Deep Learning-Based Activity Detection for Grant-Free Random Access

João Henrique Inacio de Souza, Graduate Student Member, IEEE, and Taufik Abrão, Senior Member, IEEE

Abstract—The cellular Internet of Things wireless network is a promising solution to provide massive connectivity for machine-type devices. However, designing grant-free random access (GF-RA) protocols to manage such connections is challenging, since they must operate in interference-aware scenarios with sporadic device activation patterns and a shortage of mutually orthogonal resources. Supervised machine learning models have provided efficient solutions for activity detection, noncoherent data detection, and nonorthogonal preamble design in scenarios with massive connectivity. In this article, we develop two deep learning (DL) sparse support recovery algorithms to detect active devices in massive machine-type communication random access. The DL algorithms, developed to deploy GF-RA protocols, are based on the deep multilayer perceptron and the convolutional neural network models. Unlike previous works, we investigate the impact of the type of sequences for preamble design on the activity detection accuracy. Our results reveal that sequences based on the Zadoff–Chu sequences, which present good correlation properties, achieve better activity detection accuracy with the proposed algorithms than random sequences. Besides, we demonstrate that our DL algorithms achieve activity detection accuracy comparable to state-of-the-art techniques with extremely low computational complexity.

Index Terms—Convolutional neural network, grant-free access, Internet of Things (IoT), massive machine-type communications (mMTC), multilayer perceptron, random access protocols.

I. INTRODUCTION

The massive machine-type communication (mMTC) service will provide full connectivity for Internet of Things (IoT) applications in cellular wireless networks. The machine-type devices have sporadic activation patterns, transmit small payloads with low-complexity network hardware, and are powered by short lifetime batteries. At the same time, they appear as increasing massive populations distributed over wide areas. When combined, all these characteristics impose challenges to provide connectivity, especially with regard to network access. The limited channel coherence time and the high number of devices prohibit the design of mutually orthogonal preamble sequences. Operating with fewer preambles than devices results in resource contention and network collision, which increases the access delay due to retransmissions. On the other hand, nonorthogonal preamble sequences reduce the number of collisions, at the cost of introducing interference among the active devices and degrading the system performance [1].

The random access (RA) protocols for supporting the mMTC service aim to provide connectivity under the premise of unpredictable and sporadic traffic generated by the machine-type devices. It is worth mentioning that the random aspect treated herein corresponds to the random activation pattern of the devices. In this sense, the grant-free random access (GF-RA) is an alternative to provide connectivity, dealing efficiently with the interference generated by the nonorthogonal preambles, while keeping a low network access delay [2].

Methods for sparse signal processing are broadly employed to cope with the interference and enable GF-RA procedures. These techniques are suitable for the typical mMTC scenario due to the sparse nature of the signals generated by machine-type devices. Specifically, we are interested in the sparse support and signal recovery methods. Sparse support recovery aims to estimate the indices of the active devices during a random access slot transmission. In contrast, sparse signal recovery is employed for channel estimation or data detection. Approaches to perform sparse support and signal recovery are classified in compressed sensing (CS)-based, covariance (CV)-based [2] and, more recently, machine learning (ML)-based.

The sparse support and signal recovery based on ML approaches have presented results that outperform substantially the other two CS-based and CV-based techniques, both in performance and computational complexity. These methods have extremely low running time at the cost of an intense offline training phase to learn the neural network parameters. Furthermore, according to the universal approximation theorem, any Borel measurable function can be approximated with an arbitrary nonzero amount of error by a properly designed neural network [3].

We can classify the ML-based approaches into the data-driven and model-driven categories. Data-driven approaches [4]–[7] use model-agnostic schemes that learn the mapping to the outputs purely from data. These approaches are convenient for situations involving intractable, incomplete, or poorly understood mathematical models, when it is hard to obtain accurate parameter estimate, or when the conventional approaches result in computationally inefficient algorithms. Besides, the data-driven approaches are able to extract implicit features from data, helping to understand the problem at hand [8]. On the other hand, model-driven approaches [9]–[15] are based on the domain knowledge and structures in data. One common strategy is to craft application-specific neural networks from traditional algorithms using the deep unfolding technique originally proposed in [16]. The model-driven approaches join the specialized
knowledge of classical signal processing with the flexibility and generalizability of the ML tools. However, their application needs to be carefully assessed. Highly model-dependent algorithms may result in poor inference or waste of computational resources, since they ignore the fact that system parameters are dynamic and may be unknown. Additionally, algorithms which incorporate simplistic models may not capture the main properties and dynamics of the data.

A. Literature Review

We provide an outline of the literature on sparse support and signal recovery applicable to GF-RA protocols. The approximate message passing (AMP) algorithm from the CS literature is present in many recent contributions for device activity detection and channel estimation [17], [18]. Specifically, the authors in [18] proposed a noncoherent scheme that embeds information bits into the devices’ preamble sequences, allowing joint activity and data detection. Despite the promising results of the AMP algorithm, its dependence on statistical channel information and poor stability represent technical difficulties for practical implementations.

Iterative methods using the empirical covariance matrix of the received signal are used for device activity detection in [19]. Two low-complexity algorithms to detect the set of active devices in massive MIMO unsourced RA are studied. Although the techniques achieve performance comparable to the AMP with multiple-measurement vectors and better numerical stability, the computational complexity is still high owing to the iterative structure of the estimation algorithms. Adopting a different strategy, the authors in [20] proposed an unsourced RA access scheme for millimeter-wave (mmWave) massive MIMO systems based on beam-space tree decoding. The scheme exploits the intrinsic beam division property of the mmWave channel to serve more active users and improve the system performance, outperforming the method proposed in [19].

The problem of mMTC GF-RA has been addressed in a variety of scenarios supported by different technologies. For instance, in [21], the problem of joint activity detection and channel estimation in cell-free massive MIMO was studied. The authors developed an asymptotic analysis of the MMSE estimation of sparse signal vectors, obtaining activity detection rules both at only one access point and at all access points considering cooperation between them. In another context, the same problem is tackled for satellite-enabled IoT applications in [22]. The authors proposed a Bernoulli–Rician message passing with expectation-maximization algorithm proof against unknown channel impairments.

On the matter of the ML-based approaches, neural networks obtained via deep unfolding of existing algorithms are presented in [9]–[12] and [15]. The networks of [9] and [10] are based on the AMP and, differently from the original algorithm, do not require prior information about the system parameters and channel statistics due to the transformation of the AMP parameters into trainable weights. The recursive neural network obtained from the iterative shrinkage thresholding algorithm for complex group row-sparse matrix signals in [11] converges quicker while achieving more prediction accuracy than the benchmark algorithms and robustness against ill-conditioned measurement matrices. The authors in [12] provided a framework to develop neural networks with adaptive depth, improving the efficiency of fixed-depth networks obtained via deep unfolding.

