheless, in the case of less sophisticated smaller harbor vessels (i.e., tugboat, ferry, cargo, etc.), the data from onboard automatic digital logging system is not always available. Instead, the data that is regularly available is in form of daily manual logs such as noon reports that hold a considerable amount of information[11]. Given these scenarios, this paper aims to utilize the noon report combined with positional (navigation) and weather data from external sources to achieve its aim.

The subject vessel discussed in this paper is POSH Grace. The noon report data collected from POSH Grace, combined with externally sourced data such as positional and weather data for seven months (April to October 2020), make up the dataset. POSH Grace is a tugboat that is owned by PACC Offshore Services Holdings (POSH) as shown in Figure 1. The POSH Grace vessel specifications are given in Table 1. The main activities of the tugboat consists of anchoring, assisting the large vessels in docking, and piloting around the southern sea of Singapore. The purpose of collecting the operational data from the subject vessel was to conduct a research study in predicting the fuel rate to achieve fuel efficiency and better planning by data analytics and machine learning[12,13].

Table 1. Vessel specifications.

Main particulars	**Value**
Length overall (LOA)	29 m
Displacement	665 tons
Maximum speed	12 knots
Main engines	NIIGATA 6L26HLX
Number of engines	2
Total BHP	4000 BHP
Type of propulsion	Azimuth pod
Number of propulsors	2

Figure 1. The subject vessel, POSH Grace tugboat**.**

2. Methodology

There are four steps to achieve the solution offered in this paper as shown in Figure 2. All four steps use machine learning toolkit from computer science. Machine learning is a branch of artificial intelligence that uses data and algorithms to mimic the way human learns artificially using a digital tool (i.e., computer)[14]. A model (or network) must first be designed, which is akin to a pipeline to which the data is fed at both ends. Data that is fed at both ends while training the network are often called training X and y inputs. It is to let the algorithm iterate over the data, to adjust values in the network to get y value given the X value (backpropagation). The values in the network are function coefficients and are

stored in the form of tensors. The function is often referred to as an activation function. Every iteration is evaluated by a loss function to let the algorithm know how far off it is from the desired value. A well-trained network can produce a close value of y (the prediction) to the expected value of y (the evidence/observation).

The final objective is to generate two outputs of Route Image (H) and Fuel Prediction (I) by step 4, from Job & Noon Report Metadata. Inputs (A) and (B) must first be prepared to complete all four steps.

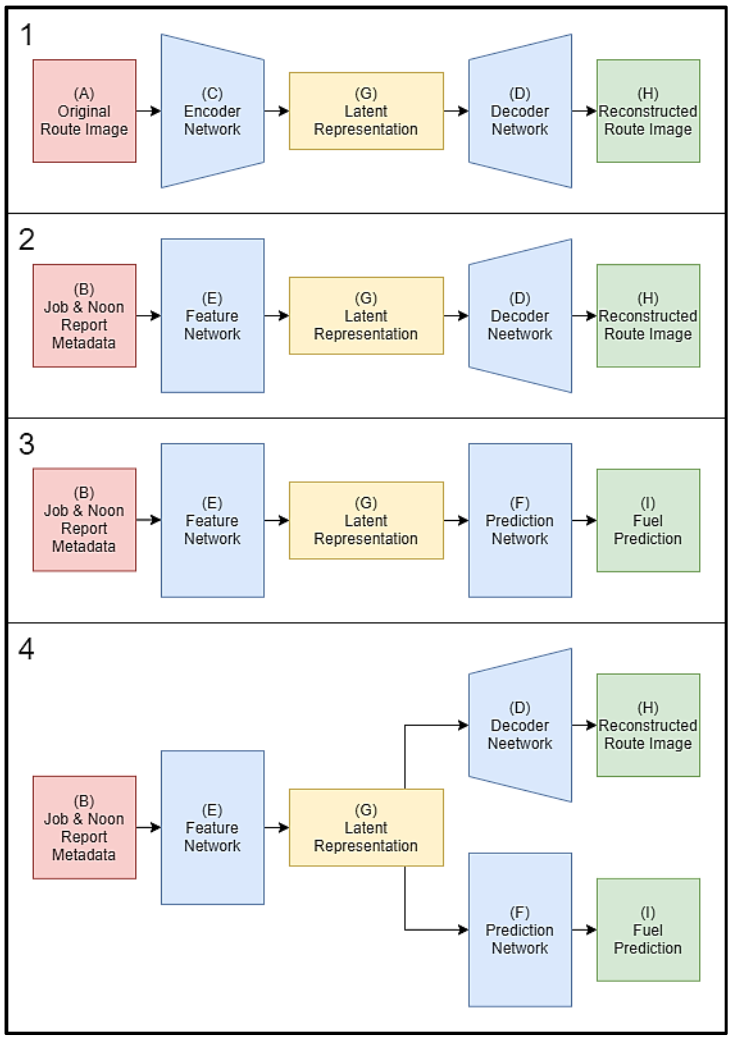


Figure 2. Block Diagram

2.1. Data Preparation

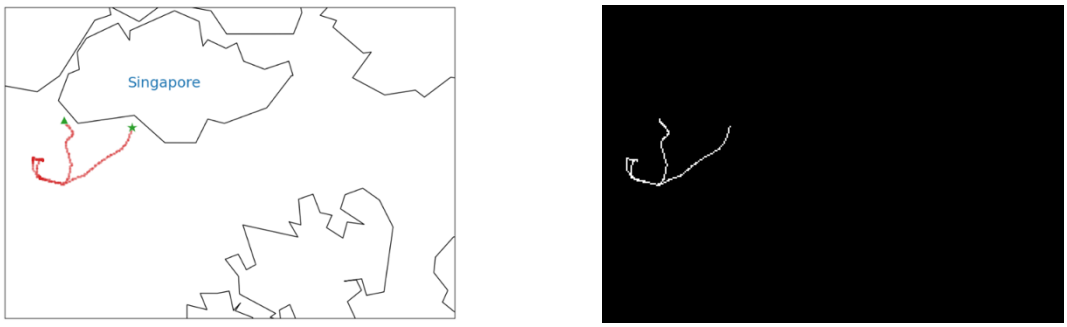
The Data Preparation sub-sections explain the methodologies to convert raw data into a dataset prior to proceeding to any steps shown in Figure 2.

2.1.1. Route Image Creation

A route of the vessel is comprised of a series of historical positions in form of GPS latitude-longitude coordinates between a timeframe from the noon report. The historical position of the vessel could be obtained from an AIS (Automatic Identification Systems) aggregator service by Marine Traffic. This is one of the external data discussed in the introduction. This work has been partially supported by Marine Traffic (www.marinetraffic.com).

The series of route positions is resampled to an interval (i.e., one minute). The series of one-minute vessel position data points are approximated and binned to a cell in a grid. Each route eventually forms

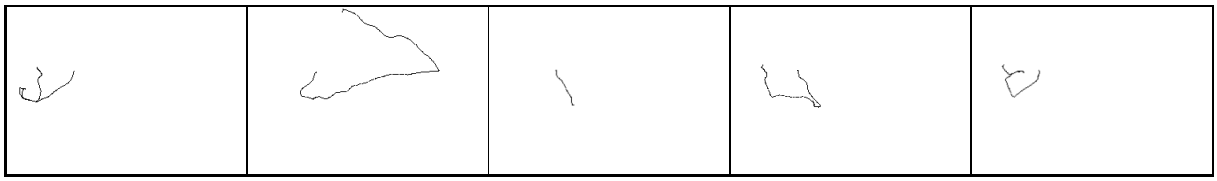
a greyscale route image as shown in Figure 3(b). The greyscale images are the Original Route Image (A) in Figure 3(a).



