Source localization in an ocean waveguide using supervised machine learning

Zhexi Zhang, Liyan Chen, Pin Tian, Junhao Su
Jacobs School of Engineering, University of California San Diego
La Jolla, California 92093
E-mail: zhz363@eng.ucsd.edu, jus074@eng.ucsd.edu

Abstract

Acoustic source localization is classically addressed with Matched-field processing (MFP). However, since MFP is necessarily based on accurately modeled ocean environment, it may not be well adapted as realistic ocean environment is far more complex and unpredictable. In this paper, the potential of applying machine learning techniques is explored regarding the source localization with unstable and fluctuating ocean acoustic signals. The sound pressure is preprocessed to a normalized sample covariance matrix as input data. Machine learning methods, such as feed-forward neural network (FNN), support vector machines (SVM) and random forests (RF), are proposed to be experimented. Both classification and regression are performed to address the range estimation problem. The performance for all methods is evaluated with mean absolute percentage error (MAPE), and the lowest MAPE of 2.31 % is obtained from SVM.

1. Introduction

Accordingly, Matched-field processing (MFP) was mostly used in solving acoustic source localization problems because it can localize the range, depth, and bearing of a point source from the signal field propagating in an acoustic waveguide. However, its main drawback is that MFP merely perform well under the circumstances of accurately modeled environments. Thus, introducing machine learning methods to approach this problem could potentially eliminate the preexisting problems. During the past decades, numerous researches have been conducted using machine learning techniques, and fields such as image processing [1] and natural language processing [2] are famously known to have benefited from applying these ground breaking techniques.

From the records, many of the acoustic source localization studies were based on neural networks, such as range and depth discrimination simulation with neural network [3], and classification of seafloor [4].

This paper will focus on applying various machine learning methods, such as Support vector machines, Random forest, and Feed-forward neural network, for source range localization. In particular, data preprocessing, source range mapping and Compact PCA are discussed in Sec. 2. Theoretical basis of the machine learning algorithms are provided in Sec. 3. The results are demonstrated in Sec. 4. In the end, conclusion is given in Sec. 5.

2. Data pre-processing

The complex environment of the ocean causes a stochastic relationship between pressure phase and amplitude received at sensor array and source range. By preprocessing, relationship between ship range and sample convariance is assumed to be determined. The processed data is used as input to machine learning algorithms such as FNN, SVMs and RF.

2.1. CSDM matrix

Since the processing need to be independent of the complex source spectra, pressure received at sensor array need to be transformed to a normalized sample covariance matrix.

DFT of input pressure obtained at frequency f and at L sensors is denoted as $p(f) = [p_1(f), ..., p_L(f)]^T$. And the sound pressure model is denoted as

$$p(f) = S(f)g(f, r) + \epsilon \tag{1}$$

where $S(f)$ is the source term, g is Green's function and ϵ is the noise.

Complex pressure is normalized by

$$\tilde{p}(f) = \frac{p(f)}{||p(f)||_2} \tag{2}$$

An averaging over N_s sanpshots forms the conjugate symmetric matrix of normalized sample covariance matrces (SCMs).

$$C(f) = \frac{1}{N_s} \sum_{s=1}^{N_s} \tilde{p_s}(f) \tilde{p_s}^H(f) \tag{3}$$

where H denotes conjugate transpose operator, $\tilde{p_s}$ denotes the sound pressure over the s^{th} snapshot.

The preprocessing makes sure that Green's functionis used for localization. In consideration of computing load and speed, only diagnal and upper triangle matrix is used as input to machine learning algorithms.

2.2. Source range mapping

Since source ranges are used as targets in supervised machine learning, they are discretized into K equal bins providing label t_n for each input vector x_n.These targets represents true range classes. And for FNN, targets are furthermore generated by

$$t_n k = \begin{cases} 1, if|t_n - r_k| \leq \frac{\Delta r}{2} \\ 0, \text{otherwise} \end{cases} \tag{4}$$

representing expected output probabilities of FNN.
Lables are used along with training sets to build models of FNN, SVMs and RF.

2.3. Compact PCA

To solve the problem of large data capacity and improve computing speed, principle component analysis (PCA) is introduced to this project. The idea of PCA was proposed by Karl Pearson (1901) and Harold Hotelling (1933). The aim of PCA is to "turn a set of possibly correlated variables into a smaller set of uncorrelated variables."[5]In general, high dimensional data sets contains correlated variables making most dimensions are meaningless. PCA method can find directions of dataset with greatest variance which is also called principle components.

