Deep Neural Network Approach to GNSS Signal Acquisition

Parisa Borhani-Darian and Pau Closas
Department of Electrical and Computer Engineering
Northeastern University
Boston, MA

***Abstract*—This paper investigates the use of data-driven models, popular in the machine learning literature, as an alternative to well-engineered signal processing blocks used in state-of-the-art GNSS receivers. Acknowledging that the latter are optimally designed and extensively tested, it is also agreed that when the nominal models do not hold the performance of the receiver might degrade. Particularly, we investigate the use of data-driven models in the signal acquisition stage of the receiver by addressing a classification problem from Cross Ambiguity Function (CAF) delay/Doppler maps. A discussion on the training of such models and future perspectives is provided. The detection results in nominal situations are then compared to the theoretical bound in the receiver operating characteristic (ROC) plots.**

***Index Terms*—Satellite-based navigation, machine learning, neural networks, signal detection.**

I. Introduction

Signal acquisition is the first action that is performed by a Global Navigation Satellite System (GNSS) receiver. The outcome of this process decides either the signal from a particular satellite is present or absent in the received signal, as well as provides a rough estimate of its associated code delay and Doppler frequency if present. All GNSS receivers [1], [2], [3], [4] implement such an acquisition process by evaluating the so-called Cross Ambiguity Function (CAF), usually in a discrete-time domain. The CAF is a two-dimensional function that is related to the correlation between the received signal and local code for every possible delay/Doppler pair and that is used in signal detection and coarse synchronization (a.k.a. acquisition).

It is customary to think of this acquisition process as a signal detection problem where two hypotheses are available: 1) the null hypothesis $\mathcal{H}_0$ is that the signal is not present or not correctly aligned with the local replica, and 2) the alternative hypothesis $\mathcal{H}_1$ is that the signal is present and correctly aligned with the local replica. In each hypothesis, three kinds of probabilities characterize the performance of the acquisition method: detection (the probability of correctly detecting signal/noise when there is signal/noise), false-alarm (the probability of wrongly detecting signal when the satellite is not present), and miss-detection (the probability of mistakenly deciding for the null hypothesis when the signal is present). The two first probabilities are used in order to obtain an important figure of merit in acquisition performance: the Receiver Operating Characteristics (ROC), a plot of the probability of detection as a function of the probability of false alarm [1], [5], [6].

Signal acquisition is based on solid statistical grounds, where the approach of maximizing the CAF (i.e., the correlation between the local replica and the incoming signal) can be seen to be optimal under certain model conditions (e.g. Gaussianity and linearity of the channel). However, experiments show (e.g. [7]) that reality is typically more challenging and that the assumed nominal model does not necessarily hold always. Recent attempts to modify the CAF to make it more robust to non-Gaussian behaviours (such as heavy-tailed noise distributions) showed outstanding performance, particularly in the context of GNSS operation under jamming [8], [9], [10], [11], [12], [13], [14]. Regardless its remarkable performance in outlier-rich data, the aforementioned robust approach does not accommodate for more complex situations such as multi-modal distributions or moderate-to-severe nonlinearities affecting the received signal. This work conjectures that such complex behaviours can be learnt by employing efficient data-driven methods, trained over large datasets. In particular, we propose to use a deep neural network (DNN) to carry out the detection (or classification) process. Prior to targeting those challenging scenarios that turn the nominal model unreliable, this paper focuses on replicating the optimal results (provided by standard model-driven methods under nominal conditions) by employing data-driven models (a.k.a. NNs). This analysis is important in order to establish a set of trained, verifiable, accurate and efficient NN kernels with expected performance guarantees. Additionally, it would validate the validity of the approach, paving the way for more advance NN models that are trained in more challenging scenarios using a combination of synthetic and real training data.

The paper is organized as follows. Section II reviews the GNSS signal model and the standard formulae of signal acquisition. Section III introduces the data-driven approach and connects it to the problem of GNSS signal acquisition. Section IV describes the training methodology, a crucial aspect of data-driven models, and Section V discusses the obtained results. The paper is concluded with final remarks and future directions in Section VI.

II. GNSS Signal Acquisition

A receiver observes signals from M satellites plus noise. After downconversion and sampling (at a rate $f_s = 1/\mathrm{T}_s$) the

978-1-7281-0244-3/20/$31.00 ©2020 IEEE

samples discrete-time signal is:

$$\begin{aligned} y[n] &= \sum_{i=1}^{M} x_{\theta,i}[n] + \eta[n] \\ x_{\theta,i}[n] &= \alpha_i b_i(n\mathrm{T}_s - \tau_i) c_i(n\mathrm{T}_s - \tau_i) \exp(j2\pi f_{d,i} n\mathrm{T}_s + j\phi_i) \end{aligned}$$

with α_i the amplitude of the i-th received signal; $b_i(\cdot)$ the data bits of the i-th navigation message; $c_i(\cdot)$ the spreading code of the i-th satellite; τ_i the time-evolving delay of the i-th satellite; $f_{d,i}$ the Doppler-shift; ϕ_i a carrier-phase term introduced by the channel; and $\eta[n]$ models the random noise at the receiver, typically complex, zero-mean and Gaussian distributed with variance σ^2.

Signal acquisition is one of the first actions a receiver needs to perform, basically deciding either the signal from a particular satellite is present or absent, as well providing a rough estimation of the code delay and Doppler frequency of the received signal in case it is deemed present [1]. Therefore, when searching for the i-th satellite this problem can be formulated as an hypothesis testing problem with two possibilities:

$$\begin{aligned} \mathcal{H}_0 &: i\text{-th satellite is not present} \\ \mathcal{H}_1 &: i\text{-th satellite is present} \end{aligned}$$

Equivalently, the two competing hypothesis are

$$\begin{aligned} \mathcal{H}_0 &: y[n] = \eta[n] \\ \mathcal{H}_1 &: y[n] = x_{\theta,i}[n] + \eta[n] \end{aligned}$$

such that $n = 0, \ldots, N-1$ index the N samples used in acquisition (i.e., coherent integration interval). Since the parameters $\boldsymbol{\theta}_i = (\alpha_i, \phi_i, \tau_i, f_{d,i})^\top$ are not known, the optimal detection framework (in the maximum likelihood (ML) sense) is the Generalized Likelihood Ratio Test (GLRT), which requires ML estimation (MLE) of the vector $\boldsymbol{\theta}_i$. Given a set of N observations, $\mathbf{y} = (y[0], y[1], \ldots, y[N-1])^\top$ the MLE of $\boldsymbol{\theta}_i$ is defined as

$$\hat{\boldsymbol{\theta}}_i = \arg\max_{\boldsymbol{\theta}_i} p(\mathbf{y}|\boldsymbol{\theta}_i) \ , \tag{1}$$

where it is typically assumed that the parameters in $\boldsymbol{\theta}_i$ are piece-wise constant within the N samples of $\mathbf{y}$ and that the codes have ideal cross-correlation properties so they can be processed independently at the receiver.

