

Article

Not peer-reviewed version

Deep Learning-Based Gain Estimation for Multi-User Software Defined Radios in Aircraft Communications

[Viraj K. Gajjar](#)* and Kurt L. Kosbar

Posted Date: 10 January 2025

doi: 10.20944/preprints202412.0720.v2

Keywords: gain estimation; digital to analog converter; software defined radio; aircraft communication; signal processing; deep learning



Preprints.org is a free multidisciplinary platform providing preprint service that is dedicated to making early versions of research outputs permanently available and citable. Preprints posted at Preprints.org appear in Web of Science, Crossref, Google Scholar, Scilit, Europe PMC.

Copyright: This open access article is published under a Creative Commons CC BY 4.0 license, which permit the free download, distribution, and reuse, provided that the author and preprint are cited in any reuse.

Article

Deep Learning-Based Gain Estimation for Multi-User Software Defined Radios in Aircraft Communications

Viraj Gajjar ^{1,2,*}  and Kurt Kosbar ¹¹ Electrical and Computer Engineering Department, Missouri University of Science and Technology, Rolla, MO 65409, USA² Fortemedia Inc, Alviso, CA 95002, USA

* Correspondence: virajkgajjar@gmail.com

[†] This article is based on the first author's doctoral dissertation. Portions of the text closely mirror or replicate content from the first author's dissertation and have been included in this manuscript.

Abstract: It may be helpful to integrate multiple aircraft communication and navigation functions into a single software defined radio (SDR) platform. To transmit these multiple signals, the SDR would first sum the baseband version of the signals. This outgoing composite signal would be passed through a digital-to-analog converter (DAC), before being up-converted and passed through a radio frequency (RF) amplifier. To prevent non-linear distortion in the RF amplifier, it is important to know the peak voltage of the composite. While this is reasonably straightforward when a single modulation is used, it is more challenging when working with composite signals. This paper describes a machine learning solution to this problem. We demonstrate that a generalized gamma distribution (GGD) is a good fit for the distribution of the instantaneous voltage of the composite waveform. A deep neural network was trained to estimate the GGD parameters, based on the parameters of the modulators. This allows the SDR to accurately estimate the peak of the composite voltage, and set the gain of the DAC and RF amplifier, "without having to generate or directly observe the composite signal." [1]

Keywords: gain estimation; digital to analog converter; software defined radio; aircraft communication; signal processing; deep learning

1. Introduction

Aerospace vehicles often require multiple connectivity applications for navigation, telemetry, communications, and a host of other applications [2]. "As shown in Figure 1, these links may be through air-to-ground, satellite, and with other high altitude platforms and networks [2]." [1] Legacy systems that use dedicated hardware devices for each link can be difficult to integrate and maintain. Interconnecting these systems can require wiring harnesses that substantially increase the weight and cost of the vehicle [3]. These costs, coupled with the complexities of modifying, integrating, troubleshooting, and maintaining new applications within hardware-dependent systems—especially those with extensive cabling—pose significant operational challenges [1,4].

"Amrhar *et al.* [5] demonstrated that an SDR-based integrated modular avionics architecture could significantly reduce the size, weight, power usage, and cost in an aircraft." [1] Software-defined radios (SDR) allow designers to support multiple communication protocols on a single hardware platform. An SDR can generate multiple baseband waveforms for the various links they support, and then in a single operation up-convert and transmit them using a single hardware transmitter [6]. By summing signals in the digital domain, no additive noise is introduced prior to the digital-to-analog converter, ensuring a clean composite waveform. The flexibility of reprogramming an SDR without hardware changes can give designers greater flexibility and assist in the troubleshooting and maintenance processes [1].

"Multiple radio waveforms can be generated in a single SDR as illustrated in Figure 2. In this system, the SDR generates waveforms for each application then sums them into a single digital waveform." [1] A digital-to-analog converter (DAC) produces an outgoing composite analog signal, which is up-converted and transmitted. To minimize quantization error, while still avoiding the

nonlinear clipping of large signals, it is important to carefully select the input gain of the DAC. The composite signal's peak amplitude is a key determinant of the optimal gain setting ($G = \frac{\alpha}{u}$), where αV represents the DAC's range and uV denotes the peak voltage of the composite signal, as further illustrated in Figure 3.

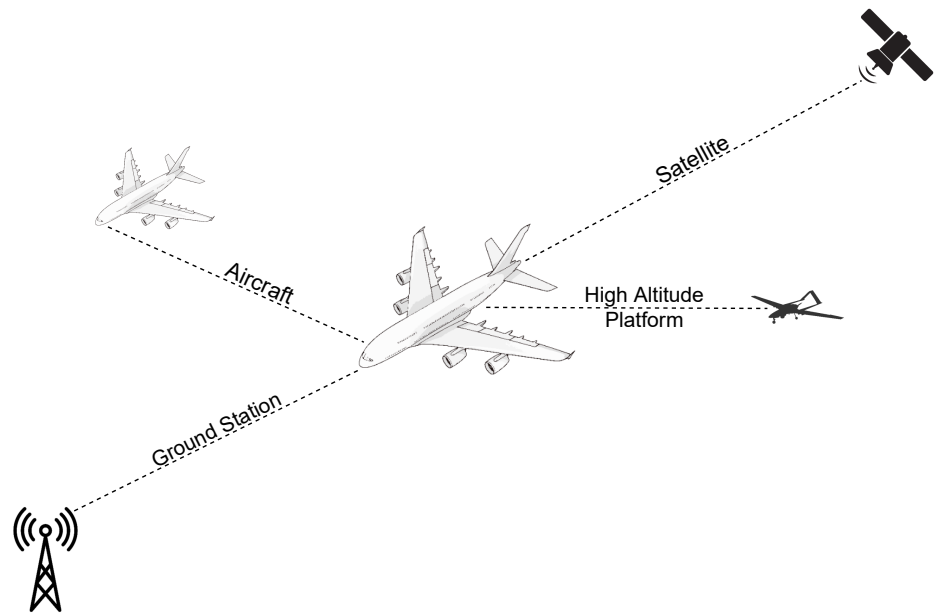


Figure 1. "Typical communication links of an aircraft."[1]

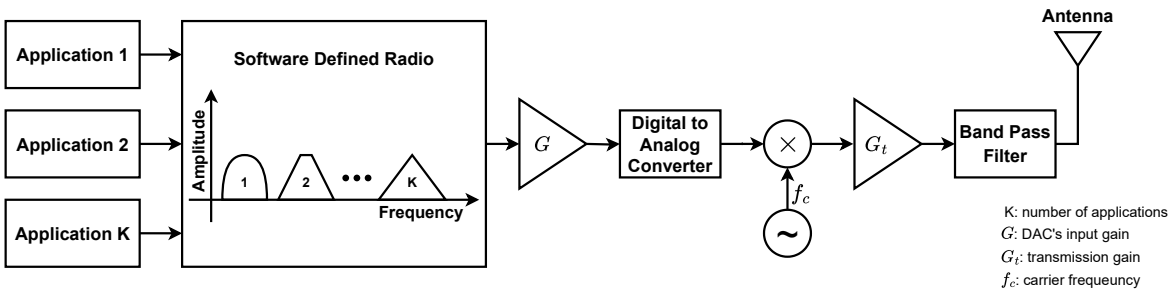


Figure 2. "SDR-Based implementation of an aircraft's communication links."[1]

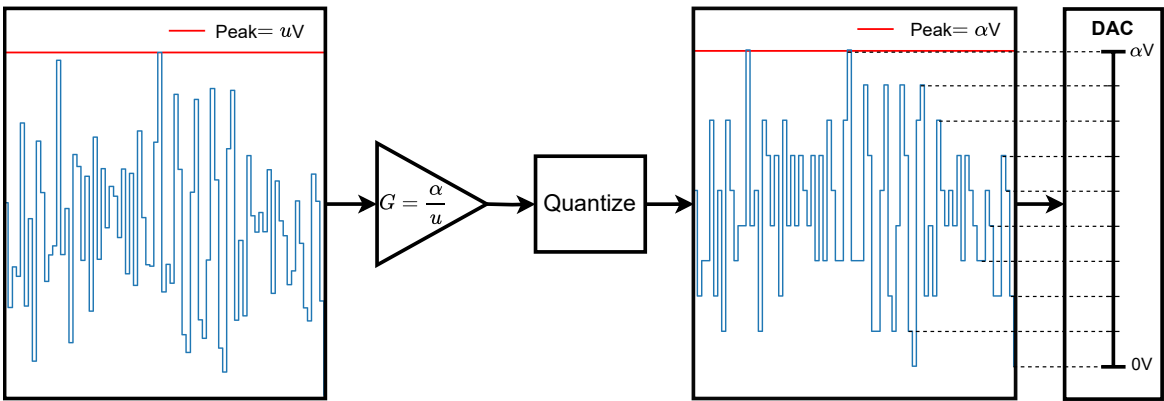


Figure 3. Time domain representation of a composite signal.

