Deep Learning-based Spreading Sequence Design and Active User Detection for Massive Machine-Type Communications

Namik Kim, Dongwoo Kim, Byonghyo Shim, and Kwang Bok Lee

Abstract

In this letter, we propose a deep learning-based spreading sequence design and active user detection (AUD) to support massive machine-type communications (mMTC) where a large number of devices access the base station using non-orthogonal spreading sequences. To design the whole communications system minimizing AUD error, we employ an end-to-end deep neural network (DNN) where the spreading network models the transmitter side and the AUD network estimates active devices. By using the AUD error as a loss function, network parameters including the spreading sequences are learned to minimize the AUD error. Numerical results reveal that the spreading sequences obtained from the proposed approach achieve higher AUD performance than the conventional spreading sequences in the compressive sensing-based AUD schemes, as well as in the proposed AUD scheme.

Index Terms

Massive machine-type communications, deep neural network, non-orthogonal multiple access, spreading sequences, active user detection.

I. INTRODUCTION

Due to the wide variety of Internet-of-Things (IoT) applications, massive machine-type communications (mMTC) has received a great deal of attention in recent years [1]. mMTC mainly

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Ministry of Education (MOE) (2018R1D1A1A02085356) and the Ministry of Science and ICT (MSIT) (2020R1A2C2102198).

N. Kim, B. Shim, and K. B. Lee are with the Department of Electrical and Computer Engineering and the Institute of New Media and Communications, Seoul National University, Seoul 08826, Korea (e-mail: {kinnimcl, bshim, klee}@snu.ac.kr).

D. Kim is with Samsung Electronics Co., Ltd., Suwon 16677, Korea (e-mail: darius.kim@samsung.com).
concerns the massive connectivity of a large number of devices (e.g., sensors, robots, vehicles, and machines) to the base station (BS). Since it is difficult to support mMTC in the conventional scheduling-based orthogonal multiple access (OMA) due to the lack of resources and the heavy signaling overhead, grant-free non-orthogonal multiple access (GF-NOMA) has received special attention in recent years [2]. Using non-orthogonal spreading sequences, GF-NOMA allows devices to transmit data without complicated scheduling. In order to support the grant-free transmission, the BS needs to identify active devices, i.e., devices transmitting data, among all potential devices. This process, often called active user detection (AUD), is an essential step for the successful data detection in GF-NOMA [3].

Since only a small portion of machine-type devices is active at a time (see Fig. 1), a transmit vector consisting of data symbols of all devices is readily modeled as a sparse vector [4]. In solving the AUD problem, compressive sensing (CS) techniques have been widely used [3], [5], [6]. Over the years, various techniques based on the CS-based AUD schemes have been proposed [3], [5], [6]. In these approaches, basically, the correlation between the received signal and the spreading sequence of each device is used to identify active devices. That is, a device whose spreading sequence is maximally correlated with the received signal is chosen in each iteration. Since the detection performance depends heavily on the correlation among spreading sequences [6], selection of the spreading sequences with low-correlation structure is crucial to improve the AUD performance.

Recently, some efforts have been made to design the low-correlation spreading sequences. In [7] and [8], iterative algorithms to find out the collection of spreading sequences having small correlation structure have been proposed. While these approaches improve the detection
performance to some extent, they do not consider the mMTC context (e.g., the number of devices, channel models, and activities of devices) so that the performance gain in practical mMTC environments is marginal.

An aim of this letter is to propose a deep learning-based communications system suited for mMTC. The key idea of the proposed scheme is to use an end-to-end deep neural network (DNN) minimizing the AUD error [9]. In a nutshell, we divide the proposed DNN into two parts: the spreading network (SN) at the transmitter side (devices) to model device activities, channels, and symbol spreading and the AUD network (AUDN) at the receiver side (BS) to estimate active devices. Due to the use of the end-to-end network from the transmitter all the way to the receiver, we can directly use the AUD error as a loss function in the entire network training. As a result of the training, we can directly obtain the spreading sequences (training parameters in the SN) minimizing the AUD error.

