

On Using Artificial Neural Networks for Multipath Radio Channel Estimation

Rostislav Karásek*, Christian Gentner†,

*Institute of Flight Guidance, DLR, Braunschweig, Germany, rostislav.karasek@dlr.de

†Institute of Communications and Navigation, DLR, Weßling, Germany

Abstract—Line spectral estimation is an important technique widely used in signal processing, e.g., radio channel parameter estimation. However, the current machine learning-based methods for line spectral estimation are incomplete, and many problems still need to be addressed. We propose an Artificial Neural Network (ANN) architecture that can directly estimate the radio channel delay parameters, including the number of delays present in the radio channel measurements. We propose a robust noise regularization technique, which successfully mitigates the problem of ANN overfitting. Finally, we propose a novel loss function significantly improving the achievable precision of the radio channel parameter estimation. We compare our results with the theoretical limit Cramer-Rao Lower Bound (CRLB) and classical approaches such as the Space-Alternating Generalized Expectation-maximization (SAGE) and Superfast Line Spectral Estimation (SLSE). Our results show that this novel loss function enables the ANN-based delay estimator to approach the CRLB for a single delay case. The proposed method still achieves a super-resolution performance for larger model orders. The ANN-based approach can be approximately 24 times faster than the SAGE algorithm and 180 times faster than the SLSE.

Index Terms—Artificial Neural Network, Convolutional Neural Network, Machine Learning, Noise Regularization, Line Spectral Estimation, Radio Channel Parameter Estimation.

I. INTRODUCTION

Although the idea of Artificial Neural Network (ANN) occurred already in the 1940s [1], the real boom started with the success of a deep learning architecture AlexNet in 2012 [2]. This established a new standard for solving classification problems. Since that, the ANN has spread to many other fields of science, e.g., into many areas of computer graphics simulations [3], [4], reinforcement learning [5] [6], and biochemistry [7]. The experiments using ANN for radio channel estimation are a recent addition to this growing ecosystem.

One of the first experiments shows Convolutional Neural Network (CNN) application for Angle of Arrival (AoA) estimation in acoustics systems [8]. This method is a dictionary-based approach, where one AoA is extracted as an index of the output layer maximum, limiting the achievable precision and number of possible signal sources. The latter approach in [9] solves the single AoA limitation by using ANN to preprocess the received signal. The resulting pseudo-spectrum is processed by the Multiple Signal Classification (MUSIC) algorithm to obtain multiple AoAs. Since the number of ANN outputs does not limit the MUSIC algorithm resolution, the presented method is no longer a dictionary-based approach.

Similar ideas are used in the following research aiming for the line spectra estimation in [10]–[12]. An ANN preprocesses the received signal, and the output is then processed by classical frequency estimation algorithms like MUSIC or root-MUSIC.

For the ANN, it is a challenging problem to directly estimate desired radio channel parameters like AoA, Time of Arrival (ToA), Time Difference of Arrival (TDoA), or Doppler frequency. This approach was briefly studied in [10] where some open problems were highlighted, e.g., how to output a variable number of radio channel parameters with a fixed ANN architecture and how to compare this set of outputted parameters with the ground truth to obtain a loss function for the ANN training. The ANN-based direct estimation of the Line-of-Sight (LoS) component delay in the multipath radio channel is proposed in [13].

In the paper, we present several novel contributions to the state-of-the-art of ANN-based line spectral estimation.

- 1) The comparison of the ANN-based estimator precision with the theoretical limit given by the Cramer-Rao Lower Bound (CRLB) and with the benchmark algorithms approaching this theoretical limit.
- 2) We propose a method to solve the problem of unknown model order.
- 3) We introduce a novel loss function for the ANN training.
- 4) We propose a noise regularization technique, which adds noise to the input dataset with the same variance as the noise present in the radio channel.
- 5) Finally, we propose three different methods for the data association of the ANN outputs with the true values, which is vital for the ANN training.

II. THEORETICAL BACKGROUND AND CONCEPTS

This section describes the general concepts and approaches necessary for modeling a Channel Impulse Response (CIR) and estimating its parameters. The classical Maximum Likelihood (ML) estimation approach helps to understand the parameter estimation problem. In particular, it can show us the theoretical limits for parameter estimation, provide an intuition for building the training dataset, and help us design a custom loss function improving the ANN training.

A. Multipath propagation

The transmitted signal propagates towards the receiver through a medium that distorts it due to its physical properties.

The most common source of distortion is Additive White Gaussian Noise (AWGN) added to the received signal by the receiver. Also, the transmitted signal propagates with a finite speed through the medium, causing a delay between transmission and reception. This delayed replica of the transmitted signal is called the LoS component. Additionally, the transmitted signal can be reflected and scattered by obstacles present in the medium creating replicas of the transmitted signal received with an additional delay.

This process is modeled using a multipath radio channel. All of the received replicas, including the LoS component, are scaled and added together at the receiver side. The multipath radio channel can be described by its impulse response called CIR

$$h(\tau) = \sum_{\ell=1}^L \alpha_{\ell} \delta(\tau - \tau_{\ell}), \quad (1)$$

where all L replicas of the transmitted signal are called Multipath Components (MPCs) [14]. Each MPC is received with a delay τ_{ℓ} and scaled by a complex amplitude α_{ℓ} . Each delay is described as a deterministic parameter given by delta distribution $\delta(\cdot)$. Finally, the received signal is given as a convolution between the transmitted signal $r(t)$ and CIR as

$$s(t) = r(t) * h(\tau) + \nu(t) = \tilde{s}(t) + \nu(t), \quad (2)$$

where $\nu(t)$ is a normally distributed uncorrelated random process with zero mean and constant variance $\sigma^2(t)$.

If the CIR is finite, we can perform sampling of the received signal without loss of information [15]. Hence, the sampled received signal can be written in a vector form as

$$\mathbf{s} = [s_0, \dots, s_m, \dots, s_{M-1}]^T, \quad (3)$$

where T marks transpose function, and m is indexing the samples with sampling period T . The total number of samples is M . The sampled transmitted signal \mathbf{r} , CIR \mathbf{h} , and noise $\boldsymbol{\nu}$ are obtained similarly to (3).

The model parameters consisting of all delays $\boldsymbol{\tau} = [\tau_1, \dots, \tau_{\ell}, \dots, \tau_L]^T$ and all complex-valued amplitudes $\boldsymbol{\alpha} = [\alpha_1, \dots, \alpha_{\ell}, \dots, \alpha_L]^T$ describe the radio channel observation

$$\mathbf{x} = \mathbf{B}(\boldsymbol{\tau}) \boldsymbol{\alpha} + \mathbf{n}, \quad (4)$$

where $\mathbf{B}(\boldsymbol{\tau})$ is a nonlinear function describing the multipath radio channel structure, and the radio channel observation is obtained from the frequency spectrum of the transmitted \mathbf{R} and received \mathbf{S} signals \mathbf{r} and \mathbf{s} , respectively, as

$$\mathbf{x} = \mathbf{R} \oslash \mathbf{S}, \quad (5)$$

where \oslash represents Hadamard division. Note that the parameters of the noise vector in (4) depends on \mathbf{r} since

$$\mathbf{n} = \mathbf{N} \oslash \mathbf{R}, \quad (6)$$

where \mathbf{N} is a frequency spectrum of $\boldsymbol{\nu}$. This work assumes \mathbf{r} with unit energy and constant complex envelope. Hence the noise transformation in (6) preserve its properties.