(a) (b)

Figure 3. The transformation of route information into a greyscale image.
(a) Original route over an illustrated map. Green triangle is origin, green star is destination.
(b) The route image (308 x 212 pixels).

In the noon report between April to October 2020, there are 405 jobs. Consequently, there are 405 route images, a few of which is shown in Figure 4. Routes from Figure 4 are used as a test dataset and excluded during training. The 400 images alone are insufficient to train a neural network in Figure 2 step 1. Hence, each image is duplicated randomly by 30 to 50 times. The total number of images for training the neural network is set to be around 15,000. The duplication of the images will unlikely be redundant as it will eventually generate non-identical samples of latent representation, which is explained in Appendix A (the reparameterization trick).

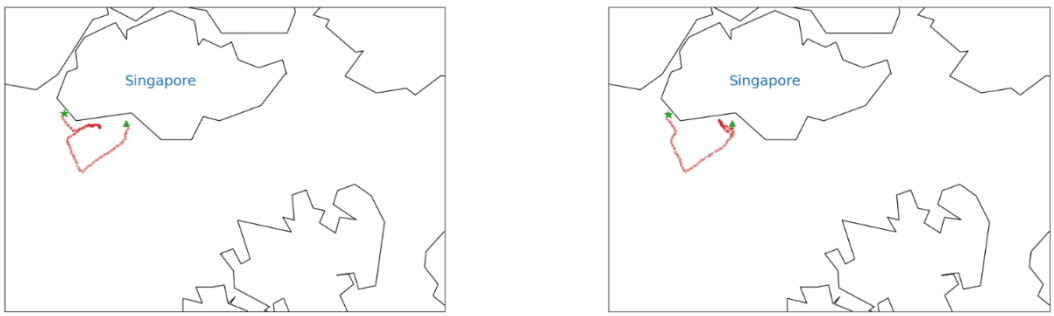


(a) (b) (c) (d) (e)

Figure 4. Sample of Original Route Images. The image greyscale is inverted for better visual.
Route number: (a) 22, (b) 264, (c) 288, (d) 383, (e) 387.

2.1.2. Route Classification

The vessel routes from jobs are classified based on the route similarity. Nonetheless, however similar two or more routes are, they are not identical in terms of vessel positional coordinate at a particular time. A sample case is shown in Figure 5.



(a) (b)

Figure 5. The route similarity comparison.
Route number: (a) 387 versus (b) 379.

For each job, the time-series positional coordinates are broken down into latitude and longitude elements. Latitude from each job is considered to measure the similarity distance using a dynamic time-warping algorithm[15]. The same method is applied to the longitude. Figure 6 shows the point-by-point associations for distance measurement. The cumulative absolute distance of latitude and longitude difference is combined by calculating a hypothenuse of the two. It is due to that the latitude and longitude are assumed to be orthogonal, thus ignoring the curvature of the earth.

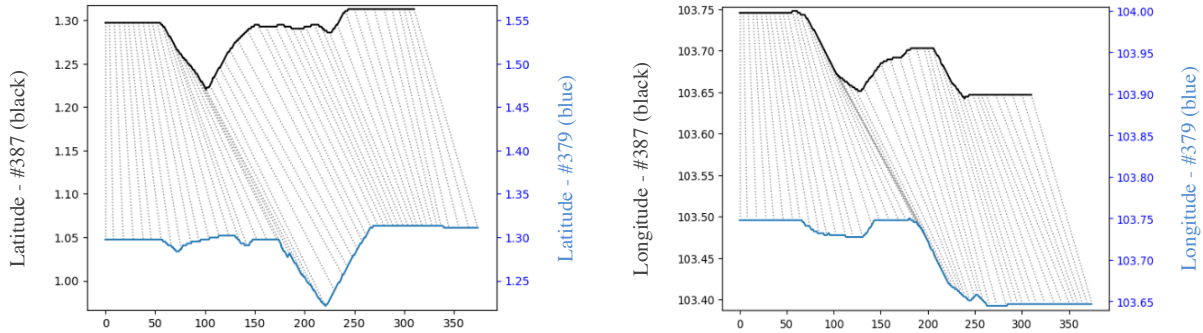


Figure 6. Dynamic time-warping correlation on #387 and #379. (a) Latitudes, (b) Longitudes.

The pair similarity distance for every pair combination is stored as a distance matrix[16]. The visualized version of the distance matrix is shown in Figure 7. Each route's distance with another is binned in a cell of a matrix (table). Hence, it is mirrored diagonally. The dark areas are the regions where the distance value is low indicating there is a close similarity with other members. In contrast, the bright areas (lines) are the part where the distance value is high. From Figure 7, it can be observed that there are members (routes) that do not have similarities with other members (the routes with white lines).

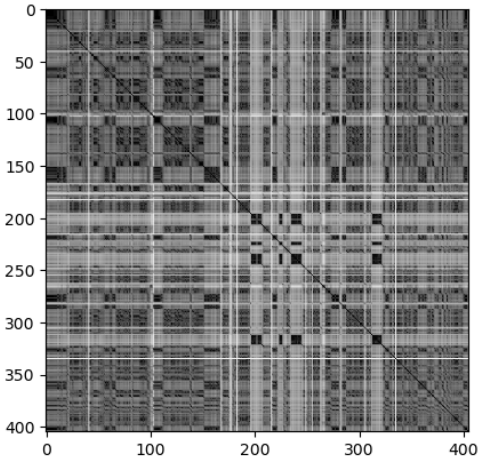
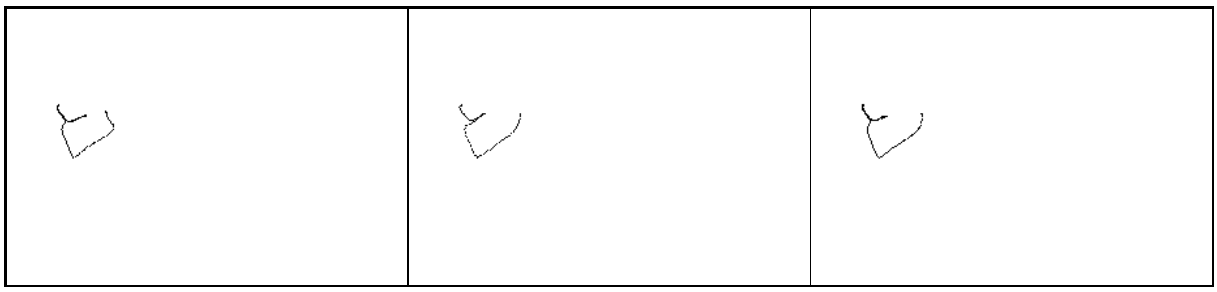


Figure 7. Visualized distance matrix.