End-to-end neural networks that mimic the noisy measurements and the sparse recovery to jointly design the preamble matrix and the receiver are proposed in [4], [6], and [13]. Different deep auto-encoder algorithms, derived from CS-based and CV-based algorithms, that achieve high channel estimation and activity detection accuracy are proposed in [13]. These end-to-end network designs demonstrate the benefits of correctly choosing the preamble sequences to attain reasonable sparse recovery accuracy. As demonstrated by the aforementioned works, designing preamble matrices matched with the receiver is a promising path to increase the sparse recovery accuracy. However, we remark that crafting preamble sequences from scratch results in additional overhead to transmit the generated sequences to the machine-type devices. Such problem is mitigated with the adoption of easy-to-generate sequences, which can be systematically obtained at the device-side. Considering this context, in [15], three types of easy-to-generate preamble sequences, one obtained from a Gaussian random distribution, another obtained from a binary random distribution, and the last one obtained from the deterministic Zadoff–Chu (ZC) sequences, are evaluated and compared in joint device activity detection and channel estimation schemes. Numerical results reveal that accurate activity detection can be achieved with non-Gaussian preamble sequences taken from known sequences sets with good correlation properties. In this sense, preamble sequences developed by concatenating multiple short ZC sequences were proposed for mobile satellite communications in [23]. The resulted sequences are proof against the satellite environment and require low-complexity detection while benefiting the multiuser access performance.

A new algorithm for device activity detection based on a convolutional neural network (CNN) was presented in [7]. The CNN performs feature extraction from the received symbols to feed a densely connected layer that outputs the activity indicators. Simulation results reveal that the CNN outperforms a conventional neural network both in activity detection accuracy and computational complexity. Algorithms based on CNNS have been achieved promising results in applications such as image and video processing, natural language processing, and anomaly detection. Compared with the conventional densely connected layers, the convolutional layers of a CNN have sparse connections between the neurons and parameter sharing. Additionally, their computations exploit the correlation between the network inputs. In [24], a framework to recover sparse signals from noisy measurements in imaging applications using a CNN was developed. The scheme learns both the representation for signals and the inverse mapping function from measurement vectors in an end-to-end network. The proposed network approximates the solution of state-of-the-art algorithms in a much lower running time, achieving a remarkable tradeoff between reconstruction time and probability of successful signal recovery.

B. Contributions

The contribution of our work is threefold.

1) We propose two robust, low-running time deep learning (DL) algorithms for sparse support recovery for efficient device activity detection in mMTC GF-RA protocols.

2) We extend the neural network design workflow by adding a feature selection step based on choosing sets of preamble sequences with good correlation properties.
3) We carry out a comprehensive numerical characterization analysis of the activity detection accuracy and computational complexity achieved by the proposed algorithms under a variety of scenarios and different settings of preamble sequences sets.

Elaborating further, we develop two DL algorithms for sparse support recovery used for efficient device activity detection in mMTC GF-RA protocols. The proposed DL data-driven neural networks do not require any statistical information of the channel or the devices’ activity, having extremely low running time and robustness against variations in parameters of the system. The first network is a deep multilayer perceptron (DMLP) composed by densely connected layers, which inputs are the received symbols during the random access slot. The second network is a CNN composed by stacked convolutional layers, which inputs are the received symbols correlated with the preamble sequences. Especially, the CNN performs the sparse support recovery by exploiting the structured correlation of the network inputs.

We extend the neural network design workflow by adding a feature selection step, where the type of the preamble sequences is selected aiming to provide input features for the networks which increase the activity detection accuracy. During this step, we evaluate the DMLP and CNN with features generated by random complex normal and Bernoulli sequences, as well as deterministic ZC sequences [25], which present good correlation properties. We demonstrate that the feature selection—in our case, represented by the choice of preamble sequences type—is as important as the network architecture design, supporting the use of well-known and easy-to-generate sequences to integrate ML-based algorithms for applications in wireless communications.

The proposed networks are subjected to a parameters selection step aiming to optimize their architectures. Then, an offline training with the backpropagation algorithm and a representative dataset is performed to learn the networks parameters. Differently from other works in the literature, we carry out a comprehensive analysis of the activity detection error via the receiver operating characteristic (ROC) and the detection error tradeoff (DET) metrics. Numerical results reveal that the DL algorithms have competitive activity detection accuracy with extremely lower running time than state-of-the-art CS-based algorithms. Especially, the feature selection demonstrates that the CNN benefits from the good correlation properties of the ZC sequences, achieving improved activity detection accuracy in the mMTC scenario. Besides, the networks are resistant to variations on the system parameters, demonstrating the robustness provided by the generalization power of the DL models.

II. SYSTEM MODEL

In this section, we describe the system model, as well as formulate the device activity detection problem. We consider the uplink of a narrow-band system constituted by one cell with $K$ machine-type devices, served by a single-antenna base-station (BS), as depicted in Fig. 1. These devices access the network with probability $p_a \ll 1$. For this reason, the number of active devices is much less than the total in the cell.

During the random access slot, the active devices transmit unique preamble sequences. These sequences identify the devices and are used at the BS to detect the set of active IoT devices. The preamble sequences have a length of $L < K$ symbols due to the limited channel coherence time. For this reason, it is impossible to design mutually orthogonal sets of preambles. In our work, we consider three types of nonorthogonal sequences for the preambles. The nonorthogonal sequences result in interference on the transmitted signal by the set of active devices during a random access slot. Details on the used sequences are given in the next subsection. The preamble sequences are defined by the vectors $a_k \in \mathbb{C}^{L \times 1}$ for $k = 1, \ldots, K$, and are normalized such that $\|a_k\|^2_2 = L, \forall k$. The preamble matrix $A \in \mathbb{C}^{L \times K}$, which is known at the BS, contains in its columns the preamble sequences of all the $K$ devices in the cell.

Let $\alpha_k \in \{0, 1\}$ for $k = 1, \ldots, K$ be an activity state indicator of the device $k$, which is equal to 1 if the device is active and 0 otherwise. We consider the activity state indicators to be i.i.d. random variables following a Bernoulli distribution with parameter $p_a$, i.e., $\alpha_k \sim p_a \delta(u) + p_a \delta(u - 1) \forall k$, where $p_a \delta(u)$ is the probability mass function of $\alpha_k$ and $\delta(u)$ is the Dirac delta function. The vector with the activity state indicators of all the $K$ devices is named activity descriptor

$$\alpha = [\alpha_1 \ldots \alpha_K]^T.$$ (1)

The channel links between each device and the BS follow the Rayleigh fading model, with channel coefficients associated to the device $k$ defined by $h_k \sim \mathcal{CN}(0, 1)$. It is worth noting that the evaluation of the proposed deep learning-based solutions under different channel conditions, e.g., line-of-sight propagation or spatially correlated channel vectors, is a promising path for future investigation.

