Machine Learning in Time Series Manatee Call Prediction Experiment and Result

Zhengyu Li, *University of Florida*

***Abstract*—The goal of this project is to detect the manatee calls from a real hydrophone taken sounds in an estuary. The methods are to use both linear filter and SPRT and CUSUM tests to predict manatee calls in the test signal by training linear models. The performances of linear and probability models are compared in terms of false and missing detected rate and computational complexity.**

***Index Terms*—CUSUM, linear filter, LMS, machine learning, SPRT, time series.**

I. Introduction

The goal of this project is to design and evaluate an optimization detection approach to detect and extract the manatee calls from hydrophone recordings from an estuary where is a noise environment. The test signal is given in a ".wav" format file with a size of 1.3 million data points. There are also two given training signals, one of them contains only the data of manatee calls and one of them contains noise only, also in ".wav" format file. There are totally 16 manatee calls in the test signal and every call have a different frequency and sound, none of them are exactly the same. In the manatee calls training signal, there are 10 different manatee calls with different types as well. The total size of the training set a is 1.2 million data points. In the noise training set, there are totally 80 thousand data points and the sound of noise is mainly in the low frequency and almost periodically but is not the exactly the stationary data. In order to predict manatee calls in test signal, two predictors for noise and manatee calls need to be trained and applied on the test signal. In order to improve the performance of predictors, SPRT and CUSUM models are added to the errors of predictors.

II. Methods

The implementation of this project includes training noise and manatee calls signals separately and parallel applying these two predictors on the test signal in order to find noise and manatee calls separately to achieve extracting and predicting manatee calls. This project is implemented in Python and the program block diagram is shown in Fig. 1.

After looking into two training sets by hearing the sound of the signal, both of them are non-stationary data so that the non-stationary predictors will be considered with a top priority to predict manatee calls. There are two procedures of the implementation. The first procedure is to use non-stationary adaptive filter model to train and test performance on the primitive signals, and the second procedure is to use probability statistical model on the error of the adaptive filter to predict performance. In the first procedure, there are two models that need to be created.

1. Linear model for manatee call

The first model is the manatee calls model. This model basically learns from the pure manatee calls which is one of our training set and adapted a list of best parameters that contains as much as information of features of common manatee calls. The manatee calls should contain many features. For instance, frequency of manatee calls is one of features. The frequency of manatee call is relatively higher than normal sound as well as the noise signal. Therefore, after adapting the manatee calls training signal, the parameter should contain this information and distinguish the manatee calls from other sound by using this and many other unique features of the manatee calls. There are only 10 manatee call samples given so that the parameter finally is expected to contain as many as possible features of these 10 different manatee calls. If some of the manatee calls in the test set have a very big different manatee call feature from the manatee call features in the training manatee calls signal, it may fail to give a very good result since that specific feature is not trained in the training procedure.

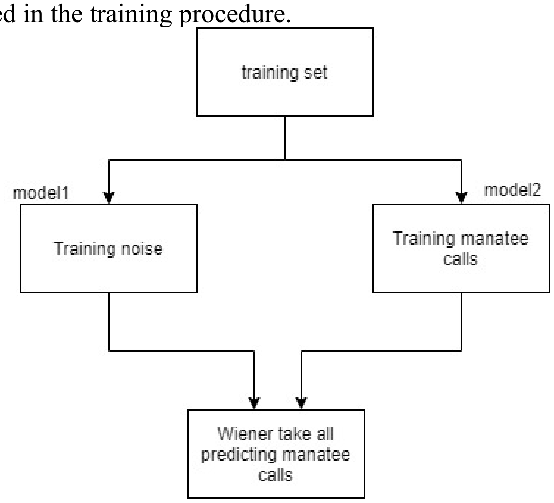

Fig. 1. Block diagram of the model.

2. Linear model for noise

Another model is the noise model. The noise model is very similar with the manatee calls model. Both noise signal and manatee call signal are non-stationary signals and the idea of training the model is to learn unique features from the signal. The parameter of model will eventually be able to contain the information of the noise signal.