The agglomerative hierarchical clustering is used to convert the distance matrix to class labels[17]. Agglomerative clustering is a hierarchical clustering, an unsupervised machine learning algorithm, that is a bottom-up approach. It starts by pairing up the individual/primary nodes (or data points) as secondary nodes. The pairing up criterion is based on the closest distance. It then joins (i.e., averages) the secondary nodes to become new primary nodes. It then again progressively pairs up and joins the now primary nodes as new secondary nodes based on the same criterion. The hypothetical shape of the clustering mimics the structure of a root of a tree. The clustering stops until either the distance threshold or the number of clusters is achieved.

Similar routes have the same class label indicating the cluster association. Figure 8 shows the other cluster members in the same cluster as route #387. The class labels (or cluster labels) are further one-hot encoded[18].



(a) (b) (c)

Figure 8. Other routes that are similar to route #387. All images are inverted greyscale image. (a) Route #98, (b) #228 & (c) #400.

2.1.3. Jobs & Noon Report Metadata

A set of secondary data that describes the primary data is known as metadata. Figure 9 shows the activity area from the position data, which forms a set of parameters (lat0, lat1, lon0, lon1). It is to denote the geographical area in which the job takes place. The activity area is a metadata as it describes the primary data.

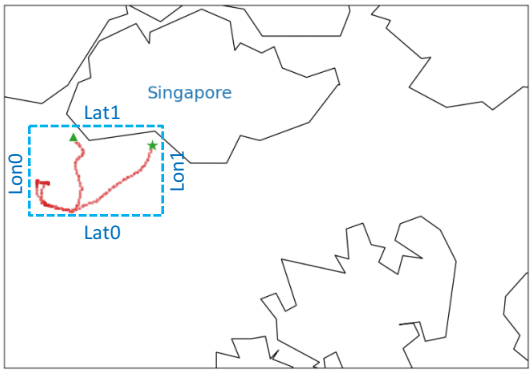


Figure 9. Activity area boundary is indicated by a dotted blue rectangle, producing a boundary of four parameters or features.

In addition to the activity area, the job duration from the noon report and the route classification classes are also metadata and are used as a complete set of Job & Noon Report Metadata (B). The total number of parameters for Jobs & Noon Report Metadata (B) depends on the number of route classification classes. Finally, fuel consumption is obtained from the noon report. The fuel consumption is used for training the prediction network (F).

2.2. Generative Modeling

Generative modeling refers to Step 1 in Figure 2. Generative modeling uses a variational autoencoder (VAE) neural network[19]. The VAE is comprised of two sub-networks: (C) Encoder Network, and (B) Decoder Network according to Figure 2. Figure 10(a) and Figure 10(b) show the encoder and decoder architectures. Several layers and layer types are used to construct both networks, which are readily available for implementation using TensorFlow framework[20]. VAE uses a combination of convolutional 2D (Conv2D) and fully connected (Dense) layers, as well as tensor shape transform layers (Flatten and Reshape)[21,22]. In the final layer of Encoder Network, a custom sampling layer (Sampling) is used. The sampling layer is discussed in section 3.

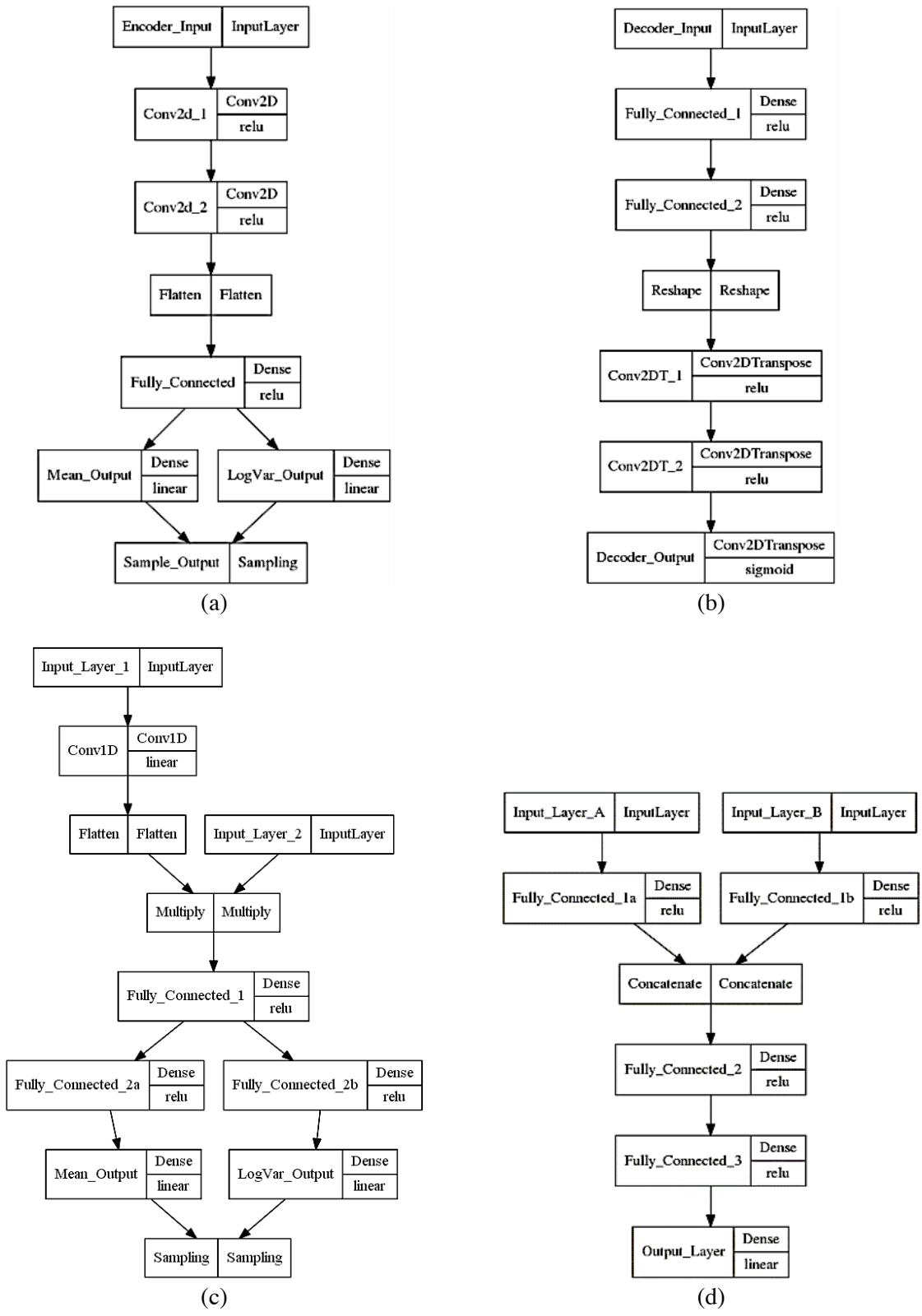


Figure 10. (a) Encoder network architecture, (b) decoder network architecture, (c) feature network architecture, and (d) prediction network architecture.