"Estimating the DAC's gain for an individual signal is relatively straightforward, as the peak amplitude is generally well-understood, even considering distortion from a pulse-shaping filter."[1]

The estimate can become challenging for composite signals due to the unpredictable superposition that occurs when an SDR must sum signals from multiple applications to form a composite signal. One method for gain estimation involves a closed-loop system, adjusting the gain after the signal has been observed for some time. The signal may suffer from high quantization noise, or clipping, during the time required for the loop to reach the optimal level. The SDR could perform this operation in advance of having to transmit data, but that could substantially increase the SDR's computational load.

To overcome this challenge, we propose a deep learning (DL) based technique to estimate statistical distribution of composite signal's amplitudes. Our method relies exclusively on the parameters used to generate the component signals such as: modulation type, power levels, data rates, normalized frequency, and the number of applications; thereby, eliminating the need to generate or observe the composite signal to estimate DAC's gain [1]. This approach dynamically adapts the gain of the DAC to rapid changes in SDR configuration, ensuring use of DAC's full range across varying scenarios.

"Deep learning (DL) has grown in popularity in recent years due to the dramatic increase in the availability of computing power [7,8]. This success has led to an increased interest in the application of DL in communications and signal processing [9,10]."[1] For instance, convolutional neural networks (CNNs) have been widely used for automatic modulation classification (AMC) by leveraging input features like constellation diagrams [11] and long-symbol rate signals [12]. Nambisan et al. [13] generated hybrid images using scalograms and IQ plots to reduce training time through transfer learning. Similarly, CNNs have been applied for channel state information (CSI) estimation by treating the time-frequency response of fast-fading channels as 2D images [14], while advanced architectures have integrated CNNs with long short-term memory (LSTM) networks for 5G wireless systems [15]. Attention-based mechanisms have also been employed for CSI estimation, using transformer encoder architectures to focus on critical features and improve performance [16]. Beyond CSI estimation, Ye et al. [17] demonstrated the use of deep neural networks (DNNs) for end-to-end channel estimation and symbol detection in orthogonal frequency-division multiplexing (OFDM) systems. Additionally, O'Shea et al. [18] trained an LSTM-based recurrent neural network for radio anomaly detection by predicting time-series data and comparing the expected error distribution with a well-characterized signal for non-anomalous behavior. Ma et al. [19] proposed an attention-based deep separation network for co-frequency modulated communication signals.

Another area where deep learning has shown promise is in regression-based applications, where deep neural network (DNN) regression models are used to estimate relationships between input and output variables. In particular, "regression analysis is useful for predicting continuous-valued estimation targets based on a given D-dimensional vector x of independent input variables [20]."[1] Sun et al. [21] demonstrated that a DNN can approximate signal processing algorithms by treating the input-output relationship as an unknown non-linear mapping, with training samples generated through optimization algorithms. This technique has also been applied in mobile terminal usage pattern estimation [22], as well as for CSI estimation [23]. The predictive power of regression analysis in estimating continuous variables makes it particularly useful for the task of estimating the statistical distribution parameters of composite signal amplitude levels in this study[1].

"This article is an extension of [24], where the authors demonstrate a method to estimate DAC's gain using statistical moments. This article demonstrates that a composite signal generated by summing signals with different modulation techniques, power levels, data rate, and normalized frequencies has voltage levels that adhere to the GGD [25]. We simulate a dataset of 100 000 composite signals, calculating their respective GGD parameters. This dataset is utilized to train a DNN for regression, where the inputs are modulation technique, power level, data rate, the normalized frequency of each component signal, and the count of signals combined. The DNN's output targets are the parameter values of the composite signal's GGD.

The remainder of this paper is organized as follows: Section 2 discusses the generation of composite signals and their statistical distribution. Section 3 outlines the application of DNN regression for estimating the statistical distribution parameters of the summed signals. Section 4 presents

simulation results for the proposed multi-output DNN regression technique and compares these with other regression methods. Finally, Section 5 concludes the paper." [1]

2. Composite Signal's Distribution and the Dataset

"This section describes how an outgoing composite signal is generated using multiple M-ary quadrature amplitude modulated (MQAM) signals. The residual sum of squares (RSS), and Kullback-Leibler (KL) divergence [26] are used to determine a theoretical distribution that closely fits the voltage distribution of the summed signal. This theoretical distribution is used to generate a dataset to train a DNN to estimate the distribution's parameters." [1]

2.1. Composite Signal

"The composite signal is generated by adding multiple component signals with varying modulation orders, data rates, power levels, and normalized frequencies. The number of component signals are chosen randomly between two and ten, inclusive. The component MQAM signal has the following form:

$$s_m(t) = A_{i,m}g(t)\cos(2\pi f_c t) + A_{q,m}g(t)\sin(2\pi f_c t) \quad (1)$$

where $s_m(t)$ is the MQAM signal, $A_{i,m}$ and $A_{q,m}$ are the in-phase and quadrature-phase amplitude levels, $g(t)$ is the pulse shaping filter, f_c is the baseband carrier frequency, and $m = 1, 2, 3, \dots, M$ is the modulation order of MQAM signal. A composite signal can then be represented as:

$$c(t) = \sum_{n=1}^N p_n s_m(t) \quad (2)$$

where $c(t)$ represents the composite signal, N is the number of MQAM signals added, and p_n denotes the power factor.

The modulation order, M , of the MQAM component signals is uniformly sampled from the set of eight values described in Table 1. The data rate, power level, and normalized frequency of the component signal are uniformly sampled from the ranges listed in Table 1, with power levels uniformly distributed on a dB scale." [1] Normalized frequency is the frequency relative to the center of the radio frequency band of interest, and distributed in increments of 1 kHz. "Figure 4 shows the power spectrum of a representative composite signal generated by adding ten component signals." [1]

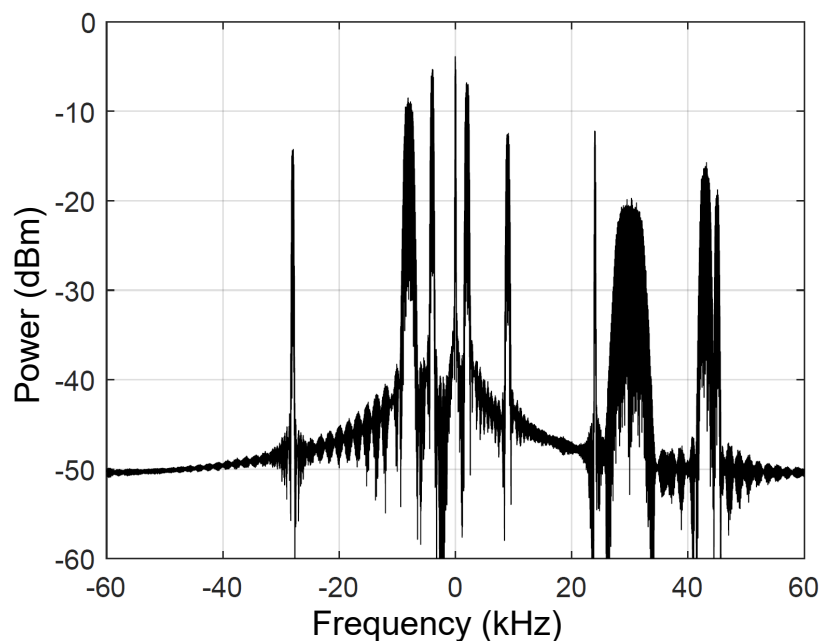


Figure 4. "Frequency domain representation of a composite signal." [1]