The method can be illustrated as following:
Assuming dataset $X = x_1, x_2, ..., x_n$ with $x_i \in \mathbb{R}^d$.
1. Computing mean value

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{5}$$

2. Computing covariance matrix

$$S = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)(x_i - \mu)^T \tag{6}$$

3. Computing eigenvalues and eigenvectors

$$Sv_i = \lambda_i$$

4. Sorting eigenvalues in descending order,the largest k eigenvalues are k principle components.
Principles of observed vector x is given by

$$y = W^T(x - \mu) \tag{8}$$

And PCA basis are reconstructed by

$$x = Wy + \mu \tag{9}$$

Compact PCA method is mostly used when sample amount is less than feature amount. Since PCA forms matrix of data with most variance, they carry most of useful information of the dataset. Thus, using a few principle components can still estimate distribution of the whole dataset and reduction in dimension greatly improves computing speed.

3. Machine learning algorithms

In this paper, three commonly used machine learning algorithms are compared with the same dataset. Consider the localization problem as a multi-class classification problem, all three method of SVMs, RF and FNN are typical solving algorithms. The following part is going to introduce principles of these three algorithms.

3.1. Support Vector Machines (SVMs)

Support vector machines are common binary classifiers used in supervised machine learning. The algorithm is based on SV learning which is a method to find near optimal of functions without knowing its statistical distribution[6]. This method enables predictions only depend on training data. Since support vector machines are binary non-

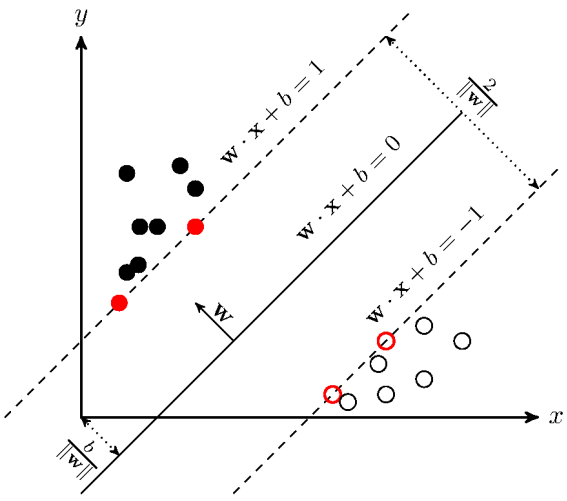


Figure 1: Support Vector Machines (SVMs)[7]

probability classifiers, the idea of SVMs is to find a separating hyperplane between two classes as illustrated in Fg.1. Assuming there exists such a separating hyperplane expressed as

$$y = w^T + b \tag{10}$$

with classes labeled by $+1$ for $y \geq 0$ and -1 for $y < 0$.

The distance between two classes can be represented by

$$\gamma = \frac{|w^T x + b|}{||w||} \tag{11}$$

Given that the aim of SVMs is to find the separating hyperplane with largest margin, the optimal solution for SVM is

$$\arg\max \frac{\gamma}{2} \tag{12}$$

Which equals to

$$\arg\min \frac{||w||^2}{2} \tag{13}$$

Here the hyperplane is assumed to be a canonical hyperplane with property

$$t_n \cdot (w^T x + b) \geq 1 \tag{14}$$

where t_n is the label corresponding to each data point. This is also a constraint condition for the optimization problem.

Since the training data is not always linear separable, other cases need to be taken into consideration. For inseparable data, slack variable $_i$ is added to the objective function in order to allow some misclassifications. And the new optimization problem can be expressed as

$$\arg\min \frac{||w||^2}{2} + C\sum \xi_i \tag{15}$$

And new constraint condition becomes

$$t_n \cdot (w^T x + b) \geq 1 - \xi_i \tag{16}$$

In many cases, classes are separated with nonlinear boundary. Then kernel trick is needed to convert data into high dimension feature spaces to find their separating hyperplane. Assuming the mapping function form data space to feature space is $\phi(x)$ and by using kernel trick we have $K(x_i, x_j) = \phi(x_i)\phi(x_j)$.