3. Output of linear models

The output information of both manatee call model and noise model are exactly the same except values. The output contains the best adaptive parameter *w*, Mean Square Error (MSE) and output signal *y*. The best adaptive parameter is the parameter *w* containing order of floating numbers after adapting by traversing the whole noise or manatee call signals in each model and learning features of the signal. Therefore, the last *w* is the best parameter. MSE is computed in order to see how big the error is between the desired signal *d* and predicted output signal *y*. MSE is the smaller the better. Small MSE means the absolute error of the predicted signal is very small indicating that the output signal *y* would be very close to the desired signal *d*. The reason using square error in order to avoid error cancellation of positive and negative errors. Output signal *y* has to be computed in the model because it is used for computing the error and MSE. Outputting y is mainly because it is very useful when verifying the training result. Output *y* as an audio file, then listen into the predicted audio file and compare with the desired signal, which is the given audio file, could be helpful on evaluating the training result. They should be very similar and the predicted audio file should at least contain most of the noise or manatee calls in each model. Therefore, in a short summary, *w* is the most significant parameter that containing adapted information and predicting the result in the test set. MSE is a metric evaluating the performance of the training model and *y* is also used for evaluating the performance of the model.

The adaptive filter in the first procedure in order to create these two models in this project is the Least Mean Square filter. The following content explains how does the LMS filter records the feature of the signal and adaptively learning from the signal.

A. Least Mean Square Method

The normalized LMS can be derived from the mathematical principle of minimal disturbance: "find the step size that disturbs the weights the least but still converges". As you can expect this will lead to a constrained optimization problem [1].

The estimated output in LMS can be computed using the equation (1) below:

$$y(n) = \sum_{k=0}^{K} w_k x_k(n) \quad (1)$$

where w_k is the weight value of x_k, and x is the input of the system. For the very first elements without enough previous data point, the x_i can be seen as zero, or the program could be started at the number of order element. The initialized weight values are randomly picked between zero and one. The method to update each weight could use equation (2):

$$w_{j+1} = \sum_{i=j-L}^{j} (w_i + 2 \times \mu \times e_i \times x_i) \qquad (2)$$

where L is the filter order, μ is the learning rate or step size that usually a very small number and e_i is the error of the data point x_i. Or using the vector format equation (3):

$$w(i+1) = w(i) \, + \, 2 \times \mu \times e(i) \times x(i) \quad (3)$$

where w(i+1), e(i), w(i), and x(i) are vectors which the length is number of filter order. For a normalized LMS algorithm, the step size controlled by the power of the input is added:

$$w(i+1) = w(i) \, + \, 2 \times \frac{\mu}{x(i)^2} \times e(i) \times x(i) \quad (4)$$

Therefore, the LMS becomes independent of the input power, which is very useful in practice [1]. The error of the above equations could be computed using equation (5):

$$e(i) = d(i) \, - \, y(i) \quad (5)$$

4. Predict manatee call and noise models

Once the manatee call model and noise model have been trained and performed well, the predicting procedure can be started. In order to parallel running two models on the test signal, the method in this project is to apply the manatee call model on the test data first and then apply the noise model on the test signal. Although these two models are not going to be executed at the same time, the predicted results will be merged together. Intuitively, the error would be large if the test signal is noise because the noise sound is very different with the manatee calls. And the error of predicting noise using the manatee call model would have a large error compared with the noise model since the noise model is trained on the noise signal so that the error would be relatively smaller than other models. This theory is the same to the noise model. It would have a large error on the manatee call signal due to lack of training.

Therefore, after testing each model on the test signal, two error lists $e_{manatee}$ and e_{noise} should be obtained from two models, respectively. In order to avoid cancellation of positive and negative errors, the square error method is used when comparing errors. In order to parallel and combine the result of two models, compare two error lists $e_{manatee}$ and e_{noise} generated by two different models and then select the model with the smaller error on each data point *i*. Then choose the correspond predicted output $y_{manatee}[i]$ if the manatee call model is selected or $y_{noise}[i]$ if the noise model is selected, to be the final predicted output $y[i]$ of the test signal.

However, since the manatee calls model only trained in manatee calls signal, it may have unknown error on noise signal in the test signal. Test signal contains both noise and manatee call signals so that the error might be large on noise signal and small on manatee calls signal. Therefore, an average smooth method needs to be applied to the error list in order to eliminate the very big different errors of signal. Focus on the average of a few data points would be better than comparing single error of a data point.