Table 1. "Range of parameters used to generate composite signals."[1]

Parameter	Range
Modulation order	{2, 4, 8, 16, 32, 64, 128, 256}
Data rate	[1, 10] kbps
Power level	[−30, 0] dBm
Nomalized frequency	[−60, 60] kHz
Number of component signals	[2, 10]

2.2. Distribution Fitting

To determine a suitable statistical distribution for representing the voltage levels of composite signals, we first fit continuous statistical distributions listed in [27] to the data using RSS [1]. The RSS score quantifies how well the fitted distribution replicates the individual sample data points by measuring the sum of the squared differences between the observed values (y_k) and the values predicted by the fitted distribution ($\hat{y}_k$). "The RSS score, R_{SS} , is defined as:

$$R_{SS} = \sum_{k=1}^K (y_k - \hat{y}_k)^2 \tag{3}$$

where K is the number of data samples, y_k is the k^{th} data sample, and $\hat{y}_k$ is an estimate of y_k from the fitted distribution."[1] The parameters for each candidate distribution are optimized by minimizing R_{SS} , which provides the closest fit of the statistical distribution to the data.

Once the parameters of the fitted distributions were optimized using RSS, KL-divergence was calculated to evaluate how well the theoretical distribution, generated with the optimal estimated parameters, aligned with the composite signal’s voltage distribution [1]. KL divergence measures the relative entropy between two probability distributions that indicates how much information is lost when the fitted distribution (Q) is used to approximate the true distribution of the signal (P). "For discrete probability distributions P and Q , the KL divergence score from Q to P is defined as:

$$D_{KL}(P \parallel Q) = \sum_{x \in \Phi} P(x) \log \left(\frac{P(x)}{Q(x)} \right) \tag{4}$$

where P represents the composite signal’s samples, and Q represents theoretical distribution with optimal estimated parameters, and Φ is the probability space."[1] A lower D_{KL} value indicates a better approximation, with $D_{KL} = 0$ meaning that the fitted distribution Q is identical to the true distribution P .

To compare the quality of each fitted distribution, we computed the mean, μ , and standard deviation, σ , for both R_{SS} and D_{KL} scores across 1000 instances of composite signals. The standard deviation, σ , describes consistency of fit; a lower σ suggests that a distribution performed consistently well across different composite signal instances.

Table 2 highlights the four distributions with the lowest R_{SS} and D_{KL} scores, signifying their close approximation to the composite signals’ behavior. Figure 5 illustrates the probability density function (PDF) of these distributions when fitted to a composite signal comprised of ten component signals, illustrating the efficacy of each in modeling the signal’s statistical properties.

Table 2. "SSE and KL-Divergence scores for the four distributions fitted to composite signals."[1]

Distribution	RSS ($\mu \pm \sigma$)	KL Divergence Score ($\mu \pm \sigma$)
Beta	0.2637±0.3544	0.0020±0.0018
Chi	0.5200±0.6494	0.0065±0.0047
Generalized Gamma	0.0922±0.2189	0.0007±0.0009
Rice	0.2342±0.4192	0.0025±0.0031

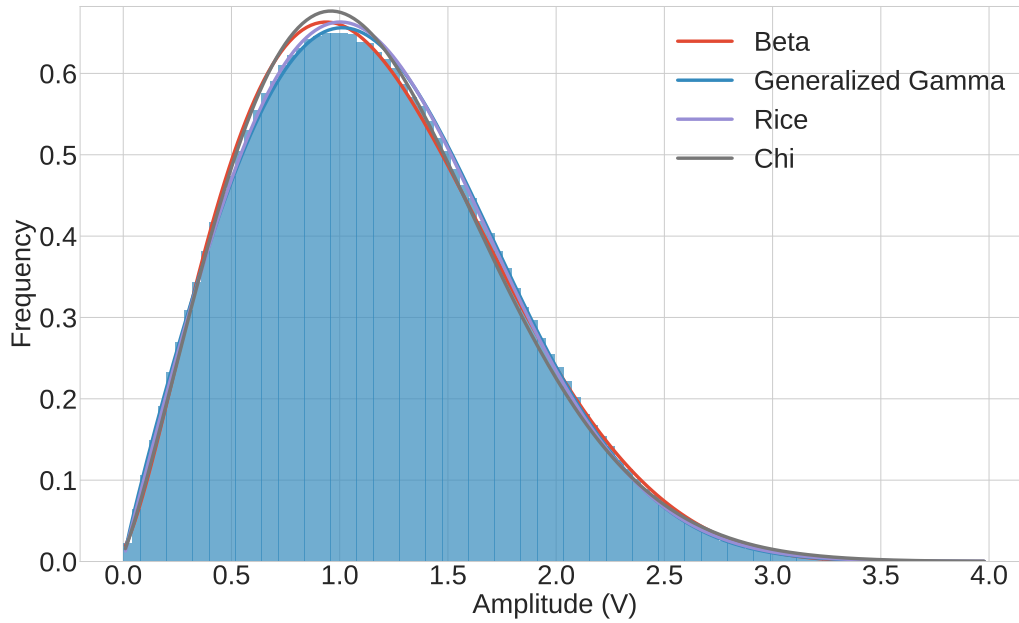


Figure 5. "Four common distributions that closely fit composite signals." [1]

Given that the GGD consistently yielded the lowest combined R_{SS} and D_{KL} scores, we use it to represent the voltage distribution of composite signals. "For a voltage level, v , the PDF and the cumulative distribution function (CDF) of v are described in (5) and (6) respectively as follows:

$$f(v; s, a, b) = \frac{\left(\frac{b}{s^a}\right) v^{a-1} e^{-\left(\frac{v}{s}\right)^b}}{\Gamma\left(\frac{a}{b}\right)} \quad (5)$$

$$F(v; s, a, b) = \frac{\gamma\left(\frac{a}{b}, \left(\frac{v}{s}\right)^b\right)}{\Gamma\left(\frac{a}{b}\right)} \quad (6)$$

where s , a , and b are the parameters of the GGD, $\Gamma\left(\frac{a}{b}\right)$ is the gamma function, and $\gamma\left(\frac{a}{b}, \left(\frac{v}{s}\right)^b\right)$ is the lower incomplete gamma function." [1]

To link the DAC's gain, G , with the PDF of the transmitted signal, we consider the DAC's clipping level, α , and aim to scale the signal's amplitude to match α based on a specific percentile of the GGD. The optimal DAC's gain, G , can be defined as:

$$G = \frac{\alpha}{F^{-1}(\theta; s, a, b)} \quad (7)$$

where $F^{-1}(\theta; s, a, b)$ represents the inverse CDF (quantile function) of the GGD for a given probability θ (e.g. the 95th percentile), indicating the voltage level below which a specified percentage of the signal's amplitude distribution lies. This calculation ensures the adjusted signal amplitude optimally utilizes the DAC's dynamic range, minimizing the risk of clipping and quantization noise.

2.3. Dataset

"To train machine learning algorithms for multi-output regression, a dataset D of R samples was simulated, where $R = 10^5$." [1] First, R composite signals were generated by summing a random number of component signals, ranging from two to ten. Where each component signal has four underlying parameters: modulation order, data rate, power level, and normalized frequency, values of which are randomly sampled from the parameter ranges described in Table 1. To ensure that categorical features have the same distance among each other in the feature space, we used the one-hot-encoding

technique to encode parameters modulation order and number of component signals. For modulation order, we employ one-hot encoding with 9 bits: 8 bits to represent the 8 possible modulation orders and 1 additional bit reserved for the case where the slot is unused (e.g., if fewer than ten signals are present). Consequently, each component signal is represented by 12 features: 9 bits for the one-hot-encoded modulation order, plus 1 feature each for data rate, power level, and normalized frequency.