It can be seen that the GLRT results in the maximization of the correlation between the received signal and a locally generated code. This correlation operation is encoded in the so-called Cross Ambiguity Function (CAF), which is nothing but the correlation between $y[n]$ and the spreading code of the i-th satellite, at a given delay/Doppler pair (in discrete-time):

$$\mathcal{C}_i(\tau, f_d) = \frac{1}{N} \sum_{n=0}^{N-1} y[n] \underbrace{c_i(n\mathrm{T}_s - \tau) \exp\{-j2\pi f_{d,i} n\mathrm{T}_s\}}_{\text{Local replica}} \ , \tag{2}$$

which can be expressed more compactly in vector notation after gathering N samples from the samples and the local code as $\mathbf{y}, \mathbf{c}_i \in \mathbb{C}^{1\times N}$ as

$$\mathcal{C}_i(\tau, f_d) = \frac{\mathbf{y}\mathbf{c}_i^H}{N} \ . \tag{3}$$

The CAF is crucial in the acquisition (and tracking) of the satellites' signals. The MLE of $\boldsymbol{\theta}_i$ can be expressed in terms of it as

$$(\hat{\tau}_i, \hat{f}_{d,i}) = \arg\max_{\tau, f_{d,i}} \left\{ |\mathcal{C}_i(\tau, f_{d,i})|^2 \right\} \tag{4}$$

$$\hat{\alpha}_i = \left| \mathcal{C}_i(\hat{\tau}_i, \hat{f}_{d,i}) \right| \tag{5}$$

$$\hat{\phi}_i = \angle \mathcal{C}_i(\hat{\tau}_i, \hat{f}_{d,i}) \ , \tag{6}$$

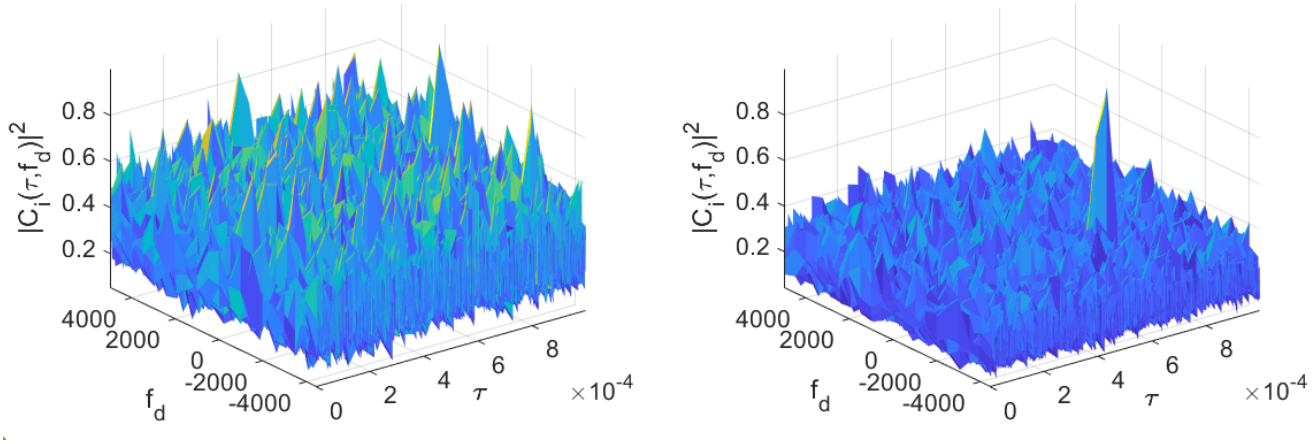
and we decide that the i-th satellite is present by setting a detection threshold β (designed for a desired false alarm probability) on the test statistic in the optimization problem in (4) such as

$$|\mathcal{C}_i(\tau, f_d)|^2 \underset{\mathcal{H}_0}{\overset{\mathcal{H}_1}{\gtrless}} \beta \ . \tag{7}$$

A. CAF Evaluation

The CAF is function which depends on the delay τ and the Doppler frequency f_d of the local replica. The optimization in (4) is performed over a grid of possible τ and f_d values, typically evaluating the CAF on a set of discrete values. Such bi-dimensional grid is referred to as the search space. The search space consist of set of cells which include the different value of delay and Doppler, which we gather in vectors $\boldsymbol{\tau} \in \mathbb{R}^{n_\tau}$ and $\mathbf{f}_d \in \mathbb{R}^{n_f}$, respectively. Typically, we have that $n_\tau > n_f$. The evaluation of this grid can be performed following several strategies that trade-off search speed and performance. Three searching strategies are typically considered: maximum search, serial search, and hybrid search strategies [1].

1) Maximum: this strategy evaluates the CAF all over the search space $\mathbb{R}^{n_\tau} \times \mathbb{R}^{n_f}$, such that each cell corresponds to a CAF value at the corresponding delay/Doppler pair. The overall maximum value of the ambiguity function is then selected and compared to the threshold β, if the maximum's value is greater than β the satellite is considered acquired, with estimated code delay and Doppler frequency corresponding to those of the maximum's cell.
2) Serial: in this strategy the ambiguity function is evaluating serially cell by cell. In each cell when the ambiguity function (7) is computed, it is immediately compared to the threshold. If the value exceeds the threshold the acquisition process stops and the value of the estimated code delay and Doppler frequency are matched to those from the cell under the test. This strategy has the benefit of reducing the number of CAF evaluations, at the expenses of some performance degradation.
3) Hybrid: this strategy evaluates the ambiguity function row-by-row (or column-by-column), and at the end of each row (column) the values of the computed ambiguity functions are compared to the threshold. As soon as the maximum value in the current row (column) exceeds the



(a) Signal is absent, $\mathcal{H}_0$

(b) Signal is present, $\mathcal{H}_1$

Fig. 1: CAF evaluation at the delay/Doppler grid in the (a) absence and (b) presence of signal with $C/N_0 = 39$ dB-Hz.

threshold the acquisition process stops and the estimated code delay and Doppler frequency are set to the corresponding cell. This strategy brings in a balance between the two approaches above.

In this work the maximum strategy is used as a searching strategy. This allows us to have a full representation of the CAF in (7), which is then fed to a NN for classification between $\mathcal{H}_0$ and $\mathcal{H}_1$ hypotheses as detailed in Section III.

B. Receiver Operating Characteristic

This part described Receiver Operating Characteristic (ROC). In first step, detection process determine the presence or absence of the signal is transmitted by satellite and the output is the random variable which is called decision variable. If the signal is present, The probability that the decision variable passes a threshold β is called the detection probability and if the signal is absent it called false alarm probability. Then the plot of detection probability (P_d) versus the false alarm probability (P_{fa}) is called the Receiver Operating Characteristic (ROC).

