RL-NCS: REINFORCEMENT LEARNING BASED DATA-DRIVEN APPROACH FOR NONUNIFORM COMPRESSED SENSING

Nazmul Karim, Alireza Zaeemzadeh, and Nazanin Rahnavard
School of Electrical and Computer Engineering
University of Central Florida
Emails: nazmul.karim18@knights.ucf.edu,{zaeemzadeh, nazanin} @eecs.ucf.edu

ABSTRACT

A reinforcement-learning-based non-uniform compressed sensing (NCS) framework for time-varying signals is introduced. The proposed scheme, referred to as RL-NCS, aims to boost the performance of signal recovery through an optimal and adaptive distribution of sensing energy among two groups of coefficients of the signal, referred to as region of interest (ROI) coefficients and non-ROI coefficients. The coefficients in ROI usually have greater importance and need to be reconstructed with higher accuracy compared to non-ROI coefficients. In order to accomplish this task, the ROI is predicted at each time-step using two specific approaches. One of these approaches incorporates a long short-term memory (LSTM) network for the prediction. The other approach employs the previous ROI information for predicting the next step ROI. Using the *exploration-exploitation* technique, a Q-network learns to choose the best approach for designing the measurement matrix. Furthermore, a joint loss function is introduced for the efficient training of the Q-network as well as the LSTM network. The result indicates a significant performance gain for our proposed method, even for rapidly varying signals and reduced number of measurements.

Index Terms-Compressed Sensing, Reinforcement Learning, LSTM, DQN, Replay Memory, Region of Interest

1. INTRODUCTION

The compressed sensing (CS) [1, 2] framework aims to recover sparse signals by acquiring significantly fewer number of measurements compared to the classical Nyquist rate. Consider a CS problem with the objective of reconstructing a time series $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_t, \ldots\}$, where $\boldsymbol{x}_t$ is the signal at time step t, from an under-sampled time series of linear measurements $\{\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_t, \ldots\}$, where $\boldsymbol{y}_t = \boldsymbol{\Phi}_t \boldsymbol{x}_t + \boldsymbol{n}_t$. Here, $\boldsymbol{\Phi}_t \in \mathbb{R}^{M \times N}$ is the measurement matrix and $\boldsymbol{n}_t$ represents the noise that corrupts our measurements at time t. Due to the time-varying nature of our setup, a fixed measurement matrix can deteriorate the reconstruction performance. Therefore, our aim is to recover the signal $\boldsymbol{x}_t$ from $\boldsymbol{y}_t$ incorporating a non-uniform sensing strategy attainable by an *adaptive* design of $\boldsymbol{\Phi}_t$.

This material is based upon work supported by the National Science Foundation under Grant No. ECCS-1810256 and CCF-1718195.

In many applications, such as dynamic MRI [3], high speed video streaming [4], and wireless sensor networks [5], it is desirable to recover signal that contains coefficients with different levels of importance. For example, a certain area or segment might be more important and informative than the rest of the image. In most cases, the signal coefficients in the *region of interest* (ROI) needs to be reconstructed more accurately. To achieve this, it is required to sample these coefficients with more sensing energy utilizing a non-uniform CS strategy [6, 7]. However, if the ROI changes over time, this task gets more complicated and reconstruction using fewer measurements becomes erroneous. In this scenario, utilizing the knowledge of previous estimations, we can predict the ROI and design the $\boldsymbol{\Phi}$ accordingly. Hence, a method to make a prediction about the ROI and adaptively design the measurement matrix is introduced in this paper.

We propose a *reinforcement learning (RL) based adaptive CS technique* that focuses on designing non-uniform measurement matrices for time-varying signals. The objective is to implement a non-uniform sampling method to boost the recovery performance that results in small reconstruction error. The technique of reinforcement learning [8] has been used in many aspects of wireless sensor networks [9, 10]. Especially, when a part of the system, an agent, interacts with another dynamic part of the system, known as environment, to come up with the optimal policy that serves the purpose of that system. The decision making ability of an agent based on the past experiences is the key part of an RL. In our work, we have formulated two mechanisms (actions) to predict the next step ROI. At each step, the task of the RL is to choose the best mechanism for deigning the measurement matrix.

Unlike adaptive CS [11, 12], the signals that we recover here are not static over time. We also take a different approach than dynamic CS methods [13, 14] that focus only on the reconstruction phase. Furthermore, there have been studies that capitalize the idea of utilization of knowledge from prior estimations and they seem to work only if the signal ROI changes very slowly [15,16]. However, our focus is to obtain adaptive design of CS measurement matrices that show effectiveness even for rapidly varying signals. In our work, an agent has the flexibility to choose between two mechanisms compared

978-1-7281-0824-7/19/$31.00 ©2019 IEEE

to only a single mechanism in other works. Furthermore, we propose a multi-task training procedure for tuning two neural networks, the Q-network and the long short-term memory (LSTM) [17] network. Although a readily available dataset is required to train an LSTM network, we devised an efficient way to tackle this challenge utilizing the experiences stored in the replay memory of the Q-network. In the end, we carried out experiments to prove the superior performing ability of our method in contrast to uniform CS method as well as other techniques.