After getting the smoothed error list of two models, one can simply compare the error of two models and pick the model with a lower error on the test signal. Since the requirement is a square wave showing the manatee calls and noise, simply set the manatee call signal as 1 and noise signal as 0 in the output $y[i]$. The result can be shown in a square wave then.

In order to find the manatee calls, check if the durations or data points when signal $y[i]$ is 1 are corresponds to the ground truth. The ground truth can be easily drawn if the time stamps of each manatee call can be found. Therefore, in this step, we must manually listen into the test signal and mark the time stamps of each manatee call. Once we hear a manatee call, record it start time and end time. We can either transform the time of ground truth to data points or transform *y[i]* to time domain in millisecond.

There are two type of failures. One is in a specific duration, there is a manatee call in the ground truth but not in *y[i]* which means missed detection. Another one is there is not a manatee call in the ground truth but the *y[i]* contains a manatee call which means detected a false call. In order to count these two different types of failures, we should manually compare each detected manatee call as well as each real manatee calls in the test signal. Then, a receiver operating characteristics (ROC) curve should be drawn in order to evaluate the performance.

5. Probability statistical model

The second procedure is to apply sequential probability ratio test (SPRT) and cumulative sum (CUSUM) tests on models created in the first procedure based on the prediction error. In order to apply SPRT, assume the probability of the error is in Gaussian distribution. Mean and standard deviation of error needed to be computed. Then using the equation (6) to compute the probability $p_{manatee}$ and p_{noise}.

$$\mathcal{N}(\mathrm{x}|\mu,\sigma_2) = \frac{1}{\sqrt{2\pi\sigma^2}} \mathrm{e}^{-0.5(\frac{(x-\mu)^2}{\sigma^2})} \quad (6)$$

After having $p_{manatee}$ and p_{noise}, select a small window start from the beginning and sum up all data points in p then take log and comparing which model has a better probability as shown in equation (7) where i is the noise error list, j is the manatee calls error list and n is the window size. The model that has a greater result would win and then the probability model should start from that model.

$$J(n) = log\frac{P_i(X(n))}{P_j(X(n))} \quad (7)$$

If $J(n)$ is greater than 0, $P_i(X(n))$ wins; otherwise, $P_j(X(n))$. Intuitively, the expected result in the SPRT should be start from the noise since there is no manatee calls at the beginning of the test signal by hearing the signal. wins. Then, CUSUM test is applied to the model. There are two labels 0 and 1 as described in the first procedure. Maintain a label start from noise which is 0 and then apply the log likelihood test on the rest of data as equation (8).

$$\Delta J(n+1) = J(n+1) - J(n) \quad (8)$$

In this step, a threshold needs to be created and tuned. And ΔJ is maintained by selecting the maximal value of ΔJ and p_{noise} if label is noise or $p_{manatee}$ if label is manatee call. To avoid ΔJ becoming a negative number, the minimal value is set to 0. Then keep picking the maximum value of ΔJ until it is greater than the tuned threshold. Try different thresholds in order to get the best result by comparing with the ground truth manually.

III. Experiments

The experiments of this project evaluate performance of both linear adaptive model in the first procedure and statistic model in the second procedure. The linear filter in this experiment is the LMS filter trained with MSE in the input space. First of all, the manatee calls model was trained using "train_signal.wav".

1. Pre-processing primitive manatee call data

Before applying LMS filter on the data, I listened in the data and found there were gaps between each manatee calls. Since there was only one model was trained for the manatee calls, the training signal was better to be pure manatee call signals without any other signals. However, in the given signal, there are background signal in the gap after each manatee call and before the next one as shown in Fig. 2. Therefore, I manually marked the timestamps of each manatee call including their start point and end point by listening the training signal. After having these data as shown in Table I, I created a new training signal by merging all pure manatee calls in the original training signal to the new training signal in Fig. 3. In this case, only the manatee calls were contained in the training signal so that the LMS filter is easier to extract features of manatee call. Otherwise, the filter may take the gap as one feature of manatee call as well and will be unable to detect any manatee calls or the accuracy is very low in the test signal then. However, if multiple models for the manatee calls were trained, there would be no need to get rid of background sound but timestamps were still needed to be figure out since it is necessary to distinguish each manatee call from the whole training signal. In this experiment, the reason that only one model for manatee call is the goal is to detect manatee call instead of a call from a specific manatee. Besides that, the test signal contains 16 different calls so that the global picture of all manatee calls should be focused on.