In order to calculate the ROC curves, first one needs to calculate the P_{fa} and P_d probabilities. The value of the detection threshold β is typically computed for a given false alarm probability, given by

$$P_{fa,K}(\beta) = \exp\left(-\frac{\beta}{2\sigma_n^2}\right) \sum_{k=0}^{K-1} \frac{1}{k!} \left(\frac{\beta}{2\sigma_n^2}\right)^k \tag{8}$$

where K indicates the number of non-coherent integrations (i.e. averages of K coherent integrations as in (7)) considered (such that $K = 1$ in absence of non-coherent integration) and $\sigma_n^2 = \frac{\sigma^2}{2N}$ is the variance of the in-phase and quadrature outputs.

Then, the P_d can be calculated as a function of β by

$$P_{d,K}(\beta) = Q_K\left(\sqrt{K\frac{\lambda}{\sigma_n^2}}, \sqrt{\frac{\beta}{\sigma_n^2}}\right) \tag{9}$$

where $\lambda = \alpha_i^2/4$ is the non-centrality parameter, and the generalized Marcum Q-function is defined as

$$Q_K(a,b) = \frac{1}{a^{K-1}} \int_b^{+\infty} x^K \exp\left(-\frac{a^2+x^2}{2}\right) I_{K-1}(ax)dx \,, \tag{10}$$

which allows for computation of the ROC curves.

III. Deep Learning Approach

In this work, the goal is to create a neural network model that is capable of recognizing the presence/absence of satellite signal. To that aim, we use as inputs the CAF evaluated at the delay/Doppler grid, which can be considered as an image. Such images (refer to Fig.1 for an exemplary situation) has certain characteristics that can be used to determine whether the signal is present or not: i) in the absence of signal from a specific satellite, the image should be composed of random values (theory telling that the CAF would be exponentially distributed in that case); and ii) in the presence of a satellite a peak should emerge from the random noise floor. This knowledge can be used to train a data-driven model (e.g. a neural network of some sort) such that a classifier can be used which learns to discriminate between $\mathcal{H}_0$ and $\mathcal{H}_1$, the hypotheses described earlier in Section II.

In this work two different structures of artificial neural networks are considered and compared: i) a Multi-Layer Perceptron (MLP), which is a neural network architecture with moderate complexity that has been widely used in the machine learning literature; and ii) a Convolution Neural Network (CNN), very popular within the computer vision community thanks to its ability to capture complex non-linear phenomena, at the expenses of a much larger complexity compared to MLPs. These models are discussed in this section after a brief overview on how the classifier is built following a probabilistic approach, in which the different NNs are in charge of delivering Bayesian estimates of the probabilities of each hypothesis given the observed data. Recall that the data

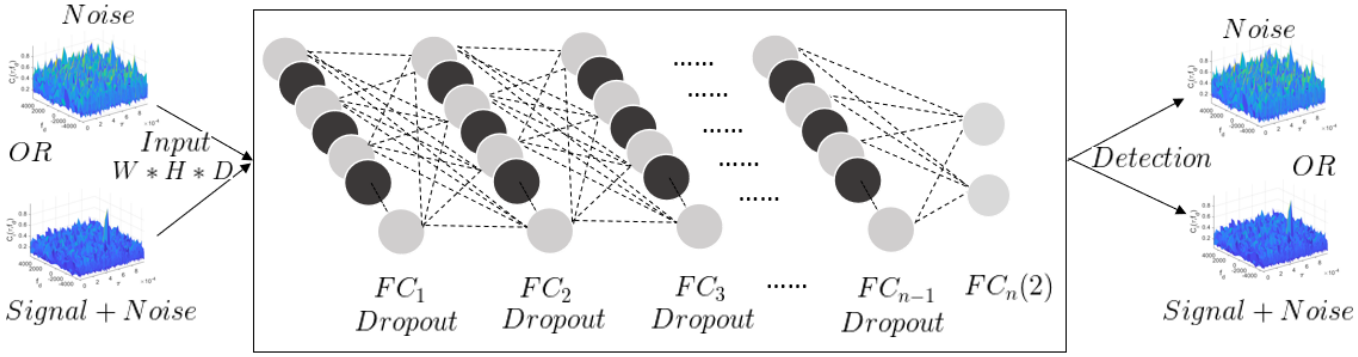


Fig. 2: Detection scheme of GNSS acquisition by using the Fully Connected Structure of Deep Learning approach.

fed to the NNs is the CAF's delay/Doppler map for the i-th satellite, which we denote with $\mathbf{Z}_i$ in the sequel. The proposed methodology works on a per-satellite basis. That is, the $\{m, n\}$ element of the input matrix is defined as

$$[\mathbf{Z}_i]_{m,n} = \left| \mathcal{C}_i \left([\boldsymbol{\tau}]_m, [\mathbf{f}_d]_n \right) \right|^2 , \tag{11}$$

where $\boldsymbol{\tau}$ and $\mathbf{f}_d$ are vectors containing the tested delay and Doppler-shifts, respectively. We use the convention that $[\mathbf{a}]_m$ represents the m-th element in vector $\mathbf{a}$ and that $[\mathbf{A}]_{m,n}$ provides a shortcut for the element of $\mathbf{A}$ in the m-th row and n-th column.

In the Bayesian sense, the information of the models is gathered in their *a posteriori* distribution after observing the data. An optimal (Bayesian) test between $\mathcal{H}_0$ and $\mathcal{H}_1$ is given by the ratio

$$\frac{p(\mathcal{H}_1|\mathbf{Z}_i)}{p(\mathcal{H}_0|\mathbf{Z}_i)} \underset{\mathcal{H}_0}{\overset{\mathcal{H}_1}{\gtrless}} 1 , \tag{12}$$

in which case we basically favor the model with largest a posteriori probability. This can be further expanded in terms of the likelihood and a priori distributions as

$$\frac{p(\mathbf{Z}_i|\mathcal{H}_1)}{p(\mathbf{Z}_i|\mathcal{H}_0)} \frac{\mathbb{P}(\mathcal{H}_1)}{\mathbb{P}(\mathcal{H}_0)} \underset{\mathcal{H}_0}{\overset{\mathcal{H}_1}{\gtrless}} 1 , \tag{13}$$

where we readily identify that $\mathbb{P}(\mathcal{H}_i)$ denotes the a priori probability of the i-th hypothesis. In the absence of better priors, we may assume equally likely hypotheses $\mathbb{P}(\mathcal{H}_0) = \mathbb{P}(\mathcal{H}_1) = 1/2$. Otherwise, we might incorporate that information in the hypothesis test, resulting on the adjustment of a threshold γ. The resulting test statistic is such that

$$\mathcal{T}(\mathbf{Z}_i) \triangleq \frac{p(\mathbf{Z}_i|\mathcal{H}_1)}{p(\mathbf{Z}_i|\mathcal{H}_0)} \underset{\mathcal{H}_0}{\overset{\mathcal{H}_1}{\gtrless}} \gamma , \tag{14}$$

which would substitute the standard acquisition test defined in (7). Since the test statistic is a ratio of probabilities, we have that $0 < \mathcal{T}(\mathbf{Z}_i) < \infty$.