Following are some common kernels [8]:

Linear kernel:$K(x_i, x_j) = x_i \cdot x_j$

Radial based function kernel: $K(x_i, x_j) = \exp(\frac{jx_i \quad x_j j^2}{2\sigma^2})$

Polynomial kernel: $K(x_i, x_j) = (x_i \cdot x_j + 1)^d$

Sigmoid kernel: $K(x_i, x_j) = \tanh(\mu x_i \cdot x_j + \nu)$

In SVMs algorithm, only data points at boundary have influence on the hyperplane and for most data points that are properly classified, they have such property that $t_n \cdot (w^T x + b) = 0$. Thus, only support vectors have been taken into consideration in computation which greatly reduce computing complexity.

3.2. Random forest

Random forest is a commonly used methodology in supervised machine learning, mainly to find prediction rules and access and rank variables with respect to their ability to predict the response[9]. Random forest is a set of decision trees[10] as shown in Fg.2.

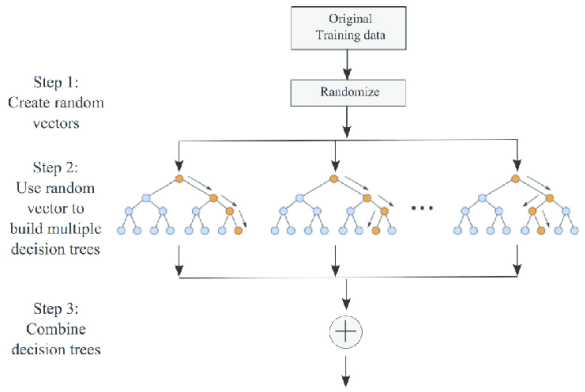


Figure 2: Ramdom Forest (RF)[11]

In classification problems, the aim of RF is to find a proper cutoff value that minimize the cost function G.

$$c \ = \arg\min G(c) \tag{17}$$

where $G(c) = \frac{n_l eft}{N} H(x_l eft) + \frac{n_r ight}{N} H(x_r ight)$, $H(\cdot)$ function is based on problem that RF is applied to and N is the total number of points.

Assuming $X = [x_1, x_2, , x_n]$ is the training set for RF algorithm, then input vector for each decision tree is randomly generated from X[12]. For each tree learner, each inner node corresponds to a variables in the input vector. Each tree is fit to their responses corresponding to their training sets, and the output of RF is an average of outputs of all trees.

$$\hat{f} = \frac{1}{B}\sum_{1} B f_b(x^0) \tag{18}$$

where B is the number of decision trees, f_b is the function of trees trained by set X_b and label Y_b that are randomly chosen from the whole training set, x' is test set or data that need its label to be predicted.

The measurement in this project is impurity function

$$H(x_m) = \frac{1}{n_m}\sum_{x_n 2 X_m} I(t_n, l_m)[1 - \frac{1}{n_m} I(t_n, l_m)] \tag{19}$$

Where l_m is assigned label for each region and expressed as

$$l_m = \arg\max \sum_{x_n 2 X_m} I(t_n, r_k) \tag{20}$$

where r_k is the source range classes and t_n is the label corresponding to point in region m, $I(\cdot)$ is the indicator function with

$$I(u,v) = \begin{cases} 0, if f = 0 \\ 1, \text{otherwise} \end{cases} \quad (21)$$

Random forest algorithm decreases variance of model without increasing bias by bagging multiple trees and averaging their decisions[13]. Given that single decision tree is highly sensitive to noise, RF algorithm trained different trees with de-correlated training set and averaging their results.

3.3. FeedForward neural network

Feedforward neural network is kind of artificial neural netwrok with no circles in connection between units. Data flow passes through the network in one direction, from input layer to hidden layer, then to output layer[14] as illustrated in Fg.3. Thus, Feedforward neural network is the simplest design of artificial neural network and was first raised. Input

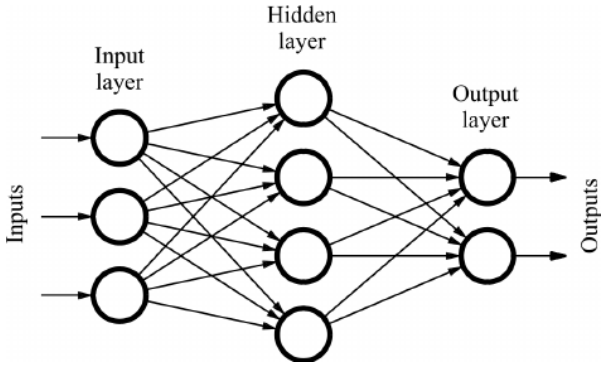


Figure 3: Feedforward Neural Networks (FNN)[15]

layer consists of multiple receptors, with each receptor for one feature. The simplest FNN has one hidden layer and is thus called single-perceptron. Hidden layer is a set of neurons, each neuron assigns inputs with respective weights and bias. Weighted inputs are accumulated at each neuron as

$$a_j = \sum_1^D w_{ji}^{(1)} + b_j^{(1)} \quad (22)$$

where D is the number of inputs.