The received signal at the BS during the random access slot is equal to the superposition of the preambles transmitted by each active device, which can be written as

$$y = \sum_{k=1}^{K} \alpha_k h_k a_k + z$$ (2)

where $z \sim \mathcal{CN}(0, \sigma^2 I)$ is the additive white Gaussian noise. Considering $x = [a_1 h_1 \cdots \alpha_K h_K]^T$, we can rewrite the received signal in a compact form

$$y = Ax + z.$$ (3)

\[1\]For simplicity but without loss of generality and since we propose a model-agnostic data-driven solution for device activity detection, a change in the channel model will not change the structure of the DL methods presented herein.
The goal of the device activity detection problem is to determine the set of active devices from the received signal and the preamble matrix $A$. This is equivalent to finding the index of the nonzero entries of the vector $x$, namely the support of $x$

$$\text{supp}(x) = \{k \in \mathbb{N}_+ \mid x_k \neq 0\}.$$  \hfill (4)

As the number of active devices is much less than the total number of devices in the cell (sporadic activation, i.e., $p_a \ll 1$), $x$ has a sparse structure. For this reason, the problem of finding the support of $x$ from $y$ and $A$ can be cast as a sparse support recovery problem.

The sparse support recovery problem is an object of study of the compressed sensing field, and has been addressed in many works using different approaches, e.g., see [26] and the references therein. In our work, the aim is to develop two DL algorithms for sparse support recovery in the context of device activity detection for mMTC. We demonstrate that a DL algorithm which inputs are the real and imaginary parts of the received signal is sufficient to efficiently perform device activity detection.

A. Nonorthogonal Preamble Sequences

The preamble sequences play a key role in the device activity detection problem, as they dictate the interference patterns on the signal transmitted by the active devices during each random access slot. In addition, the inputs of the DL activity detection algorithms developed in the sequel are based on the BS received symbols, (3), a linear transformation of the preamble matrix. Hence, the learned neural network parameters depend directly on the preamble sequence features (cross and auto-correlation functions), which control how the signals of the simultaneously active devices are interrelated.

We remark two desired properties in preamble sequences. First, the preamble sequences must be easy-to-generate, coordinate, at the device-side. Such a property is desired since in the massive machine-type GF-RA, transmitting fully learned preamble sequences as in [4], [6], and [13] to a high number of devices in the cell creates a prohibitive communication overhead. Conversely, well-known sequences as the ZC and random sequences can be generated systematically and coordinately at the device-side with low overhead. The second property is the separability among the sequences in the set. The separability governs the interference levels between the signals of the simultaneously active devices. For this reason, choosing sets of sequences with high separability is important to obtain reasonable activity detection accuracy.

In the context of sparse support recovery using CS techniques, it is common to use preambles generated by sampling random distributions. However, we show that DL algorithms with deterministic preamble sequences sets that have good correlation properties can achieve reasonable activity detection accuracy. In the following, we present three suitable types of preamble sequences sets used in our work.

Random sequences: We use two types of random sequences for the preambles. Random sequences are useful, especially in cases where the number of mutually orthogonal preambles is not sufficient for all devices, e.g., in crowded mMTC applications, because it is easy to generate large amount of unique preambles. Additionally, matrices generated by sampling sub-Gaussian random distributions, such as Bernoulli, satisfy, with high probability, the restricted isometry property (RIP), a necessary condition to guarantee the signal reconstruction of many CS algorithms [27]. The first type of random sequence is obtained by sampling a circularly symmetric complex normal distribution with zero mean and unitary variance

$$[\bar{a}_1^k]_l \sim \mathcal{C}\!\mathcal{N}(0,1)$$  \hfill (5)

for $k = 1, \ldots, K$ and $l = 1, \ldots, L$. We name this type of sequence as normal. In order to meet the preamble normalization presented previously, we scale the Normal preamble, obtaining

$$\bar{a}_k^l = \sqrt{\frac{L}{\|a_k\|_2^2}} \bar{a}_k^l.$$  \hfill (6)

The second type of random sequence type is the Bernoulli. The Bernoulli sequences are real two-valued sequences obtained by sampling a symmetric random distribution of the type

$$[\bar{a}_1^k]_l \sim p_b(u) = \frac{1}{2} \delta(u + 1) + \frac{1}{2} \delta(u - 1)$$  \hfill (7)

for $k = 1, \ldots, K$ and $l = 1, \ldots, L$. As each entry of the Bernoulli sequence has norm equal to 1, normalization is not required. Zadoff–Chu sequences: The ZC sequences are polyphase sequences with elements defined by [25]

$$[\bar{a}_1^k]_l = \exp \left[ -\frac{j \pi r (l - 1)}{L} \right], \quad l = 1, \ldots, L$$  \hfill (8)

where $j = \sqrt{-1}$ and $r \in \{1, \ldots, L - 1\}$ is a number relatively prime to $L$ named the sequence root. A ZC sequence has the ideal auto-correlation property, i.e., its auto-correlation value is equal to zero for all the shifted versions of the sequence. For this reason, a sequence and its shifted versions comprise a set of mutually orthogonal sequences. The ideal auto-correlation property holds only for sequences generated by a single root. On the other hand, sequences generated by different roots have constant cross-correlation equal to $1$ if the difference between the roots is relatively prime to $L$ [28]. Therefore, a set of nonorthogonal sequences with a three-valued cross-correlation function can be generated by taking the shifted versions of multiroot ZC sequences. We use this set of nonorthogonal sequences to generate the ZC preambles. The ZC preambles are defined by the sequences

$$\{\bar{a}_1^k\}_{r \in R}$$  \hfill (9)

and its shifted versions, considering $R \subset \{1, \ldots, L - 1\}$ the set of chosen roots. Given the number of devices in the cell and the preamble length, the minimum number of roots to generate unique preambles for all the $K$ devices is equal to

$$N_r = \left[ \frac{K}{L} \right].$$  \hfill (10)

It is worth mentioning that the set of nonorthogonal ZC preambles is composed by $|R| \geq N_r$ smaller subsets of orthogonal ones. For this reason, allocating the subset of orthogonal preambles to devices with similar activation pattern is an efficient alternative to manage the interference levels.