The trained NNs (explained below) are then providing the probabilities of each of the hypotheses in (14). Therefore, the input data would be $\mathbf{Z}_i$ and the output of the NN would be the estimated probability for the i-th satellite to be absent or present in the dataset $\mathbf{y}$ used to build $\mathbf{Z}_i$.

If the test in (14) is in favor of $\mathcal{H}_1$, then an estimate of the delay/Doppler for the i-th satellite is given by the arguments of the largest element in $\mathbf{Z}_i$. That is,

$$\{\hat{m}, \hat{n}\} = \arg \max_{m,n} [\mathbf{Z}_i]_{m,n} \tag{15}$$

such that $\hat{\tau}_i = [\boldsymbol{\tau}]_{\hat{m}}$ and $\hat{f}_{d,i} = [\mathbf{f}_d]_{\hat{n}}$.

Neural networks (NNs) are models composed of neurons, which are information processing units, for complex data processing. A NN typically contains an input layer, one or more hidden layers, and an output layer, as well as predefined activation functions that connect adjacent layers. Each layer has a specific weight, which is usually determined with backpropagation during a training process that involves large amounts of data with known labels [15], [16]. Remarkably, there are implementations of DNNs that are extremely efficient, allowing for fast, real-time execution of the DNN classifier once the network is trained. The main challenge for DNN being to have enough training data to characterize those effects in a data-driven manner, issue discussed in Section IV.

A. MLP

The first neural network structure that is used in this research is the so-called Multilayer Perceptron (MLP), which is referred to as a traditional neural network. This type of network is comprised of one or more layers of neurons (which consist of a row of neurons). Fig. 2 shows an exemplary representation of this type of network. The first layer is called the input layer, which is fed with the training dataset for learning the parameters of the (potentially several) hidden layers that are not directly exposed to the input. During training, a number of nodes in the hidden layer are randomly ignored or "dropped out", which are shown in black in Fig. 2. This process will temporarily remove them from the network with a given probability for all incoming and outgoing connections. The last layer of the MLP is called output layer, follow up by a *softmax* layer. The number of neurons in this last layer depends on the number of the classes that one wants to classify since those provide their probabilities. The softmax layer calculates a probabilities for each possible class, given a dataset. Those probabilities must add up to 1, and are used as in (14) in order

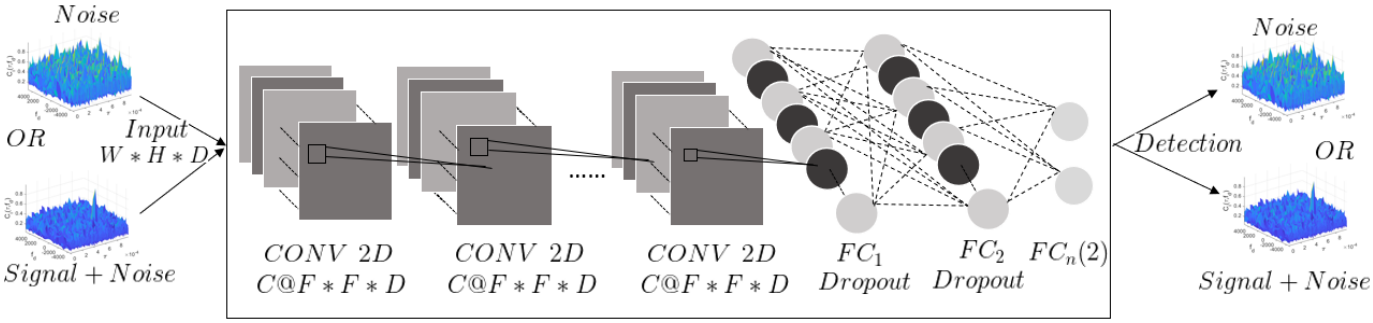


Fig. 3: Detection scheme of GNSS acquisition by using the Convolutional Neural Network Structure of Deep Learning approach.

to determine which class is the most probable given an input set $\mathbf{Z}_i$.

B. CNN

The second artificial neural network structure considered here is the so-called Convolutional Neural Network (CNN), which is one of the most popular models for deep learning in the context of learning class labels from image datasets. A CNN can have tens or hundreds of layers, where each of these layers learn to identify different features of an image [17], [18]. At each layer filters are applied to each training image and the output of each convolution image is used as an input to the next layer.

Fig. 3 illustrates a CNN structure. In contrast to other neural networks such as MLP, CNN is composed of an input layer, many hidden layers, and an output layer. During training, the input size of the CNN is fixed, the input is going through a stack of convolutional layers with the same or different filter sizes. In each convolution layer, the filter sweeps the input image from left to right and up to down by using stride with 2 pixels size, which is the number of pixels that each time the filter shifts. In the end, the convolution layers are followed by Fully Connected (FC) layers and a final *softmax* layer, which is used for classification purposes [18].

The CNN structure is shown in the box of Fig. 3 which has several convolution and fully connected layers. Each convolution layer consist of a number of filters (C), with filter size (F) and channel size (D). It transforms the input images with dimensions of $W_1H_1D_1$ through a set of convolution filters, each of these filters activates certain features from the images and creates an output with dimensions of $W_2H_2D_2$ as an input to the next layer. After each layer the batch normalization will be used to speed up learning and using the activation function to make input as a non-linearity output. The number of convolution layers are depend on the structure that is using. After the last convolution layer, the CNN architecture highlights fully connected layers in charge of a classification task, and the output of these layers is a vector with dimensions the number of classes (2 in this case) that will be predicted. The output would be the predicted probabilities for each class, as needed to compute (14).

IV. Simulation Setup and DNN training

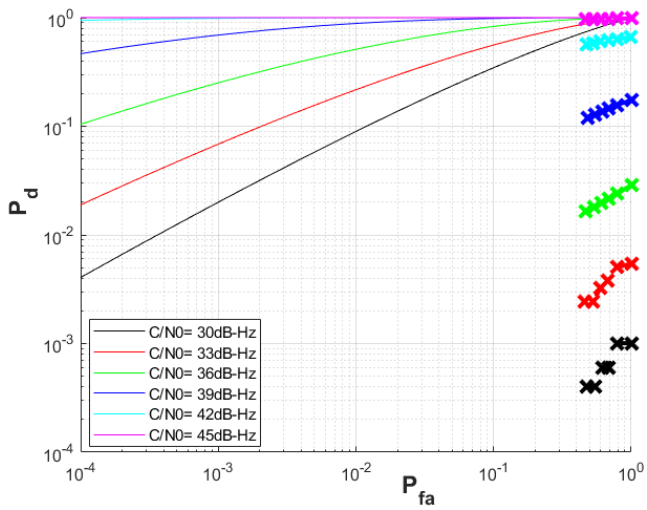
The goal of this work is framed within the described satellite signal acquisition process on a GNSS receiver. In particular, we propose to use a DNN to carry out the detection (or classification) process. The results of this work are shown in next section which are established from simulating a synthetical dataset with the two types of NNs described earlier, namely MLPs and CNNs.