The aim of the Encoder Network is to reduce the original input (route images) to a representation in a lower dimensions space, also known as latent representation in latent space. The Latent Representation (G) is a set of probability distributions comprising mean (μ), and log-variance (σ). The number of sets of probability distribution (number of pairs of mean and log-variance) determines the number of the latent space dimension.

The aim of the Encoder Network in the solution presented in this paper is to convert the input of Original Route Image (A) into its Latent Representation (G) values. The Latent Representation will later be used to train the other two networks (E) and (F), which will be explained later. Eventually, the Encoder Network is removed from the final structure as shown in Figure 2 step 4.

Figure 11 shows outputs of a 2-dimensional slice of 7-dimensional latent space from a trained VAE network. The outputs are tiled in a 2-dimensional latent space. The VAE manages to fit the route images into functions in the latent hyperspaces of μ and σ.

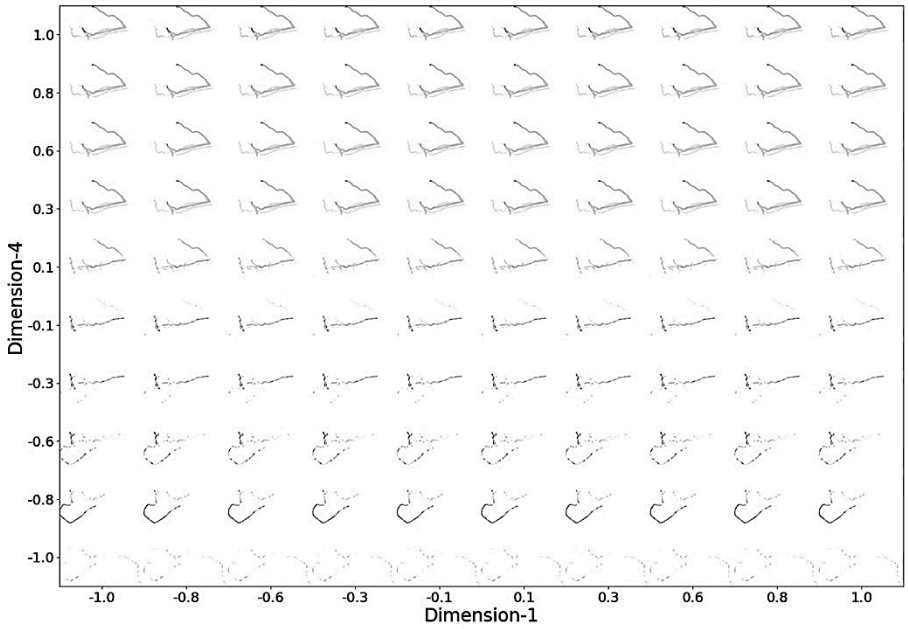


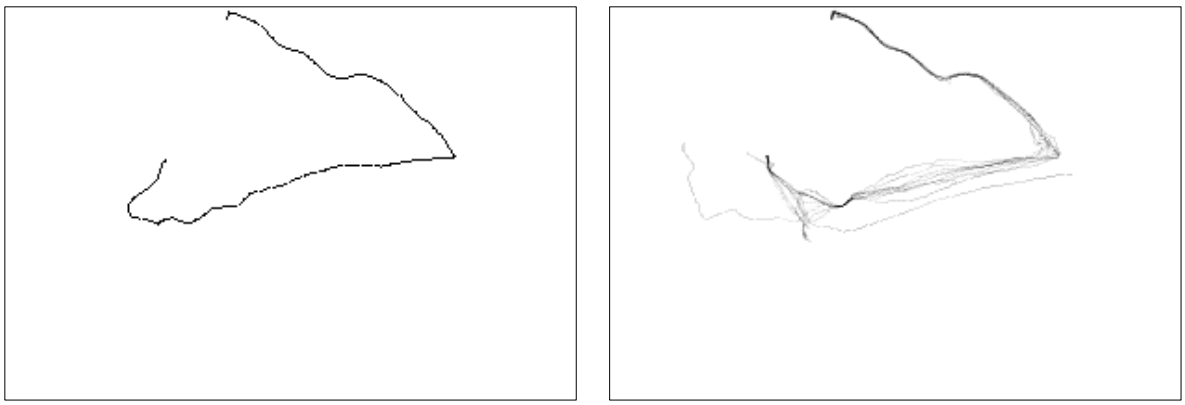
Figure 11. The visualized Latent Representation. The image is an inverted greyscale.

While training, the Decoder Network (D) accepts a sample of the Latent Representation (G) from the encoder network in attempt to reconstruct the original route image. The purpose of sampling in Encoder Network (C) is to sample out the probability distribution for training. It is due to that the backpropagation could not take place through a probabilistic function. Due to the probabilistic nature, it is highly unlikely that the sampling of probability distribution will produce identical values for a set of identically duplicated inputs as mentioned in section 2.1.1. The sampling (reparameterization trick) is discussed further in section 3.

Figure 12 shows the comparison of original and reconstructed route images from a well-trained VAE. While training VAE, a special loss function is used for the Encoder Network (C), the KL_{loss} also known as the Kullback–Leibler loss function[23].

$$KL_{loss} = -\frac{1}{2}(1 + \sigma - \mu^2 - e^{\sigma}) \quad (1)$$

KL_{loss} is a function of the probability distribution. σ is the log-variance, and μ is the mean, while e is Euler constant. The Decoder Network (D) uses binary cross-entropy as a loss function[24].



(a) (b)

Figure 12. (a) Original of route #264 versus (b) reconstructed route image comparison from step 1. Both greyscale images are inverted for better visual.

2.3. Feature Modeling

Feature modeling refers to the second step in Figure 2. The architecture of the network is shown in Figure 10(c). It is a separate neural network replacing the encoder. The Feature Network (E) accepts Job & Noon Report Metadata (B) as training X, instead of route images for Encoder Network (C). For training y, it uses Latent Representation values (the μ and σ) generated by a trained Encoder Network (C). The route classification is fed into *Input_Layer_1*, while the job duration and job activity area are fed into *Input_Layer_2*. The Feature Network uses the same TensorFlow framework as VAE.

2.4. Prediction Modeling

Fuel prediction refers to step 3 in Figure 2. It uses fuel consumption information from the noon report as training y. As training X, the Prediction Network (F) uses Latent Representation (G) generated by Encoder Network (c). Figure 10(d) shows the architecture of the Prediction Network (F). *Input_Layer_A* takes in the mean (μ), while *Input_Layer_B* takes in the log-variance (σ). The prediction network uses exclusively fully connected (Dense) layers. Both feature and fuel prediction networks use mean absolute error as a loss function[25].

2.5. Navigation & Fuel Prediction Modeling

The Navigation & Fuel Prediction Modeling uses three trained networks as shown in Figure 2 step 4. In this final step, training is not essential, although it is possible to do so (also called fine-tuning)[26]. The final step 4 produces two deliverables: the Reconstructed Route Image (H), and Fuel Prediction (I) by taking in only the Job & Noon Report Metadata (B) as input. The Original Route Image (A) is no longer needed. The decoder, feature and prediction networks make up the final model.