B. ML multipath radio channel parameter estimation

In the previous section, we described the mathematical model of the received signal. A sparse set of parameters $\boldsymbol{\theta} = [\boldsymbol{\tau}^T, \boldsymbol{\alpha}^T]^T$ can parameterize the multipath radio channel.

The CRLB gives the minimum variance any unbiased estimator can achieve. The derivation for a delay estimation is well studied and can be found, e.g., in [16]–[18]. According to [18], the minimum variance that any nonlinear channel parameter estimator can achieve is given as

$$CRB_{\hat{\tau}} = \frac{\sigma^2}{\|\boldsymbol{\alpha}\|^2} \frac{6}{M(M^2 - 1)}, \quad (7)$$

while the minimal variance of the linear parameter magnitude estimator is

$$CRB_{\|\hat{\alpha}\|} = \frac{\sigma^2}{2M}, \quad (8)$$

and the minimal variance of the linear parameter phase estimator is

$$CRB_{\angle \hat{\alpha}} = \frac{\sigma^2}{\|\boldsymbol{\alpha}\|^2} \frac{1}{2M}. \quad (9)$$

For a sufficiently large Signal-to-Noise Ratio (SNR), the CRLB performance is achievable by an ML estimator. The likelihood function

$$p(\mathbf{x}|\boldsymbol{\theta}, \sigma^2) = \frac{1}{(\pi\sigma^2)^M} \exp\left(-\frac{1}{2\sigma^2} (\mathbf{x} - \hat{\mathbf{x}})^H (\mathbf{x} - \hat{\mathbf{x}})\right), \quad (10)$$

where H marks conjugate transpose and

$$\hat{\mathbf{x}} = \mathbf{B}(\hat{\boldsymbol{\tau}}) \hat{\boldsymbol{\alpha}}, \quad (11)$$

can be used to derive ML estimators $\hat{\boldsymbol{\tau}}$ and $\hat{\boldsymbol{\alpha}}$ as

$$\hat{\boldsymbol{\tau}} = \arg \max_{\boldsymbol{\tau}} \mathbf{x}^H \mathbf{B}(\boldsymbol{\tau}) \hat{\boldsymbol{\alpha}}, \quad (12)$$

$$\hat{\boldsymbol{\alpha}} = \left(\mathbf{B}^H(\boldsymbol{\tau}) \mathbf{B}(\boldsymbol{\tau}) \right)^{-1} \mathbf{B}^H(\boldsymbol{\tau}) \mathbf{x}, \quad (13)$$

where (13) is a Best Linear Unbiased Estimator (BLUE) of linear parameters $\boldsymbol{\alpha}$ [18].

Solving (12) is a multidimensional optimization problem, which is generally hard to solve. However, the problem can be separated and each nonlinear parameter optimized independently and iteratively in a round-robin schedule. This approach is used by the Space-Alternating Generalized Expectation-maximization (SAGE) algorithm [19]. SAGE can asymptotically approach the CRLB, when the number of MPCs is known a priori. If the model order is not known, the model order and parameters have to be estimated jointly as in [20], to approach CRLB.

C. Artificial Neural Network

In general, an ANN approximates some unknown and possibly intractable nonlinear function by another nonlinear parametric function. In other words, if we have a dataset $\{\mathbf{x}^{(i)}, \mathbf{t}^{(i)}\}$ of observation $\mathbf{x}^{(i)}$ and target $\mathbf{t}^{(i)}$ variable pairs we desire to find a relation between observation and target given by a function

$$\mathbf{t} = f(\mathbf{x}). \quad (14)$$

Since the function $f(\cdot)$ is generally unknown, we can try to find its parametric approximation as

$$\mathbf{t} \approx q(\mathbf{W}, \mathbf{x}), \quad (15)$$

where the nonlinear function $q(\cdot, \cdot)$ has to be selected, and the weights \mathbf{W} have to be estimated.

Selecting the $q(\cdot, \cdot)$ architecture is based on the core idea of the ANN, which is to chain simple linear functions with well-defined nonlinear activation functions.

The process of estimating the weights is also called ANN training. The locally optimal weights are usually obtained using gradient approaches, where a backpropagation method can efficiently evaluate the gradient of any feed-forward ANN architecture [21].

Practically, the most used training method is the Adam optimizer [22]. Compared to the Stochastic Gradient Descent (SGD), the Adam optimizer also calculates the exponential moving averages of past squared gradients, which was shown to be modeling a diagonal approximation of the Fisher Information Matrix (FIM) [23], [24].

Almost arbitrary architecture, without loops, is achievable using generic ANN libraries like TensorFlow [25], Caffe [26], and PyTorch [27]. These libraries can automatically obtain the gradient of the designed ANN and use one of the standard optimizers to find the weights fitting the training dataset.

III. DATASET GENERATOR

This section describes the dataset generator that generates examples of the multipath radio channel observations and the corresponding MPC parameters describing the multipath radio channel.

The number of MPCs is selected before generating the training dataset by the $nMPC$ parameter. However, it is possible to have a random number of MPCs in each generated measurement by switching on the $rndVis$ parameter. Then, the $nMPC$ serves as the maximal possible number of MPCs. The actual number of MPCs is selected randomly and uniformly between 1 and $nMPC$. The SNR of each MPC is randomly and uniformly selected between $SNRmin$ and $SNRmax$. Note that the SNR is selected uniformly in the dB scale; hence, the SNR distribution in linear scale follows the log-uniform distribution. The $minSep$ parameter gives the minimal separation between two MPCs.

The measurements are generated directly from the multipath propagation model (4) in the frequency domain as

$$\mathbf{x} = \mathbf{B}(\boldsymbol{\tau}) \boldsymbol{\alpha} + \mathbf{n} = e^{-j2\pi\mathbf{F}\boldsymbol{\tau}^T} (\|\boldsymbol{\alpha}\| \odot e^{-j\boldsymbol{\phi}}) + \mathbf{n}, \quad (16)$$

where $\mathbf{F} = [0, \dots, m, \dots, M-1]^T$, delays $\boldsymbol{\tau}$ are randomly sampled from a uniform distribution on the interval $[0, 1)$ but assured that no pair of delays is closer than $minSep$, MPC magnitudes $\|\boldsymbol{\alpha}\|$ are obtained from randomly sampled SNR, \odot is Hadamard product, MPC phases $\boldsymbol{\phi}$ are sampled from a uniform distribution on the interval $[0, 2\pi)$, and \mathbf{n} is multivariate normally distributed noise

$$\mathbf{n} \sim \mathcal{N}(0, 2\pi^2\mathbf{I}) \quad (17)$$

with identity matrix \mathbf{I} and $\text{rank}(\mathbf{I}) = M$. Finally, the generated measurements representing the CIR are obtained from (16) using the Inverse Fast Fourier Transform (IFFT). The real and imaginary parts of the CIR are appended before being inputted to the ANN.

IV. ANN ARCHITECTURE AND TRAINING SCHEDULE FOR DELAY ESTIMATION

The architecture and training of the ANN consist of many hyperparameters having a massive impact on the achievable precision of the estimation. In this section, we describe all the aspects taken into account when designing the ANN architecture and configuring the training schedule of the proposed ANN. We use the TensorFlow library [25] to design our ANN architecture in an organized manner and leverage the automatic differentiation that simplifies the design of custom ANN loss functions we need. We aim to find an ANN architecture with minimal design choices and hyperparameters to tune while assuring robust convergence properties.