The organization of rest of the paper is as follows. System model and problem formulation have been discussed in Section 3. In Section 3, we talk about reinforcement learning and Q-learning. A detailed discussion about the approach of our proposed framework has been presented in Section 4. Then, we describe the experimental setup with simulation results in Section 5 and conclude with the discussion in Section 6.

2. SYSTEM MODEL

In compressed sensing, it has been proven that a compressible signal $\boldsymbol{x}_t \in \mathbb{R}^N$ is recoverable from relatively fewer random projections, $\boldsymbol{y}_t \in \mathbb{R}^M$. Consider a scenario where we have a vector-valued time-series that consists of compressible signals $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_t, \ldots\}$. Therefore, the set of measurement vectors can be obtained as

$$\boldsymbol{y}_t = \boldsymbol{\Phi}_t \boldsymbol{x}_t + \boldsymbol{n}_t, \quad \text{for t} \geq 1, \tag{1}$$

where $\boldsymbol{\Phi}_t \in \mathbb{R}^{M \times N}$ is employed for the mapping of signals to linear measurements at each time-step t. The noise term $\boldsymbol{n}_t \in \mathbb{R}^M$ is the additive white Gaussian noise (AWGN) with $\boldsymbol{n}_t \sim \mathcal{N}(\boldsymbol{0}, \sigma_n^2 \boldsymbol{I}_M)$, where $\boldsymbol{I}_M$ is an all-one vector. It is assumed that all the coefficients of signal $\boldsymbol{x}_t$ are not equally important. Therefore, it is necessary to identify and predict the *region of interest* (ROI) that contains the most important coefficients. For this, we aim to predict the ROI instantly based on the knowledge of the signal history. This task of instantaneous decision-making about the position of next ROI can be accomplished with reinforcement learning. After that, it is straightforward to distribute the sensing energy according to the importance levels of the coefficients, such that the coefficients in ROI are being sampled with more sensing energy.

3. REINFORCEMENT LEARNING

In reinforcement learning (RL), an agent tries to learn the optimal policy for a sequential decision-making problem through the optimization of cumulative future reward signal. For solving sequential decision-making problems, it is necessary to estimate the value of each decision or action. The action-value function defines how good an action, a, is for the agent to take if it is in a state, $\boldsymbol{s}$. In general, a policy $\pi : \boldsymbol{S} \rightarrow \boldsymbol{A}$ dictates the agent which action it needs to take, from the action space $\boldsymbol{A}$, based on the current state. Given the policy π, the action value function for a state-action pair $(\boldsymbol{s}, a)$ can be determined by

$$Q_\pi(\boldsymbol{s}, a) = \mathrm{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t r_t(\boldsymbol{s}_t, a_t) | \boldsymbol{s}_0 = \boldsymbol{s}, a_0 = a, \pi \right]. \tag{2}$$

Here, r_t is the reward at time t and the discount factor $\gamma \in [0, 1]$ decides how important the immediate reward is compared to the future rewards. Therefore, Q_π is the expected sum of future rewards for a state-action pair $(\boldsymbol{s}, a)$, also known as Q-function. The optimal policy, π^* is attainable by choosing the action that gives the optimal action-value, $Q^*(\boldsymbol{s}, a) = \max_\pi Q_\pi(\boldsymbol{s}, a)$ in each state.

The task of Q-learning [18] can be accomplished by a multi-layered neural network, also known as deep Q-network (DQN). For a given state $\boldsymbol{s}_t$, the output of DQN is a vector of action values $\boldsymbol{Q}(\boldsymbol{s}_t, a; \boldsymbol{\theta}_t)$. The two key elements required for tuning the Q-network are (i) *replay memory* which stores observed transitions (experiences) $(\boldsymbol{s}_t, a_t, \boldsymbol{s}_{t+1}, r_{t+1})$ for a later use, and (ii) a *target network* that generates target, β_t^Q, for the DQN (online network) [19]. The target network is the same as the online network except with static parameters $\boldsymbol{\theta}_t^-$, a periodically copied version of $\boldsymbol{\theta}_t$ (every τ steps). The target for online network stands as

$$\beta_t^Q = r_{t+1} + \gamma \max_{a \in A} \boldsymbol{Q}(\boldsymbol{s}_{t+1}, a; \boldsymbol{\theta}_t^-), \tag{3}$$

where $Q(\boldsymbol{s}_{t+1}, a; \boldsymbol{\theta}_t^-)$ is the action values generated by the target network. The DQN loss function is given by

$$J_t^{(DQN)} = (\beta_t^Q - Q(\boldsymbol{s}_t, a_t; \boldsymbol{\theta}_t))^2. \tag{4}$$

The objective of the deep Q-network is to come up with the optimal policy through the minimization of this loss function. In most applications, it is customary for an agent to take the approach of *exploration and exploitation* [20].