When fewer than ten component signals are present (e.g., only seven signals are added), the feature vectors for the remaining slots (signals 8, 9, and 10) are set to zero, ensuring the input vector consistently maintains $12 \times 10 = 120$ features for the maximum possible number of component signals. Moreover, we one-hot-encode the total number of added signals (ranging from 2 to 10) into 9 bits. This yields an additional 9 features, leading to a total of $p = 120 + 9 = 129$ features per composite signal.

"A GGD distribution was fit on each of the composite signal to calculate GGD's $q = 3$ parameters, shape parameters a and b , and a scale parameter s . Each sample in D has an input vector of p descriptive variables $\mathbf{x}^{(r)} = (x_1^{(r)}, \dots, x_i^{(r)}, \dots, x_p^{(r)})$, and an output vector of q target variables $\mathbf{y}^{(r)} = (y_1^{(r)}, \dots, y_j^{(r)}, \dots, y_q^{(r)})$, where $i \in \{1, \dots, p\}$, $j \in \{1, \dots, q\}$, and $r \in \{1, \dots, R\}$. Overall, the simulated dataset can be described by R samples of pairs of input and output vectors, i.e., $D = \{(\mathbf{x}^{(1)}, \mathbf{y}^{(1)}), \dots, (\mathbf{x}^{(R)}, \mathbf{y}^{(R)})\}$." [1]

Since each component waveform is generated *in* the SDR's digital domain, we do not add any noise prior to summation. This assumption reflects a key benefit of SDR-based design: the ability to combine signals noise-free before analog impairments.

3. DNN Regression

"This task involves estimating the parameters of the GGD by learning a multi-output regression model from dataset D . We find a function, Ψ , that assigns q output targets, $\mathbf{y}$, for each given input vector, $\mathbf{x}$, of length p [28]. This is expressed as

$$\begin{aligned} \Psi : S_{X_1} \times \dots \times S_{X_p} &\rightarrow S_{Y_1} \times \dots \times S_{Y_q} \\ \mathbf{x} = (x_1, \dots, x_p) &\mapsto \mathbf{y} = (y_1, \dots, y_q), \end{aligned} \quad (8)$$

where S_{X_i} denotes the sample space of X_i , $\forall i \in \{1, \dots, p\}$ and S_{Y_j} denotes the sample space of Y_j , $\forall j \in \{1, \dots, q\}$.

A DNN was used as a multi-output regression technique for GGD's parameter estimation. As shown in Figure 6, the DNN has an input dimension of $p = 129$ for input feature vector $\mathbf{x}$. The input layer is followed by four dense/hidden layers with 224, 256, 448, and 64 units respectively. The final dense layer is followed by an output layer with $q = 3$ neurons for estimating target vector $\mathbf{y}$, with the estimates denoted as $\hat{\mathbf{y}}$.

All the dense layers, the input layer, and the output layer were preceded by a batch-normalization layer to standardize inputs to a layer for each mini-batch. Mini-batch size was set to 512, and an exponentially decaying learning rate was used with an initial learning rate of 0.01. The drop-out rate of the neurons in the network was set to 0.05 to mitigate overfitting of the training data. The dense layers used the ReLU activation function, thresholding raw scores o_i at 0 (i.e. $f(o) = \max(0, o_i)$), while the output layer employed a linear activation function, leaving raw scores unchanged (i.e. $f(o) = o_i$). [1] The number of layers, their corresponding number of neurons, batch size, learning rate, and drop-out rate were determined using Bayesian hyperparameter optimization technique [29].

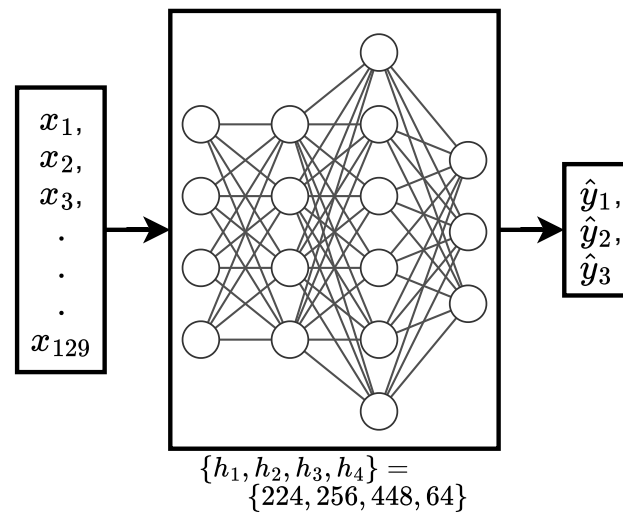


Figure 6. DNN used to estimate GGD's parameters.

4. Experiments, Results, and Analysis

This section outlines the methodology used to train the DNN and the outcomes of deploying it for estimating GGD's parameters in composite signals. We suggest using a small deep neural network (S-DNN) that achieves performance comparable to the DNN with a sufficiently large number of SDR applications. A typical example is provided to demonstrate the use of the DNN estimates.

4.1. K-Fold Cross Validation

We implemented a 10-fold cross validation technique to use all training samples while performing robust validation [30]. Ten percent of the dataset was set aside for a hold-out test set. The remaining 90% of the dataset was divided into ten equal folds. We trained the DNN ten times, using nine different folds, leaving out one fold for validation each time.

Mean Absolute Error (MAE) was used to quantify errors between the target values and the predictions at the output layer. The error values were then used to perform gradient descent to update the network's weights. Early stopping, set with a patience of 100 epochs and a precision threshold of 10^{-4} , was used to halt training when the validation set's MAE ceased to significantly improve. Figure 7 illustrates the training and validation set's average MAE trend, averaged across the ten folds, along with the minimum average validation MAE.

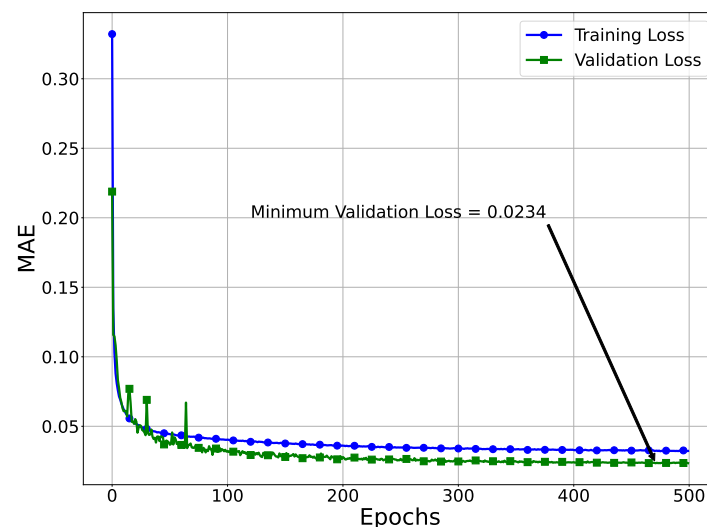


Figure 7. Average MAE training and validation loss trends across 10-fold cross validation.

4.2. Performance of the DNN

"We employed several metrics to evaluate the accuracy of the DNN in estimating the parameters of the GGD: R^2 value, explained variance, mean-squared error (MSE), and MAE. Table 3 summarizes the metric scores for each distribution parameter, along with the overall scores on all three parameters." [1] The DNN demonstrates the highest accuracy in estimating the scale parameter s , achieving an R^2 of 0.9985 and an MAE of 0.0109 on the test set. The similarity between the R^2 and explained variances scores indicates that the mean of the error is close to zero. This suggests that the estimation of the distribution parameters by the DNN are largely unbiased, which implies the DNN is less prone to overestimation or underestimation. "Figure 8 reveals the DNN's challenges in accurately predicting parameter a values beyond 0.8. The scatter for parameters a and b appears more dispersed compared to that for parameter s , with parameter b showing the largest contribution to overall MAE and MSE. This discrepancy is likely due to the higher target values for parameter b , resulting in greater error magnitudes." [1] Additionally, the smaller magnitude of parameter a compared to parameters b and s reduces its relative contribution to the loss function during training, leading to less precise estimates and contributing to the increased scatter observed in Figure 8.