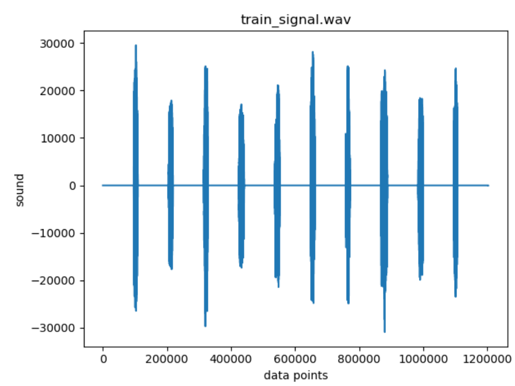

Fig. 2. Signal of manatee calls training signal.

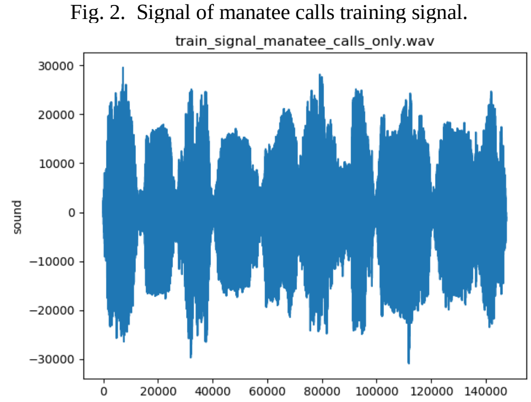

Fig. 3. Pure manatee calls in training signal.

TABLE I

MANATEE CALLS TIMESTAMP IN TRAINING SIGNAL

No. Calls	Start timestamp (s)	End timestamp (s)
1	2.002	2.276
2	4.278	4.567
3	6.569	6.843
4	8.845	9.201
5	11.203	11.506
6	13.508	13.798
7	15.800	16.093
8	18.095	18.526
9	20.528	20.855
10	22.857	23.095

This table shows the timestamp of each manatee call in the training signal.

2. *Fine tuning model order, learning rate and evaluating learning curve and MSE on manatee call data*

After preprocessing the training signal, apply LMS filter on the manatee call training signal start with model order *M = 3* as described in the guideline, and *step size = 1e-10* because the manatee call data is big and it was always out of range of double scalars when computing the MSE and then caused the running time error. Therefore, the step size had to be started with a very small value. In order to find the best model order, the step size was set the same number and model order was changed. The lower MSE the better result. This is because the MSE is the average of square errors between desired value and predicted value. The predicted signal is the same as the desired signal if the error is 0. Therefore, the signal is closed to desired signal when the average of the errors is closed to 0. However, there are some errors that are negative and some are positive. Simply computing the average would cause the error cancellation among positive and negative errors which could lead the value to be inaccuracy. For instance, a huge positive and a huge negative could cause the average to be closed to 0 but the error is pretty huge in this case. Therefore, a smaller MSE could somehow help to prove the predicted signal is good.

After finding the best model order, using the same method to find the best step size. Once find the best step size, using the step size to find the best model order on this step size again. Keep doing this until finding the best model order and step size combination. When the step size was 8e-11, the best model order was 33 where minimal MSE was 1.82 million. Then changing the model order to 33, and the best step size was 7e-11 where minimal MSE was 1.81 million. Model order of 33 also has the best step size 7e-11. Therefore, for the LMS filter, the selected parameters were 33 for model order and 7e-11 for step size as shown in Fig. 4 and 5. The relatively better MSEs were still more than 1 million which were very big due to the big training signal data.