2.5.1. Pathfinding Algorithm

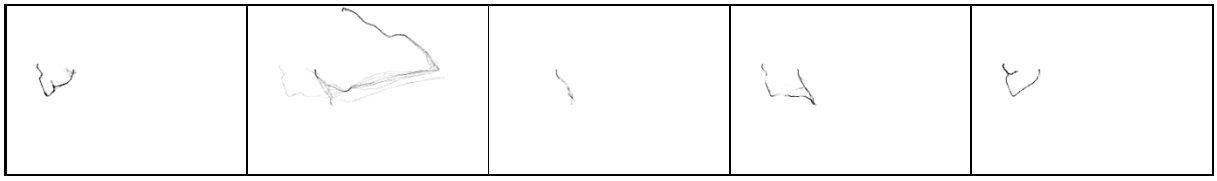
Fuel prediction is evaluated by the percentage of difference of the fuel consumption in the noon report (evidence/observation) and prediction of fuel by the Prediction Network. The Reconstructed Route Image evaluation, however, is not as straightforward and requires an application of pathfinding algorithms such as A* and Dijkstra. Hence, the evaluation of Reconstructed Route Image (H) requires deliberation and is discussed in Section 3.

3. Results & Discussion

This section separately discusses the results from step 4 in Figure 2. Section 3.1 and 3.2 independently discuss two sets of results of Reconstructed Route Images (H) and the Fuel Prediction (I) respectively.

3.1. Route Image Reconstruction

The five Reconstructed Route Images from Figure 4 are presented in Figure 13. Compare them with the original images in Figure 4. Out of the five test samples, all appear to be visually acceptable except #22 and #264, the Figure 13(a)(b). The reconstructed #22 from Figure 13(a) is considerably different from the original image Figure 4(a) due to being a unique route. In other words, #22 is a one-member cluster and the final model is not trained with any information similar to #22. Nonetheless, the final model still manages to produce result that is not extremely different from the original.



(a) (b) (c) (d) (e)

Figure 13. The reconstructed route images by final model. All greyscale images are inverted. Route number: (a) 22, (b) 264, (c) 288, (d) 383, (e) 387.

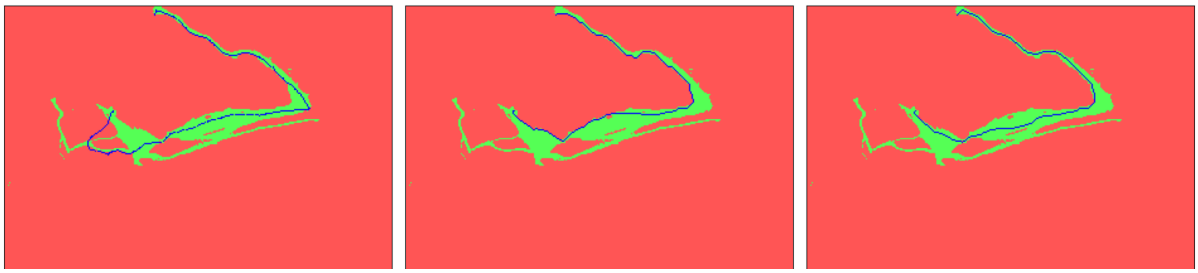
Figure 14(a) discusses the result of route #264. Despite there being less white area (areas which the vessel can visit), the result is still acceptable to become a cost map for the pathfinding algorithm. Figure 14(b), the inverted version of Figure 14(b), shows that there are grey areas which is the information from the past visits. The idea is to make the frequently visited areas lower cost to visit. In greyscale image, the darker shades are represented by lower values. In 8-bit representation, value 0 is black, value 255 is white (if normalized, 0 is black and 1 is white).

The transformed image as a cost map for the pathfinding algorithm is shown in Figure 14(c). The no-go zone or area is represented as value 0 as output by the final model. It is due to no evidence in the training data (400 images) that it has ever been visited. The no-go zone representation is a discontinuity and appears as red in Figure 14(c). In contrast, the frequently visited area appear in lighter shades by Figure 14(a), darker shades by Figure 14(b).

(a) (b) (c)

Figure 14. (a) reconstructed greyscale image as output by final model, (b) the invert of reconstructed greyscale image, (c) the visualization of cost map after transformation.

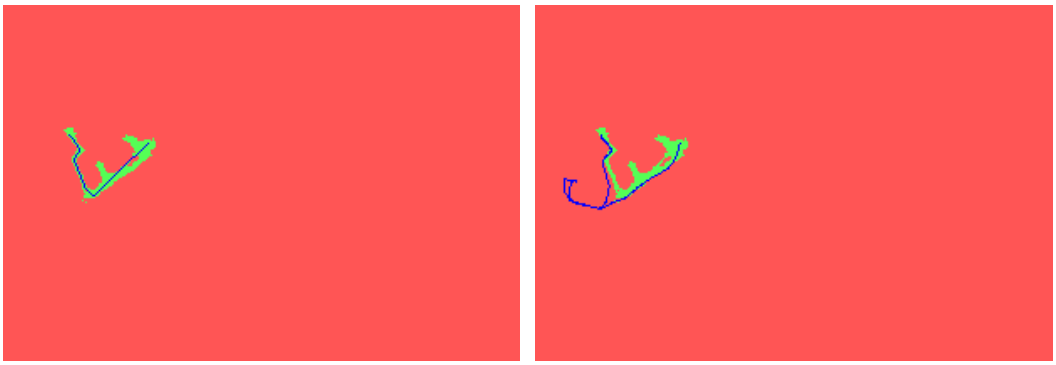
The comparison of the pathfinding algorithms using the cost map versus the observation (original route) is shown in Figure 15. Both pathfinding algorithms suggest that there are shorter paths. Provided with this information, the crew could have been advised of a shorter route. Ultimately, the crew still could have had the absolute discretion to determine the actual path. It may have been due to the adverse sea conditions that the crew may have experienced during route #264, or a certain objective to visit a certain location prior to the final location.



(a) (b) (c)

Figure 15. Application of pathfinding algorithm on reconstructed route image.
(a) Observation, (b) A*, (c) Dijkstra.

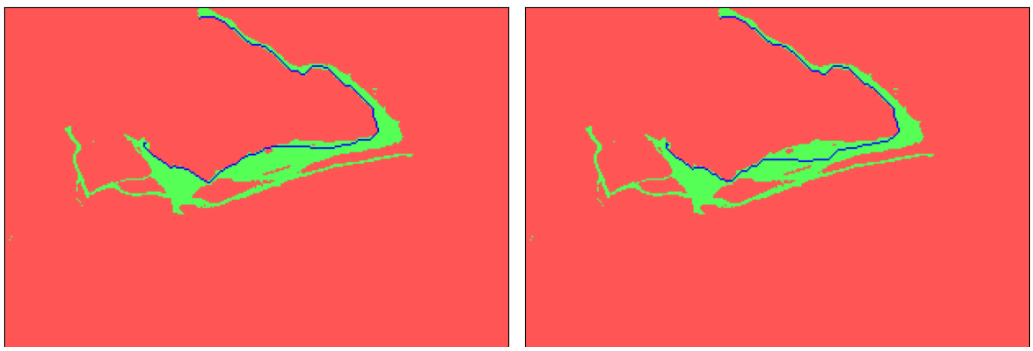
In addition to the one-member class issue discussed earlier, the solution presented in this paper is not able to make up for the shortcoming related to intermediate or multiple destinations as shown in Figure 16. If there is no explicit source of information (i.e., from the noon report) detailing or splitting the multiple trips in one job to separate trips, the solution presented in this paper will not be precise.