The proposed ANN architecture consists of a Gaussian noise layer, normalization layer, four convolutional layers, and two fully connected layers. All convolutional layers and the second last fully connected layer use the Rectified Linear Unit (ReLU) activation function. The last fully connected layer has twice as many outputs as the maximum expected number of MPCs. The first $nMPC$ outputs represent estimated delays $\hat{\boldsymbol{\tau}}$ and use the ReLU activation function. The remaining $nMPC$ outputs represent estimated weights $\hat{\boldsymbol{w}}$ and use a logistic sigmoid activation function

$$\sigma(x) = \frac{1}{1 + e^{-x}}. \quad (18)$$

The logistic sigmoid activation function assures that each weight in $\hat{\boldsymbol{w}}$ is scaled to the interval $(0, 1)$, where 0 denotes that the MPC is inactive and 1 denotes that the MPC is active. Thus, it can be used as a probability that the corresponding delay is active or inactive. The weights $\hat{\boldsymbol{w}}$ provide a simple method for coping with a changing number of MPCs while the ANN architecture is fixed.

When the ANN architecture is large enough, consisting of enough tunable parameters, it assures that it can learn an arbitrary function from the training data. On the other hand, the ANN architecture is flexible enough to memorize the training set perfectly, which leads to severe overfitting. The standard ANN architecture usually contains some regularization method [21] to resist overfitting. However, the most common regularization methods like weight or activity regularization, early stopping, or dropout are hard to tune to reduce overfitting but not cause underfitting.

Experimentally, we found out that overfitting and underfitting is a severe problem that causes the ANN not to get closer to the CRLB. With a significant amount of activity regularization tuning, it was possible to achieve delay precision about ten times worse than CRLB for one MPC in noise. However, assuming a different number of MPCs, or any model change caused a significant drop in precision. Interestingly, also the

batch size influences the achievable precision. The achievable precision decreased when the batch size was larger than 32, probably due to ANN converging to a local minimum. This behavior is undesirable since it is not possible to fully utilize the parallelization of the ANN using a Graphics Processing Unit (GPU).

We found that the noise regularization [28] is significantly more robust to small changes of the ANN architecture or change of MPCs number. Therefore, only noise regularization is assumed in this work. Generally, the noise regularization method adds a small amount of Gaussian noise to the input or output of an ANN layer. We use the TensorFlow layer *GaussianNoise* to add Gaussian noise with zero mean and Standard Deviation (STD) $stdIn$ to the input data.

We use a trick for the ANN training where the inputted CIR is noise-free, but then the Gaussian noise layer adds noise with the variance set to achieve the desired SNR. Intuitively, it can be seen as generating an infinite size training dataset since the noise added to the noise-free CIR is different in each training epoch. Thanks to this, the ANN is not overfitting the training data. Moreover, this approach solves the batch size problem. We found out that the batch size no longer influences the achievable precision with the noise regularization. Thanks to the noise regularization technique, we can fully utilize the GPU used for training (*NVIDIA GeForce GTX 980 Ti*) and use batch size $nBatch = 258$ without getting stuck in local optima.

The Gaussian noise layer is followed by the normalization layer, which normalizes all noisy CIRs to unit energy. This step is required. Otherwise the model order might be estimated based on the CIR energy, which is undesirable. Normalization of the input of the ANN is always preferable, and the proposed normalization to the unit energy is a logical choice for the signal processing task.

For training, we use the Adam optimizer [22] with exponential decay defining learning rate lr at each training *step* as

$$lr = lrInit \cdot lrRate^{step/trStep}, \quad (19)$$

where the initial learning rate $lrInit$, the decay rate of the learning rate $lrRate$, and the decay step of the learning rate $lrStep$ are adjustable parameters of the Adam optimizer.

The ANN loss function for the delay estimation aims to minimize Mean-Square Error (MSE) error between the predicted and true delays

$$L_2(\boldsymbol{\tau}, \hat{\boldsymbol{\tau}}) = (\boldsymbol{\tau} - \hat{\boldsymbol{\tau}})^T (\boldsymbol{\tau} - \hat{\boldsymbol{\tau}}). \quad (20)$$

However, this straightforward error evaluation is not possible since any delay ordering in vector $\boldsymbol{\tau}$ yields the same CIR. This is a problem because, while $\boldsymbol{\tau}$ is an unordered set, the $\hat{\boldsymbol{\tau}}$ outputted from the ANN is always an ordered set. Moreover, the number of delays is unknown and possibly differ for each CIR, but the cardinality of the ANN output $|\hat{\boldsymbol{\tau}}|$ is fixed.

Setting the maximum number of output delays large enough can fix the cardinality problem. However, we need the possibility to inactivate some of the delays to allow a lower number of delays than the number of ANN outputs. This is achieved

by doubling the number of ANN outputs, where half of the outputs represent delays $\hat{\boldsymbol{\tau}}$, and the second half represents the activation probability or weight $\hat{\boldsymbol{w}}$ of each delay.

The training of an ANN architecture requires a loss function that needs to be minimized. The minimization of the loss function is usually implemented using an effective backpropagation algorithm [29]. We propose and evaluate three possible solutions for comparing the predicted delays with the training dataset to obtain a loss function allowing reliable training of the proposed ANN.

The first and most straightforward solution is to sort the dataset delays and let the ANN learn to output the delays sorted. Then the delay MSE loss function is

$$L_2 = (\boldsymbol{\tau} - \hat{\boldsymbol{\tau}})^T ((\boldsymbol{\tau} - \hat{\boldsymbol{\tau}}) \odot \boldsymbol{w}) + L_{bce}(\boldsymbol{w}, \hat{\boldsymbol{w}}), \quad (21)$$

where the weights are trained using the binary cross-entropy function [30] defined by

$$L_{bce}(\boldsymbol{w}, \hat{\boldsymbol{w}}) = -\frac{1}{|\boldsymbol{w}|} \left(\boldsymbol{w}^T \log(\hat{\boldsymbol{w}}) + (1 - \boldsymbol{w})^T \log(1 - \hat{\boldsymbol{w}}) \right). \quad (22)$$

Notice that the true weight vector \boldsymbol{w} is used to omit the influence of nonactive delays in the first member on the right hand side (r.h.s.) of (21). Thanks to this, the delay estimation is not dependent on the precision of weight estimation. Weight estimation is based on the second part of the loss function, the second member on the r.h.s. of (21). We refer to this loss function as the ordered loss (*Ord*).

The following two proposed loss functions aim to remove the task of ordering the delays from the ANN and encode the unordered nature of delays directly to the loss function. The first proposal implements a suboptimal association between the members of $\boldsymbol{\tau}$ and $\hat{\boldsymbol{\tau}}$. It takes the first delay τ_1 from $\boldsymbol{\tau}$ and finds the best corresponding delay in $\hat{\boldsymbol{\tau}}$ as

$$j = \arg \min_{\ell} (\tau_1 - \hat{\tau}_{\ell})^2, \quad (23)$$

then $\hat{\tau}_j$ is removed from $\hat{\boldsymbol{\tau}}$, and the process is repeated for the following members of $\boldsymbol{\tau}$. This algorithm does not provide optimal associations between $\boldsymbol{\tau}$ and $\hat{\boldsymbol{\tau}}$, but it scales linearly with the model order $|\boldsymbol{\tau}| = nMPC$. Note that the same indexing is used to evaluate weight loss. The pseudocode is in Algorithm 1. However, the actual function can be implemented to apply over the whole batch, and $(\tau_{\ell} - \hat{\tau}_i)^2$ is evaluated only once. The provided pseudocode aims for readability and not optimality.