The dataset that is used consists of 10^4 snapshots of, GPS L1 C/A, I&Q samples with different Carrie-to-Noise-density ratio (C/N_0) varying between 30 to 45 dB-Hz, as well as randomly generated delays between 0 to 1 ms and Doppler shifts between -4000 to 4000 Hz. These samples are then processed to compute the CAF over a Doppler-delay grid. An analogy to images can be made for these CAFs where each Doppler/delay cell is a pixel whose value is that of the CAF, $\mathbf{Z}_i$. For instance, if 20 Doppler bins are considered to acquire a GPS L1 C/A signal, those images would be 20×1023 dimensional. These images are fed to the input layer of the DNN, whose output would be the classification between presence/absence of the i-th satellite signal, as well as its Doppler and delay values if present. In a supervised training scheme, these input/output pairs are provided by labeled data by using the aforementioned synthetic data generation.

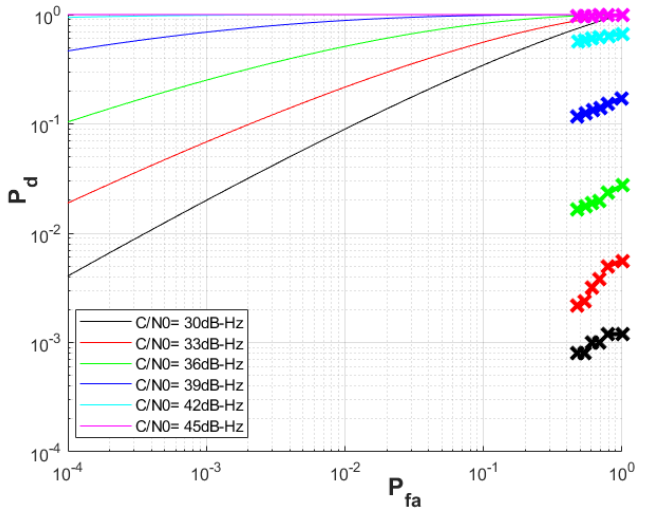
The proposed method is based on NNs, which is using a snapshot containing either signal-plus-noise or noise-only as ground truth and use the classifier network with softmax layer and dropout to classify the snapshot and predict the probability of belonging to each class.

The model implemented with MLP structures considered different number of fully connected layers. Each fully connected layer followed up with rectified linear unit (ReLU) activation function to allows for faster and more effective training by mapping negative value to zero [19]. After each layer a $1/2$ dropout probability was considered. The last fully connected layer contained two neurons, used in predicting the class for which the input image $\mathbf{Z}_i$ belongs to.

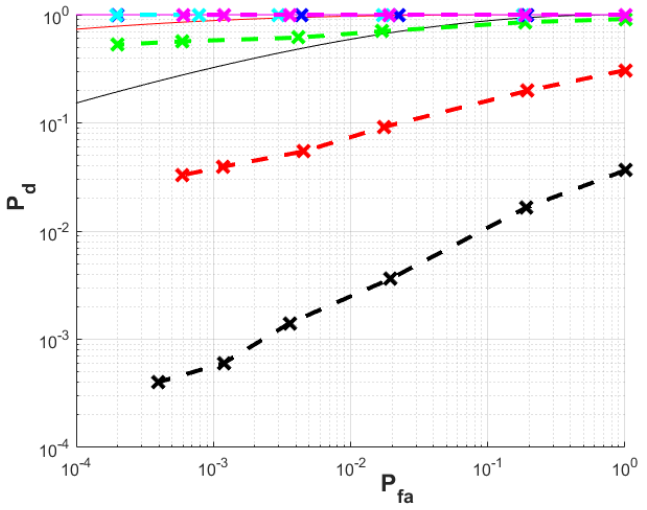
After defining the network structure, the training options were specified. The network trained with stochastic gradient descent with momentum (SGDM) optimizer with an initial learning rate of 0.001. The maximum number of epochs, which is a full training cycle on the entire training dataset, was set to