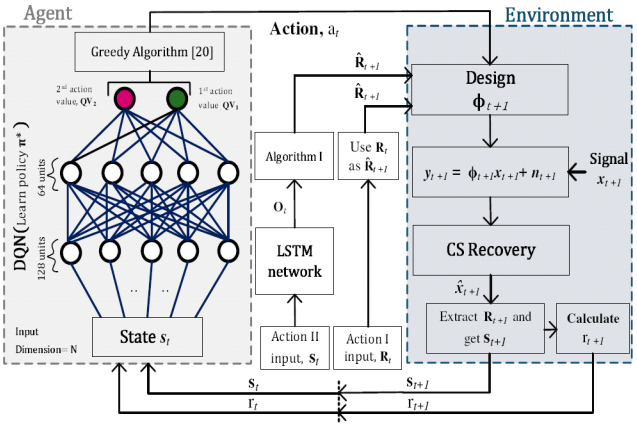

Fig. 1: The proposed RL-NCS framework where an agent tries to gain experiences from its interaction with environment and learn the optimal policy π^* for designing the measurement matrix.

4. RL-NCS: REINFORCEMENT-LEARNING-BASED NON-UNIFORM CS FRAMEWORK

In the non-uniform CS technique, it is required to design the measurement matrix in addition to sensing and reconstruction of signals. Fig. 1 depicts the way we defined these processes

in the environment part of our framework. At each time step, in the agent part (left), a neural network generates action values after receiving input from the environment (right). The measurement matrix $\mathbf{\Phi}_{t+1}$ is designed based on the course of action a_t taken by the agent. After CS recovery, the agent receives the new state $\boldsymbol{s}_{t+1}$ and the reward r_{t+1} as feedback. This feedback helps the agent to take the action that results in a higher reward. The goal of the agent is to learn the optimal policy of designing $\mathbf{\Phi}$ after a certain number of interactions with the environment. To fully realize the connection between agent and environment, it is necessary to formulate the state space ($\boldsymbol{S}$), action space ($\boldsymbol{A}$), and reward function (r).

4.1. State and Action Space

Let $\boldsymbol{R}_{t+1}$ be the set of indices of signal coefficients that are in the ROI at time $t+1$. After extracting the $\boldsymbol{R}_{t+1}$[1], we define $\boldsymbol{s}_{t+1}$, the binary state vector at time $t+1$ by its elements

$$s_{t+1}^{(l)} = \begin{cases} 1, & \text{if } l^{th} \text{ coef. of signal is in } \boldsymbol{R}_{t+1}, \\ 0, & \text{otherwise,} \end{cases} \tag{5}$$

where $s_{t+1}^{(l)}$ is the l^{th} element of $\boldsymbol{s}_{t+1}$ for $1 \leq l \leq N$.

In RL-NCS, the agent needs to decide about the strategy to design the measurement matrix at each time step. For that, we have devised the actions for the action space as following:

Action I (Direct Approach): The ROI at time t, $\boldsymbol{R}_t$, can be kept unchanged and used readily as the predicted ROI for next step, $\hat{\boldsymbol{R}}_{t+1}$. That is $\hat{\boldsymbol{R}}_{t+1} = \boldsymbol{R}_t$.

Action II (Learning Approach): For rapidly varying signals, the ROI changes relatively fast and the importance levels for most of the coefficients change over time. However, some coefficients still retain the same importance levels. Therefore, a learning mechanism can be developed to identify these two groups of coefficients. For this, we train an LSTM network, with parameters $\mathbf{\Psi}_t$, to extract valuable *correlation information from long sequence of states*. The output of the LSTM network, $\boldsymbol{O}_t$, helps us to determine $\hat{\boldsymbol{R}}_{t+1}$ using certain confidence bounds. Let $\boldsymbol{I}_t = \{1, 2, \ldots, N\} \backslash \boldsymbol{R}_t$ denote the set of indices of estimated non-ROI coefficients at time t. We can predict the $\hat{\boldsymbol{R}}_{t+1}$ using *Algorithm 1*.

Algorithm 1 Update Rule for ROI

Input: $\hat{\boldsymbol{R}}_{t+1} = \{\,\}, \boldsymbol{I}_t, \boldsymbol{O}_t,\ Th_{up} \in [0,1]\ \ \&\ \ Th_{low}\ \in [0,1]$
for steps $j \in \{1, 2, \ldots N\}$ **do**
  if $j \in \boldsymbol{R}_t$ & $\boldsymbol{O}_t^{(j)} \geq Th_{low}$ **do**
    $\hat{\boldsymbol{R}}_{t+1} \leftarrow \hat{\boldsymbol{R}}_{t+1} \cup \{j\}$
  else if $j \in \boldsymbol{I}_t$ & $\boldsymbol{O}_t^{(j)} \geq Th_{up}$ **do**
    $\hat{\boldsymbol{R}}_{t+1} \leftarrow \hat{\boldsymbol{R}}_{t+1} \cup \{j\}$
  end if
end for
Output: $\hat{\boldsymbol{R}}_{t+1}$ (predicted ROI for next time step)

[1]Depending on the application, the definition of ROI might be different for different scenarios and it can be extracted from the reconstructed signal $\hat{\boldsymbol{x}}_{t+1}$. For example, ROI can simply be the support of the signal.