(a) (b)

Figure 16. Comparison of the route with multiple trips.
(a) Suggested path, (b) observation.

It is possible to customize the pathfinding algorithm to account for external factors (i.e., wind or wave situation), by making the cost of travel to a certain direction to be more expensive. Figure 17 shows the comparison of results with and without accounting for the weather. Figure 17(b) shows the effect of a southerly wind (downward) applied to Figure 17(a). The weather data is another source of external data mentioned in the introduction and is provided by www.meteoblue.com.



(a) (b)

Figure 17. Application of weather data in pathfinding algorithm.
(a) With weather data effect, (b) Without weather data effect.

3.2. Fuel Prediction

Table 2 shows the comparison of the predicted fuel consumption versus the observation. Route #22 is under-predicted as the number of training data for the similar route pattern is scarce. The result for route #22 is expected due to being a unique route and excluded during training. The final model is unable to predict what it had never learned before, and forced to guess. The best prediction result from route #288 and the worst prediction result from route #387 are discussed further in Table 3.

Table 3 lists down the training data from routes similar to #288 for comparison. It is labelled as class #0. The final model manages to use the job duration and activity area of route #288 to approximate to a very close prediction to the actual record in the noon report (the observation).

Table 2. Fuel prediction versus observation comparison.

Route #	Prediction[a]	Observation[a]	% Difference
22	1309	1688	-22.45
264	2426	2662	-8.88
288	454	450	0.78
383	1419	1310	8.28
387	1094	1245	-12.12

[a] In liters

Table 3 also lists down the training data from routes similar to #387 for comparison. It is labelled as class #1. There is a total of four data points by class #1 (only 3 for training: #98, #228, #400). Using three training data points, the final model still manages to approximate fuel prediction to a reasonable figure of 1094 liters, -12.12% off from the actual 1245 liters.

Table 3. Comparison of routes #288 and #387 with members from their own classes.

Class #	Route #	Job Duration[a]	Observation Fuel[b]
0	**288**	**7200**	**450**
0	124	9000	474
0	232	9600	605
0	259	18000	1025
0	303	7200	515
0	348	4500	275
1	**387**	**19800**	**1245**
1	98	18000	1050
1	228	15600	985
1	400	28800	1816

[a] In seconds

[b] In liters

During deployment, the user of the final model needs to input the space and time information in which the vessel is to operate in addition to selection of a route pattern (from route classification). The activity area and duration represent the space and time information the user needs to input. Figure 18 discusses the other features (geographical features) used to produce results from Table 3. Figure 18(a)(c)(e) show the case of route #288, compared with its own class members.

Figure 18(e) shows the correlation of fuel and duration as explained by Table 3, while Figure 18(a)(c) gives the geographical perspective of the activity areas. As route #288 is not an outlying member (in terms of both fuel-distance correlation and activity area), the final model manages to produce a very good fuel prediction result. It is in addition that there are more training data points compared to route

#387. The adversity of activity areas and good duration-fuel fitting make up a good set of training features for #288.

In contrast, route #387 does not have as many data points as route #288. The activity areas are not as adverse as route #288 either, as shown in Figure 18(d). Despite the lack of adversity, these feature combinations are still able to produce a reasonable ballpark prediction of 1094 versus the actual 1245 (observation). It is shown in Figure 18(f) as #387 that it could still fit relatively well between #98 and #400. In addition, the final model manages to generate a good route image as well, as shown in Figure 13(e). Hence, this presents an opportunity to increase the number and quality of job/route metadata to increase the quality of the features.

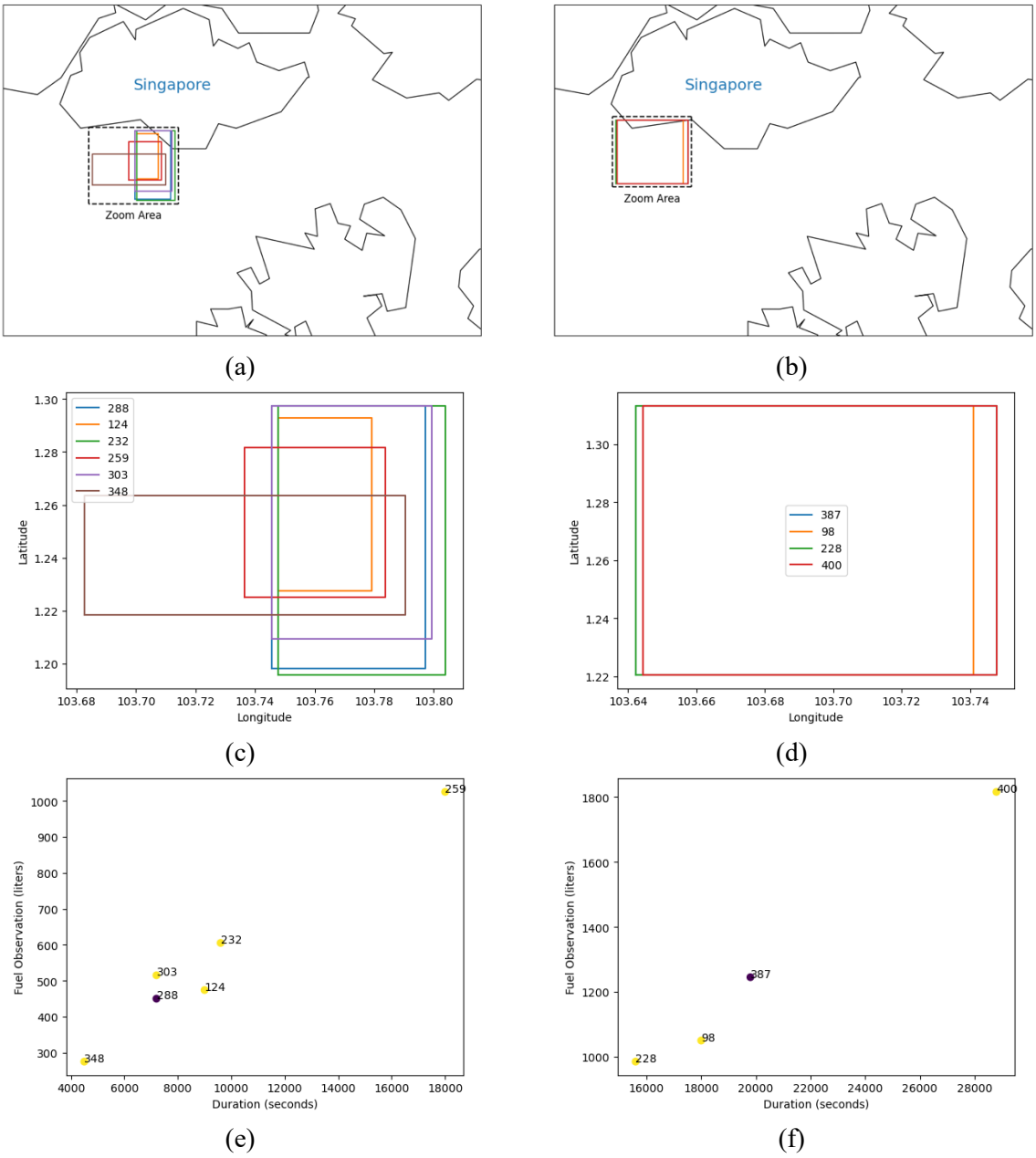


Figure 18. Comparison of features between route #288 versus route #387. Activity areas: (a) Route #288, (b) Route #387. Zoom-in (close-up) of activity areas: (c) Route #288, (d) Route #387. Fuel versus duration: (e) Route #288, (f) Route #387

