Compressed Sensing via Dictionary Learning and Approximate Message Passing for Multimedia Internet of Things

Zhicheng Li†, Hong Huang‡ and Satyajayant Misra∗

Abstract—In this paper, we present a compressed sensing based approach, which combines the dictionary learning (DL) method and the approximate message passing approach (AMP). The approach can be used for efficient communication in the multimedia Internet of Things (IoT). AMP is a signal reconstruction algorithm framework, which can be explained as an iterative denoising process. On the other hand, the DL method seeks an adaptive dictionary for realizing sparse signal representations, and provides good performance in signal denoising. We apply the dictionary learning based denoising method within the AMP algorithm framework and propose a novel DL-AMP framework. We demonstrate our framework's effectiveness for multimedia IoT devices by showing its capability in reducing required communication bandwidth for multimedia communication while improving reconstruction quality (by over 2 dB).

Index Terms—Internet of Things (IoT), compressed sensing, dictionary learning, approximate message passing, sequential generalization of K-means.

I. BACKGROUND AND INTRODUCTION

The Internet of Things (IoT) has drawn attention recently as a means of connecting the proliferating embedded devices to the Internet. IoT devices are forecast to grow to 33 billion by 2035 [1]. These devices include sensing devices, security cameras, mobile phones, and home automation/control devices. Different from traditional wireless sensor networks (WSNs), the IoT is envisioned to be deployed in a larger scale and may have a much broader geographic deployment, thus increasing its complexity.

The complexity results from many challenges: a) IoT devices are generally low-computation capability devices; b) they may run on battery or have limited power, while needing to transmit large amounts of data (e.g., cameras and other big-data applications [2], [3], [4])—stringent energy constraints; and c) given the large number of devices that will be connected, the IoT devices will suffer from congestion, packet drops, and transmission uncertainty. Thus, there is a need for developing special data transmission strategies that enable energy efficient communications, and low-power and low-cost signal processing operations. In this paper, we propose the use of an efficient compressed sensing framework as a strategy to be used by IoT devices to reduce the amount of multimedia data they need to transmit, while ensuring that the complete data can be recovered with high fidelity at the receiver.

Compressed sensing (CS) is a data acquisition and reconstruction approach that takes advantage of sparse signal structures to reduce the size of the transmitted/stored data. Multimedia data generally possesses this sparse structures. For example, images are sparse in the wavelet representation. The conventional way to deal with such signal has been to acquire all the data first, compress it and then store it, as is done in image processing. The CS technique was first proposed by Candes et al. [5], [6], in which, the original data can be accurately reconstructed from only a portion of the sampled data, sampled at rates lower than the Nyquist rate. It has sparked tremendous research interest as it can be leveraged to greatly reduce the sampling rates in signal processing applications, such as medical scanners, data communication, and cameras. This technique lends itself naturally to the multimedia IoT domain.

Given the popularity of CS, several approaches and mechanisms have been proposed to increase its efficiency. One of them is the Approximate Message Passing (AMP) algorithm, proposed by Donoho et al. [7], [8]. AMP is an iterative threshold based signal reconstruction algorithm that performs scalar denoising within each iteration; and with proper selection of the denoising function, the reconstruction quality can be one of the highest of all CS techniques, with very low reconstruction complexity. The AMP framework is becoming popular, as among iterative thresholding algorithms with low-reconstruction complexity, it converges the fastest. One of the challenges in applying image denoisers within AMP is the difficulty in dealing with the Onsager reaction term [9], because of the divergence of the involved image denoiser.

Several important AMP related results have been reported. In [10], the expectation-maximization Gaussian-mixture algorithm is presented, which is based on Gaussian mixture distribution models. In [11], a Stein’s unbiased risk estimate (SURE) based parametric denoiser is presented along with a parametric SURE-AMP algorithm. In [12], the authors applied amplitude-scale invariant Bayes estimator (ABE) and adaptive Wiener filter within AMP, and introduced AMP-ABE and AMP-Wiener algorithms. It is worth noting that the denoiser-based AMP algorithms do not have high reconstruction quality and suffer from high runtimes. To marry the best of both worlds, in this paper, we have developed an AMP denoiser with high reconstruction quality and acceptable runtime.