The last proposal is to perform an optimal assignment of estimated delays $\hat{\boldsymbol{\tau}}$ with the ground truth delays $\boldsymbol{\tau}$. Finding the optimal ordering of $\boldsymbol{\tau}$, so that (21) is minimized is a well-known assignment problem with exponential complexity if a naive approach is selected. We use the assignment algorithm known as the Hungarian algorithm or the Munkres assignment algorithm, which complexity is polynomial [31]. Concretely, we use a Python implementation that runs in $O(n^3)$ time.¹ The text refers to the Munkres assignment algorithm-based loss function, ordering $\boldsymbol{\tau}$ as the Munkres loss (*Munk*).

¹available: <http://software.clapper.org/munkres/>

Comparing the proposed ANN architecture performance with the CRLB, we found that the ANN struggles to approach this theoretical limit even in a simple single MPC scenario with a constant SNR in the whole dataset. This leads us to propose a novel ANN loss function improving the learning convergence in the vicinity of the CRLB. The proposed loss function is inspired by the ML solution to channel parameter estimation in (12). This correlation loss is defined as

$$L_C = -\mathbf{x}^H \mathbf{B}(\hat{\tau}) \hat{\alpha}, \quad (24)$$

where the amplitudes are estimated using BLUE

$$\hat{\alpha} = \left(\mathbf{B}^H(\hat{\tau}) \mathbf{B}(\hat{\tau}) \right)^{-1} \mathbf{B}^H(\hat{\tau}) \mathbf{x}. \quad (25)$$

There are two application notes regarding the proposed correlation loss function. In some cases, the Tensorflow cannot handle complex numbers while using GPU. Hence, all complex-valued variables have to be implemented using real numbers. E.g., the Moore-Penrose pseudo-inverse in (25) is implemented using two real-valued Moore-Penrose pseudo-inverses as [32]. Also, to improve the precision, we use the Numpy implementation of the Singular Value Decomposition (SVD) in a double-precision floating-point rather than the Tensorflow implementation. Finally, the standard Moore-Penrose pseudo-inverse uses a cutoff parameter to omit weak eigenvalues obtained by the SVD. However, for correct gradient backpropagation, the weak eigenvalues are substituted by Inf rather than zero.

Interestingly, the correlation loss can be viewed as unsupervised learning since we do not need the ground truth parameters to evaluate it. On the other hand, it cannot be used alone for the training. If the error between the true and estimated delay is larger than the 0.5 sampling period, the radio channel observation \mathbf{x} is weakly correlated with the estimated replica. Moreover, the first derivative of the correlation function is not guaranteed to point towards the true delay for delay error larger than the 0.5 sampling period of the radio channel observation. Hence, the ANN cannot learn to estimate delay using only (24).

The use of correlation loss for ANN-based channel parameter estimation is a promising novel idea. We train the proposed ANN architecture using a total loss

$$L = L_2 + \text{loss}W \cdot L_C. \quad (26)$$

Combining of one of the proposed L_2 loss functions with ordering of the $\hat{\tau}$ and correlation loss shows superior performance that was impossible to achieve using only (21).

V. ACHIEVED RESULTS

The main results are comparing the ANN-based channel parameter estimation method with the theoretical limits given by CRLB and with the ML estimation method SAGE [19] and the Superfast Line Spectral Estimation (SLSE) [20]. The comparison is made in three stages with an increasing level of generalization.

Algorithm 1: Adjacent loss (*Adj*)

Input: $\tau, w, \hat{\tau}, \hat{w}$

Output: L

- 1 $L_\tau = 0, L_w = 0$
 - 2 **for** $i = 1 : nMPC$
 - 3 $j = \arg \min_\ell (\tau_i - \hat{\tau}_\ell)^2$
 - 4 $L_\tau = L_\tau + w_i (\tau_i - \hat{\tau}_j)^2$
 - 5 $L_w = L_w + L_{\text{bce}}(w_i, \hat{w}_j)$
 - 6 $\hat{\tau} = \hat{\tau}.\text{remove}(j)$ // remove j-th element
 - 7 $\hat{w} = \hat{w}.\text{remove}(j)$
 - 8 $L = L_\tau + L_w / nMPC$
-

TABLE I

LIST OF ALL CONFIGURABLE PARAMETERS OF THE ANN TRAINING SCHEDULE

<i>nBatch</i>	batch size
<i>stdIn</i>	STD of input regularization noise
<i>lrInit</i>	initial learning rate
<i>lrStep</i>	learning rate decay step
<i>lrRate</i>	learning rate decay rate
<i>nEpochs</i>	number of training epochs
<i>lossFnc</i>	training loss functions
<i>lossW</i>	weight of correlation loss function
<i>valSplit</i>	fraction of dataset used as validation dataset

In the first stage, the performance is studied while estimating the delay of a single MPC in noise. In the second stage, the complexity is increased by adding an additional MPC to the CIR. The final stage is adding the last generalization step, the unknown number of MPCs in noise. According to Section III, all MPCs are generated with random delay, magnitude, and phase.

All presented results are obtained using the same feed-forward ANN architecture. The details of ANN architecture configuration are summarized in Fig. 1.

The custom normalization layer normalizes the input impulse response, including the regularization noise, so the energy equals one. All convolutional layers and the following dense layer use the ReLU activation function. The last dense layer uses ReLU activation for delay outputs and sigmoid activation for weight outputs. The weight outputs are active only for training with an unknown number of MPCs.

Also, many hyperparameters are fixed for studied cases. In this study, we focus on the estimation of well-separated MPCs. For this reason, the minimum separation between two delays is two samples of CIR. The study of performance for closely

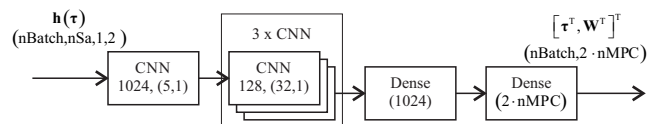


Fig. 1. The core of the ANN architecture including dimensions, number of filters, kernel sizes, and number of units of individual layers. The normalized noisy CIR is separated into the real and imaginary parts along the last axis. Then, it is inputted to the first convolutional layer as $\mathbf{h}(\tau)$.

TABLE II
LIST OF FIXED PARAMETERS

nSa	64	$nBatch$	2^7
$nData$	$2^{17}, 2^{18}$	$lrInit$	10^{-4}
$SNRmin$	15 dB	$lrStep$	$50 \frac{nData}{nBatch}$
$SNRmax$	25 dB	$lrRate$	0.9
$minSep$	2	$nEpochs$	1000
$stdIn$	$2\pi/\sqrt{2} \cdot nSa$	$lossW$	0.1

separated and hence highly correlated MPCs is a target for future work. The STD of regularization noise added during training is adding the required amount of noise to obtain the desired MPC SNR. The decay step is set as 50 times the ratio between the training data size and the batch size. As a result, the decay rate defines how much the learning rate decreases every 50 epochs. The list of fixed parameters is summarized in Table II.

After the training, the ANN model is saved and used to evaluate a newly generated dataset. The same dataset is then processed using the SAGE and SLSE algorithms. Note that the SAGE algorithm cannot estimate the true number of MPCs and is therefore provided with the number of MPCs. We do not use model order estimation methods like Akaike's Information Criterion (AIC) or Minimum Description Length (MDL) to support the SAGE algorithm [19]. The current state-of-the-art approach to channel parameter estimation, the SLSE, is a deterministic approximation approach, which provides Maximum A Posteriori (MAP) estimates of all channel parameters, including the model order [20]. When multiple MPCs are assumed, the Munkres algorithm [31] is used to associate the predicted delays with its true values.