Here, $\boldsymbol{O}_t^{(j)}$ is the j^{th} element of the LSTM output vector at time t. The role of the lower confidence bound, Th_{low}, is to select indices in $\boldsymbol{R}_t$ that can be included in $\hat{\boldsymbol{R}}_{t+1}$. On the contrary, the upper confidence bound Th_{up} is used to select indices for $\hat{\boldsymbol{R}}_{t+1}$ from set $\boldsymbol{I}_t$. Since, it is more likely that most of the coefficients retain the same importance levels as before, we set $Th_{up} > Th_{low}$. After forming $\hat{\boldsymbol{R}}_{t+1}$, the measurement matrix can be designed using the same procedure as Action I. The weighted cross-entropy loss function for the LSTM network is given by

$$J_t^{(LS)} = -\omega(\boldsymbol{O}_t^{(tar)} log(\boldsymbol{O}_t)) - (\mathbf{1} - \boldsymbol{O}_t^{(tar)}) log(\mathbf{1} - \boldsymbol{O}_t), \tag{6}$$

where $\boldsymbol{O}_t^{(tar)}$ works as the target vector for the output of LSTM, $\boldsymbol{O}_t$. A positional weight, ω, is used for the balance between true positive and false negative count.

4.2. Reward Function

We have formulated the reward function in terms of precision and recall defined as

$$Recall = \frac{TP}{|\,\boldsymbol{R}_{t+1}|}, Precision = \frac{TP}{|\hat{\boldsymbol{R}}_{t+1}|}, \tag{7}$$

where $|.|$ represents the cardinality of the set. The variable TP (true positive) indicates the number of elements in the predicted ROI, $\hat{\boldsymbol{R}}_{t+1}$ (predicted by the agent), that falls in the estimated ROI, $\boldsymbol{R}_{t+1}$. It should be noted that the $\boldsymbol{R}_{t+1}$ is extracted from the recovered signal $\hat{x}_{t+1}$ after the CS recovery (as shown in Fig. 1). Using (7), we calculate the *reward* as

$$r_{t+1} = \alpha * Precision + (2 - \alpha) * Recall. \tag{8}$$

Here, α indicate the influence of precision and recall on determining the reward at each step. Considering the objective of our work, we set $\alpha < 1$.

4.3. Multi-task Training

We have used two different neural networks as the online part of our framework as shown in Fig. 2. One is an LSTM network with 200 hidden units and the other one is a Q-network that uses two fully connected hidden layers with 128 and 64 neurons, respectively. At each time step t, we define the total loss as

$$J_t = (1 - \lambda) J_t^{(DQN)} + \lambda J_t^{(LS)}, \tag{9}$$

where $0 \leq \lambda \leq 1$ is a regularization parameter that controls the flow of LSTM loss, $J_t^{(LS)}$, to the optimizer. This way of optimization facilitates the training of LSTM without a dataset. At a certain stage of training, only the Q-network gets tuned until it finds the optimal policy. The complete RL-NCS algorithm is presented in Algorithm 2.

We have trained Q-network for T_{max} steps and LSTM network for one-third of that steps to have better generalization ability. The episode length (τ) is set to be 100 for the Q-network which also uses a learning rate of 0.05. The learning

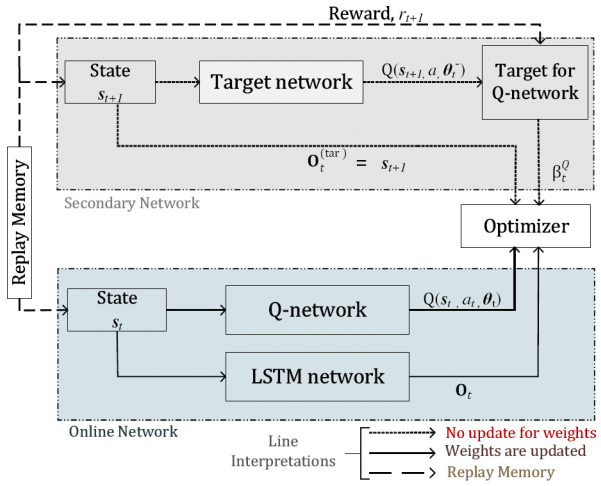

Fig. 2: Training procedure of the online network with proper utilization of replay memory. The purpose of the secondary network is to generate targets for Q-network and LSTM network.

rate has been reduced to 75% after each 5000 steps. Furthermore, we have used Rectified Linear Unit (Relu) activation for the hidden layers of the Q-network with the Sigmoid activation at the output. Throughout the training process, the agent (Q-network) follows ϵ-greedy algorithm [20] to take the best action at each step. The value of ϵ is set to be 1 and decreases to 0 with a decay rate of ϵ_{decay}. The target for Q-network is given by [3] while the state $\boldsymbol{s}_{t+1}$ works as the LSTM target. Furthermore, the output of the LSTM is used for Algorithm I to predict the ROI. Here, the LSTM tries to *learn the pattern of changes in consecutive signals* over its course of training. If, in any case, the LSTM fails to learn the pattern, the agent has the option to choose Action I. The Q-network identifies this failure using the rewards obtained from the environment and the range of this reward is between 0 to 2. For high transition probabilities, Action I works as a backup for Action II even though the performance gain might not be similar.