Then a non-linear activation function is applied to these sums so that linear combinations of inputs are transformed to non-linear combinations.

Some commonly used activation functions are listed below: Log-sigmoid transfer function:

$$f(n) = \frac{1}{1 + exp(-n)} \quad (23)$$

Hyperbolic tangent sigmoid transfer function:

$$f(n) = \frac{2}{1 + exp(-2n)} - 1 \quad (24)$$

(`https://www.mathworks.com/help/nnet/ug/multilayer-neural-network-architecture.html`)

When hidden layer is not a single layer, the network is called multi-perceptron. The previous layer's output is input of the latter layer. Multiple hidden layers provide complex relationship between input and output of the network. In most cases, sigmoid funciton is used for pattern recognition and linear functions are used for function fitting.

In this project, hidden layer activation function is chosen as:

$$f(n) = \frac{1}{1 + e \;\; n} \quad (25)$$

Output layber combines outputs of hidden layer again to formulate prediction outputs with activation function:

$$y_k(x,w) = \frac{exp(a_k(x,w)}{\sum_{j=1}^K (a_j(x,w)} \quad (26)$$

which is commonly used in multi-class classification problems[12].

Since FNN is a typical kind of supervised machine learning method, training process is required to adjust weights and bias for inputs. For big data with a large amount for each data set, high performance devices are needed. To reduce computing load, this project takes advantage of the idea of random forest that generating subset of training set as input of FNN and iterate the process for multiple times to make the model converge to an optimal solution. This trick reduce computing time without losing accuracy.

The aim of training process is to minimize dissimilarity between predicting distribution and target distribution (labels).

$$\min D_K L(t_n || y(x_n, w)) = \sum_k t_n k [\ln t_n k - \ln y_n k] \quad (27)$$

where $y_n k = y_k(x_n, w)$ and $t_n k$ is the label.

The optimization problem can be equivalently expressed as minimizing cross entropy function:

$$E_n(t_n, y(x_n, w)) = -\sum_k t_n k \ln y_n k \quad (28)$$

Cross entropy can be used as a measurement of FNN performance. Since FNN takes advantage of multiple neurons in hidden layers and activation functions, it has better performance on most non-linear problems than traditional methods.