\footnote{Separability implies low cross-correlation among any pair of preamble sequences in the set.}
B. Sequences Performance in Activity Detection Problem

In order to give an insight of the impact of the preamble design on the performance of the device activity detection scheme, we demonstrate a result on the distribution of the signal-to-interference-plus-noise (SINR) ratio of the received signal at the BS for the analyzed sequence types. The SINR gives an indirect measure of the activity detection accuracy, as it represents the ratio between the power of the signal of an active device and the power of noise plus the interference generated by the simultaneously interfering devices. Fig. 2 depicts the empirical cumulative distribution function (CDF) of the SINR for the 4th device by using the normal, Bernoulli, and ZC sequences, considering three values of activation probability, $p_a \in \{0.01, 0.1, 0.3\}$. The remain setup parameter values to generate this result are $K = 40, L = 20$, and $\sigma_n^2 = 0.1$, resulting in a signal-to-noise ratio (SNR) of 10 dB. Besides, we define the SINR correlating the received signal at the BS with the preamble of each active device, resulting

$$\text{SINR}_k = \frac{\|a_k\|^2}{\|h_k\|^2 + \|a_k^H z\|^2}$$

for $k \in \mathcal{K}_a$, where $\mathcal{K}_a$ is the set of active devices during a specific random access slot. From Fig. 2, one can observe in the set of curves that decreasing the activation probability increases the average SINR values, since it reduces the number of simultaneously interfering devices during the transmission interval. The ZC sequences achieve the best average SINR values due to their good correlation properties. At the same time, the Bernoulli sequences outperform marginally the Normal ones in terms of the SINR distribution.

III. Deep Learning Algorithms

In this section, we introduce the architectures of the DMLP and CNN algorithms devised for device activity detection. The two DL algorithms exploit different input features to perform device activity detection. Especially, the structure of the CNN exploits the correlation between the network inputs, benefiting from the good correlation properties of the ZC sequences for preamble design. The DL algorithms are represented by functions of the type $f(v_0, \theta): \mathcal{I} \to \mathbb{R}^{K \times 1}$, which learns an inverse map of the activity descriptor from the signal received during the random access slot. In these functions, we consider $v_0 \in \mathcal{I}$ the inputs in the input space $\mathcal{I}$, and $\theta$ the set with the algorithm parameters. The learning phase is implemented by a training procedure, in which the parameters $\theta$ of the function $f(v_0, \theta)$ are optimized from a set of examples containing the activity descriptor and the respective received signal at the BS.

Let $S = \{(y^{(s)}, \alpha^{(s)}) \mid s = 1, \ldots, S\}$ be a dataset containing the received signal and the activity descriptor of a random access slot. We denote the output of a DL algorithm for the sample $(y^{(s)}, \alpha^{(s)})$, as $\hat{\alpha}^{(s)}$. Taking this into account and recalling that $\alpha^{(s)} \in \{0, 1\}$, we can measure the quality of the estimate $\hat{\alpha}^{(s)}$ w.r.t. the original activity descriptor $\alpha^{(s)}$ by calculating the binary cross-entropy function. Therefore, in our DL algorithms, the loss function is defined as the average binary cross-entropy function [29], calculated over all the $S = |S|$ samples as

$$\mathcal{L}(S) = \frac{1}{SK} \sum_{s=1}^{S} \sum_{k=1}^{K} - \left[ \alpha^{(s)}_k \log \left( \hat{\alpha}^{(s)}_k \right) + \left( 1 - \alpha^{(s)}_k \right) \log \left( 1 - \hat{\alpha}^{(s)}_k \right) \right].$$

The aim of the training procedure is to minimize $\mathcal{L}(S)$ by choosing the right parameters of $f(v_0, \theta)$. In this way, the trained DL network produces an accurate estimate of the activity descriptor.

A. Deep Multilayer Perceptron

The DMLP algorithm, as depicted in Fig. 3, is implemented with densely connected layers. Its inputs are the real and imaginary parts of the received signal, while the output is an estimate of the activity descriptor. The DMLP architecture has two densely connected hidden layers with $N_n$ neurons and one densely connected output layer with $K$ neurons. The hidden layers have the ReLU activation function

$$\text{ReLU}(u) = \max(0, u).$$

Fig. 2. Empirical CDF of the SINR achieved by normal, Bernoulli, and ZC sequences for three values of $p_a$. Fig. 3. Architecture of the DMLP algorithm proposed for device activity detection. The inputs are the real and imaginary parts of the received signal, while the output is an estimate of the activity descriptor.
On the other hand, the output layer has the sigmoid activation function

$$\text{Sigmoid}(u) = \frac{1}{1 + \exp(-u)}.$$

The details of the DMLP layers are organized in Table I. It is worth noting that the numbers of inputs and outputs of the DMLP are tied respectively to the preamble length, $L$, and the number of devices, $K$. Hence, one needs to retrain a network from scratch in order to accommodate different numbers of devices with different preamble lengths.

Now, we introduce the parameters of the DMLP layers, which are the weights matrices and the bias values vectors. Given the dimensions of the inputs and outputs of each layer, we have the weights matrices $W_1 \in \mathbb{R}^{N_a \times 2L}$, $W_2 \in \mathbb{R}^{N_a \times N_a}$, and $W_3 \in \mathbb{R}^{K \times N_a}$ for the hidden layers 1 and 2, and the output layer, respectively. Similarly, the vectors with the bias values are $b_1 \in \mathbb{R}^{N_a \times 1}$, $b_2 \in \mathbb{R}^{N_a \times 1}$ and $b_3 \in \mathbb{R}^{K \times 1}$. Taking this into account, we define the set with the DMLP parameters by

$$\theta_{\text{DMLP}} = \{W_i, b_i\}_{i=1}^3.$$

Considering the DMLP parameters, we write the input–output relationship of the hidden layers in the DMLP

$$v_i = \text{ReLU}(W_i v_{i-1} + b_i), \quad i = 1, 2$$

where $v_i$ is the layer output, considering $v_0$ as the DMLP input, and the activation function is calculated for each entry of the input vector. The input–output relationship of the output layer is obtained following the same logic, substituting the layer parameters, as well as the ReLU activation by the sigmoid function.