5. NUMERICAL EXPERIMENTS

In this section, a detailed description of the executed experiments are presented along with the outcome of the experiments. To underscore the effectiveness of our method, the primary performance metric that has been used here is

$$\text{TNMSE} = \frac{1}{\tau}\sum_{t=1}^{\tau}\frac{||\boldsymbol{x}_t - \hat{\boldsymbol{x}}_t||_2^2}{||\boldsymbol{x}_t||_2^2}. \tag{10}$$

TNMSE stands for time averaged normalized mean square error over the time period τ and $\hat{\boldsymbol{x}}_t$ is the reconstructed signal while $\boldsymbol{x}_t$ is its original counterpart. $||.||_2$ denotes the ℓ_2-norm of a vector.

At each step, the estimation of the signal is obtained using the ℓ_1 minimization recovery algorithm given by

$$\hat{\boldsymbol{x}}_t = \arg\min||\boldsymbol{x}_t||_1, \;\; s.t.||\boldsymbol{y}_t - \boldsymbol{\Phi}_t\boldsymbol{x}_t||_2 \leq \mu, \tag{11}$$

where $\hat{\boldsymbol{x}}_t$ is the estimated signal determined through the minimization of $||\boldsymbol{x}||_1 = \sum_n |x^{(n)}|$. The error bound μ is assigned to have a value of $\sigma_n\sqrt{M}$.

Algorithm 2 The RL-NCS Algorithm

Input: $\boldsymbol{\theta}_t$, $\boldsymbol{\theta}_t^-$ and $\boldsymbol{\Psi}_t$: initialize weights (random normal), $\boldsymbol{s}_1$: initialize the state (binary random)
for steps $t \in \{1, 2, \ldots T_{max}\}$ **do**
  Input $\boldsymbol{s}_t$ to the Q-network to get a_t and then design $\boldsymbol{\Phi}_{t+1}$
  Extract $\boldsymbol{R}_{t+1}$ from $\hat{\boldsymbol{x}}_{t+1}$ to get $\boldsymbol{s}_{t+1}$ and r_{t+1}
  Store the experiences ($\boldsymbol{s}_t$,a_t,$\boldsymbol{s}_{t+1}$,r_{t+1}) into *replay memory,* D^{replay}.
  Sample a mini-batch of Z transitions, SE, from D^{replay}
  for each sample e=($\boldsymbol{s}_t$,a_t,$\boldsymbol{s}_{t+1}$,r_{t+1}) in SE **do**
    $\boldsymbol{O}_t^{(tar)} = \boldsymbol{s}_{t+1}$
    $\beta_t^Q = r_{t+1} + \gamma \max_{a \in A} \boldsymbol{Q}(\boldsymbol{s}_{t+1}, a; \boldsymbol{\theta}_t^-)$
  end for
  Calculate the loss using (9) and decrease λ by $\Delta\lambda$
  Update $\boldsymbol{\Psi}_t$ and $\boldsymbol{\theta}_t$ using gradient descent method
  if t mod τ = 0; then $\boldsymbol{\theta}_t^- \longleftarrow \boldsymbol{\theta}_t$ **end if**
  $\boldsymbol{s}_t \longleftarrow \boldsymbol{s}_{t+1}$
end for
Output: An optimal policy for designing $\boldsymbol{\Phi}$

After predicting the ROI, we distribute the sensing energy to the columns of $\boldsymbol{\Phi}$ according to the prediction. We set the ℓ_2-norm of the n^{th} column as

$$e^{(n)} = \sqrt{N}\,\frac{\eta^{(n)}}{\sqrt{\sum_n \{\eta^{(n)}\}^2}}, \tag{12}$$

where $\eta^{(n)}$ is the importance level for n^{th} coefficient and N represents the total number of coefficients. The distribution of the energy is done in a way such that the columns corresponding to ROI coefficients receive more energy than the non-ROI counterpart and the criteria of energy constraint, $||\boldsymbol{\Phi}||_F^2 = N$, is met. On the other hand, for uniform sampling, the columns of the matrices are scaled to have unit norm.

5.1. Simulations with sparse signals in the canonical basis

At first, signals that are sparse in the canonical basis are employed for all simulations. Therefore, the ROI is considered to be the support of the signal. Furthermore, it has been assumed that the signals in the time series are correlated. To establish the correlation in value, the evolution of the value of n^{th} signal coefficient $w_t^{(n)}$ can be modelled as $(1-\rho)w_{t-1}^{(n)} + \rho v_t^{(n)}$. Here, the correlation parameter ρ dictates the degree of correlation and assumes a value between 0 and 1. The term $v_t^{(n)}$ is modelled with zero mean Gaussian distribution, $\mathcal{N}(0, \sigma_L^2)$, for imposing variations among two consecutive time steps. For describing the ROI, we consider a binary vector $\boldsymbol{d}_t = [d_t^{(1)}, \ldots, d_t^{(N)}]^T$. A value of 1 for $d_t^{(n)}$ indicates n^{th} coefficient is in the ROI and a zero value indicates otherwise. All the values of $\boldsymbol{d}_t$ are assumed to be independent of each other.