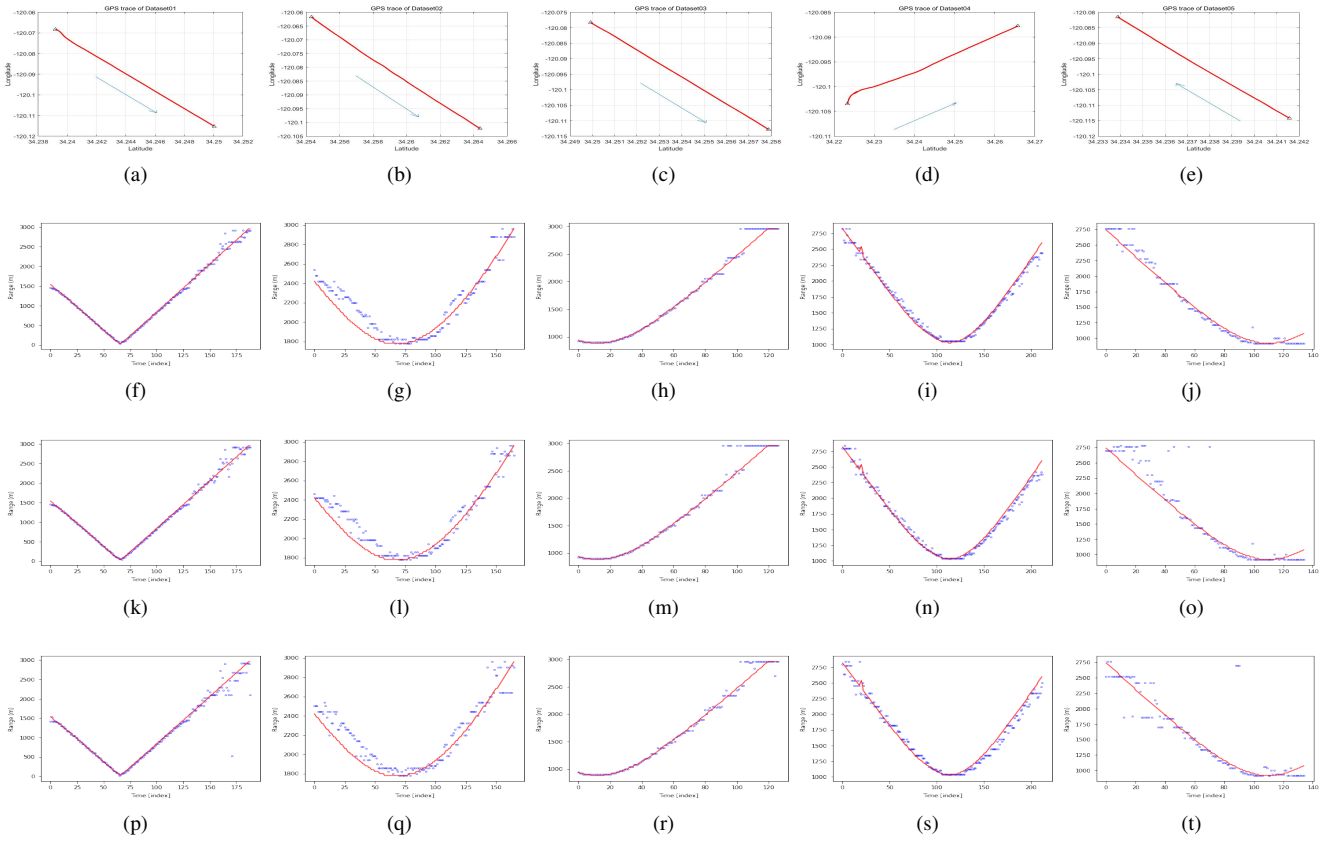


Figure 4: Comparison of three machine learning algorithms. (a)-(e) Dataset 01-05, The blue arrow is the moving direction of the ship. (f)-(j) SVM, (k)-(o) RF, (p)-(t) FNN, the blue circle and red line represents the predictions and the GPS ranges respectively.

4. Results

4.1. Dataset Selection

Firstly, the three algorithms are used on five different datasets Fg.4. And mean absolute percentage error(MAPE) is used to evaluate their performance. The formula of MAPE is shown below.[12]

$$E_{MAPE} = \frac{100}{N}\sum_{i=1}^{N}|\frac{R_{p_i} - R_{g_i}}{Rg_i}| \tag{29}$$

where R_{p_i} and R_{g_i} are the predicted range and the ground truth range, respectively. MAPE is preferred over accuracy as an error measure because it accounts for the magnitude of error in faulty range estimates as well as the requency of correct estimates. MAPE is known to be an asymmetric error measure but is adequate for the small range of outputs considered.

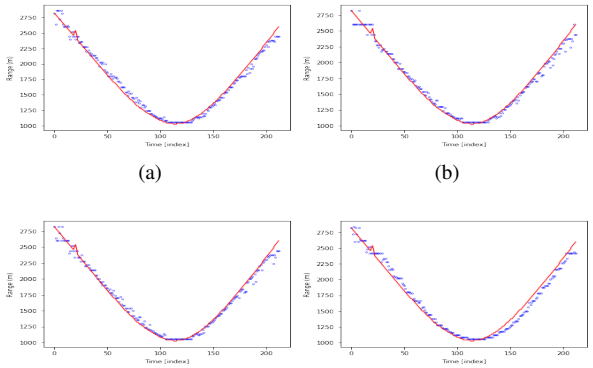
According to the Fg.5, we can see that all three algorithms have good predictions on five datasets. They have worse predictions on dataset05 and better predictions on dataset03 and 04. So, in the next parts of this section, dataset04 is used.

