


RESEARCH

Open Access



Multi-headed deep learning-based estimator for correlated-SIRV Pareto type II distributed clutter

Taha Hocine Kerbaa^{1,3}, Amar Mezache^{1,4*} , Fulvio Gini² and Maria S. Greco²

*Correspondence:
amar.mezache@univ-msila.dz

¹ Department of Electronics,
University Mohamed Boudiaf-
M'sila, 28000, M'sila, Algeria

² University of Pisa, Pisa, Italy

³ Laboratoire d'Analyse des
Signaux et Systemes (LASS),
University Mohamed Boudiaf-
M'sila, 28000, M'sila, Algeria

⁴ SISCOM Laboratory,
Department of Electronics,
University Constantine1,
25010 Constantine, Algeria

Abstract

This paper deals with the problem of estimating the parameters of heavy-tailed sea clutter in high-resolution radar, when the clutter is modeled by the correlated Pareto type II distribution. Existing estimators based on the maximum likelihood (ML) approach, integer-order moments (IOM) approach, fractional-order moments (FOM), and log-moments (log-MoM) have shown to be sensitive to changes in data correlation. In this work, we resort to a deep learning (DL) approach based on a multi-headed architecture to overcome this problem. Offline training of the artificial neural networks (ANN) is carried out by using several combinations of the clutter parameters, with different correlation degrees. To assess the performance of the proposed estimator, we resort to Monte Carlo simulation, and we observed that it has superior performance over existing approaches in terms of estimation mean square error (MSE) and robustness to changes of the clutter correlation coefficient.

Keywords: Parameter estimation, Pareto, SIRV, Correlation, Deep learning, CNN, LSTM, Autoencoder

1 Introduction

The construction of systems that distinguish objects of interest buried in background disturbance constitutes a significant challenge in radar signal processing. In maritime surveillance applications, sea clutter represents a dominant undesired signal that affects radar performance. Besides the applicability in real-time scenarios, an effective detection performance should satisfy the adaptivity to clutter power spectral density and, more broadly, to the clutter multidimensional probability density function (pdf) while maintaining a constant false alarm rate (CFAR). Consequently, the prior knowledge of clutter distribution enhances target detection capabilities in white or correlated non-Gaussian environments. Improvements in radar system technology and the increase in signal processing applications in scale, complexity, and power cancel in fact the stationarity and Gaussianity assumptions that could not adequately reflect the statistical characteristics of the environment [1]. Related research works revealed that high-resolution radar scatters contain spikes giving rise to non-Gaussian heavy-tailed measurements. Thus, different heavy-tailed distributions were proposed to model

clutter returns from high resolution and low grazing angles, this includes log-normal, Weibull, Pareto, and K-distribution [2–5]. Depending on the data acquisition geometry, the environmental conditions, and the radar parameters, the sea clutter echoes could be correlated in temporal and/or spatial components. However, the idealistic consideration of independent and identically distributed (iid) clutter returns affects the detection and leads to highly suboptimum performance. Therefore, the derivation of models characterizing non-Gaussian correlated clutter was addressed in [6, 7] based on a general approach used in [8] for formulating the multivariate pdf of a correlated non-Gaussian random vector and extended to incorporate complex random vectors for log-normal or Weibull distributed clutter.

Available approaches for simulating correlated sea clutter are primarily divided into two categories: the class of memoryless nonlinear transformation (MNLT) and the spherically invariant random process-based approach (SIRP). The MNLT technique is broadly inspected in [9], and the method's downside lies in its incapacity to manipulate both the PDF vectors and the power spectrum. On the other hand, the nonlinear mapping involved in the generation process could not satisfy the Hermitian property of the covariance matrix. As a second approach, Conte and Longo [10] suggested that the radar clutter process can be represented by a relatively straightforward approach based on the SIRP. The latter provides a mathematically tractable tool for modeling the pdf of correlated heavy-tailed non-Gaussian random variables avoiding the nonlinear transformation and ensuring the control of the PDF vectors' along with the power spectrum. Sampling SIRP results in a spherically invariant random vector (SIRV), and authors of [11, 12] detailed the properties in conjunction with consistent PDFs with the SIRV model. The distribution of the random vector associated with a clutter modeled as a SIRV process forms a compound-Gaussian model interpreted as a locally Gaussian process with a random local power variable.

Estimation of clutter model parameters forming the background level, in which the target of interest is embedded, is an essential part in CFAR detection schemes. Since the statistical models of clutter vary with the application scenario, parameter estimation methods will also change with clutter nature. As a result, several approaches have been offered based on least squares (curve fitting), matching of moments, and maximum likelihood (ML) approaches. For instance, numerous estimators have been constructed to deal with Compound Gaussian (CG) clutter, with and without additive thermal noise [13–16]. The integer-order moments approach is commonly used for parameter estimation of numerous clutter models and provides suboptimal results [14]. For noiseless case, it was shown that non-integer-order moments and log-moments could be manipulated to get closed-form estimators labeled FOME (Fractional Order Moments) and $[z\log(z)]$ methods, respectively, for K , Pareto type II (GP), and CGIG (Compounded Gaussian Inverse Gaussian) radar clutter parameters [14, 15].

In terms of sea clutter model types, several authors have considered different decision rules that are independent of true clutter parameters [17–20]. Several CFAR detectors operating in homogeneous and heterogeneous Pareto type II clutter are derived when the shape parameter or the scale parameter of the Pareto type II model is known a priori [19]. If these parameters are unknown, an alternative procedure

based upon the Bayesian approach is introduced in which a modified decision rule is given in integral form [20].

With the rapid emergence of the different machine learning paradigms, deep learning (DL) techniques have gained significant interest and motivated several advances in radar signal processing and target detection/classification applications [21–25]. In the context of parameter estimation, several deep neural networks architecture were proposed [26, 27]. In more details, a hybrid CNN-LSTM-based approach for estimating the shape parameter in a K + noise distribution for a single look case was developed in [27], and better estimation performance was achieved against the classical higher-order moment (HOME) and $[z\log(z)]$ estimators with improved execution time. Authors of [27] suggested that estimating clutter parameters could be seen as a many-to-one sequence prediction problem and confirm the effectiveness of deep neural networks in learning pertinent features from the environment returns leading to a reasonable estimation performance of the shape parameter. Furthermore, given a specific theoretical distribution, the estimation of the clutter parameters may be conveyed as a time series forecasting problem. DL methods have shown noticeable performance in solving this kind of problem.

Based on the compound-Gaussian model, the problem of radar detection in a non-Gaussian correlated environment has been considered in many works [28–32] where the clutter covariance matrix is estimated using the range clutter profile (CRP). Gini and Greco [33] proposed an iterative approximate maximum likelihood estimator (AML) for covariance matrix estimation with unknown parameters and a minimal number of iterations while maintaining the detector's CFAR property. Recently, Sangston et al. [34] proposed a parametrically dependent detector referred to as GLRT-LTD algorithm for correlated non-Gaussian clutter. The robustness of the GLRT-LTD detector was investigated via true values of the shape and scale parameters of an inverse-Gamma texture. For a realistic situation, unknown parameters should be estimated from the collected samples. Already in [34], the parameter estimation of the Pareto model was addressed using the higher-order moments approach (see eqs. (43) and (44) in [34]), leading to what is called the Adaptive GLRT (AGLRT). However, this estimator has been proposed for independent and identically distributed (iid) clutter samples. For very spiky clutter (i.e., shape parameter less than 2), moments of order 1/2 and 1 should be used, whereas moments of order 2 and 4 can be exploited only when the shape parameter is greater than 1.

As available estimators cited above cannot ensure good estimates of the parameters when the clutter data are correlated, in this work we propose the implementation of a novel multi-headed based estimator of the clutter parameters, including the CNN, stacked LSTM, CNN-LSTM, bidirectional LSTM, and LSTM Auto encoder architectures, to provide better estimates of Pareto type II clutter parameters. After the training task of the proposed approach using synthetic sea clutter data, the MSE metric test is improved against the changes of the correlation coefficient. Since the GLRT-LTD detector presented in [34] necessitates the knowledge of the clutter parameters, the application of the deep learning-based estimator plays two roles: maximization of the detection probability owing the best estimates of the parameters and improvement of the processing time due to the forward computing of the estimates.

In this work, the problem of estimating clutter parameters in the case of correlated data is addressed using a deep learning (DL) multi-headed architecture approach, and an offline training of the artificial neural networks (ANN) is carried out by using several combinations of the clutter parameters, with different correlation degrees for generalization purposes. To assess the performance of the proposed estimator, comparisons with the conventional approaches are carried out using generated correlated clutter data. The proposed deep learning estimator demonstrates its superior performance over existing approaches in terms of estimation mean square error (MSE) and robustness to changes of the clutter correlation coefficient.

The rest of the paper is structured as follows: Section 2 deals with the problem formulation of estimating parameters in correlated compound-Gaussian environments. In the case of uncorrelated samples, classical estimators of the multivariate Pareto type II clutter parameters are presented in Sect. 3. A more in-depth discussion of the appropriate deep learning models for the prediction scheme is reported in Sect. 4, along with the validation of the proposed estimator. In Sect. 5, the performance of the proposed estimator is compared with existing approaches against correlated clutter data. Finally, in Sect. 6, conclusions are drawn and further work is proposed.

2 Problem statement

Estimating parameters of the clutter model can be handled, conceptually, as a many-to-many sequence prediction where the input sequence is formed by values of the clutter returns and the output vector contains the estimates of the shape and the scale parameters of the clutter model. To this end, a prediction model is trained on a set of training sequences via deep learning algorithms.

After it has been trained, the model uses a forward calculation which is equivalent to closed form estimators to estimate the unknown parameters.