The precision evaluation is done using the Probability Density Function (PDF) of error between the estimated delays $\hat{\tau}$ and the true delays τ . The fair comparison for different SNR values is achieved by normalizing error by the corresponding $\sqrt{CRB_{\hat{\tau}}}$ obtained from (7).

A. Performance for single MPC

First, the training of the ANN using a dataset with random MPC SNR is performed. Then, 41 datasets with SNR between 15 dB and 25 dB are generated to evaluate the performance. Each dataset has fixed SNR and consists of 2^{17} impulse responses with one random MPC delay.

The comparison of the MSE for different values of SNR is shown in Fig. 2. The figure shows the MSE as a function of the MPC SNR. The black line shows the theoretical limit given by the CRLB. The blue $\hat{\tau}_{Corr}$ line shows the MSE achieved by the ANN architecture trained with the combination of the correlation loss function (24) and the delay MSE loss function (20) combined according to (26). The green $\hat{\tau}_{MSE}$ line shows the MSE achieved by the ANN architecture trained with the delay MSE loss (20). Finally, the red $\hat{\tau}_{SLSE}$ and yellow $\hat{\tau}_{SAGE}$ lines show the MSE achieved by the SLSE and SAGE algorithms, respectively.

The SAGE and SLSE closely follow the theoretical limit given by CRLB in the whole SNR span. The ANN archi-

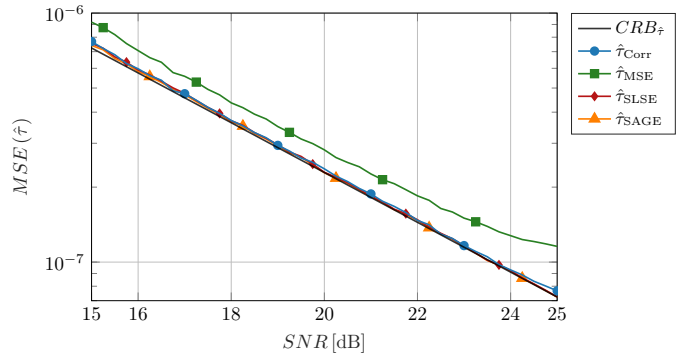


Fig. 2. Performance comparison of the proposed ANN-based method with the CRLB theoretical limit, the SAGE, and SLSE algorithms for a single MPC in noise. The proposed ANN-based method using a combination of the delay MSE and the correlation loss functions $\hat{\tau}_{Corr}$ is further compared with the ANN using only the delay MSE loss function $\hat{\tau}_{MSE}$ for training.

ecture trained using only the delay MSE loss function does not approach the CRLB. Moreover, with increasing SNR, the performance gap between the CRLB and ANN is even increasing since the ANN slowly learns to estimate the delay with the high precision achievable for the higher MPC SNR. This drawback of the ANN-based approach is successfully mitigated using the newly proposed correlation loss function (24) together with the delay MSE loss function (20). When using the weighted combination of MSE and correlation losses, according to (26), the delay estimation of one MPC in noise is approaching the CRLB theoretical limit and is comparable with both SAGE and SLSE algorithms.

If the $\sqrt{CRB_{\hat{\tau}}}$ normalizes the delay estimation error, it is possible to compare the error Cumulative Distribution Functions (CDFs) of all approaches independently of the SNR value. After normalizing the delay estimation error by the corresponding $\sqrt{CRB_{\hat{\tau}}}$, the theoretically optimal error CDF becomes the CDF of zero mean, unit variance normal distribution $\mathcal{N}(\mu = 0, \sigma^2 = 1)$. Fig. 3 shows the error CDF of the delay estimation approaches and compares them with the theoretically optimal error CDF. Note that Fig. 3 uses the error rather than the absolute value of the error because the deviation from the zero mean value would be better visible in the CDF when the error is not in absolute value.

The SAGE and SLSE error closely follows the theoretical error CDF given by the $\mathcal{N}(\mu = 0, \sigma^2 = 1)$. The ANN-based estimator error also follows a normal distribution. However, it can be noticed that the normalized error variance of the ANN-based approach using the delay MSE loss function is larger than one. When the newly proposed correlation loss is utilized during the ANN training, the variance decreases and is similar to the optimal benchmark methods SAGE and SLSE.

Since the error CDF of all delay estimation methods closely follow the normal distribution, we can compare the performance quantitatively by comparing the first two moments of the error PDF normalized by the $\sqrt{CRB_{\hat{\tau}}}$. The mean and variance of all methods are included in Table III. The experiments show that the ANN-based approach tends to

TABLE III

COMPARISON OF MEAN AND VARIANCE OF THE PROPOSED ANN-BASED ESTIMATOR WITH THE THEORETICAL CRLB AND THE BENCHMARK APPROACHES SAGE AND SLSE FOR A SINGLE MPC IN NOISE. THE OVERALL MSE ESTIMATION IS OBTAINED AS $\mu^2 + \sigma^2$.

Method	$\mu/\sqrt{CRB_{\hat{\tau}}}$	$\sigma^2/CRB_{\hat{\tau}}$	$MSE/CRB_{\hat{\tau}}$
$\hat{\tau}_{Ideal}$	0	1	1
$\hat{\tau}_{Corr}$	+1.5e-2	1.030	1.031
$\hat{\tau}_{MSE}$	-3.1e-2	1.281	1.281
$\hat{\tau}_{SLSE}$	+4.8e-4	1.022	1.022
$\hat{\tau}_{SAGE}$	+4.0e-4	1.014	1.014

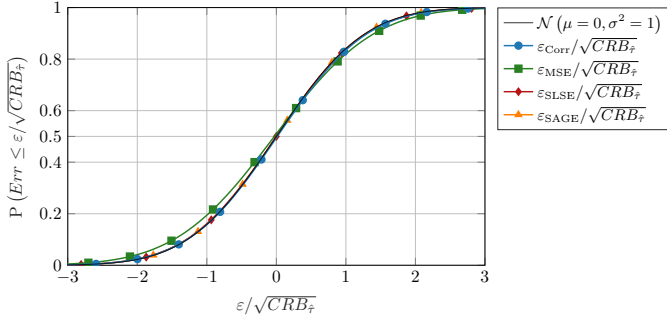


Fig. 3. Comparison of delay estimation error CDFs of all tested approaches with the theoretical error CDF for a single MPC in noise. The $\sqrt{CRB_{\hat{\tau}}}$ corresponding to the actual SNR normalizes the delay error obtained from all 41 datasets with different SNR values.

have a higher mean delay estimation error than the classical approaches.

Usually, fine training with a reduced learning rate can decrease the mean of delay estimation error. During the fine training, the sign of the mean changes. Hence, the rather high mean error might be linked with the Adam optimizer overshoot issue. Using the AMSGrad [23] optimizer to improve the convergence of the Adam Optimizer did not improve the performance.

B. Performance for two MPCs

First, the training of the ANN using a dataset with random MPC SNR is performed. Then, 41 datasets with SNR between 15 dB and 25 dB are generated to evaluate the performance. Each dataset has a fixed SNR of both MPCs and consists of 2^{15} CIRs with two random MPC delays. The minimal delay separation is set to $\tau_{\Delta_{MIN}} = 2/nSa$. The delay separation assures that the MPCs can always be distinguished as separate MPCs.

The datasets with constant SNR show the performance for the given SNR, but it does not capture the proper performance. In reality, each MPC in a CIR can have different SNR. Therefore, we create an additional dataset where each MPC has a random SNR. The SNR of each MPC is sampled uniformly in a range between 15 dB and 25 dB. The random SNR dataset serves to evaluate the overall performance of the trained model. The random SNR dataset consists of 2^{18} CIRs.