We focus on the AMP framework and combine it with a family
of dictionary learning denoisers. Dictionary learning (DL) is an effective technique that has attracted a great deal of attention in image denoising. DL based methods generally achieve lower reconstruction error than wavelet-based methods [12]. Among DL based methods, the method of optimal directions (MOD) was first presented by Engan et al. [13]. Then, Aharon et al. presented a K-means Singular Value Decomposition (K-SVD) algorithm [14]. Recently, Sahoo and Makur proposed a more effective DL algorithm called Sequential Generalization of K-means (SGK) [15], which has better performance both in runtime and reconstruction quality. Due to the good performance of DL methods in image denoising, we present a novel DL-AMP algorithm within the AMP framework. In our framework, at each iteration, the trained dictionary, obtained using a DL algorithm used in the last step, is introduced as the initial one in the current step. We name the different DL-AMP algorithms that incorporate DL algorithms described above, as MOD-AMP, K-SVD-AMP and SGK-AMP respectively—the DL-AMP family—and present their performance in our simulation results to illustrate the efficacy of the DL-AMP framework for CS.

Contributions: The main contributions for this paper are third-folds. First, we propose a novel DL-AMP CS framework developed for multimedia IoT devices. To the best of our knowledge, we are the first to use the AMP method in the IoT domain. Second, our framework is designed in a way that new developments in DL denoising methods can be easily substituted for the current DL algorithm, making the resultant algorithm faster. Third, we demonstrate that the proposed algorithm framework can reconstruct transmitted images better than the state of art, especially for images with texture [16], [17].

The paper is organized as follows. In Section II, the basic formulation of compressed sensing and the idea of IoT are introduced and formulated. In Section III, the basic framework of AMP and some DL algorithms are introduced. Finally, the DL-AMP framework is proposed. Illustrative examples are given to demonstrate the effectiveness of the proposed algorithms in Section IV, and finally conclusions are presented in Section V.

II. COMPRESSED SENSING AND THE INTERNET OF THINGS

In this section, we introduce some basic definitions and the basic framework of CS.

A. The Basic Framework of Compressed Sensing

Assume that there is an orthonormal basis $\Psi = [\psi_1 \ \psi_2 \ \ldots \ \psi_n]$, with the vectors $\{\psi_i\}$ as columns, and an $n$-dimensional signal $x$, which can be expressed as

$$x = \sum_{i=1}^{n} \theta_i \psi_i = \Psi \theta,$$

where, $\theta_i$ is the $i^{th}$ coefficient. Based on the CS theory, if $x$ is sparse in the basis $\Psi$, then, under certain conditions, we can use $m$ non-adaptive measurements of $x$ to recover the signal exactly, where $m \ll n$. Define these $m$ measurements as $y_j$ ($j = 1, 2, \ldots, m$), which are the projections of $x$. Then, the $m$-dimensional measurement is described by

$$y = \Phi x,$$

where $y = [y_1 \ y_2 \ \ldots \ y_m]$ is the measurement vector and $\Phi$ is the sensing matrix with size $m \times n$. Since $m \ll n$, if we want to recover $x$ from $y$, the solution of the inverse problem satisfying (2) may not be unique. However, due to the fact that the original signal $x$ is sparse in a certain basis $\Psi$, the optimization problem can solve the reconstruction problem above, which is formulated as follows [18]:

$$(P_0) \min_{\theta} \|\theta\|_0, \text{ subject to } y = \Phi x.$$  \hspace{1cm} (3)

It is well-known that solving $P_0$ is NP-complete. Surprisingly, it was shown that one can replace the $l_0$-norm by $l_1$-norm, and instead formulate the optimization problem as [19], [20], [21]:

$$(P_1) \min_{\theta} \|\theta\|_1, \text{ subject to } y = \Phi x.$$  \hspace{1cm} (4)

In [20], the authors showed that if the signal is sufficiently sparse, the solutions of $P_0$ and $P_1$ are the same. $P_1$ is a convex linear programming problem, and there are many efficient solution techniques for this optimization problem (DL-AMP being one).

![Fig. 1. Multitude of devices that make up the IoT.](image)

B. Data transmission in Internet of Things (IoT)

As shown in Fig. 1, the IoT will enable connections among a wide variety of things, ranging from small sensors on one end of the spectrum to the cloud of servers (data storage and analysis) on the other end. With the growth of IoT, on this spectrum, as we move closer to the sensors (and other end-devices) the device numbers will increase by several orders of magnitude, the bandwidth available to them will reduce by several orders of magnitude (Tbps for cloud servers to several hundred Kbps for end-devices), and the compute power and energy available will be lower by several orders of magnitude. These constraints call for serious attention to the development of mechanisms that reduce the computation, power, and communication loads for the end-devices. The mechanisms should also promote collaborative sensing between end-devices to reduce overall energy and bandwidth requirements. For multimedia IoT devices, these constraints become especially stringent as they will transmit more data and may additionally have to meet stringent real-time requirements.