Figure 5: MAPE of three machine learning algorithms for the five Datasets

4.2. Determine Parameters of Three Algorithm

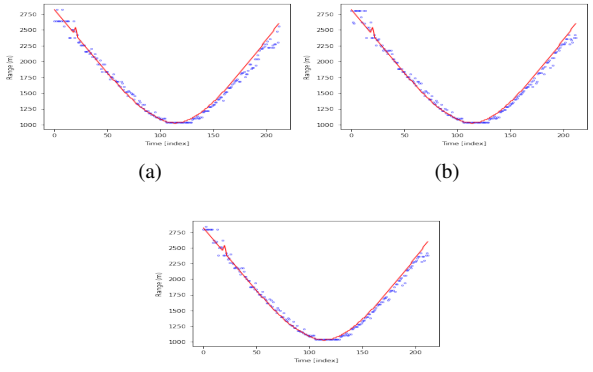
4.2.1 Kernels for SVM

There are four frequently used kernels: linear, polynomial, radial basis function(RBF) and sigmoid kernels. Penalty parameter(C) is the most important one which determines the performance of the SVMs. Gamma(γ) is another important parameter which effects the performance of polynomial, RBF and sigmoid kernel SVMs. In this evaluation, C is set to 10 and γ is set to 0.2. In Fg. 6, four kernels all have very good prediction and RBF has lowest MAPE.



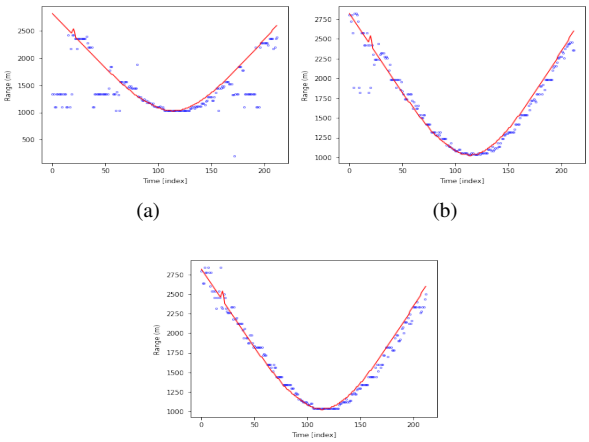
(a) (b) (c) (d)

Figure 6: Range predictions on test Dataset04 by SVM with different kernels: (a) linear(MAPE=2.84%), (b) Polynomial(MAPE=2.48%), (c) RBF(MAPE=2.31%) and (d) sigmoid(MAPE=3.62%).



(a) (b) (c)

Figure 7: Range predictions on test Dataset04 by RF with different number of decision trees: (a) 10(MAPE=2.88%), (b) 100(2.58%) and (c) 1000(2.39%).



(a) (b) (c)

Figure 8: Range predictions on test Dataset04 by FNN with different number of hidden units: (a) 10(MAPE=13.82%), (b) 50(3.83%) and (c) 100(3.41%).

4.2.2 Number of Decision Trees in RF

The number of decision trees is built before taking the maximum voting or averages of predictions. Higher number of decision trees provides better performance but makes the code slower. Normally, as high value as the processor can handle will be chosen because this makes the predictions stronger and more stable.

In this paper, 10, 100 and 1000 decision trees are employed to compare their performance because the code getting very slow when the number of decision trees is higher than 10000.

4.2.3 Number of Hidden Units in FNN

In this paper, only one hidden layer is used. So Deciding the number of neurons in the hidden layer is a very important part of deciding the overall neural network architecture. Though the layer do not directly interact with the external environment, it has a tremendous influence on the final output. Fg.8 shows the predictions on dataset04 by FNN with different number of hidden units.

Using too few neurons in the hidden layer will result in something called underfitting. Underfitting occurs when there are too few neurons in the hidden layer to adequately detect the signals in a complicated data set.

Using too many neurons in the hidden layer can result in problem, too. First, too many neurons in the hidden layer may result in overfitting. Overfitting occurs when the neural network has so much information processing capacity that the limited amount of information contained in the training set is not enough to train all of the neurons in the hidden

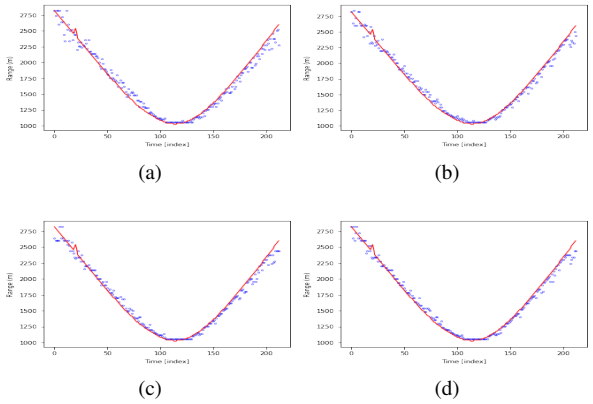
layer.