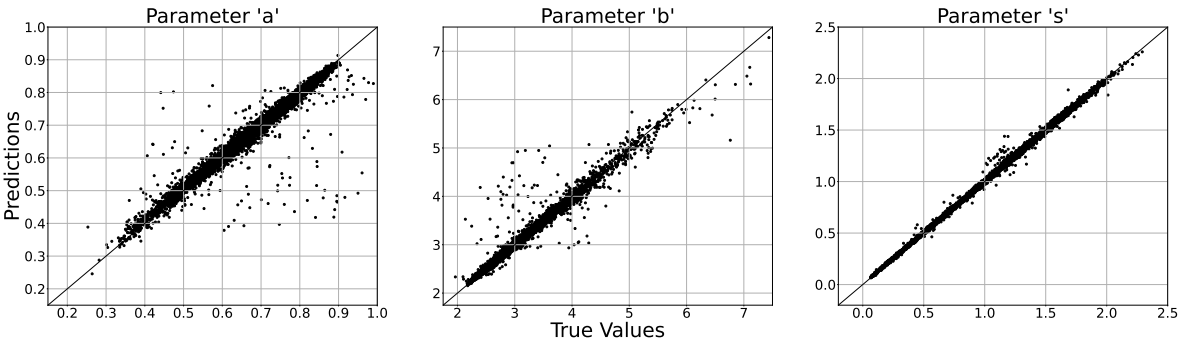


Figure 8. R^2 plot for each of the individual parameters of the GGD, estimated using the DNN.

Table 3. R^2 , explained variance, MAE, and MSE performance scores obtained by the proposed DNN.

Parameter	R^2	Explained Variance	MAE	MSE
a	0.9358	0.9358	0.0119	0.0008
b	0.9664	0.9664	0.0442	0.0133
s	0.9985	0.9985	0.0109	0.0003
Overall	0.9669	0.9669	0.0223	0.0048

Figure 9 illustrates how varying the fraction of the total dataset used for training affects the DNN's test set accuracy, as measured by the MAE. We observe a marked decrease in the overall MAE as the proportion of data used for training increases, showcasing rapid improvements in model accuracy. The rate of MAE reduction diminishes when training involves more than 50% of the data. Beyond this point, additional data yields only marginal improvements in accuracy, especially for parameters a and s . Marginal gains can be seen for parameter b until all the training data is exhausted, suggesting that parameter b 's estimates can potentially benefit from additional training data. The trends suggest that increasing the dataset size can enhance model performance, but there exists a threshold beyond which the benefits become incremental.

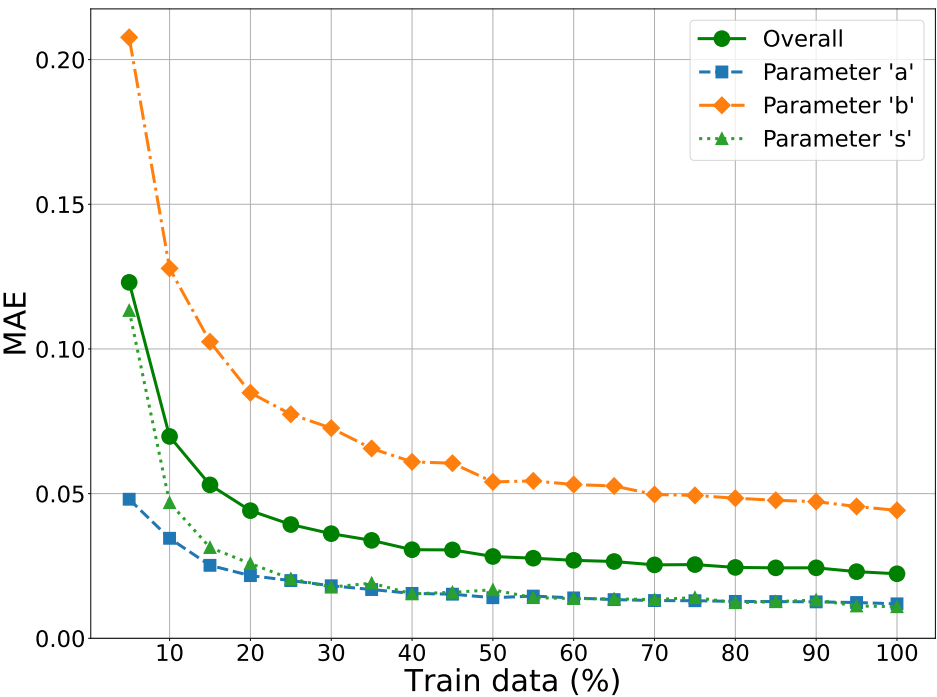


Figure 9. Trend in MAE on the hold-out test set with increasing amount of training data, expressed as a percentage of the total dataset used for training.

4.3. Model Interpretability

Model interpretability is essential, especially in critical applications like aviation. To interpret the contributions of input features to the model’s predictions we employed SHAP (SHapley Additive exPlanations) [31], which assigns an importance value to each feature, thereby explaining the model’s decisions. We grouped the input features into five primary categories: modulation order, data rate, power level, normalized frequency, and number of component signals.

Table 4 presents the normalized SHAP values (percentage importance) for each feature group across the estimated distribution parameters a , b , and s , as well as the overall importance. These values indicate the relative contribution of each feature group to the model’s predictions.

Table 4. Normalized SHAP values calculated for the hold-out test set using the DNN.

Parameter	Feature Groups (%)				
	Modulation Order	Data Rate	Power Level	Normalized Frequency	Number of Component Signals
a	42.55	1.57	46.67	1.80	7.40
b	49.94	0.68	39.35	0.75	9.28
s	38.96	0.12	48.87	0.11	11.95
Overall	44.42	0.63	44.30	0.70	9.95

The results demonstrate that modulation order and power level are the most influential features, each contributing approximately 44% to the overall model predictions. The number of component signals in a particular composite signal also has a notable impact, contributing around 10%. In contrast, data rate and normalized frequency have minimal impact on the overall estimation but are more influential when estimating the shape parameter a .

4.4. Performance as Number of SDR Applications Increase

Figure 10 and Figure 11 demonstrate how the MAE and R^2 score change as components are added to the composite signal. The performance of the DNN is worse when trying to estimate distribution

parameters for less than six component signals, especially for the shape parameters. Figure 10 shows that estimates of the shape parameter, s , become less accurate as more component signals are added to the composite signal, suggesting a trade off between shape and scale parameters as the number of SDR applications increase.