From the numerical evaluations in grant-free mMTC scenarios, we show that the spreading sequences obtained from the proposed scheme perform better than those of the conventional scheme minimizing mutual coherence in terms of the AUD error probability. We also show that the spreading sequences obtained from the proposed scheme can be applied to the conventional CS-based AUD schemes.

II. SYSTEM MODEL

We consider the uplink transmission of the mMTC system where $N$ devices access a single BS using spreading sequences of the length $M$ (see Fig. 1). We assume that the BS and all devices are equipped with one antenna and channels experience the flat fading. In most mMTC scenarios, the number of devices is larger than the length of spreading sequences (i.e., $M < N$) so that it is in general not possible to recover the transmit vector with the conventional recovery algorithm designed for overdetermined scenarios [6].

Let $x_n$ and $s_n \in \mathbb{R}^{M \times 1}$ be the symbol and spreading sequence of the $n$-th device, respectively. Also, let $\delta \in \mathbb{R}^{N \times 1}$ be the activity indicator vector ($\delta_n$ is 0 for an inactive device and 1 for an active device). The indicator $\delta_n$ of the $n$-th device follows the Bernoulli distribution:

$$\delta_n \sim \text{Bern}(p_n),$$

(1)
where \( p_n \) is the activity probability of the \( n \)-th device. Then, the received signal at the BS is given by

\[
y = \sum_{n=1}^{N} s_n h_n \delta_n x_n + w
\]

\[
= S q + w,
\]

(2)

where \( h_n \) is the channel between the \( n \)-th device and the BS, \( S = [s_1 \cdots s_N] \in \mathbb{R}^{M \times N} \) is the spreading matrix, \( q = [h_1 \delta_1 x_1 \cdots h_N \delta_N x_N] \) is the composite vector of the symbols and the channels, and \( w \sim \mathcal{N}(0, \sigma^2 I) \) is the additive white Gaussian noise (AWGN) vector. In this work, we consider the real-valued spreading sequences to make a fair comparison with the previous study using real values [8] but the extension to the complex scenarios is straightforward.

If the number of active devices, called sparsity, is \( K \), then the AUD problem can be formulated as [10]

\[
\tilde{\delta} = \text{arg} \min_{\|\delta\|_0 = K} \| y - Sq \|_2^2.
\]

(3)

To solve this problem, greedy sparse recovery algorithm such as the orthogonal matching pursuit (OMP) has been employed [6], [11]. Since the AUD performance of the greedy algorithm relies heavily on the correlation among spreading sequences, previous studies focused on the minimization of the correlation between columns in the spreading matrix [7], [8]. In [8], for example, an approach to find out the quasi-orthogonal spreading sequences minimizing the mutual coherence has been proposed:

\[
\min_{S} \max_{1 \leq i \neq j \leq N} \frac{|\langle s_i, s_j \rangle|}{\|s_i\|_2 \|s_j\|_2} \text{ and } \min_{S} \| I - S^T S \|_F^2,
\]

(4)

where \( \| \cdot \|_F \) is the Frobenius norm. It has been shown that these approaches are effective in reducing the correlation among all spreading sequences [8]. However, since the portion of active devices is very small (less than 10%) [12], it would be more effective to consider the device activities in the spreading sequence design. If we somehow minimize the correlation of spreading sequences for frequently active devices, multi-user interference can be reduced significantly, thereby achieving an improvement in the AUD performance.

### III. DNN-based Spreading Sequence Design and Active User Detection

In order to design the end-to-end system (transmitter and receiver) minimizing AUD error, we employ the autoencoder-based DNN [9]. For the network training, we use the binary cross-
entropy loss of AUD [13]:
\[
L(\delta, \hat{\delta}) = - \sum_{n=1}^{N} \left( \delta_n \log(\hat{\delta}_n) + (1 - \delta_n) \log(1 - \hat{\delta}_n) \right),
\]
where \(\hat{\delta}\) is the estimated activity indicator vector and \(\delta_n\) and \(\hat{\delta}_n\) are the element of \(\delta\) and \(\hat{\delta}\), respectively. Using the backpropagation mechanism, we can propagate the loss (the AUD error) all the way back to the transmitter and hence train the whole network parameters.