According to the three comparisons. The RBF kernel for SVM, 1000 decision trees for RF and 100 hidden units for FNN are chosen for the next part of this section.

4.3. Prediction Performance with different SNRs

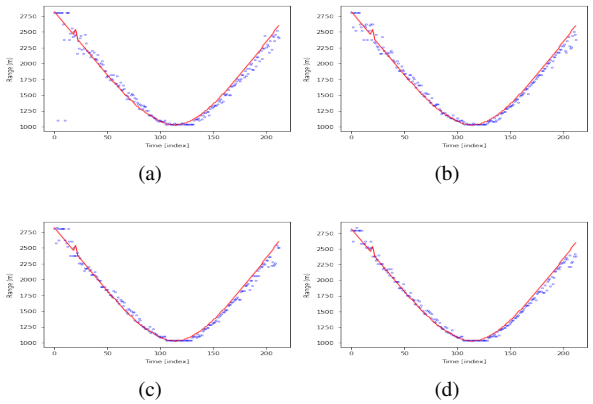
The prediction performance is examined for four SNRs(-10, -5, 0, 5dB) with three different algorithms. Fg. 9, 10 and 11 compare the range predictions by SVM, RF and FNN

It's reasonable that the MAPE decreases when the SNR getting higher. And the SVM and RF have better prediction performance than RF according to the MAPE shown in Fg. 9, 10 and 11.



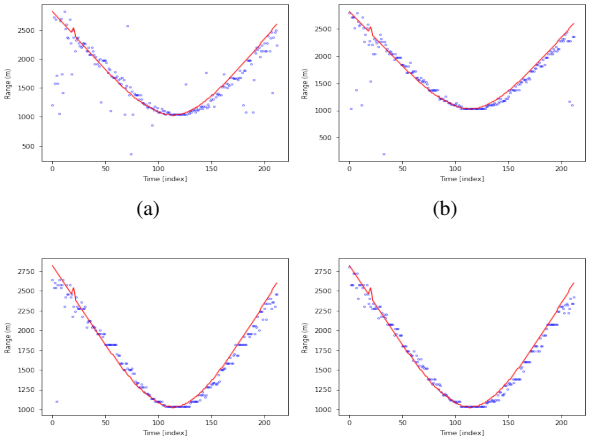
(a) (b) (c) (d)

Figure 9: Range predictions by SVM on test Dataset04 with SNR of: (a) -10(MAPE=3.22%), (b) -5(MAPE=2.67%), (c) 0(MAPE=2.63%) and (d) 5(MAPE=2.31%) dB



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

Figure 10: Range predictions by RF on test Dataset04 with SNR of: (a) -10(3.77%), (b) -5(2.81%), (c) 0(2.67%) and (d) 5(2.46%) dB



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

Figure 11: Range predictions by FNN on test Dataset04 with SNR of: (a) -10(7.93%), (b) -5(5.40%), (c) 0(3.85%) and (d) 5(3.28%) dB

4.4. Single Frequency SCM inputs for Three Algorithms

The input SCM is format at 585 Hz. It is first used to train the three algorithms separately. The RBF kernel SVM range prediction results are given in Fg.12 (a)(b); the RF range prediction results are given in Fg.12 (c)(d) and the FNN range prediction results are given in Fg.12 (e)(f).

MAPE is used to quantify the prediction performance, too. The MAPE statistics of the SVM is 9.26% at 585 Hz. The MAPE statistics of the RF is 10.55% at 585 Hz. The MAPE statistics of the FNN is 11.12% at 585 Hz.

5. Conclusion

In this paper, conventional acoustic source localization problem is addressed as a supervised learning problem. By applying several machine learning techniques, such as FNN, SVM and RF, fluctuating acoustic signals collected in unstable and complicated ocean environments are fed into each classifier which in turn produces prediction with desirable accuracy.

The results indicate that FNN performs much better for SNR above $0dB$ than for lower SNR datasets, while the other two algorithms tend to be more stable against variation in SNR. In view of datasets with high SNR, SVM and RF provide more accurate predictions with respect to MAPE.