With $\theta_{\text{DMLP}}$ in hands, it is possible to calculate the number of trainable parameters in the DMLP algorithm. Considering the dimensions of the weights matrices and the bias vectors values, the number of trainable parameters is given by

$$\Theta_{\text{DMLP}}(N_n, L, K) = N_a^2 + 2N_aL + N_nK + 2N_n + K.$$ (17)

Since the outputs of the DMLP are in the range $[0, 1]$, a hard decision module with threshold parameter $\tau \geq 0$ is positioned at the output of the algorithm to calculate the activity descriptor in its original domain. Hence, the hard decision output is

$$\hat{\alpha}_k = \begin{cases} 0, & \hat{\alpha}_k < \tau \\ 1, & \hat{\alpha}_k \geq \tau. \end{cases}$$ (18)

During the device activity detection, two different kinds of error may occur. First, a false alarm (FA) occurs when an inactive device is detected as active. Second, a miss detection (MD) occurs when an active device is detected as inactive. The probabilities of FA and MD are calculated in terms of the hard decision threshold by:

$$P_{\text{FA}}(\tau) = \Pr(\hat{\alpha}_k \geq \tau \mid \alpha_k = 0)$$ (19)

$$P_{\text{MD}}(\tau) = \Pr(\hat{\alpha}_k < \tau \mid \alpha_k = 1).$$ (20)

Given the imbalance on the number of active devices during each random access slot, the frequency of each type of error changes. The error probability as a function of $\tau$ is written in terms of the FA and MD probabilities as

$$P_e(\tau) = (1 - p_a)P_{\text{FA}}(\tau) + p_a P_{\text{MD}}(\tau).$$ (21)

The hard decision module can be optimized in order to meet different design criteria, e.g., minimize a specific type of error, or a metric which combines the two types of error with different weights. In our work, we evaluate the algorithms adjusting the hard decision threshold such that

$$\tau^* = \arg\min_{\tau \geq 0} P_e(\tau).$$ (22)

Since it is difficult to derive a closed-form expression for the output error of the proposed DL algorithms, the value of the hard decision threshold $\tau$ can be computed near-optimally $(\bar{\tau}^*)$ using a numerical approach. In our work, we use the samples into the training dataset to adjust $\tau$ in order to reasonably meet the criterion of (22). This can be done by first predicting the activity descriptors with the trained neural networks to obtain the outputs $\{\hat{\alpha}_k\}_{k=1}^S$. Then, the threshold $\tau^*$ that successfully recovers the largest number of samples is obtained by inspecting all the entries of the vectors $\{\hat{\alpha}_k\}_{k=1}^S$.

### B. Convolutional Neural Network

Now, we introduce the CNN algorithm depicted in Fig. 4. Differently from the DMLP, the CNN algorithm is implemented with convolutional layers. Particularly, we need to adjust the input dimension aiming to match it with the number of outputs. In order to accomplish this while exploiting the good correlation properties of the ZC preamble sequences, we use a correlator stage at the input of the network, which produces the signal

$$y_{\text{corr}} = A^H y.$$ (23)

This correlator stage can be seen as an additional layer with fixed weights instead of trainable parameters. Then, the inputs of the CNN are the real and imaginary parts of the correlated received signal, organized as two vectors of dimension $K \times 1$. The output is an estimate of the activity descriptor.

The CNN architecture has two hidden layers of the type 1-D convolution with $N_f$ feature maps each one, filters of length $N_w$, and ReLU as the activation functions. The output layer has one feature map, filters of length $N_w$, and sigmoid as the activation function. The details on the layers’ parameter values of the CNN algorithm are described in Table II. From Table II, one can see that the CNN architecture does not depend directly on system parameters like the number of devices or the preamble length. In addition, differently from the DMLP, the CNN has a flexible structure that allows inputs of variable size due to the window-based computations across the layers. Hence, a trained CNN can be reused for other system configurations by only changing properly the correlator stage.
The parameters of the CNN layers are the filters and the bias values w.r.t. each feature map. Let the superscript \( \ell \) be the index of the feature map. The parameters of the first hidden layer are \( W_1^{\ell} \in \mathbb{R}^{N_w \times 2} \) and \( b_1^{\ell} \in \mathbb{R}, \ell = 1, \ldots, N_f \). Next, the parameters of the second hidden layer are \( W_2^{\ell} \in \mathbb{R}^{N_w \times N_f} \) and \( b_2^{\ell} \in \mathbb{R}, \ell = 1, \ldots, N_f \). Finally, the parameters of the output layer are \( W_3^{\ell} \in \mathbb{R}^{N_w \times N_f} \) and \( b_3^{\ell} \in \mathbb{R} \). With these definitions, we assume the set with the CNN parameters as

\[
\theta_{\text{CNN}} = \left\{ W_1^{\ell}, b_1^{\ell} \right\}_{\ell=1}^{N_f}, \left\{ W_2^{\ell}, b_2^{\ell} \right\}_{\ell=1}^{N_f}, \left\{ W_3^{\ell}, b_3^{\ell} \right\}
\]  

(24)

Given the respective dimensions of the CNN parameters, we calculate the number of trainable parameters in the algorithm by

\[
\Theta_{\text{CNN}}(N_f, N_w) = N_w^2 N_w + 3N_f N_w + 2N_f + 1.
\]  

(25)

It is important to stress that, differently from the DMLP, the number of trainable parameters in the CNN does not depend explicitly on the number of devices and the preamble length. This is a good advantage of the CNN, as its size does not grow rapidly with the scenario parameters. However, \( N_f \) and \( N_w \) must follow the changes of the parameters in the scenario in order to achieve efficient results on device activity detection.

Let the convolution between the matrices \( M \) and \( N \) be

\[
[M \ast N]_t = \sum_{i=1-t}^{t-1} M_{t-i} N^T_{t-i}
\]  

(26)

where \( \bar{M} \in \mathbb{R}^{n \times p} \) is a zero-padded version of \( M \). The input–output relationship of the hidden layers is given by

\[
[v_i^{\ell}]_t = \text{ReLU} \left( [W_i^{\ell} \ast V_{i-1}]_t + b_i^{\ell} \right)
\]  

(27)

for \( i = 1, 2 \) and \( \ell = 1, \ldots, N_f \). We consider \( v_i^{\ell} \) as the output of the \( \ell \)th feature map in the layer \( i \), while \( V_i \) is a matrix which columns are the output vectors of all the feature maps, or channels, in the layer \( i \)

\[
V_i = \begin{bmatrix} v_1^i & \ldots & v_{N_f}^i \end{bmatrix}.
\]  

(28)

The input–output relationship of the output layer can be obtained easily by substituting the layer parameters and the ReLU activation by the sigmoid. As the DMLP, the CNN has a hard decision module to calculate the estimate of the activity descriptor in the original domain.

---

**Fig. 4.** Architecture of the CNN algorithm proposed for device activity detection. The inputs of the convolutional layers are the real and imaginary parts of the correlated received signal, while the network output is an estimate of the activity descriptor.