The basic structure of the proposed end-to-end network is illustrated in Fig. 2. In essence, the proposed network consists of two subnetworks (SN and AUDN). The SN models the transmitter side (symbol spreading, device activities, and channels) and the AUDN estimates the activity indicator vector from the received signal. The estimated activity indicator vector generated from the end-to-end network is
\[
\hat{\delta} = g(x; \Theta),
\]
where \(x = [x_1 \cdots x_N]\) is the input symbol vector, \(g\) is the mapping function between the input \(x\) and the output \(\hat{\delta}\) of the network, and \(\Theta\) is the set of all network parameters including the spreading sequences. Note that parameters in \(\Theta\) are updated during the training phase using the stochastic gradient descent (SGD) algorithm:
\[
\Theta_j = \Theta_{j-1} - \eta \nabla_\Theta L(\Theta_{j-1}),
\]
where \(\Theta_j\) is the parameters in the \(j\)-th training iteration, \(\eta\) is the learning rate, and \(\nabla_\Theta L(\cdot)\) is the gradient of the loss function \(L\) [9]. After the training phase, the trained parameters in the SN are used as the spreading sequences (see Fig. 2).

A. SN Architecture

In Fig. 3, we depict the SN structure in the transmitter. Instead of using the deterministic spreading sequences, we assign the trainable vector \(s'_n \in \Theta\) to the \(n\)-th device \((n = 1, \cdots, N)\).
Hence, the spreading sequences are a part of the network parameters to be updated by the SGD algorithm. To model the device activities, the symbol $x_n$ is multiplied by the activity indicator $\delta_n$ which follows the Bernoulli distribution, i.e. $\delta_n \sim \text{Bern}(p_n)$. 
B. AUDN Architecture

The main goal of the AUDN is to identify the activity indicator vector $\delta$ from $y$. Fig. 4 depicts the structure of the AUDN. Because only a small number of machine-type devices are active at a time, the activity indicator vector can be well modeled as a sparse vector [4]. In solving this sparse recovery problem, we employ an iterative hard thresholding-based network (IHT-Net), a DNN-based sparse recovery algorithm based on iterative hard thresholding (IHT) [14]:

$$\min_{\|q\|_0=K} \|y - Sq\|_2^2. \quad (8)$$

In the IHT algorithm, the sparse vector $q$ is iteratively updated as

$$q^{(t+1)} = H_K[(I_N - S^T S)q^{(t)} + S^T y], \quad (9)$$

where $q^{(t)}$ is the estimate after $t$ iterations and $H_K[\cdot]$ is the hard thresholding operator to enforce the sparsity $K$ of the output vector. Similar to the IHT algorithm, one iteration in the IHT-Net is mapped to the neural layer of a DNN. Specifically, the $t$-th layer of the IHT-Net can be expressed as

$$\delta^{(t+1)} = \text{ReLU} [\Psi^{(t)} \delta^{(t)} + \beta^{(t)}], \quad (10)$$

where $\delta^{(t)}$ and $\delta^{(t+1)}$ are the input and the output of the layer and $\Psi^{(t)} \in \Theta$ and $\beta^{(t)} \in \Theta$ are the trainable weight and bias of the layer. Note that the hard thresholding operator in the IHT is replaced by the rectified linear unit (ReLU).

In order to improve the accuracy of AUD when a massive number of devices are used, we adjust the number of nodes based on the number of devices. It is well-known from the universal approximation theorem that a DNN can approximate the desired function, provided that sufficiently many hidden nodes are available [15]. This implies that the AUDN with enough nodes can carry out the accurate AUD process even when the number of devices is large. In light of this, we set the width of layers in proportion to the number of devices $N$ (e.g., $\delta^{(t)} \in \mathbb{R}^{5N \times 1}$).