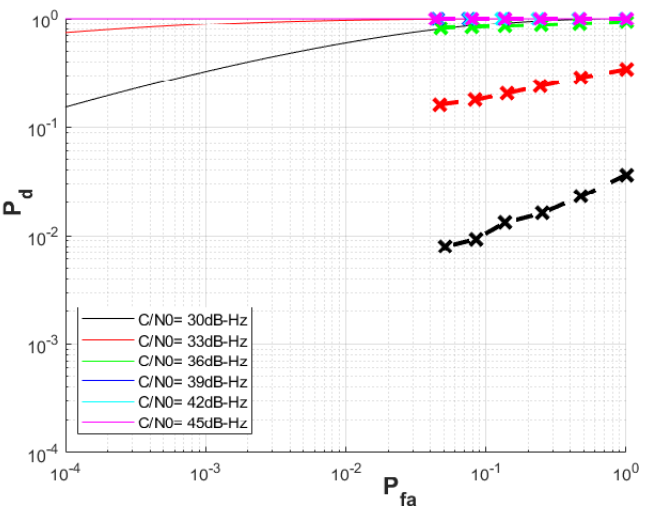
(a) MLP, 1 ms coherent integration only



(b) CNN, 1 ms coherent integration only



(c) MLP, 10 non-coherent integrations



(d) CNN, 10 non-coherent integrations

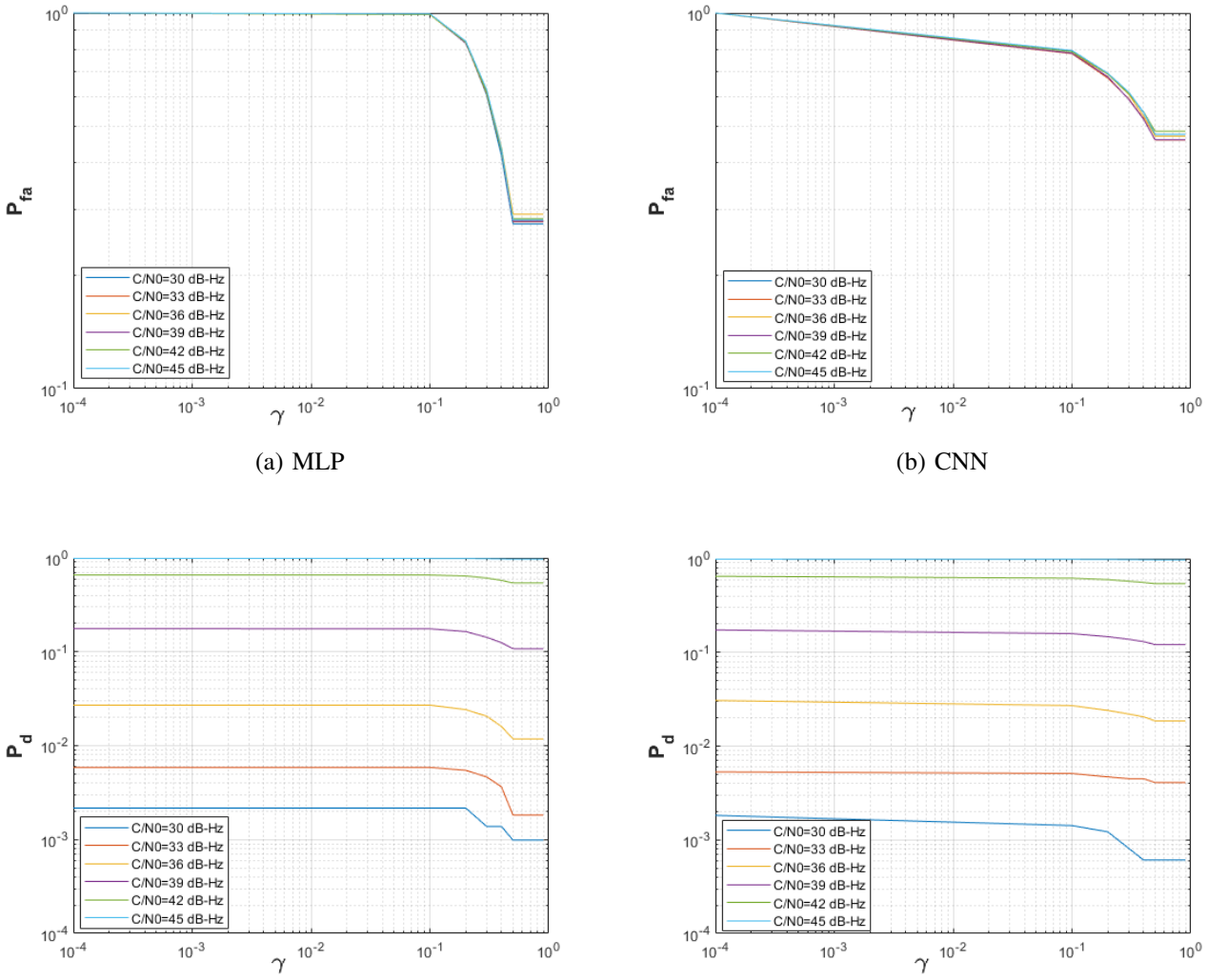
Fig. 4: ROC curves for MLP and CNN using CAF generated with (a,b) 1 ms coherent integration and (c,d) 1 ms coherent integration and 10 non-coherent integrations.

30 and at every epoch the data was shuffled. After 20 epochs the learning rate dropped by factor of 0.1. the training progress shows the mini-batch loss and accuracy and the validation loss and accuracy. The loss is the cross-entropy loss and the accuracy was defined as the percentage of inputs that the network classified correctly.

The CNN structure that was used is very similar to the VGG 16 structure [17]. The network has 13 convolution layers and 3 fully connected layers. Each convolution layer was followed by a batch normalization layer and a ReLu activation function. The batch normalization layers, normalizing the activation and gradients propagation through a network and using it between the convolution layer and ReLu layers to speed up network training [20]. Each fully connected layer follow up with ReLU activation function and a dropout layer with the probability of 0.5. The last fully connected layer contains two neurons to predict each image belongs to which class since two type of classes are exist in this work. After defining the network structure, the training options were specified, which were the same as for the MLP training options.

V. Result

The machine learning method was implemented using the two described types of NNs. In training, both were trained using either coherent integration data (i.e., 1 ms coherent) or non-coherent integrations (i.e., 1 ms coherent and 10 ms non-coherent).



(a) MLP (b) CNN

(c) MLP (d) CNN

Fig. 5: $P_d(\gamma)$ and $P_{fa}(\gamma)$ under 1 ms coherent integration for MLP and CNN models.

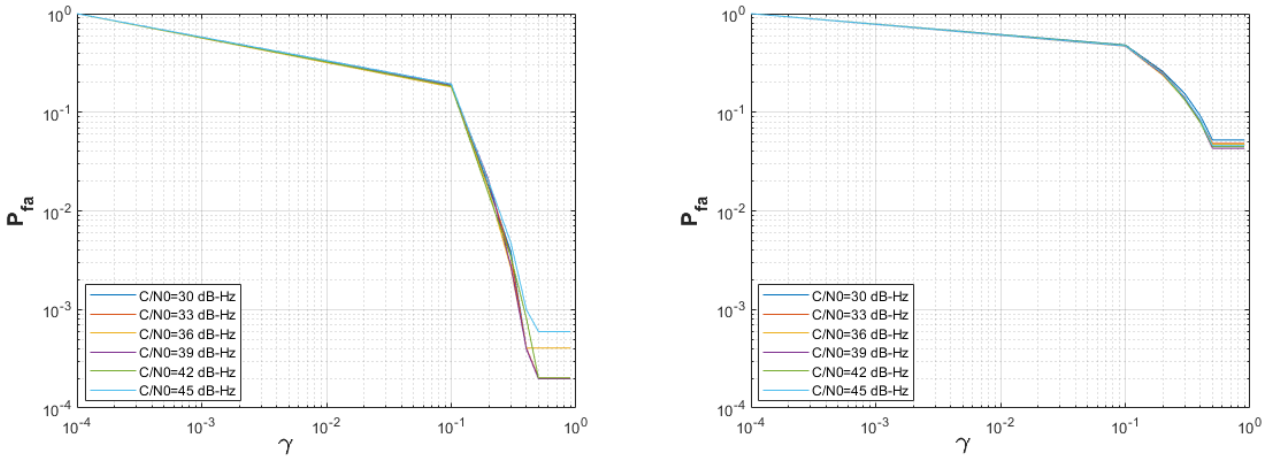
Fig. 4 to 6 shows the obtained result for both MLP and CNN under the two integration configurations. Notice that in this case, they are trained and tested under the same integration configurations.

From Fig. 4, it can be observed that for low C/N_0 values the performance of NNs is not attaining the theoretical ROC curves. However, as C/N_0 increases, such an approach is able to reach theoretical limits. An explanation could be that for low C/N_0 the various NNs cannot extract the relevant features from the *image*, $\mathbf{Z}_i$. When C/N_0, either because of an actual power increase or longer integration times, the NN is able to perform classification on whether the signal from the i-th satellite is present or not.