The learning curve was shown in Fig. 6. Since MSE was computed by mean square of a list of errors, the error list became bigger and bigger and then caused the computational complexity was higher as the iteration number was bigger. Thus, computing an entire training signal would be very costly. In my experiment, it took half hour to one hour to compute MSE of each data point in the entire training signal. Therefore, MSE was computed every 1000 points in order to reduce computational complexity. And the global picture of MSE was important instead of the MSE on each data point.

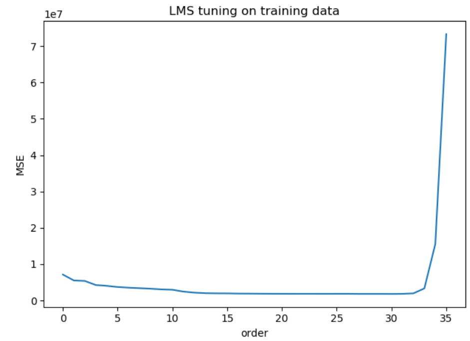

Fig. 4. Fine tuning model order at step size *7e-11*.

At this point, a list of optimization weight parameters *w* were computed at the best model order and step size. Using the last parameter *w* because the last *w* accumulated the most features of the training signal and contained all 10 different types of the manatee calls. Since the MSE were very big so that it could not say the training result were good. To verify the training quality, the output audio by LMS filter needed to be generated. Therefore, the next step is to use the selected *w* to compute the desire by multiply the input signal, and output the predicted output signal to audio file. Manually listening this predicted output audio and compared to the original desire audio signal. The result were both predicted and desired audio signals containing similar and high quality of manatee calls. Therefore, the training process was not too bad and can re-produce the manatee call signals as we desired.

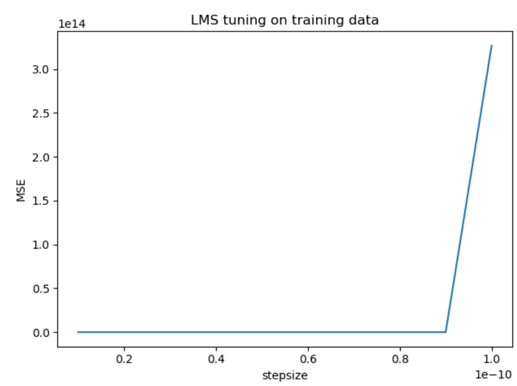

Fig. 5. Fine tuning step size at model order 33.

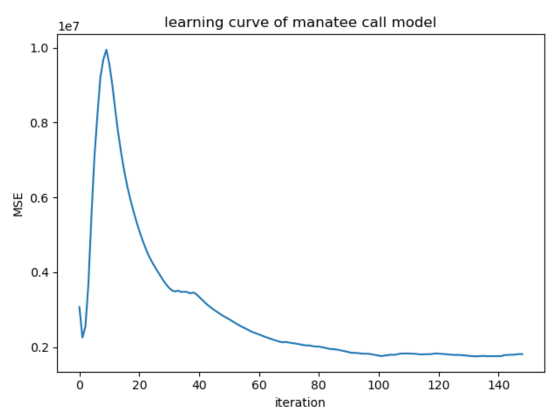

Fig. 6. Learning curve of the manatee calls model

3. *Fine tuning model order, learning rate and evaluating*

learning curve and MSE on noise data

The next model is the linear noise model. The sound wave of noise is shown in the Fig. 7. From this figure, the lower frequency background sound with the small frequency and deep color in the figure, which was listened like pure water noise, seems like stationary signal. However, the higher frequency noise in the signal was not so that this overall noise signal is not a stationary signal. After listening to the test signal, I believed that the noise in the test signal also including some other noises except the stationary noise. Therefore, LMS filter was still applied on the noise. The same procedures, strategies, and methods applied to on the manatee calls model implemented in this model. Since the noise data set was smaller, epoch was set to 2 in order to using the data more carefully and efficiently. After tuning model order and step size. The best parameters are 9 for model order and 1e-10 for step size shown in Fig. 8. The learning curve at the best parameters is shown in Fig. 9. The lowest MSE was 527,000.