The comparison of the MSE for different values of SNR is shown in Fig. 4. The figure shows that the SAGE and SLSE closely follow the theoretical limit given by the CRLB in the

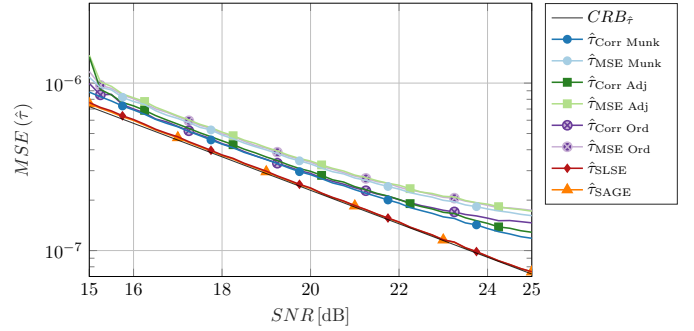


Fig. 4. Performance comparison of the proposed ANN-based method with the CRLB theoretical limit, the SAGE, and SLSE algorithms for two MPCs in noise. The proposed ANN-based method using a combination of the MSE and the correlation loss functions $\hat{\tau}_{Corr}$ is further compared with the ANN using only the delay MSE loss function $\hat{\tau}_{MSE}$ during the training. Three different methods of data association during the training are also compared.

whole SNR span. The ANN architecture trained using only the delay MSE loss function (20) have again worse performance than the architecture trained using the combination of the delay MSE and the correlation loss functions in (21).

Three different methods of data association during the training are compared. When using the ordered loss marked as *Ord*, the ANN must learn to order the outputted delays in ascending order.

The suboptimal data association method marked as *Adj* is associating each true delay value with the best corresponding prediction one at the time. The *Adj* data association method performs better than the *Ord* method but needs a longer training time.

The last data association method (*Munk*) uses an optimal Munkres algorithm to associate the predicted delays with the true values during the training. The ANN trained using the optimal data association provides the best performance for the cost of the slowest training.

Even after doubling the training dataset size, none of the trained models follow the CRLB closely. At this point, it is not obvious if further increasing of the dataset size can solve this issue or fine-tuning of the model hyperparameters is required.

The random SNR dataset evaluates the performance of the trained ANN models in a realistic scenario where each MPC has a random SNR value. The delay error of each MPC is normalized using the corresponding $\sqrt{CRB_{\hat{\tau}}}$. Fig. 5 shows the normalized error CDF of the delay estimation approaches and compares them with the theoretically optimal error CDF.

The SAGE and SLSE error closely follows the theoretical error CDF given by the $\mathcal{N}(\mu = 0, \sigma^2 = 1)$. The ANN-based estimator error also follows a normal distribution. However, it can be noticed in Fig. 5 that the normalized error variance of the ANN-based approaches is larger than one.

The quantitative performance comparison is in Table IV. Similarly to the single MPC experiment, the ANN-based approach tends to have a higher mean delay estimation error than the classical approaches. Interestingly, the performance

TABLE IV
COMPARISON OF MEAN AND VARIANCE OF THE PROPOSED ANN-BASED ESTIMATOR WITH THE THEORETICAL CRLB AND THE BENCHMARK APPROACHES SAGE AND SLSE FOR TWO MPCs IN NOISE. THE OVERALL MSE ESTIMATION IS OBTAINED AS $\mu^2 + \sigma^2$.

Method	$\mu/\sqrt{CRB_{\hat{\tau}}}$	$\sigma^2/CRB_{\hat{\tau}}$	MSE/ $CRB_{\hat{\tau}}$
$\hat{\tau}_{ideal}$	0	1	1
$\hat{\tau}_{Corr Munk}$	-5.0e-2	1.319	1.321
$\hat{\tau}_{MSE Munk}$	-3.3e-2	1.598	1.599
$\hat{\tau}_{Corr Adj}$	+1.7e-2	1.408	1.408
$\hat{\tau}_{MSE Adj}$	-9.5e-2	1.651	1.660
$\hat{\tau}_{Corr Ord}$	-6.2e-2	1.377	1.381
$\hat{\tau}_{MSE Ord}$	+9.2e-3	1.654	1.654
$\hat{\tau}_{SLSE}$	-1.0e-4	1.030	1.030
$\hat{\tau}_{SAGE}$	+3.0e-5	1.023	1.023

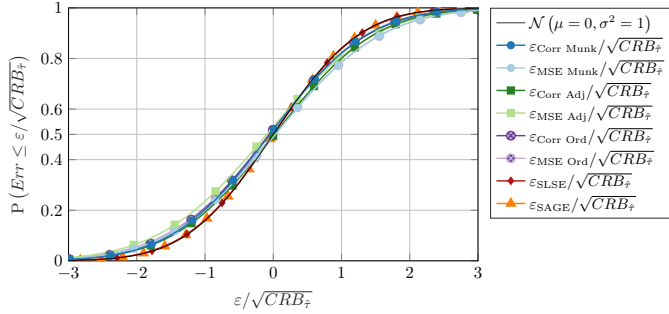


Fig. 5. Comparison of delay estimation error CDFs of all tested approaches with the theoretical error CDF for two MPCs in noise. The $\sqrt{CRB_{\hat{\tau}}}$ corresponding to the actual SNR normalizes the delay error obtained from the random SNR dataset.

is similar for all tested ANN-based methods. It seems that the data association provides only a small improvement for the two MPCs scenario. However, the combination of the delay MSE loss with the proposed correlation loss always improves the performance. Again, using the AMSGrad [23] option did not improve the convergence of the Adam Optimizer.

C. Performance for an unknown number of MPCs

Similar to the previous approach, the ANN training is done using a dataset with random MPC SNR. However, the number of MPCs is unknown and random. In general, the number of MPCs can be arbitrary, but the number of MPCs is randomly chosen between one and two for this first study.

The evaluation uses 41 datasets with SNR between 15 dB and 25 dB. Each dataset has a fixed SNR of all MPCs present in the CIR and consists of 2^{15} CIRs. The number of MPCs is also randomly set to one or two MPCs. The minimal delay separation is set to $\tau_{\Delta_{MIN}} = 2/nSa$.

As in the scenario with a fixed number of MPCs, we create an additional dataset where each MPC has a random SNR. The SNR of each MPC is sampled uniformly in a range between 15 dB and 25 dB. The random SNR dataset consists of 2^{18} CIRs.

The comparison of the MSE for different values of SNR is shown in Fig. 6. The figure shows that the SAGE and SLSE closely follow the theoretical limit given by the CRLB in the whole SNR span. The combination of the MSE and

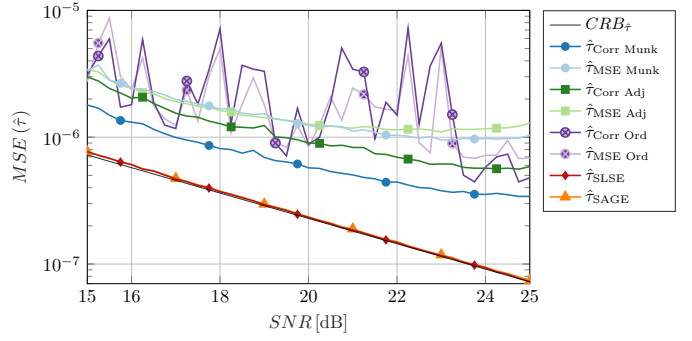


Fig. 6. Performance comparison of the proposed ANN-based method with the CRLB theoretical limit, the SAGE, and SLSE algorithms for an unknown number of MPCs in noise. The proposed ANN-based method using a combination of the MSE and the correlation loss functions $\hat{\tau}_{Corr}$ is further compared with the ANN using only the delay MSE loss function $\hat{\tau}_{MSE}$ during the training. Three different methods of data association during the training are also compared.

correlation loss functions outperforms the ANN trained using only the delay MSE loss function. The *Adj* data association method performs better than the *Ord* method but needs a longer training time. Moreover, the *Ord* method has varying performance for different SNR values. This inconsistency seems to be mitigated when the ANN does not need to learn to order the MPCs on its own and uses one of the presented association methods. Since this problem was not observed for the fixed number of MPC scenarios, it seems that the ANN cannot associate the MPCs and estimate the model order jointly. The best performance is again obtained using the optimal *Munk* association technique.