Let us assume coherent pulses echoes from a radar transmitter, then the m complex samples from the scalar time series $\{z[m], m = 1, 2, \dots\}$ can be assembled into an m -dimensional vector $\mathbf{z} = [z[1], z[2], \dots, z[m]]^T$ where T denotes the transpose operator. The detection problem is considered as a binary hypothesis testing on a complex signal containing the in-phase and quadrature components such that:

$$\begin{aligned} H_0 : \mathbf{z} &= \mathbf{c} \\ H_1 : \mathbf{z} &= \mathbf{s} + \mathbf{c} \end{aligned} \tag{1}$$

Under the null hypothesis, the measured vector \mathbf{z} contains only clutter returns \mathbf{c} , i.e., $\mathbf{z}|H_0 = \mathbf{c}$. For a compound-Gaussian modeled clutter, the SIRV \mathbf{c} consists of the product of two independent random variables such that:

$$\mathbf{c} = \sqrt{\tau} \mathbf{x} \tag{2}$$

where the speckle \mathbf{x} is a complex Gaussian variate with zero mean, unit variance and represents the properties of the coherent radar sensor. The covariance matrix $E\{\mathbf{x}\mathbf{x}^H\} = \mathbf{M}$ is positive definite Hermitian, then, in shorthand $\mathbf{x} \sim \mathcal{CN}(0, \mathbf{M})$ and normalized such that $\text{Tr}\{\mathbf{M}\} = m$, where $\text{Tr}\{\mathbf{M}\}$ is the trace of \mathbf{M} . The texture τ is a positive random quantity and represents the local power of the clutter in the cell under test (CUT). The average clutter power is given by $E\{\tau\}$. We assume here that the texture and the speckle are

statistically independent, and the clutter covariance matrix \mathbf{M} is assumed to be perfectly known. Given a specific value of τ , $\mathbf{c}|\tau \sim \mathcal{CN}(0, \tau\mathbf{M})$, the multivariate m th order pdf under the null hypothesis can be derived:

$$p_{\mathbf{z}|\tau, H_0}(\mathbf{z}|\tau, \mathbf{H}_0) = p_{\mathbf{c}|\tau}(\mathbf{z}|\tau) = \frac{1}{\pi \tau^m |\mathbf{M}|} \exp\left(-\frac{\mathbf{z}^H \mathbf{M}^{-1} \mathbf{z}}{\tau}\right) \tag{3}$$

$|\cdot|$ denotes the determinant of a matrix.

The pdf of $\mathbf{z}|H_0$ is calculated by averaging $p_{\mathbf{z}|\tau, H_0}(\mathbf{z}|\tau, \mathbf{H}_0)$ over τ as:

$$p_{\mathbf{z}|H_0}(\mathbf{z}|\mathbf{H}_0) = E_{\tau} \{p_{\mathbf{z}|\tau, H_0}(\mathbf{z}|\tau, \mathbf{H}_0)\} = \int_0^{\infty} \frac{1}{\pi \tau^m |\mathbf{M}|} \exp\left(-\frac{\mathbf{z}^H \mathbf{M}^{-1} \mathbf{z}}{\tau}\right) p_{\tau}(\tau) d\tau \tag{4}$$

The texture τ is modeled by an inverse-gamma pdf:

$$p_{\tau}(\tau) = \frac{1}{\Gamma(\lambda)} \left(\frac{\lambda}{\eta}\right)^{\lambda} \frac{1}{\tau^{\lambda+1}} \exp(-1/b\tau) \tag{5}$$

where $\lambda > 0$ is the shape parameter, $\eta > 0$ is the scale parameter and $\Gamma(\cdot)$ is the Gamma function. With $b = \frac{\eta}{\lambda}$, (5) becomes:

$$p_{\tau}(\tau) = \frac{1}{(\lambda)} \frac{b^{-\lambda}}{\tau^{\lambda+1}} \exp(-1/b\tau) \tag{6}$$

From [34], the expression of the GLRT-LTD is given by:

$$q_0 - q_1 = \frac{|\mathbf{p}^H \mathbf{M}^{-1} \mathbf{z}|^2}{\mathbf{p}^H \mathbf{M}^{-1} \mathbf{p}} \underset{H_0}{\overset{H_1}{\geq}} \left(1 - e^{-T/m+\lambda}\right) \left(\frac{\lambda}{\eta} + q_0\right) \tag{7}$$

The right-hand side of (7) is the adaptive threshold which is formulated in terms of T , m , λ , η and q_0 , where $q_0(\mathbf{z}) = \mathbf{z}^H \mathbf{M}^{-1} \mathbf{z}$ is obtained from the m correlated data, $q_1(\mathbf{z}) = q_0(\mathbf{z} - \mathbf{s})$, $\mathbf{s} = \alpha \mathbf{p}$ is the target vector, α is the unknown complex amplitude and \mathbf{p} is the steering vector. T is the threshold multiplier which is computed from the desired probability of false alarm P_{FA} . The P_{FA} of the detector (7) is the following [34]

$$P_{FA} = \exp\left(-T \frac{m + \lambda - 1}{m + \lambda}\right) \tag{8}$$

The GLRT-LTD algorithm depends on the clutter parameters. Replacing the unknown values of the clutter parameters λ and η with their estimates leads to what is called the Adaptive GLRT-LTD detector. For a realistic case, we need to collect many samples to estimate the clutter parameters.

In the literature, there are no estimators that consider explicitly data correlation. Most existing estimators of clutter parameters were derived under the assumption of independent and identically distributed (iid). If iid Pareto type II clutter is assumed, methods that employ the maximum likelihood principle or the method of moments are found in [35]. Those estimators are implemented in this work to identify the impact of the correlation coefficient on the estimation accuracy and a novel approach based on a trained deep learning predictor is proposed to estimate the

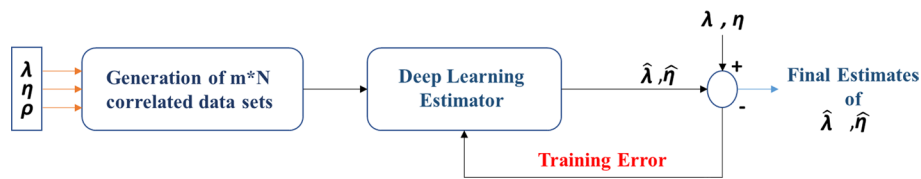


Fig. 1 Synoptic of deep learning-based estimator of correlated Pareto type II clutter

shape and scale parameters of correlated Pareto type II model. As shown in Fig. 1, this approach covers two stages: an offline training process for estimation purposes and validation stage of the estimator. The offline part consists of generating massive training data using different couples of $\{\lambda, \eta\}$ and train the deep neural network as a supervised many to many sequence prediction problems. In addition, the correlation coefficient ρ is also considered in the adaptive scheme of the Deep learning approach for a generalization purpose. After validating the DL-based estimator, the discrimination of the sea clutter with different shape parameters from objects of interest is carried out by replacing the real parameter values with their estimates in the GLRT-LTD test statistic.

An overview of the standard estimators of the shape and scale parameter of the Pareto type II distribution is presented in the next section.

3 Parameter estimation of multivariate Pareto type II clutter model

In this section, four estimation methods of Pareto type II clutter parameters are presented named IOM (Integer-Order Moments), NIOM (Non-Integer-Order Moments), $[z\log(z)]$, and MLE (Maximum Likelihood Estimator). Let y_1, y_2, \dots, y_N be a random sequence for the Pareto type II clutter given at the output of the envelope detector [36], with

$$p_Y(y) = \frac{2y\eta\Gamma(\lambda + 1)}{\lambda(\frac{\eta}{\lambda}y^2 + 1)\Gamma(\lambda)} \tag{9}$$

Moments of order k should be determined to estimate the two unknown parameters λ and η from the data using IOM and NIOM techniques.

$$\mu_k = \left(\frac{\lambda}{\eta}\right)^{k/2} \frac{\Gamma(1 + k/2)\Gamma(\lambda - k/2)}{\Gamma(\lambda)} \tag{10}$$

which converges only if $\lambda > k/2$.

3.1 Integer-order moments (IOM) method

From (10), the texture parameters, i.e., λ and η , can be obtained by equating the analytical expectations of moments of order 2 and 4 to their sample counterparts, i.e.,

$$\hat{\mu}_k = \frac{1}{N} \sum_{i=1}^N y_i^k. \text{ Hence,}$$

$$\begin{cases} r\mu = \frac{\hat{\mu}_4}{\hat{\mu}_2^2} \\ \hat{\lambda} = \frac{2r\mu - 1}{r\mu - 1} \\ \hat{\eta} = \frac{\hat{\mu}_2}{\hat{\mu}_4} (2r\mu - 1) \end{cases} \tag{11}$$

Recall that the constraint $\lambda > k/2$, for small values of λ , estimation of (1) is unachievable via (11). Thus, it is suggested to use FOM method for values of k which are not necessary integers [36].

Degraded estimation performances are observed for low sample sizes in spiky clutter situations.

3.2 Fractional order moments (FOM) method

Moments of order 1 and 1/2 are chosen to estimate numerically the underlying parameters. Hence,

$$\mu_1 = \left(\frac{\lambda}{\eta}\right)^{1/2} \frac{\Gamma(3/2)\Gamma(\lambda - 1/2)}{\Gamma(\lambda)} \tag{12}$$

$$\mu_{1/2} = \left(\frac{\lambda}{\eta}\right)^{1/4} \frac{\Gamma(5/4)\Gamma(\lambda - 1/4)}{\Gamma(\lambda)} \tag{13}$$

Solving the above equations simultaneously and after some algebraic operations we get

$$\begin{cases} \frac{\Gamma(3/2)^2\mu_{1/2}}{\Gamma(5/4)\mu_1^2} = \frac{\Gamma(\hat{\lambda}-1/4)}{\Gamma(\hat{\lambda})\Gamma(\hat{\lambda}-1/2)^2} \\ \hat{\eta} = \hat{\lambda} \left(\frac{\mu_1\Gamma(\hat{\lambda})}{\Gamma(3/2)\Gamma(\hat{\lambda}-1/2)} \right)^{-2}, \lambda > 1/2 \end{cases} \tag{14}$$

Numerical routines are required to compute λ and η leading to a slow convergence time.

3.3 [zlog(z)] method

It has been shown that the estimates obtained by this approach are extremely close to the MLE [35]. Here, we need to determine first the expectations $E\{\log(y^2)\}$ and $E\{y^2\log(y^2)\}$ [37]. The [zlog(z)] estimator is given in a closed form provided by

$$\begin{cases} \hat{\lambda} = \left(\frac{E\{y^2\log(y^2)\}}{y} - E\{y^2\log(y^2)\} - 1 \right)^{-1} + 1 \\ \hat{\eta} = \hat{\lambda} \left(\frac{\mu_{1/2}\Gamma(\hat{\lambda})}{\Gamma(5/4)\Gamma(\hat{\lambda}-1/4)} \right)^{-4}, \lambda > 1/4 \end{cases} \tag{15}$$

This approach relatively reduces the estimation time with a slight degradation in estimation performance compared to the MLE method.