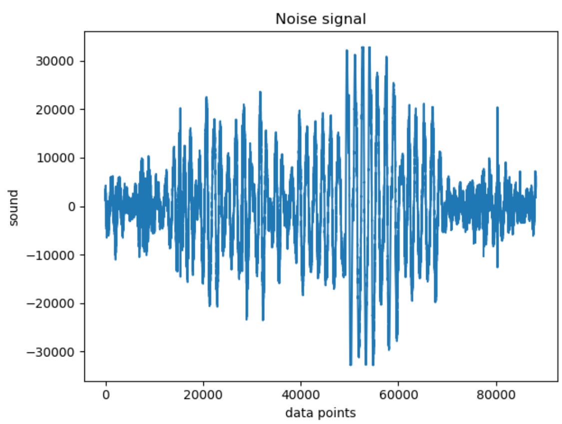

Fig. 7. Fine tuning noise model order at step size *1e-10*.

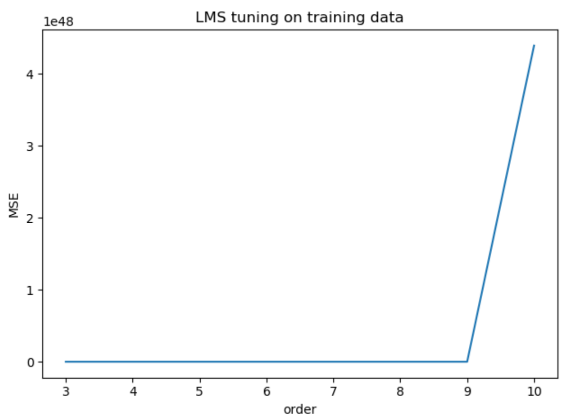

Fig. 8. Fine tuning noise model with order 9, step size *1e-10*.

At this point, both the noise model and the manatee calls model were trained. Then these two models need to be applied together on the test signal. As mentioned in the method section, these two models were implemented separately and each model could get an output and error list y_n, e_n for the noise model, and y_m, e_m for the manatee calls model. In output the noise model, it should have a better error when the signal was noise, and worse error when the signal was a manatee call, vice versa for the manatee calls model.

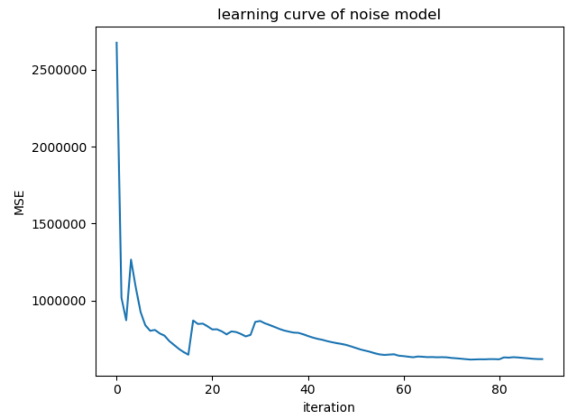

Fig. 9. Learning curve of the model under fine-tuned parameters

4. *Apply trained linear models on test signal*

Therefore, in order to combine two outputs from two models together and detected both noise and manatee call signals using their models, the error of each data point in these two models was compared. On the same data point, if e_m less than e_n, we could say the predicted error on this data point using the manatee call model was better than using the noise model. As mentioned in the method section, an average smooth method needs to be applied on the error list. The window size of the average method needs to be tuned manually by looking into the result generated. In terms of the result, the ground truth timestamps of each manatee call in the test signal had to be manually marked as marked the manatee call training signal. Table II shown the timestamps of test signal.

TABLE I
MANATEE CALLS TIMESTAMP IN TRAINING SIGNAL

No. Calls	Start timestamp (s)	End timestamp (s)
1	1.05	1.30
2	2.10	2.30
3	3.35	3.60
4	4.70	5.30
5	5.70	6.05
6	7.70	8.30
7	8.50	8.90
8	9.60	9.90
9	11.60	11.80
10	14.90	15.30
11	15.50	15.80
12	18.30	18.75
13	19.50	19.80
14	20.60	20.85
15	24.95	25.35
16	25.60	25.90

5. *Evaluation of prediction performance on test signal*

After visualizing the manually marked manatee calls as 1 and other noise as 0, the ground truth of manatee calls was shown in Fig. 10.