The presented performance was obtained using the training dataset with 2^{18} CIRs. The obtained performance is worse compared to the known model order setting, but it still provides interesting precision. Even if the MSE is approximately 4.7 times larger than the CRLB for the 25 dB SNR, the delay error STD is approximately 26.8 times smaller than the spacing between CIR samples. This clearly shows that the ANN-based multipath radio channel estimator provides a super-resolution performance.

Fig. 7 shows the normalized error CDF of the delay estimation methods and compares them with the theoretically optimal error CDF. Again, the random SNR dataset is used to show a realistic performance in a case where each MPC in a CIR has different SNR. The delay error of each MPC estimate is normalized using the corresponding $\sqrt{CRB_{\hat{\tau}}}$.

The SAGE and SLSE error closely follows the theoretical error CDF given by the $\mathcal{N}(\mu = 0, \sigma^2 = 1)$. The ANN-based estimator error also follows a normal distribution. Again, it has larger variance than one.

The quantitative performance comparison is in Table IV. It is interesting that the individual approaches' performance differences are more severe than for the fixed number of MPCs scenarios. The data association improves the performance significantly when the number of MPCs has to be estimated by the ANN. Combining of the delay MSE loss with the proposed

TABLE V

COMPARISON OF MEAN AND VARIANCE OF THE PROPOSED ANN-BASED ESTIMATOR WITH THE THEORETICAL CRLB AND THE BENCHMARK APPROACHES SAGE AND SLSE FOR AN UNKNOWN NUMBER OF MPCs IN NOISE. THE OVERALL MSE ESTIMATION IS OBTAINED AS $\mu^2 + \sigma^2$.

Method	$\mu/\sqrt{CRB_{\hat{\tau}}}$	$\sigma^2/CRB_{\hat{\tau}}$	MSE/ $CRB_{\hat{\tau}}$
$\hat{\tau}_{Ideal}$	0	1	1
$\hat{\tau}_{Corr Munk}$	-9.1e-3	2.924	2.924
$\hat{\tau}_{MSE Munk}$	+1.1e-1	7.005	7.018
$\hat{\tau}_{Corr Adj}$	+1.9e-1	4.656	4.691
$\hat{\tau}_{MSE Adj}$	+4.6e-2	7.463	7.466
$\hat{\tau}_{Corr Ord}$	+6.0e-3	6.677	6.677
$\hat{\tau}_{MSE Ord}$	-3.2e-1	5.667	5.770
$\hat{\tau}_{SLSE}$	+9.9e-4	1.024	1.024
$\hat{\tau}_{SAGE}$	-5.1e-4	1.034	1.035

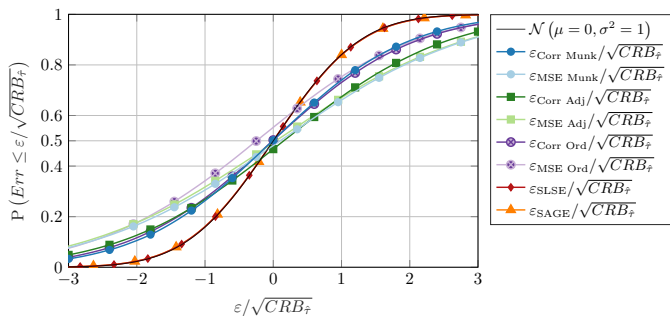


Fig. 7. Comparison of delay estimation error CDFs of all tested approaches with the theoretical error CDF for an unknown number of MPCs in noise. The $\sqrt{CRB_{\hat{\tau}}}$ corresponding to the actual SNR normalizes the delay error obtained from the random SNR dataset.

correlation loss function improves the performance significantly when the optimal Munkres data association method is used.

D. Comparison of the calculation complexity

During the ANN training stage, the algorithm has to evaluate the loss and gradients to update all the trained model weights. For this reason, the training step is slower than the evaluation using the pre-trained model. Hence, we compare the speed of the ANN approach with the benchmark methods only for the model evaluation. Additionally, we compare the relative speed of the training for the presented training approaches with different loss functions. Since all the trained ANN models consist of the same number of trained parameters, the comparison shows the complexity of the different loss functions.

During the training stage, the Munkres algorithm-based data association perform one training epoch with two MPCs approximately 31% slower than the *Ord* approach, which does not performs any data association during the training. The *Adj* method is only 10% slower than the *Ord* training method.

However, if we use the additional correlation loss function, the difference changes. With the combined loss functions, the Munkres data association is approximately 17% slower than the *Ord* approach, and the *Adj* method is 13% slower than the *Ord* training method.

TABLE VI

THE COMPARISON OF THE TRAINING EPOCH DURATION OF DIFFERENT ANN TRAINING METHODS IN SECONDS.

	Ord	Adj	Munk
MSE	104	115	150
MSE and Corr	144	166	174

TABLE VII

COMPARISON OF THE EVALUATION TIME OF THE ANN-BASED APPROACH WITH THE BENCHMARK METHODS

	SLSE	SAGE	MSE	MSE and Corr
CPU $nBatch = 1$	15.3 ms	2.02 ms	4.86 ms	5.02 ms
GPU $nBatch = 1$			4.52 ms	6.52 ms
GPU $nBatch = 2^7$			85.2 μs	106 μs
GPU $nBatch = 2^8$			69.2 μs	85.2 μs

The training with the additional correlation loss is slower, but it provides a better performance than without it. Hence, the training with the proposed correlation loss is preferred.

The duration of a training epoch with the maximal model order of two MPCs using delay MSE loss and the combination of the delay MSE and correlation loss with batch size equal to 2^7 and dataset size of 2^{18} is compared in Table VI.

The increase of the training complexity for a low number of MPCs is almost negligible. However, it might be preferred for a large number of MPCs to use the *Adj* association method, which scales linearly with the number of MPCs.

Comparison of the evaluation time is difficult since the benchmark algorithms are sequential optimization methods utilizing CPU while the ANN approach leverages parallel computing using GPU. The strength of the ANN comes from the possibility to evaluate multiple CIRs in parallel while evaluating a single measurement at the time can be slow.

For this reason, we show the evaluation speed for the benchmark algorithms and the trained ANNs. We use different batch sizes for GPU-based evaluation and a CPU to evaluate the ANN. The average duration of processing one CIR is in Table VII. Using the combination of the MSE and correlation loss is slightly slower, which should not be since the loss function calculation is performed only during the training. The actual implementation of the method can cause this discrepancy. However, the ANN output ordering type does not affect the evaluation speed.

VI. CONCLUSION

This paper proposes an Artificial Neural Network (ANN) architecture and a training schedule learning to estimate the multipath radio channel parameters from a noisy Channel Impulse Response (CIR). The presented results show that the proposed ANN-based multipath radio channel estimator can approach the theoretical limit Cramer-Rao Lower Bound (CRLB) for a single Multipath Component (MPC) scenario. Note that the CRLB gives the Mean-Square Error (MSE) of a theoretically optimal estimator. When the multipath radio channel model order is known and fixed to two MPCs, the MSE is approximately 1.3 times larger than the CRLB.