3.4 MLE method

The vector of parameters $\hat{\theta} = [\hat{\lambda}, \hat{\eta}]$ which maximizes the log-likelihood function $L(\theta)$, i.e. $\hat{\theta}$ is the solution of the following equation

$$\frac{\delta}{\delta\theta} \ln f_{Y|\theta}(y|\theta) = 0 \tag{16}$$

From the definition of the MLE method, the likelihood function is computed first on the basis of (9) and partial derivatives with respect to λ and η are required to find the maximum of the log-likelihood function. From [36], it is shown that the ML estimate of $\hat{b} = \frac{\hat{\lambda}}{\hat{\eta}}$ is a solution of the following function.

$$\frac{N \sum_{i=1}^N \frac{z_i^2}{\hat{b}z_i^2+1}}{N - \hat{b} \sum_{i=1}^N \frac{z_i^2}{\hat{b}z_i^2+1}} - \sum_{i=1}^N \log(\hat{b}z_i^2 + 1) = 0 \tag{17}$$

From (17), the ML estimate of $\hat{\lambda}$ is given in a closed form as a function of $\hat{\eta}$

$$\hat{\lambda} = \frac{N}{\hat{b} \sum_{i=1}^N \frac{z_i^2}{\hat{b}z_i^2+1}} - 1 \tag{18}$$

where the scale parameter is simply determined by $\hat{\eta} = \hat{b}\hat{\lambda}$.

This is a time-consuming approach because of the optimization of nonlinear equation for the scale parameter.

4 Multi-headed deep learning estimator

In recent years, different machine learning paradigms have received extensive attention as a multidisciplinary subject. Deep learning [38], a subfield of machine learning, belongs to the class of representation learning aiming to construct models and representations from large unprocessed data. The particularity of deep learning resides in its ability to learn hierarchical feature representations directly from the raw data and thus reducing the hand-crafted features extraction considerably. Deep learning has presented super-human skills in many applications, including, and not limited to: object detection/recognition, anomaly detection, time series forecasting, and natural language processing. Since we are tackling the issue of estimating clutter parameters and the clutter returns are 1-dimensional signals, one can shrink the estimation process to a regression problem as a many-to-many sequence prediction. We will expose the most popular deep learning models dedicated to sequential data problems in the following subsection.

4.1 Holistic view of appropriate DL models for sequential data

Given the variety of sequential or time series problems across diverse domains, several neural network approaches have emerged [39, 40]. Among these models, recurrent neural networks or RNNs for short have gained great interest owing to the short-term memory obtained from the recurrent feedback connections and provide a powerful tool for dealing with correlated sequential data. However, traditional RNNs suffer from two main limitations. First, the temporal order involved in the input sequence affects the model's outputs which is primarily based on the previous context. For this, an appropriate solution proposed in [41] consists of presenting each training sequence forwards and backwards to two recurrent networks connected to the same output layer. The second drawback is the difficulty of learning long-term dependencies that are encoded in

the data because of the vanishing gradient [42]. To deal with this, an alternative recursive architecture with a specialized cell structure was introduced, leading to long short-term memory network (LSTM) [43]. As the length of the input variable changes in the sequence-to-sequence prediction applications, an RNN Encoder-Decoder architecture was developed in [44] and based on one RNN network acting as an encoder which maps the variable-length source sequence to a fixed-length vector and a second RNN network is used to map the vector representation back to a variable-length target sequence. Besides, incorporating LSTM cells instead of RNNs in the Encoder-Decoder design leads to Encoder-Decoder LSTM architecture [45]. Moreover, Autoencoder (AE), an unsupervised learning model trained using supervised learning methods, also called a self-supervised learning method, is a special type of neural network that seeks to learn a compact representation of a set of data intended as a feature vector to another supervised learning model. A variety of AE models are proposed in the literature and are principally used for anomaly detection, feature extraction, and dimensionality reduction [46]. The implementation of the AE using the Encoder-Decoder LSTM yield the LSTM-Autoencoder (LSTM-AE). The LSTM-AE may learn a compressed version of sequential data [47].

Furthermore, a tremendous interest in another deep learning architecture alternating convolutional and subsampling layer has been addressed in recent years. The convolutional neural networks (CNNs), which are feed-forward artificial neural networks with many hidden layers trained with a massive size of labeled datasets, have demonstrated to be a powerful tool for numerous engineering applications, especially for 2D signals. Recently, 1D CNNs have been proposed to deal with the time series forecasting and classification concerns and have immediately achieved extraordinary accuracy performance, making them a viable option for 1D signal processing applications [39].

A more in-depth explanation of the different architectures is addressed in the following subsections.

4.2 Long short-term memory

Recurrent neural networks take their appellation from the involved recurrence connections, which provide them with a memory of past activations allowing the temporal dynamics learning and making this kind of recursive networks appropriate for modeling sequential data. RNNs are defined as a topology stimulating a discrete-time dynamical system with an input x^t , a hidden state h^t and an output y^t . The drawback of the conventional RNNs is the exploding or vanishing gradient, which occurs when the gradient tends to be extremely small, leading to the short-term memory problem in long sequences. The long short-term memory architecture (LSTM) is proposed as an efficient attempt to solve this problem. The LSTM is constituted of three states to control the interactions between the different memory units. These states are referred to as the input, forget and output gates. Specifically, the forget gate removes redundant or irrelevant data and chooses to retain or ignore its previous status. Whereas the input state analyzes the new data and checks whether the input signal may alter the memory cell state, the output gate, conversely, handles the input data with the cell state [43]. In other words, the idea is to have an input vector added with the

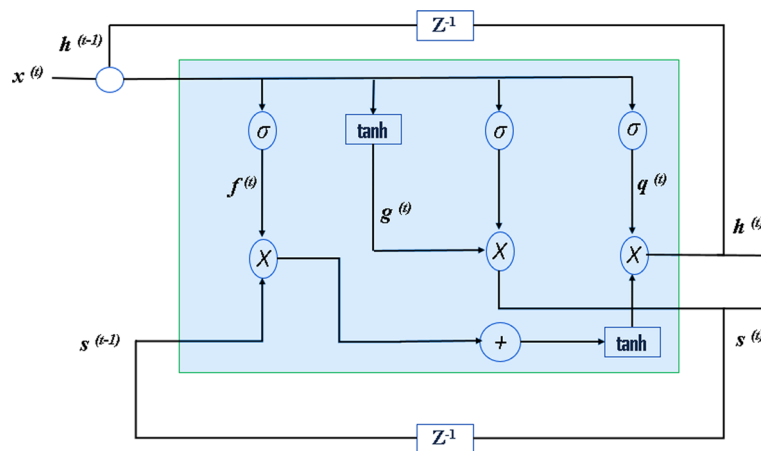


Fig. 2 Block diagram of the LSTM cell architecture

previous output vector passed through a neural network which outputs the values to keep with 1 and the values to forget with 0.

The block diagram of Fig. 2 illustrates the different components of a cell memory of the LSTM network.

Unlike the unidimensional, bidirectional LSTMs process the data in both forward (positive time direction) and backward (negative time direction) senses. The idea is to duplicate the first recurrent layer in the network into two side-by-side layers, then providing the input sequence as-is to the first layer, and the second layer processes a reversed copy of the input [48].

It is worth noting that timesteps in the input sequence are treated one at a time the same way as the unidirectional architecture. The difference is that the network steps through the input sequence are processed in both directions at the same time.

4.2.1 LSTM-autoencoders

Autoencoders (AEs) are an elegant variation of neural networks trained in an unsupervised learning paradigm to reconstruct an informative representation of its input and often involved in different implications such as clustering or used as generative models. AE consists of three layers arrangement, based on feed-forward neural networks, for encoding and decoding purposes. The encoding phase comprises the mapping of the dataset to a hidden layer and learns a compressed representation of the data sequence. In contrast, the decoding task covers the reconstruction of the input sequence from the latent variables of the encoding phase. The encoding and decoding processes can be formulated as follows:

$$h(x) = f(W_1x + b_1) \tag{19}$$

$$\hat{x} = g(W_2h(x) + b_2) \tag{20}$$

where x is the input sequence, $h(\cdot)$ and $f(\cdot)$ are the encoding and decoding functions, respectively, $h(x)$ is the hidden encoded vector and \hat{x} is the reconstructed vector of the output layer. Additionally, W_1 , W_2 are the encoder and decoder weight matrix while b_1 , b_2 represents the bias vectors of each phase [49].

During the training, the AE model tries to minimize an objective function which is nothing but the reconstruction error expressed in terms of the difference between the input and the reconstructed output (i.e., minimize $\|x - \hat{x}\|^2$). One way to get lower reconstruction error is to stack different AE layers, and the obtained architecture is referred to as stacked autoencoder (SAE). SAE allows the extraction of useful high-level features and other advantages of abstraction and invariance, and thereby better generalization skills of the model are expected [49].

Since RNNs are well suited for modeling sequential data compared to the traditional feed-forward multiple layer perceptron (MLP) involved in the AE architecture, numerous LSTM-based autoencoder (LSTM-AE) networks were proposed where the classical MLP in AE are replaced by LSTM cell organized in encoder-decoder architecture as shown in Fig. 3.

LSTM-AE is the combination of two LSTM layers acting as encoder and decoder. The LSTM encoder layer is trained to produce the representation vector, and the latter is fed to the second LSTM decoder layer that should reconstruct the input data. The LSTM-AE cost function is the mean square error between the original input sequence and the reconstructed vector. The LSTM-AE can be applied either for input sequence reconstruction and/or prediction purposes [47, 49]. Regardless the desired task, once the AE has been trained, the decoder can be removed, and the encoder can be kept for other supervised learning processes as a standalone model. In [49], Stacked LSTM-AE (LSTM-SAE) was proposed for modeling multistep time series forecasting problems as a robust forecasting module that converts time series observation into representative features which can be used for other tasks such as prediction or analysis. The implemented LSTM-SAE adopts a shallow architecture with one LSTM for the encoder and decoder layers. The pre-training phase consists of building three LSTM-SAE blocks where each block is constituted of the past trained and saved LSTM encoder as a hidden layer. A fine-tuning phase is then carried out by adding an output layer for a supervised regression problem where the label is the corresponding variable to the input sequence. The evaluation is then completed using out-of-sample

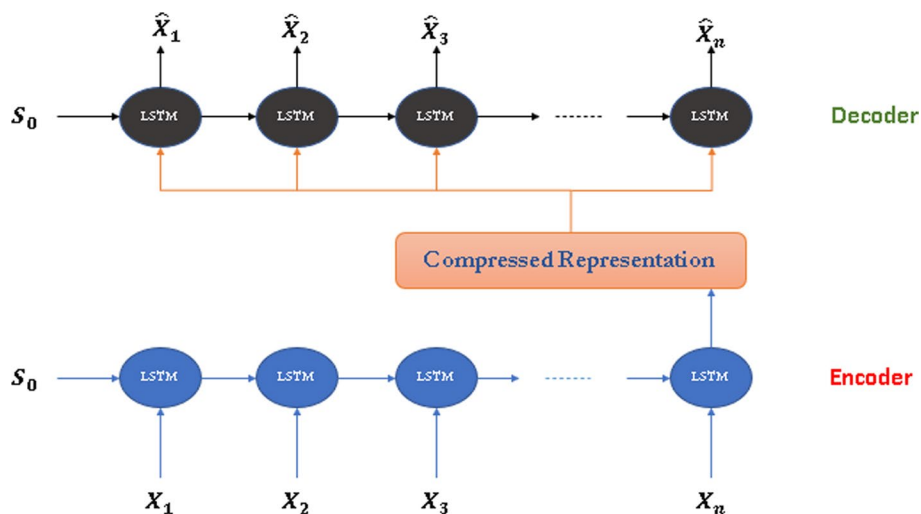


Fig. 3 LSTM-AE architecture

testing data. This is the same approach we used in this paper, with a minor modification in the number of blocks.