In order to establish the correlation among the ROI of two consecutive time steps, a Markov chain process is defined for each signal coefficient. That is, the modelling of transition probabilities, $tp_{01} = \text{P}\{d_t^{(n)} = 1 | d_{t-1}^{(n)} = 0\}$ and $tp_{10} =$

$\kappa * tp_{01}/(1-\kappa)$, is achieved by Markov chain process. The term κ denotes the sparsity level and can be expressed as $\kappa = \mathrm{P}\{d_t^{(n)} = 1\}$. Finally, the coefficients of the signal at each time step can be formulated as $x_t^{(n)} = w_t^{(n)} d_t^{(n)} + b_t^{(n)}(1 - d_t^{(n)})$. Small (or non-ROI) coefficients $b_t^{(n)}$ are modelled with $\mathcal{N}(0, \sigma_S^2)$. The simulation parameters for the CS environment are set as follows. The correlation parameter is set as $\rho = 0.2$. Standard deviation(SD) of large and small coefficients are $\sigma_L = 5$ and $\sigma_S = 0.01$, respectively. Furthermore, the value of $\eta^{(n)}$ is 0.7 for ROI coefficients and 0.3 for non-ROI coefficients. All other simulation parameters are $\gamma = 0.1$, $\alpha = 0.5, \omega = 5, Th_{up} = 0.8, Th_{low} = 0.1, \tau = 100, \lambda = 1$, $\Delta\lambda = 1/10000$, $\epsilon_{decay} = 1/10000$ and $T_{max} = 30000$, $N = 200$.

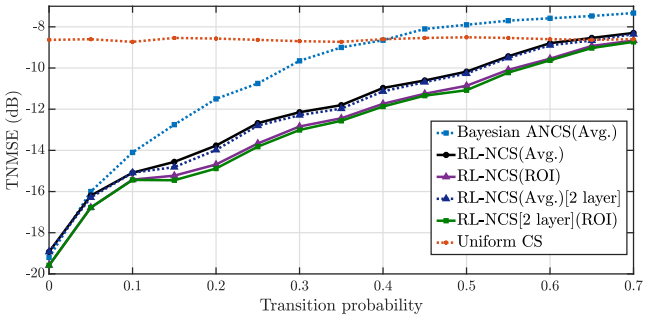

Fig. 3: Performance evaluation of l_1 recovery and proposed sampling method (RL-NCS) for different values of tp_{01}. For setup: $N = 200$, $M = 60$, SNR = 20dB, and total sensing energy = N.

For performance evaluation, TNMSE (dB) for different values of transition probabilities is shown in Fig. 3. We have averaged the reconstruction error over the whole episode. It can be observed that the proposed sampling method outperforms the uniform CS upto $tp_{01} \leq 0.62$. Unlike Bayesian ANCS [16], RL-NCS has significant performance gains over uniform CS for high transition probabilities. We have also shown the TNMSE only for the coefficients in ROI and it is observable that ROI coefficients are reconstructed with more accuracy compared to Non-ROI counterpart. A slight improvement is noticed in the performance gain for an extra LSTM layer. In order to comprehend this improvement, the percentage value of recall for different tp_{01} is shown in Fig. 4. We can see that two-layered LSTM helps the agent to infer the next ROI more accurately. Furthermore, it is also shown that the trend of choosing second action increases for signals with rapid variations.

Fig. 5 shows the recovery performance of uniform and non-uniform CS for different number of measurements. It can be seen that a performance gain up to 8.75 dB (for M = 60) is obtainable by employing RL-NCS. In addition, the reconstruction performance of l_1 recovery algorithm boosts up even with less number of measurements. The difference in performance gain is easily discernible. For example, uniform sampling requires 63% more measurements than RL-NCS to achieve a -15 dB gain. Furthermore, the noise associated with the sampling step usually increases the reconstruction error.

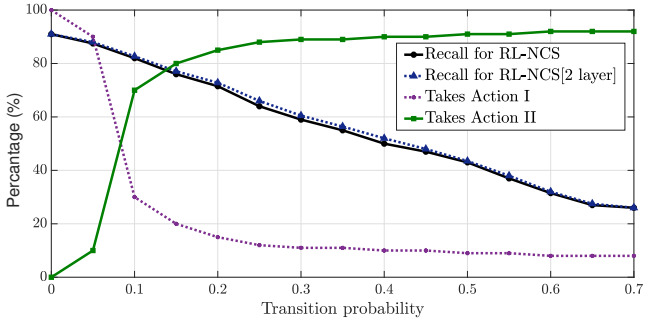

Fig. 4: The percentage of recall and percentage of chosen type of action (by agent) for different transition probability, tp_{01}.

Our proposed method is effective even in the regime of low SNR. Fig. 6 depicts the performance gain achieved by our method over other sampling method for different input SNRs.

