Optimizing Machine Learning Algorithms for Dynamic Direction Finding

John M. Willis, Satheesh B. Venkatakrishnan, and John L. Volakis
Florida International University, Miami, Florida, USA (jwill567@fiu.edu)

***Abstract*—We build, analyze, and compare six machine learning algorithms for direction-finding on dynamic systems. Each model is optimized to decrease its computational complexity to make quicker angle-of-arrival estimates and maintain accuracy. The extracted angle determination compares favorably in terms of accuracy and speed to the MUSIC algorithm, with some models exceeding its performance.**

Keywords— direction finding; machine learning; optimization

I. Introduction

Machine learning (ML) has the potential to reduce time of design and in this particular case, to increase angle of arrival accuracy. Direction finding (DF) applications are well suited for ML, and numerous algorithms have been tested and proven effective[1]–[3]. This is because ML can perform complex calculations quicker, especially for real-time, dynamic systems. Notably, DF systems with fast-moving targets must limit the number of snapshots. This snapshot limit can exclude the use of specific conventional techniques where time is critical.

In previous research, individual ML methods have been investigated and compared to conventional beamforming and subspace techniques. However, most research is limited to a single model, with a few considering two or three [4]. Fewer studies take the application beyond the azimuthal plane. This work compares six ML algorithms and determines which produces the quickest, most-accurate results for dynamic 2-D DF. Even though ML reduces the computational complexity as compared to conventional methods, it can still have significant overhead. For this reason, this work also optimizes models to perform at high speeds for dynamic DF.

II. Building The ML Models For DF

Machine Learning can be split into regression and classification models. A regression model maps the input to a continuous range of output values, whereas classification models work with discrete spacing, called classes. All of our models were built using classification algorithms. Classification algorithms were selected because they are generally faster. Fast computation is often most critical with moving targets, and a coarser angular determination is, typically, an acceptable tradeoff.

A. Input Selection

DF techniques for array systems can be grouped into beamforming and subspace methods. Both methods use the autocorrelation (AC) matrix as input. The AC matrix is an element-by-element correlation, and as the array size increases, the size of the matrix grows exponentially. Additionally, each matrix value is complex with a real and imaginary component. For example, a 4x4 element array has a 16x16 AC matrix with a total of 256 complex values. Furthermore, as shown in (1), the matrix is symmetric and Toeplitz. Consequently, when used as an ML input, over half the values are redundant [5]. Taking advantage of redundancy can reduce the 512 entries down to 240.

$$\mathrm{R}_{xx} = \begin{bmatrix} E[X_1X_1] & E[X_1X_2] & E[X_1X_3] & E[X_1X_4] \\ E[X_2X_1] & E[X_2X_2] & E[X_2X_3] & E[X_2X_4] \\ E[X_3X_1] & E[X_3X_2] & E[X_3X_3] & E[X_3X_4] \\ E[X_4X_1] & E[X_4X_2] & E[X_4X_3] & E[X_4X_4] \end{bmatrix} \quad (1)$$

B. Output Selection

For DF classification, the resolution of the algorithm is directly tied to the number of classes [6]. Of course, the more classes used to cover a region, the finer the resolution. For example, a 10° spacing in azimuth and elevation over a hemisphere would require 324 sectors or classes. The type of antenna, array configuration, and system application can reduce the needed number of classes. For our application a uniform rectangular array is used that is assumed to rotate its beam. With an angular rotation of π rad/s, we opted for a tighter resolution versus full azimuthal coverage. Instead, ten azimuthal classes were used over the half-power beamwidth and nine elevation classes over the 90° to 120° range. This sectorization of space gave an azimuthal resolution of 2.5° and an elevation resolution of 6.7°. The sectors for the 90 classes can be seen in Figure 1.

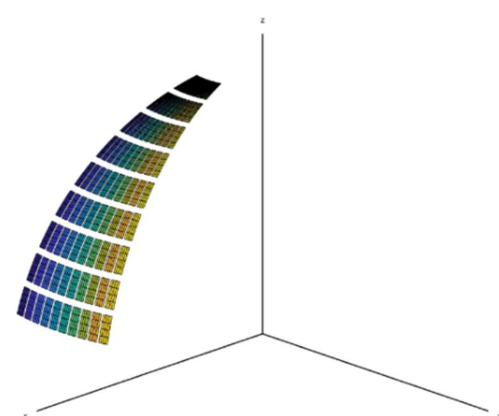

Figure 1. Output sectors for 10 classes over the array's half-power beamwidth and nine classes over 90° to 120° in elevation. The pattern is rotated around the z-axis providing full 360° coverage.

C. Models and Training Data Creation

Six ML algorithms were considered: (1) k-nearest neighbors (KNN), (2) tree, (3) discriminant analysis, (4) support-vector machine (SVM), (5) random forest, and (6) ensemble KNN. These six algorithms were chosen to compare distance-, distribution-, and ensemble-based classification models. The models were all trained through a supervised learning approach on data created in MATLAB simulation. The training data simulated a target transmitting at one-degree increments from -45° to 45° in azimuth and 60° to 120° in elevation. At each degree, additive white Gaussian noise was added to the incoming signal giving SNR = 0, 2, 4, 6, 8, and 10. The above settings resulted in a training set of 99918 truncated-autocorrelation matrices covering the region.