4.3 Convolutional neural networks (CNNs)

Convolutional neural networks are a particular type of feed-forward neural networks that map features from input data in a hierarchical way and used for data processing. They consist of a chain of layers alternating convolutional and subsampling operations to the input data, followed by a classifier. Besides the immunity to small data transformations and the ability to process large inputs, CNNs combine the feature extraction and the classification tasks in the same process breaking all records in computer vision and many other domains. CNNs were first fashioned specifically to process 2D signals, and their applications for 1D scarce data require a suitable 2D to 1D conversion and various feature transformations were introduced [39]. However, due to the high computational complexity along with a large amount of data necessary for the training, 2D CNNs may not be an adequate solution for many 1D data applications. A compact 1D CNN architecture was recently proposed in [39] to deal with the drawbacks of 2D CNNs in 1D applications and achieve high-performance levels in diverse signal processing applications.

CNNs work the same way whether inputs have 1, 2, or 3 dimensions. The difference is the structure of the input data and how the convolution kernel moves across the data. The input layer of the 1D CNN receives a 1D time series sequence instead of 2 or 3D data, and 1D filters are replaced instead of 2D filters. The convolutional layer performs a sequence of convolutions, which are simply linear weighted sums of 1D sequences passed through the activation function followed by the max-pooling or other pooling operation. Consequently, the network processes 1D signals instead of 2D matrices for both kernels and feature maps, resulting in a low computational complexity. Related explanations of the Forward and Backpropagation learning algorithms in CNN layers are provided in [39].

Considering the various deep architectures seen in this section, it should be possible to conceive an intelligent estimator for Pareto modeled clutter using a mixture of CNN, LSTM, BLSTM and Autoencoder designs.

The following subsection of this paper deals with the design and the validation of a multi-headed deep learning-based estimator.

4.4 The multi-headed deep learning-based estimator

4.4.1 Data generation/pre-processing

The performance accuracy of deep learning methods relies highly on the quality and the amount of the available labelled dataset. DL algorithm inputs could be of various types, and the most commonly used data inputs are in the form of tensors. For the problem of estimating the Pareto II clutter parameters, a combination of random values of the correlation coefficient, scale, and shape parameters together with data of normalized clutter power were considered for generalization purposes.

The data generation of correlated samples is performed using SIRV model as follows:

- **Setting the following parameters:**

the number of pulses N , window size m , the number of Monte Carlo trials n , and the number of vectors nb ;

- Generation of random coefficient of correlation ρ between 0 and 1;
- Generation of random values of the shape parameter λ in the range of [1.1, 8] and the scale parameter η between [1, 12];

- For $i = 1: nb$

- Repeat for each value of ρ

- Repeat for each value of the couple $\{\lambda, \eta\}$;
- Compute the covariance matrix of the speckle: $M_x = \text{toepliz}(\rho^m)$ and the Cholesky factor of M_x : $L = \text{chol}(M_x)$;
- Generate the SIRV Pareto data as follows:

- Compute the complex speckle x

$$x = L * (\text{randn}(m, N) + j * \text{randn}(m, N)) / \sqrt{2}$$

- Compute the Inverse Gaussian texture τ as $\tau = 1/\text{gamrnd}(\lambda, \eta, 1, N)$

- The SIRV c is obtained by Eq. (2) (i.e., $c = \sqrt{\tau}x$) which is a matrix (N^*m).
- Calculate the power of c so that the input matrix contains real values.
- Flatten the matrix c and append λ, η (i.e., $z_i = \{c_{i,1}, c_{i,2}, \dots, c_{i,(N*m)}, \lambda_{i,(N*m)+1}, \eta_{i,(N*m)+2}\}$).
- Concatenate the current vector with the previously generated vectors in the same matrix.
- Save the matrix

$$\begin{bmatrix} c_{1,1}c_{1,2} & \dots & c_{1,(m*N)}\lambda_{1,(N*m)+1}\eta_{1,(N*m)+2} \\ c_{2,1}c_{2,2} & \dots & c_{2,(N*m)}\lambda_{2,(N*m)+1}\eta_{2,(N*m)+2} \\ \vdots & \ddots & \vdots \\ c_{nb,1}c_{nb,2} & \dots & c_{nb,N*m}\lambda_{nb,(N*m)+1}\eta_{nb,(N*m)+2} \end{bmatrix}$$

4.4.2 Estimation architecture design and validation

The basic idea for constructing a multi-headed deep learning-based estimator is to present a combination of different features to the deep neural network with varying designs of architecture, and the role of each block is either to model/extract valuable features and/or to predict values from the original dataset. The proposed approach consists of building four deep architectures, namely an LSTM-SAE, 1D CNN, BLSTM, and 1D CNN-LSTM models. Then, the model's outputs are concatenated to form the input for the final prediction block, which consists of a single LSTM layer followed by an MLP to produce final estimates of the shape and scale Pareto type II clutter, as illustrated in Fig. 4.

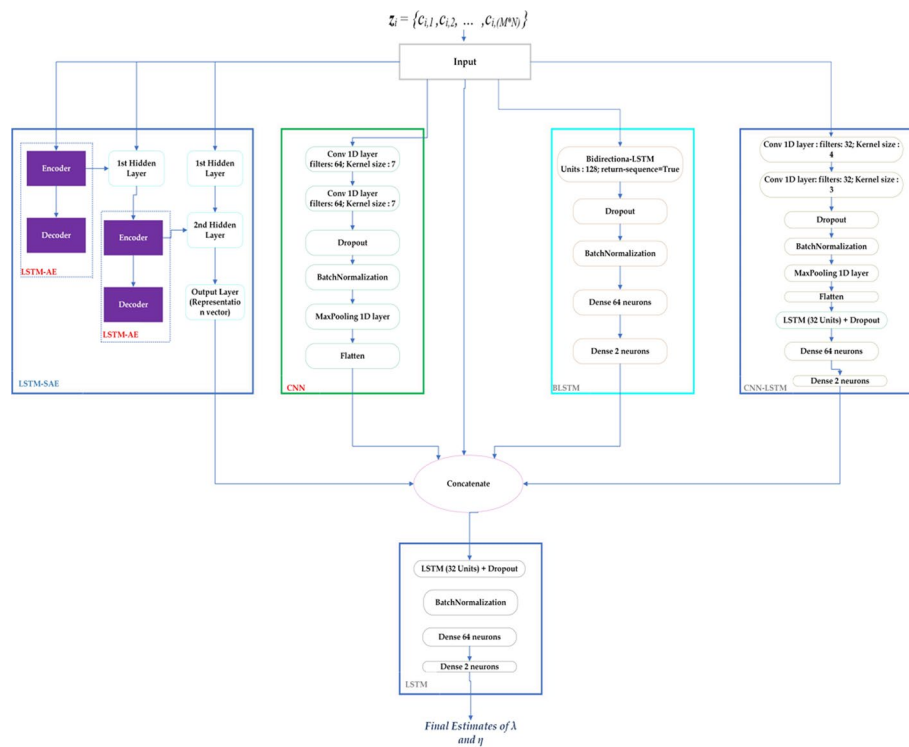


Fig. 4 The proposed multi-headed deep learning estimator

The two first heads of the estimator are mainly developed to produce a new representation and extract new features from the input sequences. The LSTM-SAE is composed of two hidden layers of encoded data following a similar approach of [49] and trained in a self-supervised manner where the desired output is the same as the expected input sequence. The first LSTM-AE block in the stack is trained, and its LSTM encoder layer is saved to be used as input for the second LSTM-AE block in the stack, which is also saved and load as a second hidden layer for the final LSTM-SAE model. The LSTM encoder-decoder architecture is designed using 64 LSTM units for both encoding and decoding, and the RepeatVector layer is used as an adapter to fit the encoder and decoder of the network. The output of that architecture is a reconstructed version of the generated Pareto type II samples.

The second head is the convolutional architecture constituted of two consecutive 1D-convolutional layers where each Conv-Layer consists of 64 filters, followed by a MaxPooling operation. The principal task achieved by this head is to extract useful features. The output is then flattened to be concatenated with the representation vector from the LSTM-SAE.

The two last heads are designed for prediction purposes. To this end, hybrid 1D CNN-LSTM architecture and a BLSTM network are considered. The bidirectional LSTM model is formed using 128 LSTM cells trained forward and backward to output intermediate estimates of the shape and scale parameters of the Pareto model. The last head is a combined architecture of 1D CNN and LSTM networks. It consists of two consecutive convolutional layers of 32 filters in each Conv-Layer, followed by a MaxPooling.

The flattened output of the last convolutional layer is fed to the input nodes of a hidden LSTM layer composed of 32 cell units. The output layer is MLP which is incorporated to provide another intermediate predicted value of the unknown Pareto distribution parameters.

Finally, the outputs of the above-mentioned models are concatenated with the original set of data and fed to an LSTM Layer composed of 64 units followed by two dense layers of 128 and 2 neurons, respectively, to provide the final estimates of the shape and scale parameters. It is worth noting that the ReLU function is the activation function for all the models mentioned above.

The dataset is generated following the process previously described and consists of 250,000 vectors; each vector contains $N \times m$ simulated samples associated to the values of λ and η as targets. The dataset is randomly organized into a training set (80%) and a validation set (20%).

The final multi-headed deep learning-based estimator is obtained after different trials and errors process. Different architectures were considered with diverse models, layers, activation functions, and optimizers. The elaboration of the proposed intelligent estimator was validated after the comparison with several deep learning models. Table 1 summarizes the overall mean square error (MSE) in the training and validation phases along with the configuration of the different architecture designs. Dropout and batch normalization layers were incorporated for the above models to increase the generalization and improve prediction performances.