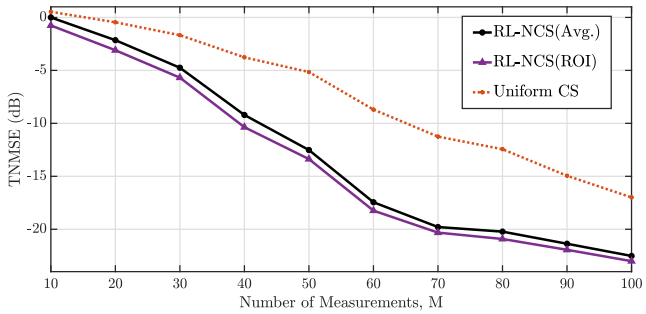

Fig. 5: Time averaged recovery error (in dB) for different number of measurements. For setup: $N = 200$, tp_{01} = 0.02, Input SNR = 20dB.

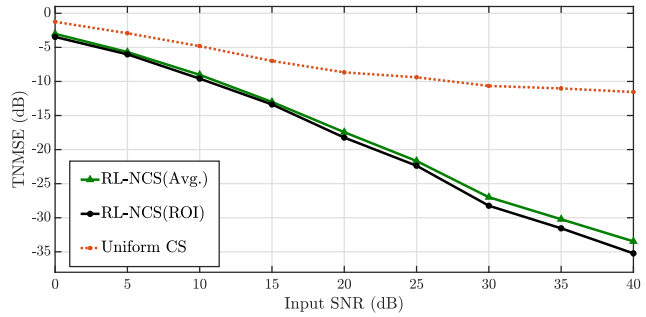

Fig. 6: Performance evaluation of recovery algorithm for different values of input SNR. For setup: $N = 200$, $M = 60$, tp_{01} = 0.02 and $T = 30$.

5.2. Simulations with sparse signals in the DCT domain

We have also carried out experiments with signals that are sparse in the DCT domain. Let, $\boldsymbol{z}_t = \boldsymbol{\Theta}\boldsymbol{x}_t$ be the sparse representation of the signal $\boldsymbol{x}_t$, obtained by the DCT transform matrix $\boldsymbol{\Theta}$. To establish the correlation between signals, same procedure explained in Section 5.1 has been followed. Since ROI and support of the signal are not the same in this case, a new set of binary Markov processes is used to model the variation of ROI. This Markov process uses the same parameters κ and tp_{01} to impose a certain degree of randomness along with the correlation. Therefore, we have a setup where support of $\boldsymbol{z}_t$ and ROI in $\boldsymbol{x}_t$ change at the same rate. To re-

construct $\hat{\boldsymbol{z}}_t$ we employ

$$\hat{\boldsymbol{z}}_t = \arg\min ||\boldsymbol{z}_t||_1, \quad s.t. ||\boldsymbol{y}_t - \boldsymbol{\Phi}_t \boldsymbol{\Theta}^T \boldsymbol{z}_t||_2 \leq \mu \qquad (13)$$

that eventually leads us to $\boldsymbol{x}_t = \boldsymbol{\Theta}^T \boldsymbol{z}_t$.

Depending on the applications, there are different methods for figuring out the ROI of signal. Furthermore, we also assume that the ROI detection method may give faulty observations. The performance of RL-NCS for different number of measurements are depicted in Fig. 7. Considering only the ROI coefficients, the RL-NCS algorithm has a significant performance gain over uniform CS even though it loses out on total reconstruction error. In Fig. 8, the performance for non-uniform CS is shown for different fault rates in ROI detection. As the estimation of ROI gets more erroneous, it gets harder for the agent to fix the location of ROI. However, even for 60% fault rate, RL-NCS performs better than uniform CS when it comes to the reconstruction of ROI coefficients.

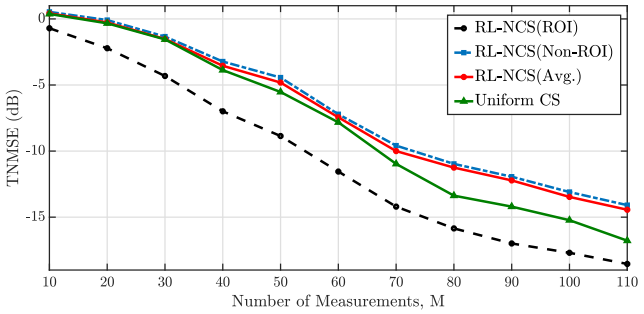

Fig. 7: TMSNE error (in dB) for different number of measurements (M). For setup: $N = 200$, $T = 30$, SNR = 20 dB and fault-rate = 0.10.

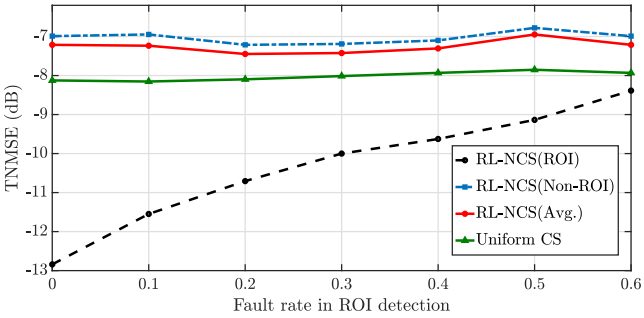

Fig. 8: TMSNE error (in dB) for different fault-rate (%) in ROI detection.