---

**TABLE III**

| Training algorithm | ADAM estimation |
|--------------------|-----------------|
| Training samples   | \( S_{\text{tr}} = 4.5 \cdot 10^5 \) (90%) |
| Validation samples | \( S_{\text{val}} = 0.5 \cdot 10^5 \) (10%) |
| Batch size         | 128             |
| Maximum epochs     | \( 10^5 \)       |
| Epochs for early stopping | 5             |

**TABLE IV**

Training Time (Seconds) of the DL Algorithms According to the Preamble Length

| \( L \) | DMLP | CNN |
|--------|------|-----|
| 7      | 2.49 \cdot 10^4 | 1.25 \cdot 10^4 |
| 13     | 3.88 \cdot 10^2 | 1.72 \cdot 10^3 |
| 17     | 3.97 \cdot 10^2 | 1.78 \cdot 10^3 |
| 21     | 3.82 \cdot 10^2 | 1.38 \cdot 10^3 |

**IV. NUMERICAL RESULTS**

In this section, we cover the performance and complexity numerical results associated to both proposed DL activity detection algorithms and the preambles sequences design, as well as compare with two baseline device activity detection techniques.

**A. DL Training Procedure**

The steps to train both DMLP and CNN algorithms for solving the device activity detection problem are described herein. The training procedure aims to optimize \( \theta_{\text{DMLP}} \) and \( \theta_{\text{CNN}} \) from a dataset \( S \) to obtain accurate activity detection, being carried out before the deployment of the device activity detection scheme. The dataset is obtained by sampling random distributions to generate the signals according to the definitions in Section II and evaluating (3). In our case, the dataset is constituted by \( S = 5 \cdot 10^5 \) samples, from which \( S_{\text{tr}} = |S_{\text{tr}}| = 4.5 \cdot 10^5 \) are used for training and \( S_{\text{val}} = |S_{\text{val}}| = 0.5 \cdot 10^5 \) used for validation. Each dataset is generated with a single realization of the preamble matrix. The DL algorithms are trained with the dataset samples using the *adaptive moment* (ADAM) estimation algorithm. The training stops if the *loss function* \( \mathcal{L}(S_{\text{tr}}) \) in (12) does not improve after five epochs. Table III contains the information on the training setup of the DL algorithms.

Table IV contains the average training time of the proposed DL algorithms for different preamble lengths. The training is carried out at a workstation equipped with a Nvidia GeForce
940MX GPU, an Intel Core i5-7200 U CPU @2.5 GHz and 8 GB of RAM.

**B. DL Input Parameters Tuning**

In the following, we describe the procedure to tune the architectural parameters of the DL algorithms. In order to get high activity detection accuracy and prevent overfitting, we carefully tune the number of neurons in the hidden layers of the DMLP and the number of feature maps in the hidden layers of the CNN. The parameters are tuned by hand in the ranges $N_n \in \{80, 160, 320, 400, 800, 1600\}$ and $N_f \in \{20, 40, 80\}$. The tuning procedure consists of generating and assessing the learning curves of the algorithms, which contain the loss values along the epochs, calculated by the binary cross-entropy function in (12) on the training and validation datasets. Ideally, the loss values calculated for the validation set must follow the values calculated for the training one, indicating improvement on the generalization capability of the algorithm. If the validation loss worsens, or the gap between the losses of the two datasets increases, there is no more gain on increasing the parameter value.

Fig. 5 depicts the learning curves for the DL algorithms varying their architectural parameters. At first glance, we see that, except for the case where $N_n = 800$, the DL algorithms attain convergence. Analyzing the curves, we observe that the minimum training loss always reduces increasing the parameters value, as expected. Fig. 5(a) illustrates the learning curves of the DMLP for $N_n \in \{80, 160, 320\}$, while the Fig. 5(c) depicts the validation loss for $N_n \in \{320, 400, 800\}$. Combining these results, one can infer that the DMLP attains overfitting with $N_n > 400$. Furthermore, we observe that the validation loss improves marginally from $N_n = 320$ to $N_n = 400$. Hence, we choose $N_n = 320$ for the DMLP algorithm. On the other hand, the validation loss of the CNN, depicted in Fig. 5(b), always decreases with the epochs. Despite that, the gap between the validation and training losses increases for $N_f > 40$.

**C. Performance: Error, FA, and MD Rates**

Numerical results on the activity detection accuracy of the proposed DL activity detection algorithms for GF-RA protocols are assessed and compared with two baseline algorithms available in the literature, the least absolute shrinkage and selection operator (LASSO) method [30] and the AMP algorithm [31]. The adopted AMP algorithm is implemented with the complex soft thresholding denoising function [32], while the LASSO method is computed via the coordinate descent optimization (CD) algorithm [33]. Both thresholding factor of AMP and regularization parameter of the LASSO are tuned for each evaluated scenario in order to develop a fair comparison, in which the baseline algorithms achieve good performance-complexity tradeoffs. Three figure of merit are evaluated: error rate, FA rate, and MD rate. It is worth mentioning that the numerical results in the following are generated using an independent evaluation dataset that contains different samples from those into the training dataset mentioned in Section IV-A. Table V summarizes the setup used to produce the evaluated scenarios and respective numerical results. The scenario parameters are consistent with other recently published papers on ML-based approaches for GF-RA [4], [6]. In addition, all the signals...
are generated by sampling random distributions following the definitions in Section II.

Fig. 6 depicts the dependence of the error rate w.r.t. the hard decision threshold ($\tau$) of the DL algorithms considering different values of undersampling ratio. One can notice that there exists a value of $\tau$ that minimizes the error rate attained by both DMLP and CNN algorithms, emphasizing the importance of optimizing this parameter. Table VI presents the values of the near-optimal hard decision threshold ($\tilde{\tau}$) for each setting of the DL algorithms implemented to generate the numerical results in the sequence.

Fig. 7 depicts the error rate versus the undersampling ratio $\Delta = L/K$ for both proposed DL algorithms, as well as the baseline algorithms. We evaluate each DL algorithm using the three types of sequences. Besides, we include the 95% confidence interval of the DL algorithms considering different values of undersampling ratio. One can notice that there exists a value of $\tau$ that minimizes the error rate attained by both DMLP and CNN algorithms, emphasizing the importance of optimizing this parameter. Table VI presents the values of the near-optimal hard decision threshold ($\tilde{\tau}$) for each setting of the DL algorithms implemented to generate the numerical results in the sequence.