Moreover, the proposed method can jointly estimate the multipath radio channel model order and delay parameters with a super-resolution performance. The ANN combining the newly proposed correlation loss function with the Munkres ordering of the MPCs learns to estimate the model order and delays with the MSE three times larger than the CRLB. The ANN overfitting problem is successfully mitigated by using a noise regularization-based approach. The calculation complexity is comparable to the classical optimization approaches, Space-Alternating Generalized Expectation-maximization (SAGE) and Superfast Line Spectral Estimation (SLSE). However, when parallel computing using Graphics Processing Unit (GPU) is used, the evaluation is approximately 24 and 180 times faster than the SAGE and SLSE, respectively. The possibility of parallelization is beneficial for a use case where multiple different radio channel measurements have to be processed simultaneously, e.g., for multipath assisted positioning where the Maximum Likelihood (ML)-based processing is too slow for real-time deployment. The achieved results extend the current state-of-the-art of the machine learning-based line spectral estimation approach and show a promising direction for further research. Further research could explore the performance of the parameter estimation when Dense Multipath Components (DMCs) are present in the radio channel or use a Recurrent Neural Network (RNN) to track the MPCs over time. However, how the learning process scales for the increasing number of MPCs needs to be studied first to show if the ANN-based radio channel estimation method can be faster than the classical approaches in a general setting while approaching CRLB.

REFERENCES

- [1] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The Bulletin of Mathematical Biophysics*, vol. 5, no. 4, pp. 115–133, 12 1943.
- [2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 5 2017.
- [3] A. Sanchez-Gonzalez, J. Godwin, T. Pfaff, R. Ying, J. Leskovec, and P. W. Battaglia, "Learning to simulate complex physics with graph networks," 2020.
- [4] T. Karras, M. Aittala, S. Laine, E. Härkönen, J. Hellsten, J. Lehtinen, and T. Aila, "Alias-free generative adversarial networks," 2021.
- [5] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, 1 2016.
- [6] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev, J. Oh, D. Horgan, M. Kroiss, I. Danihelka, A. Huang, L. Sifre, T. Cai, J. P. Agapiou, M. Jaderberg, A. S. Vezhnevets, R. Leblond, T. Pohlen, V. Dalibard, D. Budden, Y. Sulsky, J. Molloy, T. L. Paine, C. Gulcehre, Z. Wang, T. Pfaff, Y. Wu, R. Ring, D. Yogatama, D. Wünsch, K. McKinney, O. Smith, T. Schaul, T. Lillicrap, K. Kavukcuoglu, D. Hassabis, C. Apps, and D. Silver, "Grandmaster level in StarCraft II using multi-agent reinforcement learning," *Nature*, vol. 575, no. 7782, pp. 350–354, 10 2019.
- [7] J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Židek, A. Potapenko, A. Bridgland, C. Meyer, S. A. A. Kohl, A. J. Ballard, A. Cowie, B. Romera-Paredes, S. Nikolov, R. Jain, J. Adler, T. Back, S. Petersen, D. Reiman, E. Clancy, M. Zielinski, M. Steinegger, M. Pacholska, T. Berghammer, S. Bodenstein, D. Silver, O. Vinyals, A. W. Senior, K. Kavukcuoglu, P. Kohli, and D. Hassabis, "Highly accurate protein structure prediction with AlphaFold," *Nature*, 7 2021.
- [8] S. Chakrabarty and E. A. P. Habets, "Broadband doa estimation using convolutional neural networks trained with noise signals," in *2017 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. IEEE, 10 2017.
- [9] S. Adavanne, A. Politis, and T. Virtanen, "Direction of arrival estimation for multiple sound sources using convolutional recurrent neural network," in *2018 26th European Signal Processing Conference (EU-SIPCO)*. IEEE, 9 2018.
- [10] G. Izacard, B. Bernstein, and C. Fernandez-Granda, "A learning-based framework for line-spectra super-resolution," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 5 2019.
- [11] G. Izacard, S. Mohan, and C. Fernandez-Granda, "Data-driven estimation of sinusoid frequencies," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32. Curran Associates, Inc., 2019.
- [12] Y. Jiang, H. Li, and M. Rangaswamy, "Deep learning denoising based line spectral estimation," *IEEE Signal Processing Letters*, vol. 26, no. 11, pp. 1573–1577, 11 2019.
- [13] Y.-S. Hsiao, M. Yang, and H.-S. Kim, "Super-resolution time-of-arrival estimation using neural networks," in *2020 28th European Signal Processing Conference (EUSIPCO)*. IEEE, 1 2021.
- [14] P. Bello, "Characterization of randomly time-variant linear channels," *IEEE Trans. Commun. Syst.*, vol. 11, no. 4, pp. 360–393, 12 1963.
- [15] B. Boashash, *Time-Frequency Signal Analysis and Processing*, 2nd ed. London, United Kingdom: Elsevier, 2015.
- [16] J. Francos and B. Friedlander, "Bounds for estimation of complex exponentials in unknown colored noise," *IEEE Transactions on Signal Processing*, vol. 43, no. 9, pp. 2176–2185, 1995.
- [17] P. Stoica, A. Jakobsson, and J. Li, "Cisoid parameter estimation in the colored noise case: asymptotic Cramer-Rao bound, maximum likelihood, and nonlinear least-squares," *IEEE Transactions on Signal Processing*, vol. 45, no. 8, pp. 2048–2059, 1997.
- [18] A. Richter, "Estimation of radio channel parameters: Models and algorithms," Ph.D. dissertation, Technischen Universität Ilmenau, Ilmenau, Germany, 2 2005.
- [19] B. Fleury, M. Tschudin, R. Heddergott, D. Dahlhaus, and K. I. Pedersen, "Channel parameter estimation in mobile radio environments using the SAGE algorithm," *IEEE Journal on Selected Areas in Communications*, vol. 17, no. 3, pp. 434–450, 3 1999.
- [20] T. L. Hansen, B. H. Fleury, and B. D. Rao, "Superfast line spectral estimation," *IEEE Transactions on Signal Processing*, vol. 66, no. 10, pp. 2511–2526, 5 2018.
- [21] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Berlin, Heidelberg: Springer-Verlag, 2006.
- [22] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2017.
- [23] S. J. Reddi, S. Kale, and S. Kumar, "On the convergence of adam and beyond," 2019.
- [24] J. Martens, "New insights and perspectives on the natural gradient method," 2020.
- [25] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Watrenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org.
- [26] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," *arXiv preprint arXiv:1408.5093*, 2014.
- [27] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer,

- F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 8024–8035.
- [28] J. Rothfuss, F. Ferreira, S. Walther, and M. Ulrich, “Conditional density estimation with neural networks: Best practices and benchmarks,” 2019.
- [29] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning representations by back-propagating errors,” *Nature*, vol. 323, no. 6088, pp. 533–536, 10 1986.
- [30] D. R. Cox, “The regression analysis of binary sequences,” *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 20, no. 2, pp. 215–242, 1958.
- [31] H. W. Kuhn, “The hungarian method for the assignment problem,” *Naval Research Logistics Quarterly*, vol. 2, no. 1-2, pp. 83–97, 3 1955.
- [32] A. Falkenberg, “Method to calculate the inverse of a complex matrix using real matrix inversion,” Tech. Rep., 2007.