After testing window size from 1,000 to 10,000 with the step size 1000, the window size that had the best performance is around 7500 to 8000. The predict result of applying two models on the test signal, calculating errors and selecting an appropriate window size of average the error is shown in the Fig. 11.

In the ground truth, there are 3 calls before 200,000; 2 calls after 200,000 and still have a large distance before 400,000; 3

calls around 400,000; 1 call between 400,000 and 600,000; 2 calls between 600,000 and 800,000; 3 calls between 800,000 and 1,000,000 and 2 calls after 1,000,000, totally 16 calls.

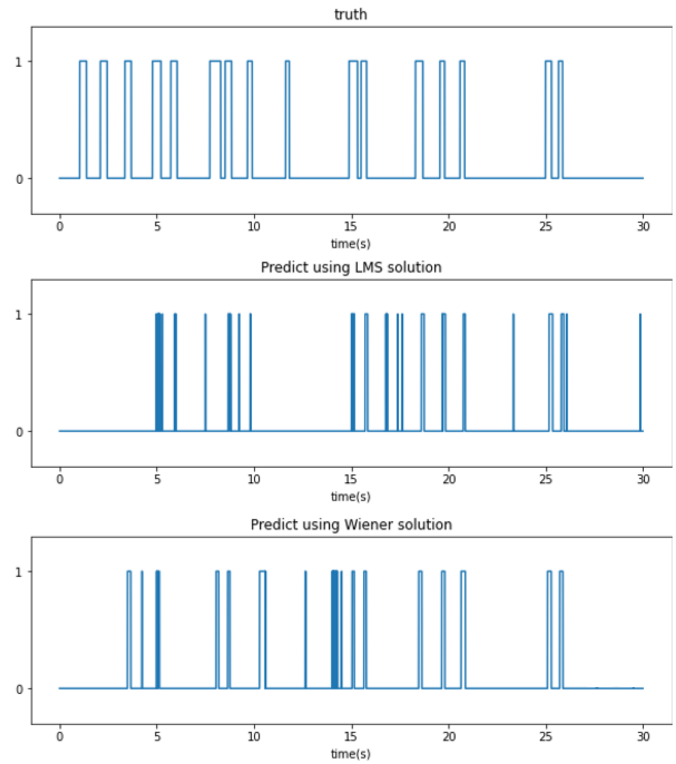

Fig. 10. Manually marked manatee calls in test signal

In the predicted manatee calls, there are 2 calls before 200,000, 2 calls after 200,000 and still have a large distance before 400,000; 3 calls around 400,000; 1 call between 400,000 and 600,000; 1 call around 600,000; 2 calls between 600,000 and 800,000 and 1 call after 1,000,000, totally 14 calls.

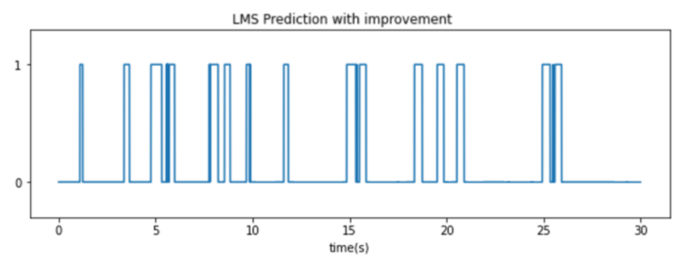

Fig. 11. Predicted manatee calls

In order to evaluate the model, two type of failures need to be figured out. The first one is the false predicted call. There are 2 out of 14 false in this model which are the 6th and 9th call in Fig. 11. Another error is the missing predicted call. There are 4 miss predicted calls where are the 2nd, 9th, 12nd and 16th call in Fig. 10. Therefore, the false rate is 14.2% and missing rate is 25%.