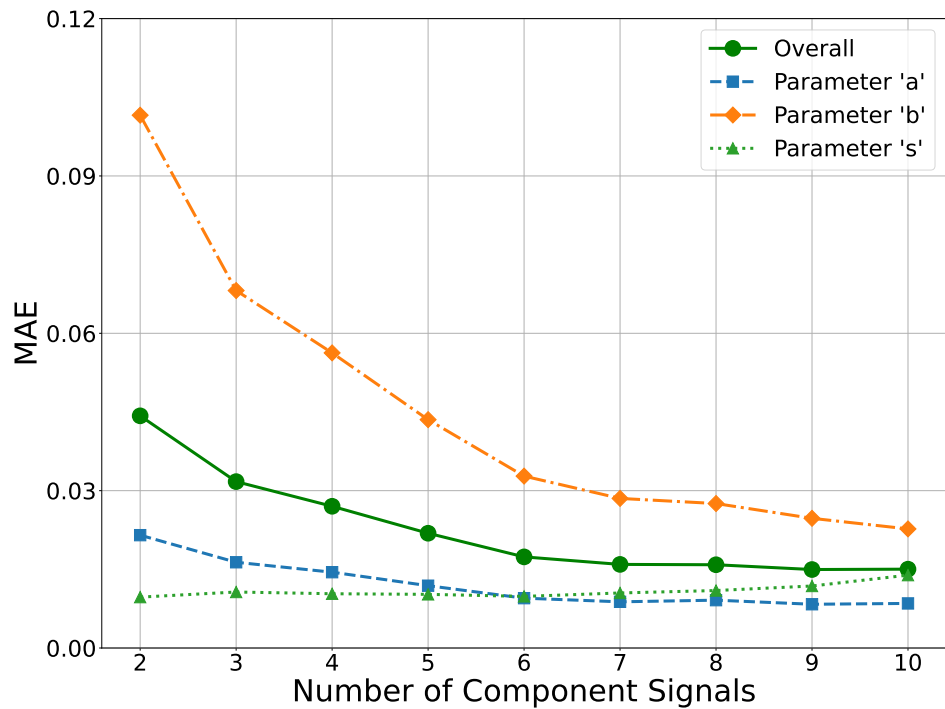


Figure 10. Trends in MAE as more component signals are added to the composite signal.

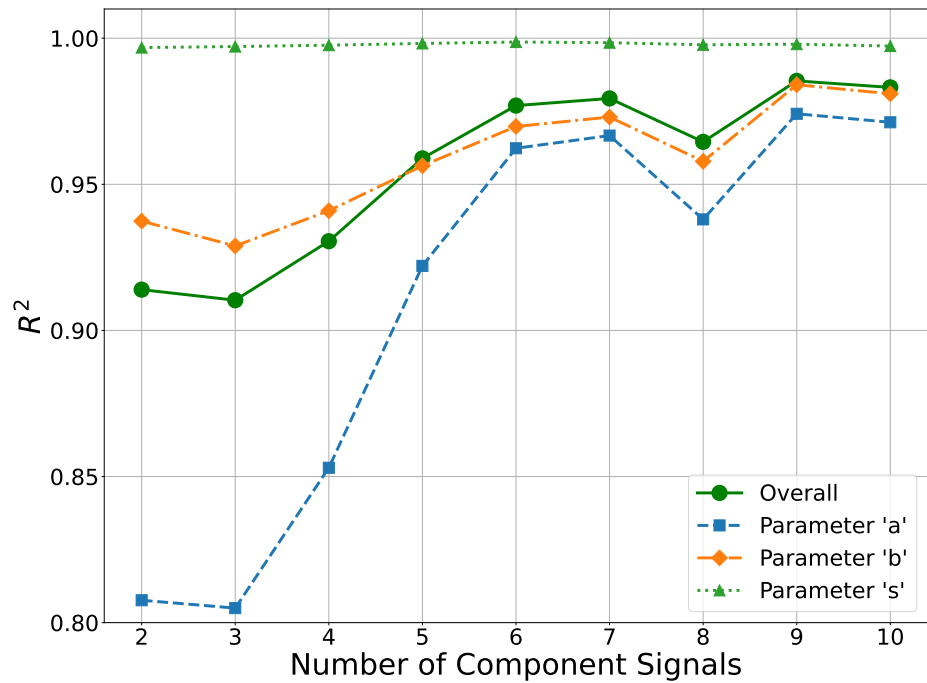


Figure 11. Trends in R^2 score as more component signals are added to the composite signal.

4.5. Small DNN

"Implementing a DNN with 992 hidden-layer neurons across four hidden layers poses significant computational training costs. To assess efficiency, its performance was compared with that of the S-DNN, with 100 neurons distributed over three hidden layers (50, 30, and 20 neurons, respectively)."[1]

Despite using only a tenth of the neurons, the S-DNN achieved an R^2 of 0.9270, which is 4.13% lower than that of the DNN.

Figure 12 compares R^2 performance of the S-DNN with the DNN as the number of component signals increase. The SNN has significantly lower accuracy estimating shape parameter a , when there are fewer than six component signals. When seven or more component signals are used, the SNN has comparable performance to the DNN. This provides an alternative to use a model with considerably fewer parameters when an SDR is supporting sufficiently large number of applications.

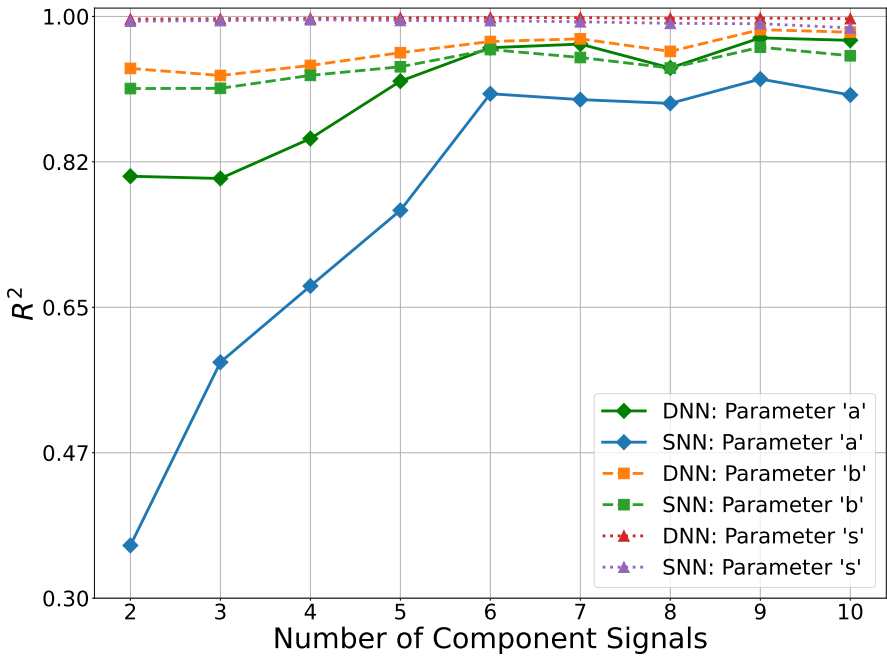


Figure 12. R^2 score comparison for S-DNN and DNN.

4.6. Implementation Details, Computational Resources, and Model Complexity

We used MATLAB to synthesize the raw composite signals, and Python with the SciPy library [27] to fit distributions to the generated composite signals. The deep learning models were built and trained using TensorFlow/Keras libraries [32,33] in Python. All experiments were performed on a personal computer equipped with an Intel Core i7-12700 CPU at 2.10 GHz, 32 GB of RAM, and an NVIDIA GeForce RTX 3060 TI GPU. Table 5 lists the training time for a training set of 90,000 samples and the total number of floating point operations (FLOPs) required per inference for the two models.

Table 5. Training time and number of FLOPs for the DNN and S-DNN models.

Model	Training Time (s)	FLOPs
DNN	596	459548
S-DNN	407	17096

The modest number of FLOPs, especially for the S-DNN, combined with manageable training times, indicate that the proposed models are practical for implementation in real-world SDR systems. The reduced computational complexity of the S-DNN makes it particularly suitable for systems with limited computational resources or when rapid adaptation is required.

4.7. Comparison with Other Techniques

The performance of the proposed neural networks is compared with following multi-output regression techniques [1]:

1. Linear Regression (LR) [34]

- 2. K-Neighbors Regression (K-NR) [35]
- 3. Decision Tree Regression (DTR) [36]
- 4. Random Forest Regression (RFR) [37]
- 5. Gradient Boosted Regression Trees (GBRT) [38]
- 6. Support Vector Regression (SVR) [39]

Table 6 presents R^2 scores achieved by these techniques, along with those obtained from S-DNN and DNN. We used Bayesian hyperparameter optimization was employed to search for optimal settings for these methods [29]. K-NR shows the lowest overall R^2 score, indicating its relative ineffectiveness in capturing the relationships between signal parameters and the distribution parameters. Except for K-NR, the techniques demonstrated strong predictive capabilities for the s parameter, with the lowest R^2 of 0.6994 achieved by LR.