The ROC curves are described by the successful detection rate, or true positive rate, versus the false positive rate, or 1 - true negative rate. These results are generated by increasing the value of the hard decision threshold in the range $\tau \in [0, 1]$. For $\Delta = 0.525$, the CNN using the ZC sequences has a huge improvement on the MD rate, if compared with the random sequences. Again, this is due to the correlator stage at the CNN input, the good correlation properties of the ZC sequences, and the CNN structure. Additionally, the ROC curve of the CNN using the ZC sequences is tight to the LASSO. At the same time, the ROC of AMP degrades significantly the AMP algorithm. The LASSO method has the best activity detection accuracy for $\Delta > 0.175$ at the expense of an extremely high computational complexity, as we demonstrate in Section IV-E. Moreover, the behavior of the error curves for the evaluated algorithms is similar for $p_a < 0.1$, with decreased error rates as a consequence of the reduced number of active devices, and therefore, the level of interference.

The ROC curves are described by the successful detection rate, or true positive rate, versus the false positive rate, allowing a comparison of the detection algorithms according to the two types of error.

TABLE VI
NEAR-OPTIMAL VALUES OF THE HARD DECISION THRESHOLD ($\tilde{\tau}$) FOR EACH SETTING OF THE DL ALGORITHMS IN Fig. 7

| $\Delta$ | DMLP Normal | DMLP Bernoulli | DMLP ZC | CNN Normal | CNN Bernoulli | CNN ZC |
|----------|-------------|---------------|---------|------------|---------------|--------|
| 0.175    | 0.466       | 0.501         | 0.486   | 0.517      | 0.529         | 0.487  |
| 0.325    | 0.502       | 0.517         | 0.486   | 0.506      | 0.477         | 0.456  |
| 0.425    | 0.502       | 0.519         | 0.497   | 0.459      | 0.439         | 0.512  |
| 0.525    | 0.516       | 0.522         | 0.501   | 0.490      | 0.503         | 0.466  |

4The confidence interval for a mean is computed by $CI_{1/2} = m \pm z_{\epsilon/2}s/\sqrt{n}$, where $1 - \epsilon$ is the confidence level, $z_{\epsilon/2}$ is the value such that the area to the right of it under the standard normal curve is $\epsilon/2$, $m$ is the sample mean, $s$ is the sample standard deviation, and $n$ is the number of samples. Notice that $z_{0.025} \approx 1.96$ for a confidence level of 95%.

5The ROC curves are described by the successful detection rate, or true positive rate, versus the FA rate, or false positive rate, allowing a comparison of the detection algorithms according to the two types of error.
Fig. 8. ROC curves of the DL algorithms using the three types of preamble sequences, and the baseline algorithms.

Fig. 9. DET curves of the DL algorithms using the three types of preamble sequences, and the baseline algorithms. $p_a = 0.1$. (a) $\Delta = 0.525$. (b) $\Delta = 0.175$.

D. Robustness of the DMLP and CNN Algorithms

Fig. 10 depicts the error rate versus the activation probability of the DL and the baseline algorithms. We evaluate each DL algorithm using the three types of sequences and two values of preamble length. Such figure of merit demonstrates the robustness of the DL activity detection algorithms in the face of variations in scenario parameters, since the networks are trained for a single-fixed activation probability $p_a = 0.1$. As expected, the error rate increases with the activation probability, as high numbers of active devices incur in increased interference power at the received signal. Moreover, regarding the type of sequences, for $\Delta = 0.525$, again the best detection accuracy is achieved with ZC sequences. Both DMLP and CNN algorithms present similar results using such sequences. Next, the Bernoulli sequences outperform marginally the normal ones. Especially, the CNN results in significant improvements using the ZC sequences set, achieving detection error rates close to the DMLP. Lastly, when the undersampling ratio is reduced substantially, e.g., $\Delta = 0.175$, the performance of the DMLP and the CNN algorithms operating under each one of the three types of sequences is tight for $p_a \geq 0.1$.

E. Complexity of the DL and Baseline Algorithms

Fig. 11 depicts the running time in seconds versus the number of devices of the evaluated algorithms for two values of undersampling ratio. This metric accounts only for the time required...
by the algorithms to estimate the set of active devices, since the neural networks are trained before the system deployment, having no impact on the DL algorithms complexity. In this result, we set \((K, L) \in \{(40, 7), (40, 21), (400, 71), (400, 211)\}\). The DL algorithms present extremely low running time values on the order of 10 to 100 \(\mu s\). The DMLP algorithm has slightly shorter running time than the CNN. Compared with the baseline algorithms, the proposed algorithms result in running time values at least two orders of magnitude less. As expected, the complexity of all the algorithms increases with the number of devices, owing to the growth in the number of inputs. Despite that, the running time changes marginally with the undersampling ratio, except for the LASSO method. Such fact is due to the accelerated convergence of the CD algorithm caused by the amount of information provided by longer preamble sequences.

V. CONCLUSION

In this article, we propose two DL sparse support recovery algorithms to detect active devices in GF-RA protocols. We develop a DMLP algorithm for device activity detection based on densely connected layers and a CNN algorithm built with 1-D convolution layers. At the same time, we analyze the impact of the type of sequences used for preamble design on the activity detection accuracy. We evaluate the accuracy achieved with random preambles generated by a complex normal and a Bernoulli distributions, as well as by deterministic ZC sequences. The numerical results demonstrate that the DMLP reaches the best detection error rate values among the evaluated algorithms. Despite that, the CNN with the ZC preamble sequences in place of random ones achieves detection error rates comparable to the DMLP, due to the good correlation properties of these sequences and the structure of the CNN that exploits the correlation properties among the network inputs. On the matter of the computational complexity, both proposed DL algorithms present extremely low running time, at the expense of the training burden to determine the suitable parameter values of the networks. Moreover, both the proposed DL activity detection algorithms attain promising and much better performance-complexity tradeoff than state-of-the-art techniques such as the LASSO method and the AMP algorithm.