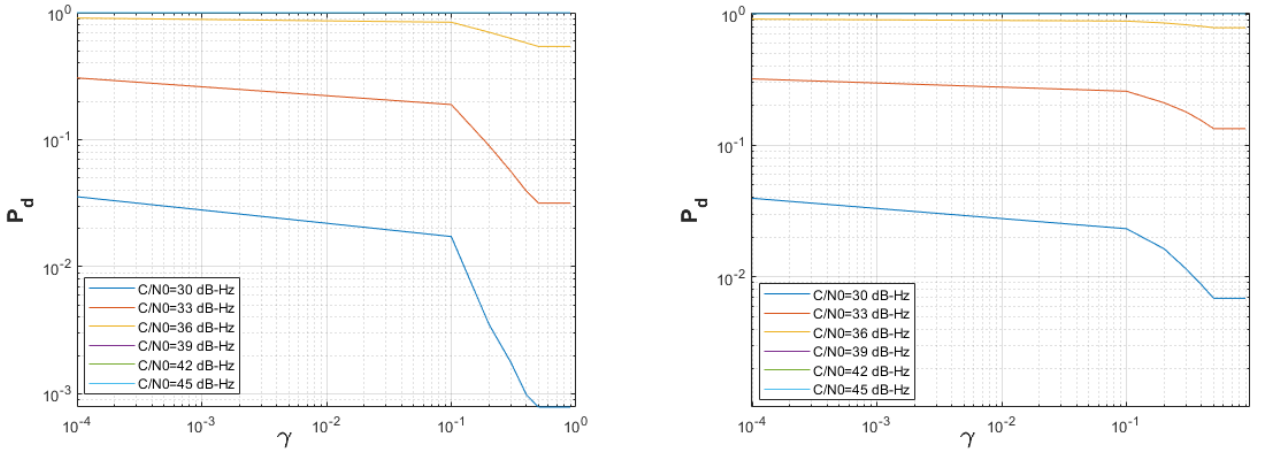
As for a comparison between MLP and CNN architectures, although their performance is very similar as per Fig. 4, the results show that MLP slightly outperforms CNN classification results.

This considerations are supported by the results in Fig. 5, where detection and false alarm probabilities are plotted against the classification threshold (i.e. $P_d(\gamma)$ and $P_{fa}(\gamma)$), for different value of C/N_0 under 1 ms coherent integration. Likewise in Fig. 6, where non-coherent values are shown. In essence, those probability plots show that CNN achieves larger P_{fa} values than MLP, which causes a degradation in their ROC curves.

The impact of low C/N_0 on ROC performance is further explained by the histograms of the test in (14) under both hypotheses, shown in Fig 7. Recall that one would like to have the histograms under $\mathcal{H}_0$ and $\mathcal{H}_1$ as *separate* as possible, which happens for large C/N_0 but clearly does not for low C/N_0 values. Whereas in the former the empirical distributions can be clearly distinguished (notice these were plot for $\log \mathcal{T}(\mathbf{Z}_i)$ to be visible), the former are overlapped such that a sample from $\mathcal{T}(\mathbf{Z}_i)$ cannot me meaningfully discerned among



(a) MLP (b) CNN



(c) MLP (d) CNN

Fig. 6: $P_d(\gamma)$ and $P_{fa}(\gamma)$ under 1 ms coherent and 10 non-coherent integrations for MLP and CNN models.

both distributions.

In the last set of results, we aim at understanding the generalization properties of these models under different receiver configurations, namely the integration intervals. The ROC curves for both MLP and CNN models for such experimentare shown in Fig. 8. For both models, during training the network is using CAF generated with 10 ms non-coherent integrations and then for the testing part it is using the CAF generated with 15 non-coherent integration time. These results should be compared those reported in Fig. 4. At the light of the plots, it seems that MLP has less generalization properties than CNN. it shown that in both MLP and CNN models the Neural networks performance increase by increasing the number of the integration time even if the network is trained with less integration interval when compared to Fig. 4(c), 4(d).

Further research is needed in understanding how learning from a configuration (e.g., coherent integration) can be transferred to other receiver configurations (e.g., non-coherent integrations).

VI. Conclusion

This article investigated the use of machine learning models (also known as neural networks, NNs) to learn the classification task of detecting signals from GNSS satellites. Traditional schemes based on correlation are optimal, and thus NNs are not expected to outperform those in nominal conditions. The use of NNs to substitute traditional approaches could bring benefits when nominal physics-based models do not hold. The article presented a proof-of-concept validating the use of NNs for detection purposes in GNSS. Two architectures, MLP and CNN were discussed, out of which MLP seemed to provide slightly better results, presumably due to the simplicity of the classification task. Future work will show the potential of such approach in learning more complex behaviours and, ultimately, being able to carry acquisition under contested environments.