4. Conclusion

Arguably, the very popular neural network applications are for dimensionality reduction purposes such as image classification. An image of certain pixel size that contains so much information is reduced to a tag for object A, B, etc. The neural network for regression takes a step up by attempting to find higher dimensional representations of lower dimension samples (points). Generative modeling takes it even further by creating what did not exist before, such as the reconstructed route image in the attempt to recreate the original. The reconstructed image is not the exact copy of the original, it is the slightly different version of the original.

The application of the solution presented in this paper attempts to utilize generative modeling to encompass valuable past human experiences in maritime navigation. The crew's experience is deemed as a gold standard from which the model is learning. The results presented in this paper demonstrate that not only is it possible to profile the navigation route and fuel consumption, but also able to perform well. The solution presented in this paper can become a tool on which other methods are applied, such as various pathfinding algorithms and its derivations/customizations.

A harbor tug movement is driven by the type of job and the number of jobs that the crew has on hand. For the harbor tug with a certain size and jobs, there is a rule-based expert system used to plan the job assignment depending on the time and location of the tug and type of job. The solution presented in this paper may be integrated with such a rule-based system to make up a semi-autonomous system. The implementation of the solution presented in this paper in such a system would have alleviated some load from the crew and staff, putting them into the supervisory role rather than the executor role. Finally, the other crews or other non-technical users can also benefit from the knowledge base in the form of a trained neural network, not only for route planning but also for fuel estimation.

Acknowledgement

The Authors wished to acknowledge the resources supported by SMI (R-SMI-A403-0001) and MOE (R-MOE-A403-C002/MOE2018-TIF-1-G-008).

Appendix A

Appendix A explains the Network Configurations of Figure 10. The configuration of the VAE network is as shown in Figure A1 for every layer in Figure 10(a)(b) for both Encoder Network and Decoder Network. Encoder Network accepts greyscale Original Route Image (A) of 308 x 212 pixels. It produces two outputs: Mean Output (μ) and LogVar (σ). The Sample Output (z) is derived from sampling the probability distribution (2) the sampling layer, where ε is the random value of a normal distribution (mean = 0, standard deviation = 1)[27]. The z is necessary to train the Decoder Network (D). This sampling technique is also known as reparameterization trick[28].

$$z = \mu + e^{\sigma/2} \times \varepsilon \tag{2}$$

In Figure A2(a), the configuration of the last 3 layers of Feature Network (E) output aligns with Encoder Network (C) output. It is to be trained with Job & Noon Report Metadata (B), and to be able to predict the latent representation values. The Feature Network (E) accepts 90 input parameters. The first set is the route classification input (Input_Layer_1) with 85 classes (85 route patterns). The second set is 5 job/route metadata (job duration and activity area) as an input to Input_Layer_2. Multiply layer performs element-wise multiplication.

Figure A2(b) is the Prediction Network (F) configuration. The output is one parameter of Fuel Prediction (I) output trained with consumed fuel information from the noon report. The input is a set of probability distribution (μ and σ) from the Feature Network (E). The latent representation is multi-dimensional. The network configuration used in this paper uses 7 dimensions of μ and σ.

Figure 10 also states “relu” in various locations. Relu (or ReLU) refers to rectified linear unit activation function used in the corresponding layers[29]. The function of ReLU is described in (3).

$$f(x) = \max(0, x) \tag{3}$$

Model: "Encoder"

Layer (type)	Output Shape	Param #	Connected to
Encoder_Input (InputLayer)	[(None, 212, 308, 1)]	0	[]
Conv2d_1 (Conv2D)	(None, 106, 154, 8)	520	['Encoder_Input[0][0]']
Conv2d_2 (Conv2D)	(None, 53, 77, 16)	8208	['Conv2d_1[0][0]']
Flatten (Flatten)	(None, 65296)	0	['Conv2d_2[0][0]']
Fully_Connected (Dense)	(None, 16)	1044752	['Flatten[0][0]']
Mean_Output (Dense)	(None, 7)	119	['Fully_Connected[0][0]']
LogVar_Output (Dense)	(None, 7)	119	['Fully_Connected[0][0]']
Sample_Output (Sampling)	(None, 7)	0	['Mean_Output[0][0]', 'LogVar_Output[0][0]']

Total params: 1,053,718
Trainable params: 1,053,718
Non-trainable params: 0

(a)

Model: "Decoder"

Layer (type)	Output Shape	Param #
Decoder_Input (InputLayer)	[(None, 7)]	0
Fully_Connected_1 (Dense)	(None, 16)	128
Fully_Connected_2 (Dense)	(None, 65296)	1110032
Reshape (Reshape)	(None, 53, 77, 16)	0
Conv2DT_1 (Conv2DTranspose)	(None, 106, 154, 16)	16400
Conv2DT_2 (Conv2DTranspose)	(None, 212, 308, 8)	8200
Decoder_Output (Conv2DTranspose)	(None, 212, 308, 1)	73

Total params: 1,134,833
Trainable params: 1,134,833
Non-trainable params: 0

(b)

Figure A1. VAE network parameters.
(a) Encoder, (b) Decoder.

Model: "Feature"

Layer (type)	Output Shape	Param #	Connected to
Input_Layer_1 (InputLayer)	[(None, 90, 1)]	0	[]
Conv1D (Conv1D)	(None, 1, 5)	455	['Input_Layer_1[0][0]']
Flatten (Flatten)	(None, 5)	0	['Conv1D[0][0]']
Input_Layer_2 (InputLayer)	[(None, 5)]	0	[]
Multiply (Multiply)	(None, 5)	0	['Flatten[0][0]', 'Input_Layer_2[0][0]']
Fully_Connected_1 (Dense)	(None, 1024)	6144	['Multiply[0][0]']
Fully_Connected_2a (Dense)	(None, 256)	262400	['Fully_Connected_1[0][0]']
Fully_Connected_2b (Dense)	(None, 256)	262400	['Fully_Connected_1[0][0]']
Mean_Output (Dense)	(None, 7)	1799	['Fully_Connected_2a[0][0]']
LogVar_Output (Dense)	(None, 7)	1799	['Fully_Connected_2b[0][0]']
Sampling (Sampling)	(None, 7)	0	['Mean_Output[0][0]', 'LogVar_Output[0][0]']

Total params: 534,997
Trainable params: 534,997
Non-trainable params: 0

(a)

Model: "Predictor"

Layer (type)	Output Shape	Param #	Connected to
Input_Layer_A (InputLayer)	[(None, 7)]	0	[]
Input_Layer_B (InputLayer)	[(None, 7)]	0	[]
Fully_Connected_1a (Dense)	(None, 128)	1024	['Input_Layer_A[0][0]']
Fully_Connected_1b (Dense)	(None, 128)	1024	['Input_Layer_B[0][0]']
Concatenate (Concatenate)	(None, 256)	0	['Fully_Connected_1a[0][0]', 'Fully_Connected_1b[0][0]']
Fully_Connected_2 (Dense)	(None, 512)	131584	['Concatenate[0][0]']
Fully_Connected_3 (Dense)	(None, 64)	32832	['Fully_Connected_2[0][0]']
Output_Layer (Dense)	(None, 1)	65	['Fully_Connected_3[0][0]']

Total params: 166,529
Trainable params: 166,529
Non-trainable params: 0

(b)

Figure A2. Feature and Prediction/Predictor network parameters.
(a) Feature, (b) Predictor.