REFERENCES

[1] J. Ding, D. Qu, and J. Choi, “Analysis of non-orthogonal sequences for grant-free RA with massive MIMO,” IEEE Trans. Commun., vol. 68, no. 1, pp. 150–160, Oct. 2020.
[2] X. Chen, D. W. K. Ng, W. Yu, E. G. Larsson, N. Al-Dhahir, and R. Schober, “Massive access for 5G and beyond,” IEEE J. Sel. Areas Commun., vol. 39, no. 3, pp. 615–637, Mar. 2021.
[3] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning, Cambridge, MA, USA: MIT Press, 2016.
[4] S. Li, W. Zhang, Y. Cui, H. V. Cheng, and W. Yu, “Joint design of measurement matrix and sparse support recovery method via deep auto-encoder,” IEEE Signal Process. Lett., vol. 26, no. 12, pp. 1778–1782, Oct. 2019.
[5] W. Kim, Y. Ahn, and B. Shim, “Deep neural network-based active user detection for grant-free NOMA systems,” IEEE Trans. Commun., vol. 68, no. 4, pp. 2143–2155, Jan. 2020.
[6] N. Kim, D. Kim, B. Shim, and K. B. Lee, “Deep learning-based spreading sequence design and active user detection for massive machine-type communications,” IEEE Wireless Commun. Lett., vol. 10, no. 8, pp. 1618–1622, Jun. 2021.
[7] X. Wu, S. Zhang, and J. Yan, “A CNN architecture for learning device activity from MMV,” IEEE Commun. Lett., vol. 25, no. 9, pp. 2933–2937, Jun. 2021.
[8] N. Shlezinger, N. Farsad, Y. C. Eldar, and A. J. Goldsmith, “Model-based machine learning for communications,” 2021, arXiv:2101.04726.
[9] M. Borgerding, P. Schniter, and S. Rangan, “AMP-inspired deep networks for sparse linear inverse problems,” IEEE Trans. Signal Process., vol. 65, no. 16, pp. 4293–4308, May 2017.
[10] Y. Qiang, X. Shao, and X. Chen, “A model-driven deep learning algorithm for joint activity detection and channel estimation,” IEEE Commun. Lett., vol. 24, no. 11, pp. 2508–2512, Jul. 2020.
[11] Y. Shi, S. Xia, Y. Zhou, and Y. Shi, “Sparse signal processing for massive device connectivity via deep learning,” in Proc. IEEE Int. Conf. Commun. Workshops, 2020, pp. 1–6.

[12] W. Chen, B. Zhang, S. Jin, B. Ai, and Z. Zhong, “Solving sparse linear inverse problems in communication systems: A deep learning approach with adaptive depth,” IEEE J. Sel. Areas Commun., vol. 39, no. 1, pp. 4–17, Jan. 2021.

[13] Y. Cui, S. Li, and W. Zhang, “Jointly sparse signal recovery and support recovery via deep learning with applications in MIMO-based grant-free random access,” IEEE J. Sel. Areas Commun., vol. 39, no. 3, pp. 788–803, Aug. 2021.

[14] X. Shao, X. Chen, Y. Qiang, C. Zhong, and Z. Zhang, “Feature-aided adaptive-tuning deep learning for massive device detection,” IEEE J. Sel. Areas Commun., vol. 39, no. 7, pp. 1899–1914, May 2021.

[15] Y. Shi, H. Choi, Y. Shi, and Y. Zhou, “Algorithm unrolling for massive access via deep neural network with theoretical guarantee,” IEEE Trans. Wireless Commun., vol. 21, no. 2, pp. 945–959, Feb. 2022, doi: 10.1109/TWC.2021.3100500.

[16] K. Gregor and Y. LeCun, “Learning fast approximations of sparse coding,” in Proc. 27th Int. Conf. Int. Conf. Mach. Learn., 2010, pp. 399–406.

[17] L. Liu and W. Yu, “Massive connectivity with massive MIMO—Part I: Device activity detection and channel estimation,” IEEE Trans. Signal Process., vol. 66, no. 11, pp. 2933–2946, Mar. 2018.

[18] K. Senel and E. G. Larsson, “Grant-free massive MTC-enabled massive MIMO: A compressive sensing approach,” IEEE Trans. Commun., vol. 66, no. 12, pp. 6164–6175, Aug. 2018.

[19] A. Fengler, S. Haghifatshoar, P. Jung, and G. Caire, “Non-Bayesian activity detection, large-scale fading coefficient estimation, and unsourced random access with a massive MIMO receiver,” IEEE Trans. Inf. Theory, vol. 67, no. 5, pp. 2925–2951, Mar. 2021.

[20] J. Che, Z. Zhang, Z. Yang, X. Chen, C. Zhong, and D. W. K. Ng, “Unsourced random massive access with beam-space tree decoding,” IEEE J. Sel. Areas Commun., vol. 40, no. 4, pp. 1146–1161, Apr. 2022, doi: 10.1109/JSAC.2022.3143241.

[21] M. Guo and M. C. Gursoy, “Joint activity detection and channel estimation in cell-free massive MIMO networks with massive connectivity,” IEEE Trans. Commun., vol. 70, no. 1, pp. 317–331, Jan. 2022.

[22] Z. Zhang et al., “User activity detection and channel estimation for grant-free random access in LEO satellite-enabled Internet of Things,” IEEE Internet Things J., vol. 7, no. 9, pp. 8811–8825, May 2020.

[23] L. Zhen, H. Qin, B. Song, R. Ding, X. Du, and M. Guizani, “Random access preamble design and detection for mobile satellite communication systems,” IEEE J. Sel. Areas Commun., vol. 36, no. 2, pp. 280–291, Feb. 2018.

[24] A. Mousavi and R. G. Baraniuk, “Learning to invert: Signal recovery via deep convolutional networks,” in Proc. IEEE Int. Conf. Acoust., Speech Signal Process., 2017, pp. 2272–2276.

[25] D. Chu, “Polyphase codes with good periodic correlation properties,” IEEE Trans. Inf. Theory, vol. 18, no. 4, pp. 531–532, Jul. 1972.

[26] J. A. Tropp and S. J. Wright, “Computational methods for sparse solution of linear inverse problems,” Proc. IEEE Proc. IRE, vol. 98, no. 6, pp. 948–958, Apr. 2010.

[27] Y. C. Eldar and G. Kutyniok, Compressed Sensing: Theory and Applications. Cambridge, U.K.: Cambridge Univ. Press, 2012.

[28] B. Popović, “Generalized chirp-like polyphase sequences with optimum correlation properties,” IEEE Trans. Inf. Theory, vol. 38, no. 4, pp. 1406–1409, Jul. 1992.

[29] K. P. Murphy, Machine Learning: A Probabilistic Perspective. Cambridge, MA, USA: MIT Press, 2012.

[30] R. Tibshirani, “Regression shrinkage and selection via the lasso,” J. Roy. Statist. Soc.: Ser. B (Methodological), vol. 58, no. 1, pp. 267–288, Jan. 1996.

[31] D. L. Donoho, A. Maleki, and A. Montanari, “Message-passing algorithms for compressed sensing,” Proc. Nat. Acad. Sci., vol. 106, no. 45, pp. 18914–18919, Sep. 2009.

[32] A. Maleki, L. Anitori, Z. Yang, and R. G. Baraniuk, “Asymptotic analysis of complex LASSO via complex approximate message passing (CAMP),” IEEE Trans. Inf. Theory, vol. 59, no. 7, pp. 4290–4308, Mar. 2013.

[33] S. J. Wright, “Coordinate descent algorithms,” Math. Program., vol. 151, no. 1, pp. 3–34, Mar. 2015.