Table 6. Comparison of R^2 scores obtained using different multi-output regression techniques.

Method	a	b	s	Overall
LR	0.4926	0.4484	0.6994	0.5468
K-NR	0.4795	0.5297	0.4447	0.4846
DTR	0.5578	0.7218	0.9400	0.7398
RFR	0.7651	0.8527	0.9745	0.8641
GBRT	0.8381	0.8982	0.9928	0.9097
SVR	0.5908	0.6814	0.9142	0.7288
S-DNN	0.8311	0.9548	0.9952	0.9270
DNN	0.9358	0.9664	0.9985	0.9669

The shape parameters a and b proved more challenging to estimate, requiring the use of more complex models like RFR and GBRT. While ensemble methods performed comparably to the proposed neural networks for parameter s , the neural networks displayed superior performance in estimating the shape parameters. This is critical, as the estimation of shape parameters significantly impacts the prediction of composite signal’s peak amplitudes.

4.8. Typical Example

"To demonstrate a normative scenario, consider a composite signal generated by adding the signals described in Table 7. Figure 13 compares the CDF of the composite signal’s voltage levels, as estimated by the DNN,"[1] with the ground truth, showing that the two distributions closely align. Additionally, we examine three scenarios detailing the probabilities of surpassing specific voltage levels. For instance, the estimated distribution indicates that 1% of the voltage levels exceed 1.26 V. Such insights are valuable when determining the DAC’s gain. "For example, if the DAC’s operational range is α V and G is the gain, then setting $G = \frac{\alpha}{1.26}$ ensures that only 1% of the signal samples will be clipped."[1]

Table 7. Features of six component signals used to generate a composite signal.

Modulation Order	Data rate (bps)	Power level (dBm)	Normalized Frequency (kHz)
64	6753	-23.65	-9
32	5566	-23.05	-44
4	1460	-26.23	-21
8	5518	-22.09	37
16	9751	-10.05	28
256	7993	-5.43	14

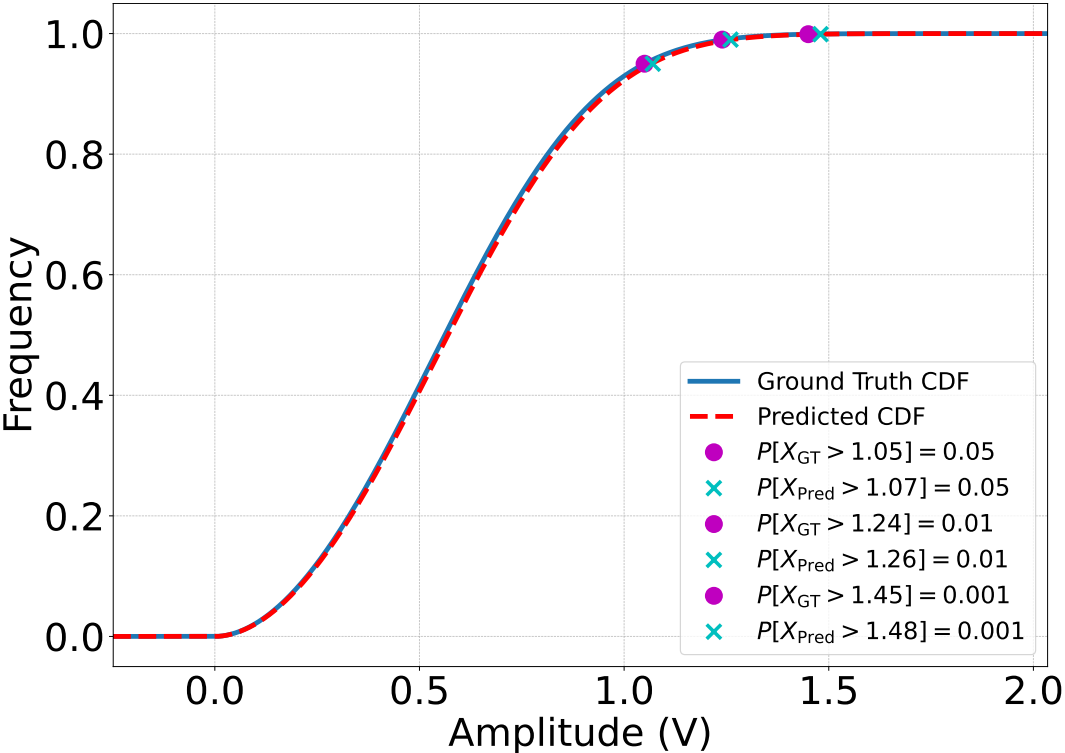


Figure 13. "CDF of the combined signal estimated using the DNN."

5. Conclusion

This study introduces a deep learning based approach to determine the optimal gain of a DAC in scenarios where an SDR generates multiple baseband signals for aircraft communications. We identified an analytical distribution that closely models the voltage distribution of the composite signal, using the RSS fitting method, and found that the GGD fits the composite signal particularly well. We trained a DNN on simulated composite signal data to estimate the parameters of the GGD. This trained DNN allows rapid prediction of the composite signal’s voltage distribution as it varies with changes in the SDR configuration. Utilizing the predicted voltage distribution, the DAC’s gain can be dynamically adjusted, maximizing its range utilization and minimizing potential signal distortion.

Author Contributions: Conceptualization, K.K.; data curation, V.K.; formal analysis, V.K.; investigation, V.K.; methodology, V.K.; project administration, K.K.; software, V.K.; supervision, K.K.; validation, V.K. and K.K.; visualization, V.K.; writing - original draft, V.K.; writing - reviewing & editing, V.K. and K.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The methodology used to simulate the data is described in the manuscript, further inquiries can be directed to the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

SDR	Software Defined Radio
DAC	Digital to Analog Converter
RF	Radio Frequency
GGD	Generalized Gamma Distribution
DL	Deep Learning
DNN	Deep neural network
MQAM	M-ary Quadrature Amplitude Modulated

RSS	Residual Sum of Squares
KL	Kullback-Leibler
PDF	Probability Density Function
CDF	Cumulative Density Function
S-DNN	Small-Deep Neural Network
MAE	Mean Absolute Error
MSE	Mean Squared Error
SHAP	SHapley Additive exPLanations
FLOP	Floating Point Operation
LR	Linear Regression
K-NR	K-Neighbors Regression
DTR	Decision Tree Regression
RFR	Random Forest Regression
GBRT	Gradient Boosted Regression Trees
SVR	Support Vector Regression