III. Model Optimization

The aforementioned six models with new simulated test data on a rotating platform were compared. The initial computation speed for the six models, averaged over 5000 calculations, is depicted by the blue lines in Figure 2. Only the tree model was faster than the MUSIC algorithm baseline (red dashed line). Several techniques were applied to these initial models to decrease the angle determination timing.

A. Splitting the Model

The first speed-up technique was splitting the models into two, one for azimuth and another for elevation. Breaking the single 90-class model resulted in using ten azimuthal classes and nine elevation classes. When the models were separated, SVM and discriminant analysis calculation times improved dramatically. However, the other four models had longer calculation times. This contradiction is likely a function of the overhead of the algorithm as compared to the savings from class reduction. The two improved models are more computationally involved. This is because, SVM requires the calculation of multiple hyperplanes, and discriminant analysis does a statistical distribution for every class. Despite the inconclusive results, all the models saw a slight increase in misclassification.

B. Dimensionality Reduction

Dimensional reduction was applied to the feature space through principal component analysis (PCA). When reducing the feature space, some less relevant data must be discarded. Four different PCA transformations were applied to reduce the relevant data to 99%, 90%, 80%, and 50%. These reductions reduced the original 240 feature vector to 27, 16, 12, and 6, respectively. The 99% feature reduction decreased the calculation time for all the models. Surprisingly, each model also saw an improvement in the classification rate. However, further reduction from 99% to 50% did not produce a substantial impact on speed. That said, it did produce a slight increase in the misclassification rate for each step. The 90% reduction was chosen as the optimal compromise between speed and performance.

C. Snapshot Reduction

DF systems, especially those utilizing array processing methods, may require many snapshots for an accurate estimation. The MUSIC algorithm, in particular, needs hundreds of snapshots to resolve between two targets. All training data used the ergodic average of 500 snapshots for the AC matrix. Reducing snapshots can help improve an algorithm's calculation speed and allow it to track fast-moving targets in real time. This snapshot reduction helped increase decision speed for each model with negligible impact on the accuracy. Notably, reducing the KNN model down to 200 snapshots only increased the misclassification rate by 0.013%.

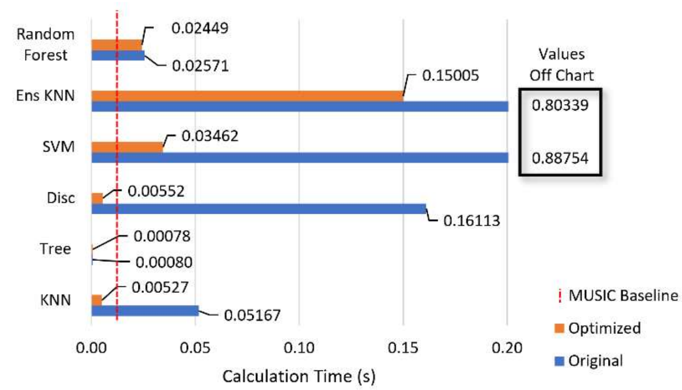

Figure 2. Calculation times the ML models for 5000 calculations. Blue lines: original calculation speed; Orange lines: optimized calculation speed; Red dash: MUSIC algorithm (baseline).

IV. Conclusion

Multiple approaches were considered to reduce the computational burden for conventional direction-finding techniques. ML was used, with various tweaks to provide dramatically improved results based on application type. Each of the six considered models had improved calculation speeds, and some were faster and more accurate than the MUSIC algorithm. The optimized calculation speeds are shown by the orange lines in Figure 2.

References

[1] M. Yang et al., "Machine-Learning-Based Fast Angle-of-Arrival Recognition for Vehicular Communications," IEEE Trans. Veh. Technol., vol. 70, no. 2, pp. 1592–1605, 2021, doi: 10.1109/TVT.2021.3054757.

[2] L. L. Wu and Z. T. Huang, "Coherent SVR Learning for Wideband Direction-of-Arrival Estimation," IEEE Signal Process. Lett., vol. 26, no. 4, pp. 642–646, 2019, doi: 10.1109/LSP.2019.2901641.

[3] G. R. Friedrichs, M. A. Elmansouri, and D. S. Filipovic, "A Compact Machine Learning Architecture for Wideband Amplitude-Only Direction Finding," IEEE Trans. Antennas Propag., no. DL, pp. 1–1, 2021, doi: 10.1109/tap.2021.3137505.

[4] A. Khan, S. Wang, and Z. Zhu, "Angle-of-Arrival Estimation Using an Adaptive Machine Learning Framework," IEEE Commun. Lett., vol. 23, no. 2, 2019, doi: 10.1109/LCOMM.2018.2884464.

[5] Z. T. Huang, L. L. Wu, and Z. M. Liu, "Toward Wide-Frequency-Range Direction Finding with Support Vector Regression," IEEE Commun. Lett., vol. 23, no. 6, pp. 1029–1032, 2019, doi: 10.1109/LCOMM.2019.2910253

[6] A. M. Elbir, "DeepMUSIC: Multiple Signal Classification via Deep Learning," IEEE Sensors Lett., vol. 4, no. 4, pp. 2020–2023, 2020, doi: 10.1109/LSENS.2020.2980384