6. *Apply and evaluate prediction performance after adding SPRT and CUSUM models*

The following experiment is the second procedure which is using SPRT and CUSUM probability statistical model on the test signal. First of all, since the statistical test made based on the prediction error from the created model in the first procedure, the statistical information including mean and standard deviation of error of training signal and test signal needed to be computed. Then, as the method section explained, a log likelihood test needed to be done in order to determine which model had the most probability at the beginning of the test signal. $J(n) = log\frac{P_{manatee}(X(0,100))}{P_{noise}(X(0,100))}$ were computed and the result was -152.43. Since the result was a negative number when computing log using $p_{manatee}$ over p_{noise}, the noise model was chosen as beginning model. Then keep computing the probability of noise model to make sure the accumulated J of the model would not exceed the threshold. As the threshold became bigger, the detected manatee call became less. This was because a large threshold had a strict transfer model requirement. The method to see the performance is to check false and missing rate compared with ground truth manually. The best threshold after tuning was 25000 in my experiment. The result is shown in Fig. 12

There are 0 false prediction in this result and 4 missing predictions which are the 1st, 2nd, 8th and 13th manatee calls in Fig. 10. After combining result of linear model and CUSUM statistic model, the ROC curve is shown in Fig. 13.

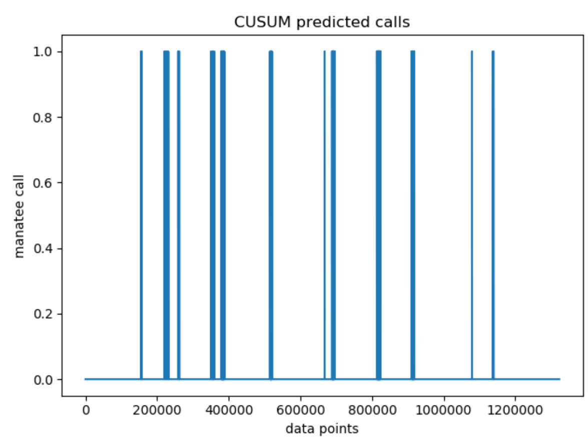

Fig. 12. CUSUM predict result

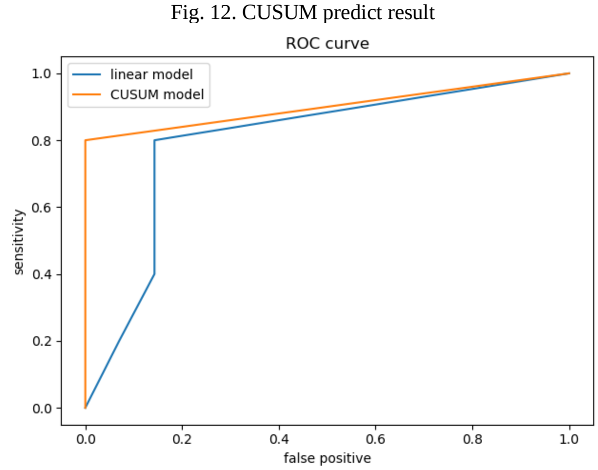

Fig. 13. ROC curve of linear and statistical model

The CUSUM model has a better ROC curve and false rate compared with LMS linear model. The missing detection rate of CUSUM is the same as the linear model. The computational complexity of CUSUM is also better than LMS. The training and calculating errors procedure are all the same for both of these two methods because CUSUM method applies on the error of linear models. For the rest procedures of these two methods, the smooth procedure in linear model takes window size multiply number of elements computational complexity, but in CUSUM model, the probability computation takes only number of elements complexity. Therefore, the overall computational complexity of CUSUM model is better than linear model on this train and test signal.

The last step was to output the result to an audio signal and listen to it. A new output signal was created by using the predicted value from different linear models depends on the label value. The audio file performed the same clearly manatee calls along with the similar timestamps as the test signal on all correctly predicted manatee calls.

IV. Conclusions

The manatee calls in a noise environment can be detected using linear predictor and probability statistical model. The performance of predictor could be better by combining linear and probability model. Analysis of primitive training and testing signal is important because we have to understand the given data before getting started to choose a method and filter. After analyzing data structure, linear and non-stationary methods were chosen on this data. Fine tuning procedure has to be done carefully in order to achieve the best performance. Evaluation of performance also need to be done very carefully and scientifically by comparing both performance using false and missing errors in order to get a better predicted result and computational complexity in order to obtain a feasible and efficient algorithm and a better running time. Listening to the predicted audio file is significantly since it is the best way to see the performance of the model.