As depicted in Table 1, it is seen that the proposed DL-based estimator exhibits better performance than the other topologies for both the training and the validation datasets. The Adam optimizers exhibit the best performance in reason of its convergence rapidity along with the RMSprop with near-optimal performance.

Table 1 Comparison of different architectures configuration and MSE accuracy

Model	Configuration					
	Layers/blocks	Number of neurons/filters/units	Activation function	Optimizers	Training accuracy (MSE)	Validation accuracy (MSE)
Stacked LSTM	4 LSTMs Layers + 2 Dense Layers	LSTM: 128, 64, 64, 32 Dense: 64, 2	Sigmoid ReLU	SGD	1.11	1.24
				RMSprop	1.08	1.19
				Adam	1.05	1.14
BLSTM	BLSTM + 2 Dense layers	BLSTM: 128 units Dense: 100, 2	Sigmoid Tanh ReLU	SGD	1.33	1.87
				RMSprop	1.28	1.85
				Adam	1.26	1.85
CNN-LSTM	2 CNN Layers + 2 LSTM Layers + 2 Dense Layers	Conv1D:64, 64 LSTM: 32, 32 Dense: 64, 2	Sigmoid Tanh ReLU	SGD	0.98	1.12
				RMSprop	0.86	0.89
				Adam	0.86	0.87
Multi-head	LSTM-SAE + CNN + BLSTM + CNN-LSTM + LSTM + Dense	See Fig. 4	ReLU	SGD	0.51	0.71
				RMSprop	0.39	0.59
				Adam	0.31	0.34

5 Performance assessment

With regards to the Deep Learning-based estimator presented above, we proceed in this section to carry out numerical simulations to evaluate the effectiveness of the proposed estimation method and confirm its applicability. For this purpose, we compare the results obtained using the multi-headed deep learning estimator against the existing FOME, $[z\log(z)]$, and the MLE methods by means of the Mean Square Error (MSE) metric test.

Besides the fact that the moment orders vary depending on the clutter spikiness, the applicability hypothesis of the integer-order moment estimator in real-time scenarios is disproved due to the large number of pulses required for accurate estimation of the clutter parameter. That is why we omit the comparison of the proposed Deep learning estimator with the integer-order moment method described in Sect. 3.

The dataset used for the test task is constructed following the same approach used for the training database. The window size is set to $m=24$ with $N=4, 8$ and 16 integrated pulses, and we considered different values of the correlation coefficient, from data almost uncorrelated to strongly correlated ($\rho=0.01, 0.1, 0.3, 0.6, 0.9, 0.99$) to see the effect of the correlation on the estimation task. Pareto samples are generated in the range of values for λ between $[1.1-6]$ so that different states of the clutter spikiness are included. In the overall test phase, the estimates of λ are averaged over $n=1000$ Monte Carlo trails, and the computations were executed on an Intel® Core™ i5-3230 M CPU @ 2.60 GHz RAM 8.0 GB.

As previously stated, for each value of λ between $[1.1, 6]$, we generate an m by N matrix with a fixed correlation coefficient. After that, the matrix is flattened to produce a single sample vector. This process is repeated 1000 times for each value of the couple (λ, η) , resulting in a $50,000 \times 384$ matrix in the test dataset.

For low correlation coefficient ($\rho=0.1$), Fig. 5 illustrates the MSE curves of the FOME, $[z\log(z)]$, MLE, and the DL-based estimator for a window size of 24 and 16 integrated pulses. For very spiky clutter, comparable results are obtained using the four estimators with slightly enhanced performance of the DL-based estimator for $\lambda \leq 1.3$. As the spikiness of the clutter decreases, the proposed method exhibits improved performance compared to the other estimators with instabilities of the MLE and its closed-form counterpart $[z\log(z)]$ estimators. For the parameter η , it is clearly seen that the multi-headed deep learning attains the lowest error for $\eta < 8$ with overlapped curves of the other classical estimators. For large values of η , the MLE gives equivalent results as the DL and $[z\log(z)]$ estimators.

As the correlation coefficient increase ($\rho=0.3$), the estimation error of the $[z\log(z)]$, MLE, and the FOME increases while the DL estimator maintains its superiority for high values of the shape parameter as depicted from Fig. 6 with a slight superiority of the MLE for $\lambda=1.4$. Note that for $\rho=0.3$ and large values of η , similar results to those obtained for $\rho=0.1$ are observed, and globally, precise estimation accuracy is achieved using the proposed estimation approach. For $\rho=0.6$, as shown in Fig. 7 that the proposed estimator achieves improved performance for the overall range of the shape and scale parameters. For highly correlated data ($\rho=0.9$), the estimation accuracy of the multi-headed deep learning-based approach exhibits better performance than the classical approaches as illustrated in Fig. 8. For a more informative