References

1. Gajjar, V.K. Machine Learning Applications in Plant Identification, Wireless Channel Estimation, and Gain Estimation for Multi-User Software-Defined Radio. Doctoral dissertation, Missouri University of Science and Technology, 2022.
2. Baltaci, A.; Dinc, E.; Ozger, M.; Alabbasi, A.; Cavdar, C.; Schupke, D. A survey of wireless networks for Future Aerial Communications (FACOM). *IEEE Communications Surveys & Tutorials* **2021**, *23*, 2833–2884. <https://doi.org/10.1109/COMST.2021.3103044>.
3. Furse, C.; Haupt, R. Down to the wire [aircraft wiring]. *IEEE Spectrum* **2001**, *38*, 34–39. <https://doi.org/10.1109/6.898797>.
4. Sámano-Robles, R.; Tovar, E.; Cintra, J.; Rocha, A. Wireless avionics intra-communications: Current trends and design issues. In Proceedings of the Eleventh ICDIM, Porto, Portugal, 2016; pp. 266–273.
5. Amrhar, A.; Kisomi, A.A.; Zhang, E.; Zambrano, J.; Thibeault, C.; Landry, R. Multi-Mode reconfigurable Software Defined Radio architecture for avionic radios. In Proceedings of the ICNS, Herndon, VA, USA, 2017; pp. 2D1–1–2D1–10.
6. Price, N.; Kosbar, K. Decoupling hardware and software concerns in aircraft telemetry SDR systems. In Proceedings of the Proc. ITC, Glendale, AZ, USA, 2018; pp. 474–483.
7. Pouyanfar, S.; Sadiq, S.; Yan, Y.; Tian, H.; Tao, T.; Reyes, M.P.; Shyu, M.; Chen, S.; Iyenger, S.S. A survey on deep learning: Algorithms, techniques, and applications. *ACM Computing Surveys (CSUR)* **2019**, *51*, 1–36. <https://doi.org/10.1145/3234150>.
8. Dong, S.; Wang, P.; Abbas, K. A survey on deep learning and its applications. *Computer Science Review* **2021**, *40*. <https://doi.org/10.1016/j.cosrev.2021.100379>.
9. França, R.P.; Monteiro, A.; Arthur, R.; Iano, Y. An overview of deep learning in big data, image, and signal processing in the modern digital age. In *Trends in Deep Learning Methodologies*; Academic Press, 2020; pp. 63–87.
10. Dai, L.; Jiao, R.; Adachi, F.; Poor, H.V.; Hanzo, L. Deep learning for wireless communications: An emerging interdisciplinary paradigm. *IEEE Wireless Communications* **2020**, *27*, 133–139. <https://doi.org/10.1109/MWC.001.1900491>.
11. Peng, S.; Jiang, H.; Wang, H.; Alwageed, H.; Zhou, Y.; Sebdani, M.M.; Yao, Y. Modulation classification based on signal constellation diagrams and deep learning. *IEEE Transactions on Neural Networks and Learning Systems* **2019**, *30*, 718–727. <https://doi.org/10.1109/TNNLS.2018.2850703>.
12. Meng, F.; Chen, P.; Wu, L.; Wang, X. Automatic modulation classification: A deep learning enabled approach. *IEEE Transactions on Vehicular Technology* **2018**, *67*, 10760–10772. <https://doi.org/10.1109/TVT.2018.2868698>.
13. Nambisan, A.; Gajjar, V.; Kosbar, K. Scalogram Aided Automatic Modulation Classification. In Proceedings of the Proc. ITC, Glendale, AZ, USA.
14. Soltani, M.; Pourahmadi, V.; Mirzaei, A.; Sheikhzadeh, H. Deep learning-based channel estimation. *IEEE Communications Letters* **2019**, *23*, 652–655. <https://doi.org/10.1109/LCOMM.2019.2898944>.
15. Luo, C.; Ji, J.; Wang, Q.; Chen, X.; Li, P. Channel State Information Prediction for 5G Wireless Communications: A Deep Learning Approach. *IEEE Transactions on Network Science and Engineering* **2020**, *7*, 227–236. <https://doi.org/10.1109/TNSE.2018.2848960>.

16. Luan, D.; Thompson, J. Attention Based Neural Networks for Wireless Channel Estimation. In Proceedings of the 2022 IEEE 95th Vehicular Technology Conference: (VTC2022-Spring), 2022, pp. 1–5. <https://doi.org/10.1109/VTC2022-Spring54318.2022.9860803>.
17. Ye, H.; Li, G.Y.; Juang, B.H. Power of Deep Learning for Channel Estimation and Signal Detection in OFDM Systems. *IEEE Wireless Communications Letters* **2018**, *7*, 114–117. <https://doi.org/10.1109/LWC.2017.2757490>.
18. O'Shea, T.J.; Clancy, T.C.; McGwier, R.W. Recurrent Neural Radio Anomaly Detection, 2016, [arXiv:cs.LG/1611.00301].
19. Ma, H.; Zheng, X.; Yu, L.; Zhou, X.; Chen, Y. A novel end-to-end deep separation network based on attention mechanism for single channel blind separation in wireless communication. *IET Signal Processing* **2023**, *17*, e12173, [<https://ietresearch.onlinelibrary.wiley.com/doi/pdf/10.1049/sil2.12173>]. <https://doi.org/10.1049/sil2.12173>.
20. Jiang, C.; Zhang, H.; Ren, Y.; Han, Z.; Chen, K.C.; Hanzo, L. Machine learning paradigms for next-generation wireless networks. *IEEE Wirel. Commun.* **2017**, *24*, 98–105. <https://doi.org/10.1109/MWC.2016.1500356WC>.
21. Sun, H.; Chen, X.; Shi, Q.; Hong, M.; Fu, X.; Sidiropoulos, N.D. Learning to Optimize: Training Deep Neural Networks for Interference Management. *IEEE Transactions on Signal Processing* **2018**, *66*, 5438–5453. <https://doi.org/10.1109/TSP.2018.2866382>.
22. Donohoo, B.K.; Ohlsen, C.; Pasricha, S.; Xiang, Y.; Anderson, C. Context-aware energy enhancements for smart mobile devices. *IEEE Trans. Mob. Comput.* **2014**, *13*, 1720–1732. <https://doi.org/10.1109/TMC.2013.94>.
23. Gajjar, V.; Kosbar, K. CSI estimation using artificial neural network. In Proceedings of the Proc. ITC, Las Vegas, NV, USA, 2019; pp. 413–422.
24. Gajjar, V.; Kosbar, K. Rapid gain estimation for multi-user software defined radio applications. In Proceedings of the Proc. ITC, Las Vegas, NV, USA.
25. Stacy, E.W. A generalization of the gamma distribution. *Ann. Math. Stat.* **1962**, *33*, 1187–1192.
26. Kullback, S.; Leibler, R. On information theory and sufficiency. *The Annals of Mathematical Statistics* **1951**, *22*, 79–86.
27. Continuous Statistical Distributions — SCIPY V1.13.0 Manual.
28. Borchani, H.; Varando, G.; Bielza, C.; Larranaga, P. A survey on multi-output regression. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* **2015**, *5*, 216–233. <https://doi.org/10.1002/widm.1157>.
29. Brochu, E.; Cora, V.M.; de Freitas, N. A Tutorial on Bayesian Optimization of Expensive Cost Functions, with Application to Active User Modeling and Hierarchical Reinforcement Learning. *CoRR* **2010**, *abs/1012.2599*, [1012.2599].
30. Ojala, M.; Garriga, G.C. Permutation tests for studying classifier performance. *Journal of machine learning research* **2010**, *11*.
31. Lundberg, S.M.; Lee, S.I. A unified approach to interpreting model predictions. In Proceedings of the Proceedings of the 31st International Conference on Neural Information Processing Systems, Red Hook, NY, USA, 2017; NIPS'17, p. 4768–4777.
32. Abadi, M.; Agarwal, A.; Barham, P.; Brevdo, E.; Chen, Z.; Citro, C.; Corrado, G.S.; Davis, A.; Dean, J.; Devin, M.; et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems, 2015. Software available from tensorflow.org.
33. Chollet, F.; et al. Keras. <https://keras.io>, 2015.
34. Montgomery, D.C.; Peck, E.A.; Geoffrey, G.G. Multiple linear regression. In *Introduction to Linear Regression Analysis*, 6 ed.; John Wiley and Sons, Inc.: Hoboken, NJ, USA, 2021; pp. 72–85.
35. Kramer, O. *K-Nearest Neighbors*; Springer: Berlin, Germany, 2013; pp. 13–23.
36. Dea'ith, G. Multivariate regression trees: A new Technique for modelling species-environment relationships. *Ecology* **2002**, *83*, 1105–1117. [https://doi.org/10.1890/0012-9658\(2002\)083\[1105:MRTANT\]2.0.CO;2](https://doi.org/10.1890/0012-9658(2002)083[1105:MRTANT]2.0.CO;2).
37. Segal, M.R. Machine learning benchmarks and random forest regression **2004**.
38. Prettenhofer, P.; Louppe, G. Gradient Boosted Regression Trees in Scikit-Learn. 23 February 2014.
39. Brudnak, M. Vector-valued support vector regression. In Proceedings of the IEEE Int. Joint Conf. on Neural Network Proc., Vancouver, BC, Canada, 2006; pp. 1562–1569.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.