References

[1] Ingicco T, van den Bergh G D, Jago-on C, Bahain J-J, Chacón M G, Amano N, Forestier H, King C, Manalo K, Nomade S, Pereira A, Reyes M C, Sémah A-M, Shao Q, Voinchet P, Falguères C, Albers P C H, Lising M, Lyras G, Yurnaldi D, Rochette P, Bautista A and de Vos J 2018 Earliest known hominin activity in the Philippines by 709 thousand years ago *Nature* **557** 233–7

[2] Tijan E, Jović M, Aksentijević S and Pucihar A 2021 Digital transformation in the maritime transport sector *Technological Forecasting and Social Change* **170** 120879

[3] Agarwala P, Chhabra S and Agarwala N 2021 Using digitalisation to achieve decarbonisation in the shipping industry *Journal of International Maritime Safety, Environmental Affairs, and Shipping* **5** 161–74

[4] Rosenbaum A, Hartley S and Holder C 2011 Analysis of diesel particulate matter health risk disparities in selected US harbor areas *Am J Public Health* **101 Suppl 1** S217–23

[5] Jain A K, Mao J and Mohiuddin K M 1996 Artificial neural networks: a tutorial *Computer (Long Beach Calif)* **29** 31–44

[6] Grifoll M, Martorell L, Castells M and de Osés F X M 2018 Ship weather routing using pathfinding algorithms: the case of Barcelona – Palma de Mallorca *Transportation Research Procedia* **33** 299–306

[7] Gu Y, Goez J C, Guajardo M and Wallace S W 2021 Autonomous vessels: state of the art and potential opportunities in logistics *International Transactions in Operational Research* **28** 1706–39

[8] Foead D, Ghifari A, Kusuma M B, Hanafiah N and Gunawan E 2021 A Systematic Literature Review of A* Pathfinding *Procedia Computer Science* **179** 507–14

[9] Shu-Xi W 2012 The Improved Dijkstra's Shortest Path Algorithm and Its Application *Procedia Engineering* **29** 1186–90

[10] Patle B K, Babu L G, Pandey A, Parhi D R K and Jagadeesh A 2019 A review: On path planning strategies for navigation of mobile robot *Defence Technology* **15** 582–606

[11] Bialystocki N and Konovessis D 2016 On the estimation of ship's fuel consumption and speed curve: A statistical approach *Journal of Ocean Engineering and Science* **1** 157–66

[12] Tay Z Y, Hadi J, Konovessis D, Loh D J, Tan D K H and Chen X 2021 Efficient Harbor Craft Monitoring System: Time-Series Data Analytics and Machine Learning Tools to Achieve Fuel Efficiency by Operational Scoring System *Volume 6: Ocean Engineering* (American Society of Mechanical Engineers)

[13] Tay Z Y, Hadi J, Chow F, Loh D J and Konovessis D 2021 Big Data Analytics and Machine Learning of Harbour Craft Vessels to Achieve Fuel Efficiency: A Review *Journal of Marine Science and Engineering* **9**

[14] Sarker I H 2021 Machine Learning: Algorithms, Real-World Applications and Research Directions *SN Computer Science* **2** 160

[15] Giorgino T 2009 Computing and Visualizing Dynamic Time Warping Alignments in R: The dtw Package *Journal of Statistical Software* **31** 1–24

[16] Weyenberg G and Yoshida R 2015 Chapter 12 - Reconstructing the Phylogeny: Computational Methods *Algebraic and Discrete Mathematical Methods for Modern Biology* ed R S Robeva (Boston: Academic Press) pp 293–319

[17] Davidson I and Ravi S S 2005 Agglomerative Hierarchical Clustering with Constraints: Theoretical and Empirical Results *Knowledge Discovery in Databases: PKDD 2005* ed A M Jorge, L Torgo, P Brazdil, R Camacho and J Gama (Berlin, Heidelberg: Springer Berlin Heidelberg) pp 59–70

[18] Harris S L and Harris D M 2016 3 - Sequential Logic Design *Digital Design and Computer Architecture* ed S L Harris and D M Harris (Boston: Morgan Kaufmann) pp 108–71

[19] Kingma D P and Welling M 2014 Auto-Encoding Variational Bayes

[20] Abadi M, Agarwal A, Barham P, Brevdo E, Chen Z, Citro C, Corrado G S, Davis A, Dean J, Devin M, Ghemawat S, Goodfellow I J, Harp A, Irving G, Isard M, Jia Y, Józefowicz R, Kaiser L, Kudlur M, Levenberg J, Mané D, Monga R, Moore S, Murray D G, Olah C, Schuster M, Shlens J, Steiner B, Sutskever I, Talwar K, Tucker P A, Vanhoucke V, Vasudevan V, Viégas F B, Vinyals O, Warden P, Wattenberg M, Wicke M, Yu Y and Zheng X 2016 TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems *CoRR* **abs/1603.04467**
[21] Bengio Y and Lecun Y 1997 Convolutional Networks for Images, Speech, and Time-Series
[22] Sainath T N, Vinyals O, Senior A and Sak H 2015 Convolutional, Long Short-Term Memory, fully connected Deep Neural Networks *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* pp 4580–4
[23] Asperti A and Trentin M 2020 Balancing reconstruction error and Kullback-Leibler divergence in Variational Autoencoders *CoRR* **abs/2002.07514**
[24] Ramos D, Franco-Pedroso J, Lozano-Diez A and Gonzalez-Rodriguez J 2018 Deconstructing Cross-Entropy for Probabilistic Binary Classifiers *Entropy* **20**
[25] Chai T and Draxler R 2014 Root mean square error (RMSE) or mean absolute error (MAE)? *Geosci. Model Dev.* **7**
[26] Zhou Z, Shin J, Zhang L, Gurudu S, Gotway M and Liang J 2017 Fine-tuning Convolutional Neural Networks for Biomedical Image Analysis: Actively and Incrementally *Proc IEEE Comput Soc Conf Comput Vis Pattern Recognit* **2017** 4761–72
[27] Rezende D J, Mohamed S and Wierstra D 2014 Stochastic Backpropagation and Approximate Inference in Deep Generative Models
[28] Kingma D P, Salimans T and Welling M 2015 Variational Dropout and the Local Reparameterization Trick
[29] Xu B, Wang N, Chen T and Li M 2015 Empirical Evaluation of Rectified Activations in Convolutional Network