6. CONCLUSION

We proposed a framework with a view to achieving non-uniform compressed sensing for time-varying signals. To overcome the limitation of adaptation with fast varying signals, an LSTM network along with a manually tailored action is introduced. A deep Q-network approach has also been employed here whose purpose is to choose the best action for designing measurement matrix at each time step. With the joint training method, we introduced an elegant solution for training an LSTM-based learning mechanism while obtaining an optimal policy from Q-network in the end. Moreover, we have carried out experiments for sparse signals both in the canonical basis and in the DCT domain. The results presented here show that our method achieves significant performance gain over uniform CS.

7. REFERENCES

[1] Candes EJ, Wakin MB. An introduction to compressive sampling [a sensing/sampling paradigm that goes against the common knowledge in data acquisition]. IEEE signal processing magazine. 2008;25(2):21-30.

[2] Donoho DL. Compressed sensing. IEEE Transactions on information theory. 2006 Apr 1;52(4):1289-306.

[3] Vaswani N. Kalman filtered compressed sensing. In2008 15th IEEE International Conference on Image Processing 2008 Oct 12 (pp. 893-896). IEEE.

[4] Asif MS, Reddy D, Boufounos PT, Veeraraghavan A. Streaming compressive sensing for high-speed periodic videos. In2010 IEEE International Conference on Image Processing 2010 Sep 26 (pp. 3373-3376). IEEE.

[5] Zaeemzadeh A, Joneidi M, Rahnavard N, Qi GJ. Co-SpOT: Cooperative spectrum opportunity detection using bayesian clustering in spectrum-heterogeneous cognitive radio networks. IEEE Transactions on Cognitive Communications and Networking. 2017 Dec 27;4(2):206-19.

[6] Shahrasbi B, Rahnavard N. Model-based nonuniform compressive sampling and recovery of natural images utilizing a wavelet-domain universal hidden Markov model. IEEE Transactions on Signal Processing. 2016 Sep 29;65(1):95-104.

[7] Rahnavard N, Talari A, Shahrasbi B. Non-uniform compressive sensing. In2011 49th Annual Allerton Conference on Communication, Control, and Computing (Allerton) 2011 Sep 28 (pp. 212-219). IEEE.

[8] Mnih V, Kavukcuoglu K, Silver D, Graves A, Antonoglou I, Wierstra D, Riedmiller M. Playing atari with deep reinforcement learning. arXiv preprint arXiv:1312.5602. 2013 Dec 19.

[9] Chincoli M, Liotta A. Self-learning power control in wireless sensor networks. Sensors. 2018 Feb;18(2):375.

[10] Le TT, Moh S. Reinforcement-Learning-Based Topology Control for Wireless Sensor Networks. Proceedings of the Grid and Distributed Computing 2016. 2016:22-7.

[11] Malloy ML, Nowak RD. Near-optimal adaptive compressed sensing. IEEE Transactions on Information Theory. 2014 Jul;60(7):4001-12.

[12] Braun G, Pokutta S, Xie Y. Info-greedy sequential adaptive compressed sensing. IEEE Journal of selected topics in signal processing. 2015 Jun;9(4):601-11.

[13] Ziniel J, Schniter P. Dynamic compressive sensing of time-varying signals via approximate message passing. IEEE transactions on signal processing. 2013 Nov 1;61(21):5270-84.

[14] Shahrasbi B, Talari A, Rahnavard N. TC-CSBP: Compressive sensing for time-correlated data based on belief propagation. In2011 45th Annual Conference on Information Sciences and Systems 2011 Mar 23 (pp. 1-6). IEEE.

[15] Vaswani N. LS-CS-residual (LS-CS): compressive sensing on least squares residual. IEEE Transactions on Signal Processing. 2010 Aug;58(8):4108-20.

[16] Zaeemzadeh A, Joneidi M, Rahnavard N. Adaptive non-uniform compressive sampling for time-varying signals. In2017 51st Annual Conference on Information Sciences and Systems (CISS) 2017 Mar 22 (pp. 1-6). IEEE.

[17] Greff K, Srivastava RK, Koutnk J, Steunebrink BR, Schmidhuber J. LSTM: A search space odyssey. IEEE transactions on neural networks and learning systems. 2016 Jul 8;28(10):2222-32.

[18] Watkins CJ, Dayan P. Q-learning. Machine learning. 1992 May 1;8(3-4):279-92.

[19] Mnih V, Kavukcuoglu K, Silver D, Rusu AA, Veness J, Bellemare MG, Graves A, Riedmiller M, Fidjeland AK, Ostrovski G, Petersen S. Human-level control through deep reinforcement learning. Nature. 2015 Feb;518(7540):529.

[20] Tokic M. Adaptive -greedy exploration in reinforcement learning based on value differences. InAnnual Conference on Artificial Intelligence 2010 Sep 21 (pp. 203-210). Springer, Berlin, Heidelberg.