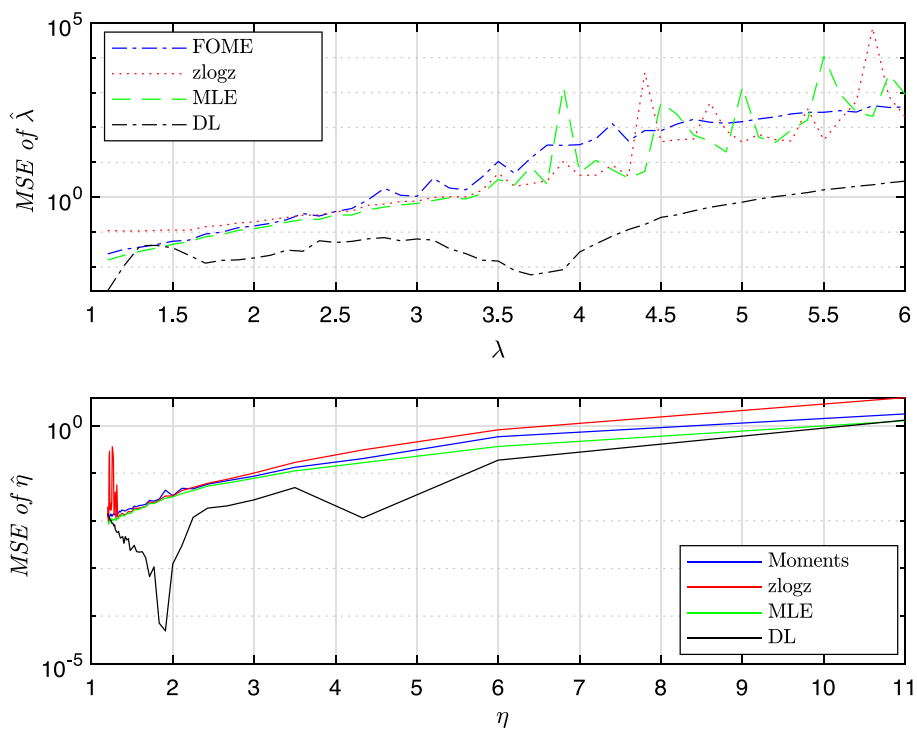


Fig. 5 MSE of the estimates of λ and η for $\rho=0.1, m=24, N=16$

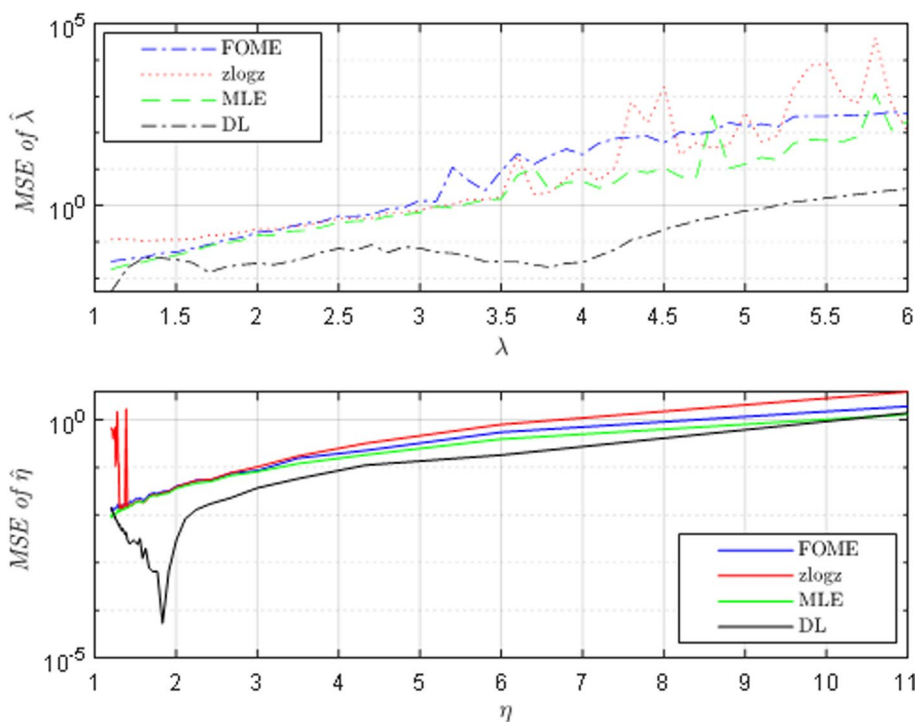


Fig. 6 MSE of the estimates of λ and η for $\rho=0.3, m=24, N=16$

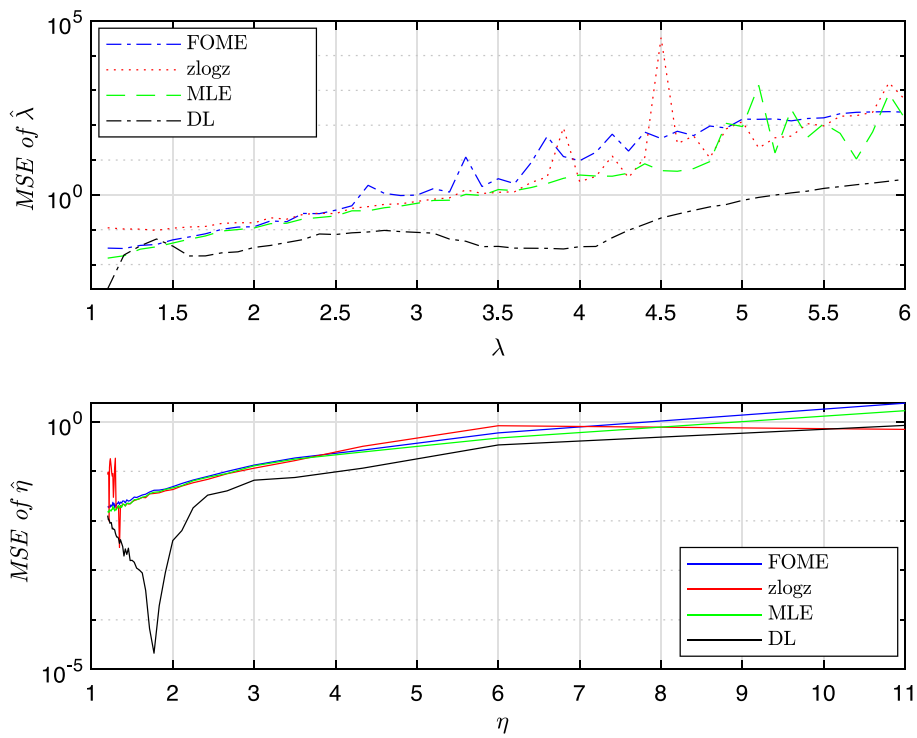


Fig. 7 MSE of the estimates of λ and η for $\rho=0.6, m=24, N=16$

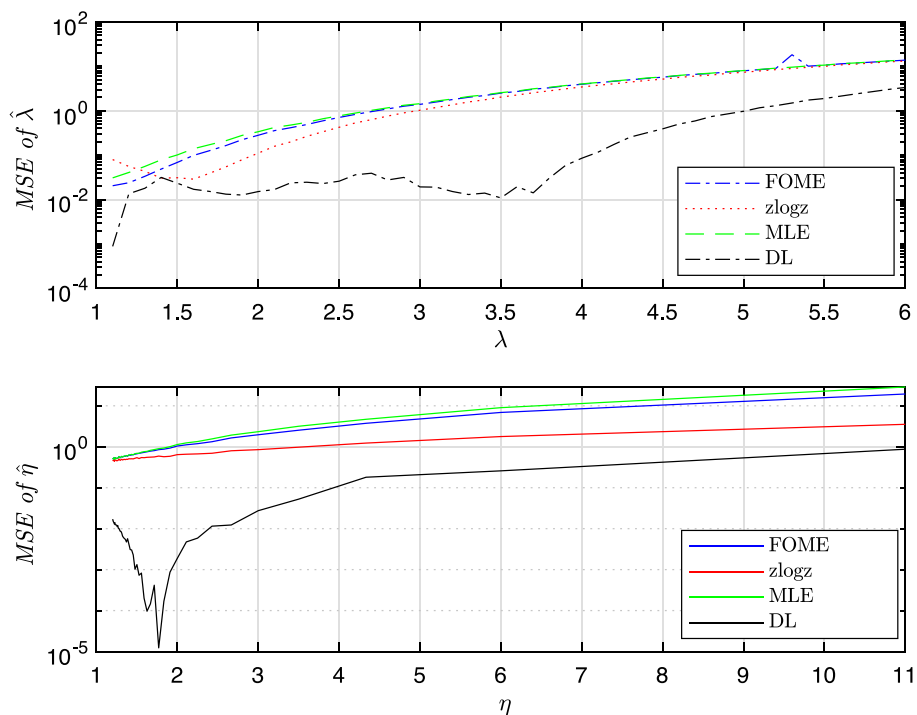


Fig. 8 MSE of the estimates of λ and η for $\rho=0.9, m=24, N=16$

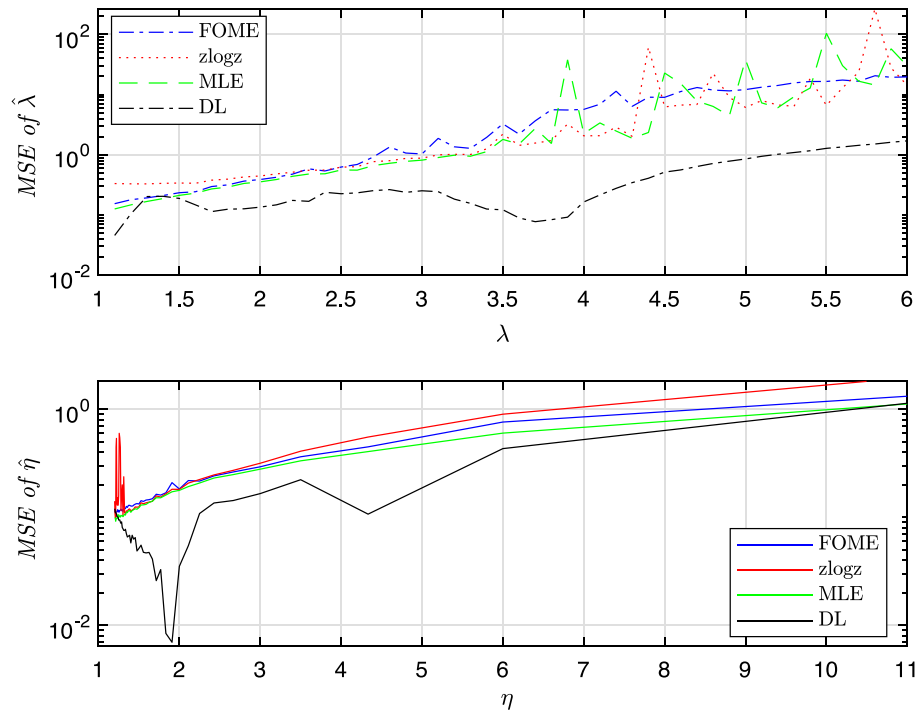


Fig. 9 Bias of estimated λ and η for $\rho=0.1, m=24, N=16$

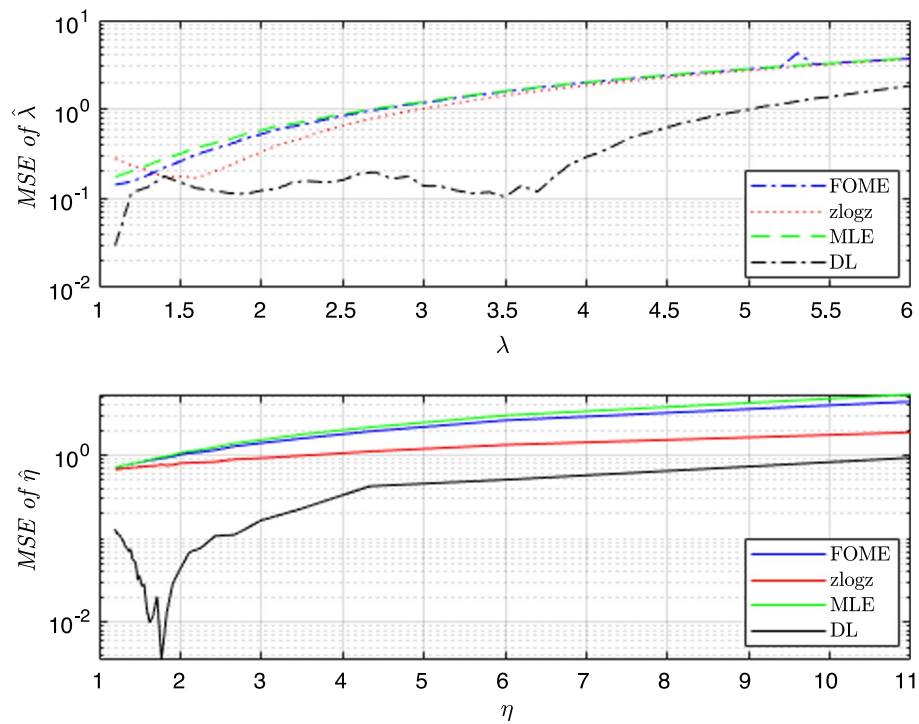


Fig. 10 Bias of estimated λ and η for $\rho=0.9, m=24, N=16$

illustration, Figs. 9 and 10 show the bias of the shape and scale parameters for $\rho = 0.1$ and $\rho = 0.9$, respectively. As expected, the results obtained from the bias criterion appears to be identical to those obtained by the MSE. Regardless of the correlation coefficient, for large shape parameter values, the DL estimator performance exceeds the log and moments-based estimator. On the other hand, the bias confirms the superiority of the DL approach in estimating the scale parameter in the overall studied cases.

It is worth noticing that the Multi-headed deep learning approach estimates both the shape and scale parameters separately but simultaneously, while the estimation of the scale parameter for the $[z\log(z)]$ and FOME depends on the estimate of the shape parameter. Conversely, for the case of the MLE, the shape parameter is estimated from the scale parameter. Thus, it is important to compare the results of the proposed approach against those obtained using the moment of order 1/2 (see Eq. 13). Figure 11 shows improved results of the multi-headed DL estimator for $\eta > 1.8$ for uncorrelated data. On the other hand, for correlated pulses, Fig. 12 illustrates the effectiveness of the proposed approach to estimate the scale parameter for the overall range of η values. Consequently, the estimation of one parameter from another using the existing approaches produces imprecise results, leading to a considerable error impeding the application of such estimators in real radar scenarios.

In order to visualize the impact of the correlation on the estimation, MSE curves of the scale and shape estimators with $\rho = 0.01$ and $\rho = 0.99$ are plotted together in the same figure. For $m = 24$ and $N = 16$, we clearly see in Fig. 13 that, for low values of the shape parameter and uncorrelated data, better estimation performance is achieved for the conventional approaches. In contrast, for the correlated data case with high values of λ , degraded performance with constated instability MSE curves are depicted.