In comparison among frequencies, single frequency data tend to be more stochastic in predictions. While multi-frequency data gives prediction source range with less MAPE. In conclusion, machine learning methods are proved to be a strong candidate in acoustic source local-

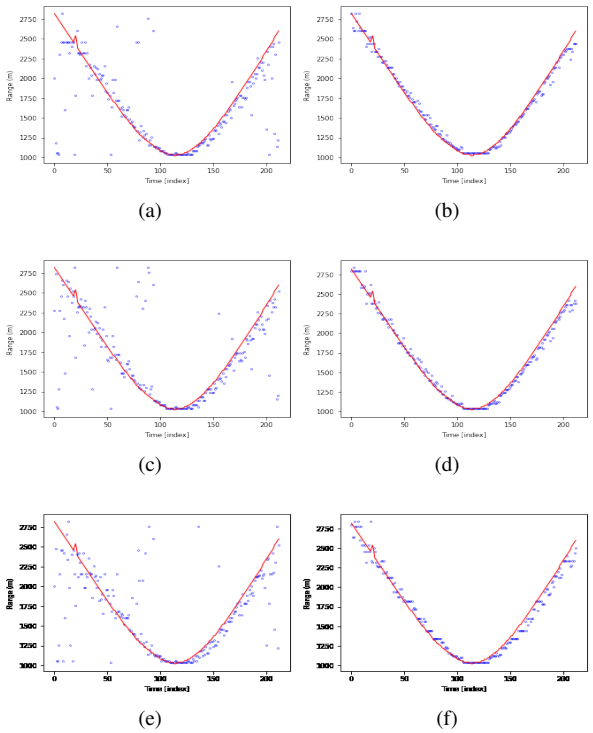


Figure 12: Range predictions on test Dataset04 by SVM(a,b), RF(c,d) and FNN(e,f). (a)(c)(e) 585Hz, (b)(d)(f) 305-711Hz

ization problems due to its excellence in handling data collected from unknown environments and robustness in dealing with various types of models.

References

[1] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[2] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel P. Kuksa. Natural language processing (almost) from scratch. *CoRR*, abs/1103.0398, 2011.

[3] John M. Ozard, Pierre Zakarauskas, and Peter Ko. An artificial neural network for range and depth discrimination in matched field processing. *The Journal of the Acoustical Society of America*, 90(5):2658–2663, 1991.

[4] Z. H. Michalopoulou, D. Alexandrou, and C. de Moustier. Application of neural and statistical classifiers to the problem of seafloor characterization. *IEEE Journal of Oceanic Engineering*, 20(3):190–197, Jul 1995.

[5] Matthew Turk and Alex Pentland. Eigenfaces for recognition. *Journal of cognitive neuroscience*, 3(1):71–86, 1991.

[6] Ingo Steinwart and Andreas Christmann. *Support vector machines*. Springer Science & Business Media, 2008.

[7] Aoife Lowery, Christophe Lemetre, Graham Ball, and Michael Kerin. *MicroArray Technology-Expression Profiling of MRNA and MicroRNA in Breast Cancer*. INTECH Open Access Publisher, 2011.

[8] Bernhard Schölkopf and Alexander J Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2002.

[9] Anne-Laure Boulesteix, Silke Janitza, Jochen Kruppa, and Inke R König. Overview of random forest methodology and practical guidance with emphasis on computational biology and bioinformatics. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2(6):493–507, 2012.

[10] Mahesh Pal. Random forest classifier for remote sensing classification. *International Journal of Remote Sensing*, 26(1):217–222, 2005.

[11] Milad Malekipirbazari and Vural Aksakalli. Risk assessment in social lending via random forests. *Expert Systems with Applications*, 42(10):4621–4631, 2015.

[12] Haiqiang Niu, Peter Gerstoft, and Emma Reeves. Source localization in an ocean waveguide using supervised machine learning. *arXiv preprint arXiv:1701.08431*, 2017.

[13] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An introduction to statistical learning*, volume 6. Springer, 2013.

[14] David J Montana and Lawrence Davis. Training feedforward neural networks using genetic algorithms. In *IJCAI*, volume 89, pages 762–767, 1989.

[15] Ramón Quiza and J Paulo Davim. Computational methods and optimization. In *Machining of hard materials*, pages 177–208. Springer, 2011.