In this work, we exploit the batch normalization (BN) and residual networks (ResNets) to handle the problem occurring in the training phase [16], [17]. Since devices have different activities in mMTC, a variation of the received signal (the sum of all device signals) is large, which slows down the training speed and also hinders convergence because the network should handle a large variation of input data. In order to mitigate this, we add the BN layer to the data
\[ \dot{\delta}^{(t)} = \Psi^{(t)} \delta^{(t)} + \beta^{(t)} = [\dot{\delta}_1^{(t)} \ldots \dot{\delta}_{5N}^{(t)}] \] so that the normalized data \( \dot{\delta}_1^{(t)}, \ldots, \dot{\delta}_{5N}^{(t)} \) have zero means and unit variances for each batch size \( B \):

\[ \dot{\delta}_i^{(t)} = \frac{\dot{\delta}_i^{(t)} - \mu_{B,i}}{\sigma_{B,i}}, \quad \text{for } i = 1, \ldots, 5N, \]

(11)

where \( \dot{\delta}^{(t)} = [\dot{\delta}_1^{(t)} \ldots \dot{\delta}_{5N}^{(t)}] \) is the output of the BN layer and \( \mu_{B,i} = \frac{1}{B} \sum_{b=1}^{B} \dot{\delta}_i^{(t)[b]} \) and \( \sigma_{B,i} = \frac{1}{B} \sum_{b=1}^{B} (\dot{\delta}_i^{(t)[b]} - \mu_{B,i})^2 \) are the mini-batch mean and variance. Since a large number of layers are used in the AUDN, the entire network suffers from the vanishing gradient problem. Note that as the gradient propagates backward all the way to the SN, new gradient obtained by multiplying local gradients gets smaller, making it difficult to update the weight of early layers (i.e., layers of the SN). In order to alleviate this so-called vanishing gradient problem, we exploit the ResNet architecture that adds the direct links between stacked layers. The \( t \)-th layer in the ResNet is given by

\[ \delta^{(t+1)} = \text{ReLU}[\dot{\delta}^{(t)} + \delta^{(t-1)}]. \]

(12)

Since the layers are directly connected by the ResNet (see Fig. 4), the gradients can be transferred across the layers with much less distortion, resulting in an improvement in the training accuracy.

After the training phase is completed, the estimated activity indicator \( \hat{\delta}_n \) is mapped to the unit interval \((0, 1)\) using the sigmoid activation \( f(x) = \frac{1}{1 + e^{-x}} \). Finally, if \( \hat{\delta}_n \) is greater than the threshold \( \alpha \), we declare the \( n \)-th device as an active device. The threshold value \( \alpha \) minimizing the AUD error is given by

\[ \alpha^* = \arg \min_{\alpha} \left( \left| \{(b, n) \mid \hat{\delta}_n^{[b]} \leq \alpha \text{ if } \delta^{[b]}_n = 1 \} \right| + \left| \{(b, n) \mid \hat{\delta}_n^{[b]} > \alpha \text{ if } \delta^{[b]}_n = 0 \} \right| \right), \]

(13)

where \( b \) is the data index [18].

C. Comments on Complexity

The AUDN performs the matrix-vector multiplication \( O(N^2) \) in each hidden layer, resulting in the complexity of \( O(LN^2) \) where \( L \) is the number of hidden layers. The proposed spreading sequences are obtained after the training process of the entire network of \( O(LN^2) \) complexity; thus, the total complexity is \( O(LN_{\text{train}}N^2) \) where \( N_{\text{train}} \) is the number of training iterations.
IV. Numerical Results

A. Simulation Setup

We simulate the uplink mMTC system with 64 devices ($N = 64$). The length of spreading sequences is set to $M = 32$ and each spreading sequence $s_n$ is normalized (i.e., $\|s_n\|_2 = 1$). An additive white Gaussian noise (AWGN) channel model with the noise variance $\sigma^2$ is assumed for the sake of simplicity. The average symbol SNR is set to $1/\sigma^2$. We test two scenarios: activity probabilities are the same and different. We henceforth call these cases homogeneous and heterogeneous activities, respectively.

For comparison, we employ the conventional spreading sequences in [8] and the Gaussian random spreading sequences in which the elements follow the normal distribution $\mathcal{N}(0, 1)$. To evaluate the AUD performance, AUDN is used at the receiver and is trained with the same training data for each set of spreading sequences. As an AUD performance measure, we use the activity error rate (AER) considering both missed detection and false alarms: $\text{AER} = 1 - \frac{|\Omega \cap \hat{\Omega}|}{|\Omega \cup \hat{\Omega}|}$ where $\Omega$ and $\hat{\Omega}$ are the support\(^\dagger\) of $\delta$ and $\hat{\delta}$, respectively.