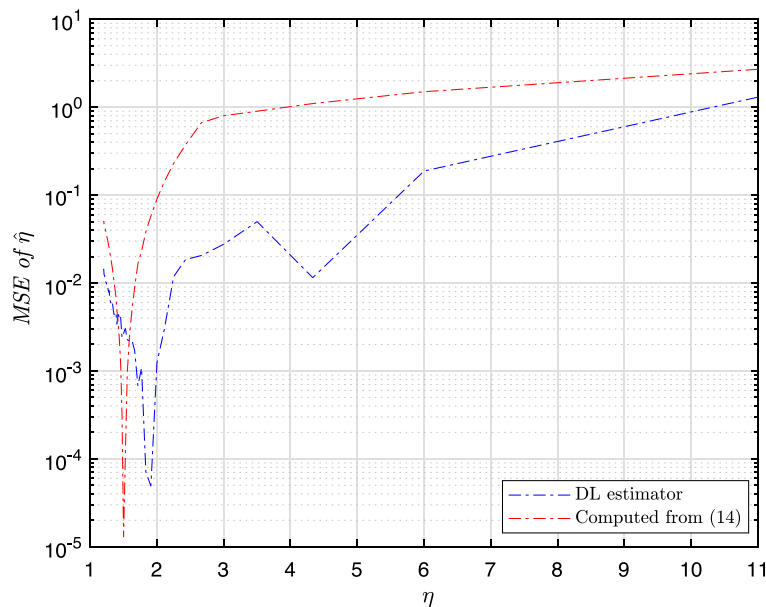


Fig. 11 MSE comparison of estimated η for $\rho = 0.1$, $m = 24$, $N = 16$

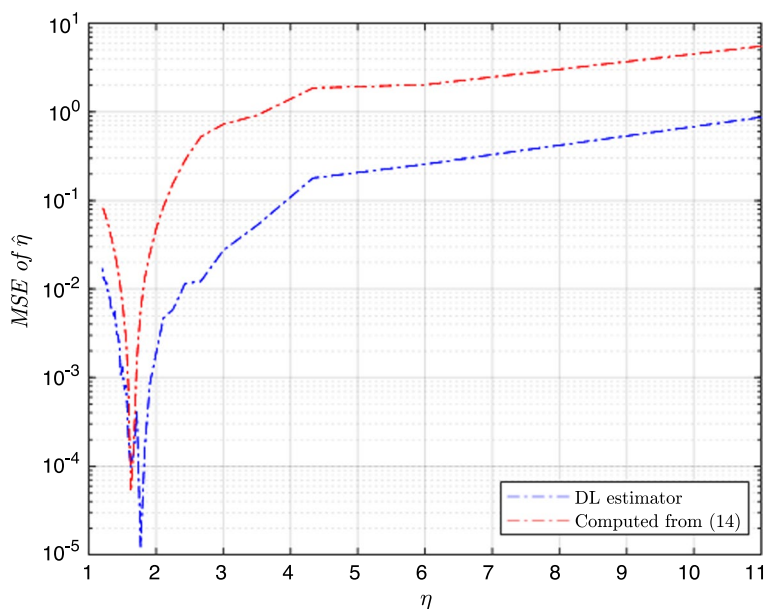


Fig. 12 MSE comparison of estimated η for $\rho=0.9, m=24, N=16$

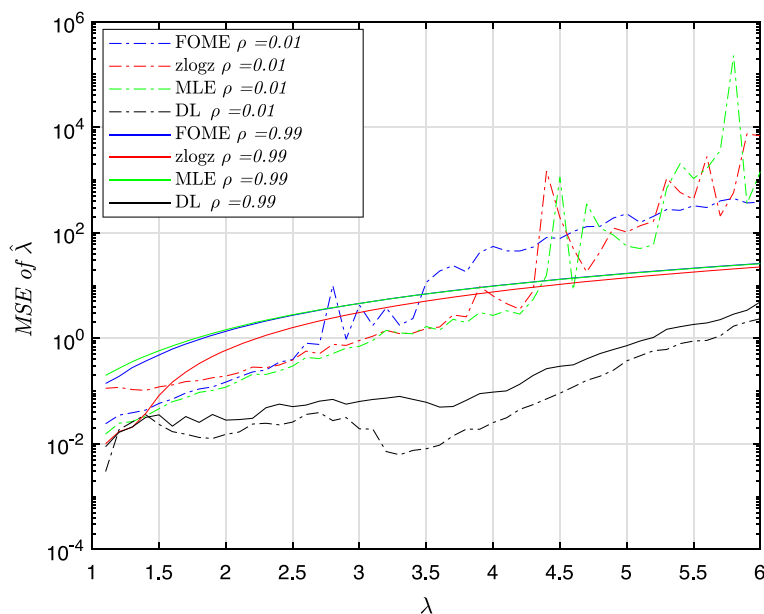


Fig. 13 MSE comparison of estimated λ for $\rho=0.01$ (dashed lines) and $\rho=0.99$ (solid lines) with $m=24, N=16$

Here, the DL-based estimator attains the lowest MSE values and ensures the stability and the accuracy for both cases with enhanced results for the uncorrelated data.

With reduced number of pulses ($M=8, 4$), the estimation performance degrades for the overall estimators compared to the results obtained for large number of pulses as depicted in Fig. 14 and 15. The proposed approach keeps its superiority for large and small number of pulses; then, it can be applied in various situations (Fig. 15).

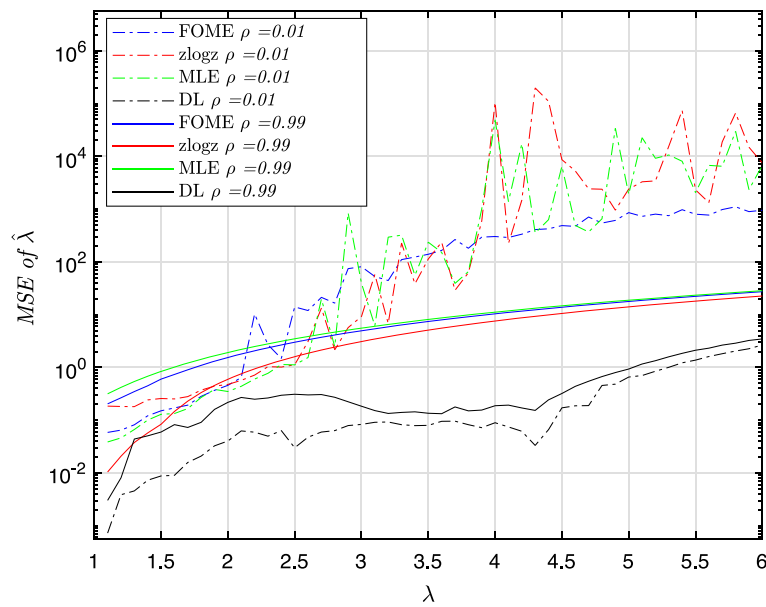


Fig. 14 MSE comparison of estimated λ for $\rho=0.01$ (dashed lines) and $\rho=0.99$ (solid lines) with $m=24, N=8$

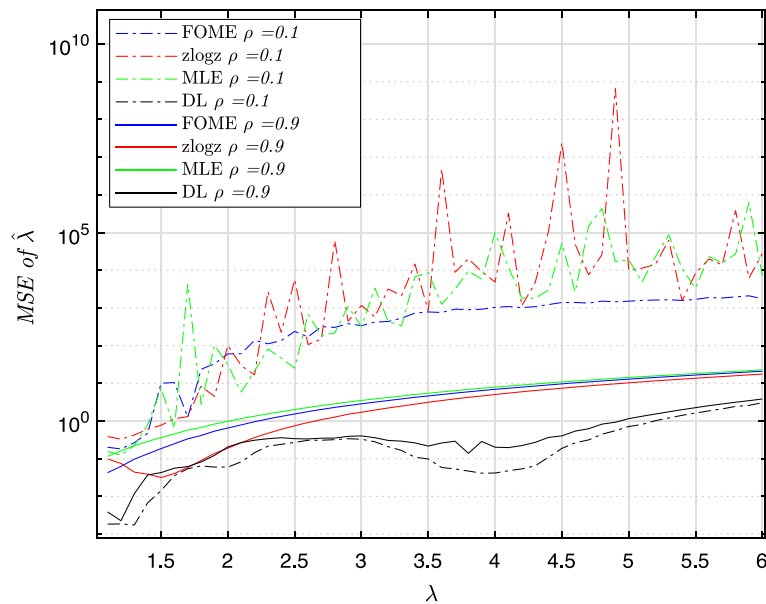


Fig. 15 MSE comparison of estimated λ for $\rho=0.1$ (dashed lines) and $\rho=0.9$ (solid lines) with $m=24, N=4$

Concerning the computation complexity of the algorithms presented in this work, the MLE and FOME approaches require the use of an optimization algorithm to produce estimates of the unknown parameters which introduce a heavy computational burden to the estimation task and limit their real-time application. In contrast, the proposed approach is trained offline leading to an improved execution time for the tested data. The increase in the number of neurons and layers of the deep learning approach results in the increase in the neural network parameters to be processed and somehow, reduces

Table 2 Average elapsed time in (s) for different values of λ , 100 runs and different estimation method

Shape parameter	Estimation approach			
	MLE	FOME	DLE	zlog(z)
$1.1 \leq \lambda < 3$	6.1753	4.334	34.502×10^{-3}	15.375×10^{-3}
$3 \leq \lambda < 4.5$	6.4641	4.869	35.925×10^{-3}	17.781×10^{-3}
$4.5 \leq \lambda < 6$	8.7823	6.211	37.638×10^{-3}	19.224×10^{-3}

the speed of the algorithm execution. The [zlog(z)] closed-form estimator eliminates the optimization process resulting in a faster execution time as depicted in Table 2.

Nonetheless, for correlated echoes, the [zlog(z)] method exhibits degraded performance, which again limit its applicability. The compromise between the time execution and the accuracy performance of the proposed intelligent algorithm confirms its superiority against the conventional FOME, [zlog(z)] and MLE methods.

The estimation accuracy depends highly on the clutter spikiness, the number of pulses, and the correlation between the samples. From the results obtained previously, it is worth stating that the classical FOME, [zlog(z)] and MLE approaches are well suited for spiky uncorrelated echoes and degraded performance is observed for strongly correlated data with high values of the shape parameter. The particularity of the proposed multi-headed deep learning-based estimator resides in its ability to estimate accurately and simultaneously the shape and the scale parameters for both correlated and uncorrelated data with reduced computational complexity.

6 Conclusions and further works

This paper focuses on the estimation of the shape and the scale parameters of Pareto Type II distributed clutter. The proposed approach relies on a mixture of supervised and semi-supervised DL architectures. Simulated data were generated to investigate the performance of the proposed estimator and compare its performance with that of existing estimators. Numerical results show the superiority of the proposed multi-headed DL-based estimator over methods based on the method of moments and the maximum likelihood. Conventional FOME, [zlog(z)] and ML approaches are well suited for spiky uncorrelated clutter, but with increasing correlation coefficient and for high values of the shape parameter, these methods exhibit inaccurate estimates. The proposed multi-headed DL-based estimator has better performance with reduced execution time. The proposed estimator can be implemented in a CFAR detector in the presence of correlated non-Gaussian clutter. The estimate of Pareto clutter parameters in the presence of non-negligible additive white Gaussian noise (AWGN) is a problem that also can be explored by making use of DL approach.

Author contributions

All authors have contributed toward this work as well as in compilation of this manuscript. All authors read and approved the final manuscript.