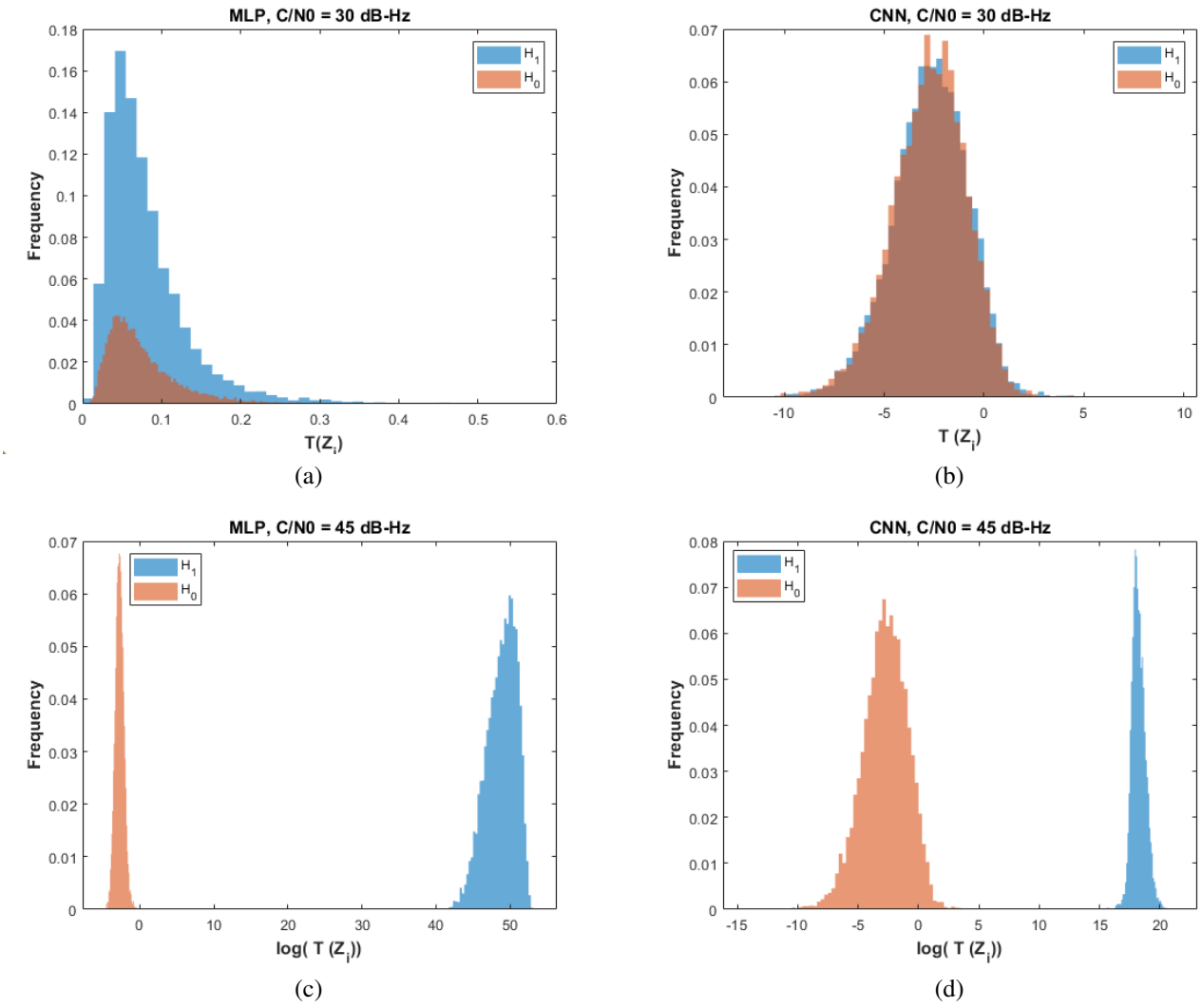


Fig. 7: Histogram of $\mathcal{T}(\mathbf{Z}_i)$ under $\mathcal{H}_0$ and $\mathcal{H}_1$ hypotheses for each NN and two different C/N_0 values.

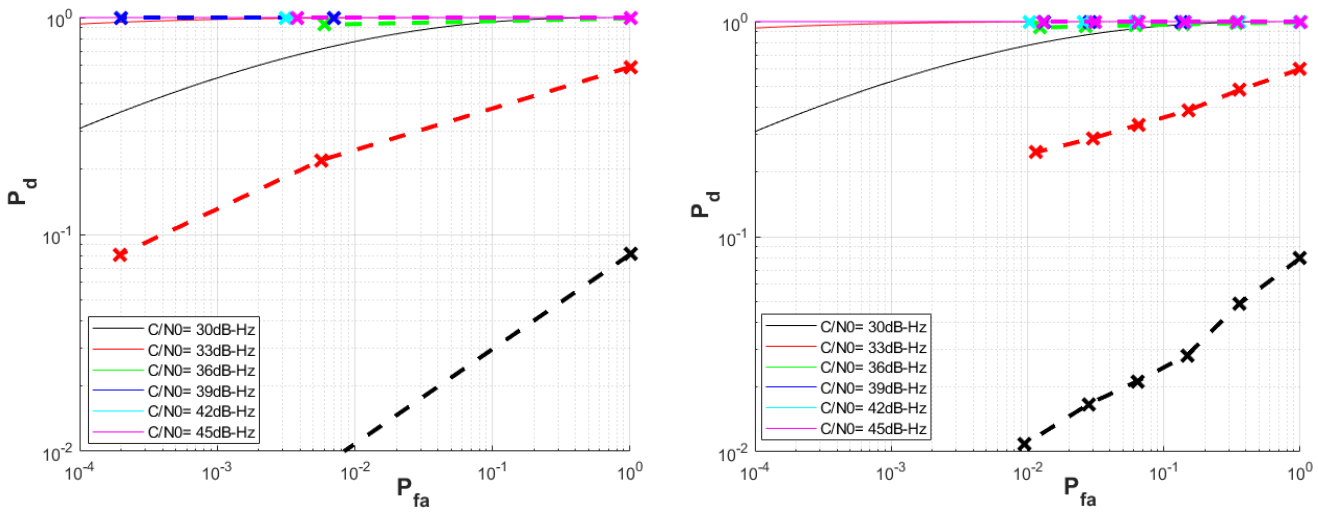


Fig. 8: ROC curves for MLP and CNN using different configurations to generate CAF inputs for training and testing.

ACKNOWLEDGMENT

This work has been partially supported by the NSF under Awards CNS-1815349 and ECCS-1845833.

REFERENCES

[1] D. Borio, "A statistical theory for GNSS signal acquisition," *PhD Dissertation Polytecnico di Torino*, 2008.

[2] E. Kaplan and C. Hegarty, *Understanding GPS: principles and applications*, Artech house, 2005.

[3] P. Misra and P. Enge, "Global positioning system: signals, measurements and performance second edition," *Global Positioning System: Signals, Measurements And Performance Second Editions,*, 2006.

[4] J. B. Tsui, *Fundamentals of global positioning system receivers: a software approach*, vol. 173, John Wiley & Sons, 2005.

[5] H. Mathis, P. Flammant, and A. Thiel, "An analytic way to optimize the detector of a post-correlation FFT acquisition algorithm," *Quadrature*, vol. 1000, pp. 1, 2003.

[6] A. Whalen, *Detection of signals in noise*, Academic press, 2013.

[7] A. Lehner and A. Steingass, "A novel channel model for land mobile satellite navigation," in *Institute of Navigation Conference ION GNSS*, 2005, pp. 13–16.

[8] D. Borio and P. Closas, "A fresh look at GNSS anti-jamming," *Inside GNSS*, vol. 12, pp. 54–61, 2017.

[9] D. Borio, "Robust signal processing for GNSS," in *Proc. of the 2017 European Navigation Conference (ENC)*, Lousanne, Switzerland, May 2017, pp. 150–158.

[10] D. Borio, "Myriad non-linearity for GNSS robust signal processing," *IET Radar Sonar and Navigation*, vol. 11, no. 10, pp. 1467–1476, Oct. 2017.

[11] D. Borio and P. Closas, "Complex signum non-linearity for robust GNSS signal mitigation," *IET Radar Sonar and Navigation*, pp. 1–10, Apr. 2018.

[12] D. Borio, H. Li, and P. Closas, "Huber's non-linearity for GNSS interference mitigation," *Sensors*, vol. 18, no. 7, pp. 2217, 2018.

[13] D. Borio and P. Closas, "Robust transform domain signal processing for GNSS," *Navigation*, 2019.

[14] H. Li, D. Borio, and P. Closas, "Dual-domain robust GNSS interference mitigation," in *Proceedings of the International Technical Meeting of The Satellite Division of the Institute of Navigation (ION GNSS+ 2019)*, Miami, FL, 16-20 Sept 2019.

[15] T. O'Shea, T. Roy, and T. Clancy, "Over-the-air deep learning based radio signal classification," *IEEE Journal of Selected Topics in Signal Processing*, vol. 12, no. 1, pp. 168–179, 2018.

[16] Q. Yan, W. Huang, and C. Moloney, "Neural networks based sea ice detection and concentration retrieval from GNSS-R delay-Doppler maps," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 10, no. 8, pp. 3789–3798, 2017.

[17] S. Liu and W. Deng, "Very deep convolutional neural network based image classification using small training sample size," in *2015 3rd IAPR Asian conference on pattern recognition (ACPR)*. IEEE, 2015, pp. 730–734.

[18] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[19] "Activation functions in neural networks," https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6.

[20] "Create simple deep learning network for classification," https://www.mathworks.com/help/deeplearning/examples/create-simple-deep-learning-network-for-classification.html.