Data samples are constructed by combining symbols and activity indicators determined by the activity probabilities (see Fig. 3). In the training phase, data samples with SNR in the range of 15 dB to 20 dB are used. We use 480,000 samples with 40 epochs for training, 60,000 samples for validation, and 60,000 samples for testing [14]. We employ an Adam optimizer for the SGD optimization. In the simulations, the number of hidden layers, the batch size, and the threshold value are set to $L = 10$, $B = 200$, and $\alpha = 0.4$, respectively. The learning rate $\eta$ starts from 0.01 and is divided by 10 for every 10 epochs.

B. Homogeneous Activities

In the homogeneous activity scenario, the activity probabilities for all devices are the same. Fig. 5 presents the average AER with different spreading sequences as a function of $p_n$ at SNR = 20 dB. The average AER of the Gaussian random sequences is much larger due to the high correlation among the spreading sequences. On the other hand, the average AER of the proposed sequences is lower than that of the conventional sequences for each activity probability $p_n$.

\(^\dagger\)If $\delta = [1 \ 0 \ 1 \ 0]$, then the support is $\Omega = \{1, 3\}$. 

C. Heterogeneous Activities

In this section, we examine whether the proposed approach can generate activity-specific spreading sequences and thus improve the AUD performance when the activity probabilities are different.

In Fig. 6, we evaluate the average AER of the AUDN as a function of SNR. The activity probability \( p_n \) is modeled by the uniform distribution on the interval of \([0.01, 0.2]\) for scenario 1 and \([0.01, 0.1]\) for scenario 2. We observe that the performance of the proposed sequences is significantly better than that of the conventional sequences. For example, the proposed sequences achieve about 3 dB gain in scenario 1 and 0.5 dB gain in scenario 2 over the conventional sequences at high SNR.
In Fig. 7, we plot the average cross-correlation of each device as a function of $p_n$. The average cross-correlation of the $i$-th device is defined as $\mu_{i,\text{avg}} = \frac{1}{N-1} \sum_{j=1, j \neq i}^{N} |\langle s_i, s_j \rangle|$. If $\mu_{i,\text{avg}}$ is small, the spreading sequence of the $i$-th device is less correlated with other spreading sequences. In terms of the AUD performance, it is desirable to have the spreading sequences such that the cross-correlation between any two active devices is as small as possible [8]. We observe that the proposed approach assigns the spreading sequences with lower $\mu_{i,\text{avg}}$ to the devices with higher activity probabilities. Since the average cross-correlation of devices with higher activity probabilities is reduced, active devices are less correlated, thereby improving the performance of the proposed approach in Fig. 6.

Fig. 8 depicts the average AER of the OMP and compressive sampling matching pursuit
(CoSaMP) in the heterogeneous activity scenario [19]. When compared to Fig. 6, we observe that the AUDN outperforms the greedy algorithms by a large margin (e.g. more than 4 dB gain for scenario 1). Further, the performance gap between the conventional and the proposed sequences becomes more severe in a wide range of SNR because the greedy algorithms depend heavily on the correlation among spreading sequences. In the CoSaMP, for example, the proposed sequences achieve about 2.9 dB gain in scenario 1 and 1.3 dB gain in scenario 2 over the conventional sequences.

V. CONCLUSION

In this letter, we proposed a deep learning-based spreading sequence design and AUD scheme for an mMTC system. To design the communications system minimizing AUD error, we employed an end-to-end DNN. By properly training the whole network, we can obtain the spreading sequences and the AUD scheme optimized for mMTC environments. Numerical results demonstrated that the AUD performance of the proposed scheme is significantly better than that of the conventional schemes in the heterogeneous activity scenario. Further, we observed that the spreading sequences obtained from the proposed end-to-end DNN can improve the AUD performance even when we use the conventional greedy algorithms.

REFERENCES