Availability of data and materials

The datasets used and analyzed during the current study are available from the corresponding author on reasonable request.

Declarations

Competing interests

MSG is the editor-in-chief of the journal at time of publication and was not involved in making a decision on the paper. All other authors declare no competing interests.

Received: 31 May 2022 Accepted: 20 January 2023

Published online: 13 July 2023

References

1. W.L. Melvin, J.A. Scheer, *Principles of Modern Radar: Radar Applications*, Vol. 3 (IET, 2013)
2. G. Lampropoulos, A. Drosopoulos, N. Rey et al., High resolution radar clutter statistics. *IEEE Trans. Aerosp. Electron. Syst.* **35**(1), 43–60 (1999)
3. A. Farina, F. Gini, M. Greco, L. Verrazzani, High resolution sea clutter data: statistical analysis of recorded live data. *IEE Proc. Radar Sonar Navig.* **144**(3), 121–130 (1997)
4. K.J. Sangston, K.R. Gerlach, Coherent detection of radar targets in a non-Gaussian background. *IEEE Trans. Aerosp. Electron. Syst.* **30**(2), 330–340 (1994)
5. K. Ward, C. Baker, S. Watts, Maritime surveillance radar. I. Radar scattering from the ocean surface, in *IEE Proceedings F-Radar and Signal Processing*, vol. 137, pp. 51–62 (IET, 1990)
6. A. Farina, A. Russo, F. Scannapieco, S. Barbarossa, Theory of radar detection in coherent Weibull clutter, in *IEE Proceedings F (Communications, Radar and Signal Processing)*, vol. 134, pp. 174–190 (IET, 1987)
7. B. Cantrell, Radar detection in non-Gaussian, correlated clutter. *Nav. Res. Lab. Rep.* **9015**, 21 (1986)
8. A. Martinez, P. Swaszek, J. Thomas, Locally optimal detection in multivariate non-Gaussian noise. *IEEE Trans. Inf. Theory* **30**(6), 815–822 (1984)
9. R. Tough, K. Ward, The correlation properties of gamma and other non-Gaussian processes generated by memoryless nonlinear transformation. *J. Phys. D Appl. Phys.* **32**(23), 3075 (1999)
10. E. Conte, M. Longo, Characterisation of radar clutter as a spherically invariant random process, in *IEE Proceedings F-Communications, Radar and Signal Processing*, vol. 134, pp. 191–197 (IET, 1987)
11. E. Conte, M. DiBisceglie, M. Longo, M. Lops, Canonical detection in spherically invariant noise. *IEEE Trans. Commun.* **43**(2/3/4), 347–353 (1995)
12. E. Jakeman, P. Pusey, A model for non-Rayleigh sea echo. *IEEE Trans. Antennas Propag.* **24**(6), 806–814 (1976)
13. S. Bocquet, L. Rosenberg, C.H. Gierull, Parameter estimation for a compound radar clutter model with trimodal discrete texture. *IEEE Trans. Geosci. Remote Sens.* **58**(10), 7062–7073 (2020)
14. D.R. Iskander, A.M. Zoubir, Estimation of the parameters of the k-distribution using higher order and fractional moments [radar clutter]. *IEEE Trans. Aerosp. Electron. Syst.* **35**(4), 1453–1457 (1999)
15. I. Chalabi, A. Mezache, Estimators of compound gaussian clutter with log-normal texture. *Remote Sens. Lett.* **10**(7), 709–716 (2019)
16. D. Blacknell, R. Tough, Parameter estimation for the k-distribution based on $\log(z)$. *IEE Proc. Radar Sonar Navig.* **148**(6), 309–312 (2001)
17. A. Gouri, A. Mezache, H. Oudira, Radar cfar detection in Weibull clutter based on $\log(z)$ estimator. *Remote Sens. Lett.* **11**(6), 581–589 (2020)
18. R. Ravid, N. Levanon, Maximum-likelihood cfar for Weibull background, in *IEE Proceedings F-Radar and Signal Processing*, vol. 139, pp. 256–264 (IET, 1992)
19. G.V. Weinberg, L. Bateman, P. Hayden, Constant false alarm rate detection in Pareto type ii clutter. *Digit. Signal Process.* **68**, 192–198 (2017)
20. G.V. Weinberg, S.D. Howard, C. Tran, Bayesian framework for detector development in Pareto distributed clutter. *IEE Proc. Radar Sonar Navig.* **13**(9), 1548–1555 (2019)
21. S. Haykin, W. Stehwien, C. Deng, P. Weber, R. Mann, Classification of radar clutter in an air traffic control environment. *Proc. IEEE* **79**(6), 742–772 (1991)
22. B. Yonel, E. Mason, B. Yazıcı, Deep learning for passive synthetic aperture radar. *IEEE J. Sel. Top. Signal Process.* **12**(1), 90–103 (2017)
23. A. Coluccia, G. Ricci, Radar detection in k-distributed clutter plus thermal noise based on knn methods, in *2019 IEEE Radar Conference (RadarConf)*, pp. 1–5 (IEEE, 2019)
24. D. Roy, S. Srivastava, A. Kusupati, P. Jain, M. Varma, A. Arora, One size does not fit all: multi-scale, cascaded rnns for radar classification. *ACM Trans. Sens. Netw.* **17**(2), 1–27 (2021)
25. J. Zhao, R. Jiang, X. Wang, H. Gao, Robust CFAR detection for multiple targets in k-distributed sea clutter based on machine learning. *Symmetry* **11**(12), 1482 (2019)
26. G. Wang, H. Ding, C. Wang, N. Liu, Estimation of sea clutter distribution parameters using deep neural network, in *Artificial Intelligence in China*, pp. 326–333 (Springer, Singapore 2020)
27. T.H. Kerbaa, A. Mezache, F. Gini, M.S. Greco, CNN-LSTM based approach for parameter estimation of k-clutter plus noise, in *2020 IEEE Radar Conference (RadarConf20)*, pp. 1–6 (IEEE, 2020)
28. E. Conte, M. Lops, G. Ricci, Adaptive detection schemes in compound-Gaussian clutter. *IEEE Trans. Aerosp. Electron. Syst.* **34**(4), 1058–1069 (1998)
29. F. Gini, A cumulant-based adaptive technique for coherent radar detection in a mixture of k-distributed clutter and Gaussian disturbance. *IEEE Trans. Signal Process.* **45**(6), 1507–1519 (1997)
30. F. Gini, M. Greco, Suboptimum approach to adaptive coherent radar detection in compound-Gaussian clutter. *IEEE Trans. Aerosp. Electron. Syst.* **35**(3), 1095–1104 (1999)

31. F. Gini, M. Greco, M. Diani, L. Verrazzani, Performance analysis of two adaptive radar detectors against non-Gaussian real sea clutter data. *IEEE Trans. Aerosp. Electron. Syst.* **36**(4), 1429–1439 (2000)
32. F. Gini, M.V. Greco, A. Farina, Clairvoyant and adaptive signal detection in non-Gaussian clutter: a data-dependent threshold interpretation. *IEEE Trans. Signal Process.* **47**(6), 1522–1531 (1999)
33. F. Gini, M. Greco, Covariance matrix estimation for cfar detection in correlated heavy tailed clutter. *Signal Process.* **82**(12), 1847–1859 (2002)
34. K.J. Sangston, F. Gini, M.S. Greco, Coherent radar target detection in heavy-tailed compound-Gaussian clutter. *IEEE Trans. Aerosp. Electron. Syst.* **48**(1), 64–77 (2012)
35. S. Bocquet, Parameter estimation for Pareto and k distributed clutter with noise. *IET Radar Sonar Navig.* **9**(1), 104–113 (2015)
36. A. Balleri, A. Nehorai, J. Wang, Maximum likelihood estimation for compound-Gaussian clutter with inverse gamma texture. *IEEE Trans. Aerosp. Electron. Syst.* **43**(2), 775–779 (2007)
37. S. Bocquet, Simulation of correlated Pareto distributed sea clutter, in *2013 International Conference on Radar*, pp. 258–261 (IEEE, 2013)
38. Y. LeCun, Y. Bengio, G. Hinton, Deep learning. *Nature* **521**(7553), 436–444 (2015)
39. S. Kiranyaz, O. Avci, O. Abdeljaber, T. Ince, M. Gabbouj, D.J. Inman, 1d convolutional neural networks and applications: a survey. *Mech. Syst. Signal Process.* **151**, 107398 (2021)
40. B. Lim, S. Zohren, Time-series forecasting with deep learning: a survey. *Philos. Trans. R. Soc. A* **379**(2194), 20200209 (2021)
41. P. Baldi, S. Brunak, P. Frasconi, G. Soda, G. Pollastri, Exploiting the past and the future in protein secondary structure prediction. *Bioinformatics* **15**(11), 937–946 (1999)
42. S. Hochreiter, Y. Bengio, P. Frasconi, J. Schmidhuber, et al., *Gradient Flow in Recurrent Nets: The Difficulty of Learning Long-Term Dependencies. A Field Guide to Dynamical Recurrent Neural Networks* (IEEE Press, 2001)
43. S. Hochreiter, J. Schmidhuber, Long short-term memory. *Neural Comput.* **9**(8), 1735–1780 (1997)
44. K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio, *Learning phrase representations using rnn encoder–decoder for statistical machine translation*. arXiv preprint [arXiv:1406.1078](https://arxiv.org/abs/1406.1078) (2014)
45. I. Sutskever, O. Vinyals, Q.V. Le, Sequence to sequence learning with neural networks, in *Advances in Neural Information Processing Systems*, pp. 3104–3112 (2014)
46. P. Baldi, Autoencoders, unsupervised learning, and deep architectures, in *Proceedings of ICML Workshop on Unsupervised and Transfer Learning*, pp. 37–49 (JMLR Workshop and Conference Proceedings, 2012)
47. N. Srivastava, E. Mansimov, R. Salakhudinov, Unsupervised learning of video representations using lstms, in *International Conference on Machine Learning*, pp. 843–852 (PMLR, 2015)
48. A. Graves, J. Schmidhuber, Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural Netw.* **18**(5–6), 602–610 (2005)
49. A. Sagheer, M. Kotb, Unsupervised pre-training of a deep lstm-based stacked autoencoder for multivariate time series forecasting problems. *Sci. Rep.* **9**(1), 1–16 (2019)

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Submit your manuscript to a SpringerOpen[®] journal and benefit from:

- Convenient online submission
- Rigorous peer review
- Open access: articles freely available online
- High visibility within the field
- Retaining the copyright to your article

Submit your next manuscript at ► [springeropen.com](https://www.springeropen.com)