CS is one mechanism that can help meet all these constraints for the following reasons [22]. The use of CS on the end-devices can help reduce the sampling rate and the amount of data to be transferred, thus requiring less computation, power, and bandwidth. In addition, it has been shown that the use of collaborative CS between neighboring devices can help reduce the load on all
dimensions further [23], [24]. Thus, the development new and more efficient CS techniques for IoT is important, which is our aim in this paper. In the next section, we present our efficient DL-AMP framework for using CS at an individual end-device. We do not present the collaborative sensing scenario, which we will study in the future.

III. COMPRESSED SENSING AND APPROXIMATE MESSAGE PASSING IN IOT

A class of applications for multimedia IoT will be the transfer of two-dimensional images. For example, video monitoring of building perimeter and motion-activated cameras may transmit images or JPEG-based videos to the data store or user computers. We will illustrate our framework in the rest of the paper using this application as a use-case. In this section, we show how the message can be transmitted using our novel CS framework.

A. The Basic Framework of Denoiser Based Approximate Message Passing

Before introducing the AMP framework, for better understanding the following definitions and lemmas are presented, borrowing from [9].

Definition 1 [9]: Assume \(x_0\) is an original noiseless signal and \(y = x_0 + \sigma \epsilon\) is the observations of \(x_0\) with noise, where \(\epsilon \sim \mathcal{N}(0, I)\), \(I\) is the identity matrix and \(\sigma > 0\) denotes the standard deviation of the noise. Define that \(D_{\sigma}\) is a family of denoisers related to the standard deviation of the noise \(\sigma\):

\[D_{\sigma}(x_0 + \sigma \epsilon) = x,\]

where \(y = x_0 + \sigma \epsilon\) is the input, and \(x\) is a denoising estimate of \(x_0\).

Definition 2 [9]: \(D_{\sigma}\) is called a proper family of denoisers of level \(\kappa\) \((\kappa \in (0, 1))\) for the class of signals \(C\) if

\[\sup_{x_0 \in C} \sigma \leq \frac{\|D_{\sigma}(x_0 + \sigma \epsilon) - x_0\|_2}{\|\epsilon\|_2},\]

for every \(\sigma > 0\), where \(\sigma\) is the expected value calculator. Note that the expectation is with respect to \(\epsilon \sim \mathcal{N}(0, I)\).

Lemma 1 [9]: Let \(C\) denote a \(k\)-dimensional subspace of \(\mathbb{R}^n\) \((k < n)\). Also, let \(D_{\sigma}(y)\) be the projection of \(y\) onto subspace \(C\) denoted by \(P_{C}(y)\). Then,

\[\sigma \leq \frac{\|D_{\sigma}(x_0 + \sigma \epsilon) - x_0\|_2}{\|\epsilon\|_2} = \frac{k}{n} \sigma^2,\]

for every \(x_0 \in C\) and every \(\sigma^2\). Hence, this family of denoisers is a proper family of level \(k/n\).

Definition 3 [9]: We call a denoiser monotone if for every \(x_0\) its risk function

\[R(\sigma^2, x_0) \equiv \sigma \leq \frac{\|D_{\sigma}(x_0 + \sigma z) - x_0\|_2}{\|\epsilon\|_2},\]

is a non-decreasing function of \(\sigma^2\). 

Based on the above definitions and lemma, we formulate the AMP framework:

\[x^{t+1} = D_{\sigma^t}(x^t + A^* z^t),\]

\[z^t = y - A x^t + z^{t-1} D_{\sigma^t-1}(x^{t-1} + A^* z^{t-1}) / m,\]

\[(\sigma^t)^2 = \frac{\|z^t\|_2^2}{m},\]

where \(x_0\) is the original noiseless signal, \(x^t\) is the estimate of \(x_0\) at iteration \(t\), and \(z^t\) is an estimate of the residue. Then, \(x^t + A^* z^t\) can be written as \(x_0 + \nu^t\), where \(\nu^t\) can be considered as i.i.d. Gaussian noise. \(\sigma^t\) is an estimate of the standard deviation of that noise. \(D_{\sigma^t}\) is defined in Definition 1. \(D_{\sigma^t-1}\) denotes the divergence of the denoiser \(D_{\sigma^t-1}\). The term \(z^{t-1} D_{\sigma^t-1}(x^{t-1} + A^* z^{t-1}) / m\) is the Onsager correction term, which has a major impact on the performance of the algorithm. The explicit calculation of this term is not always straightforward, since in many practical cases denoisers do not have explicit formulations. Hence, it is also not possible to calculate \(D_{\sigma^t-1}\). However, Metzler et al. [9] have shown that the Onsager correction term can be approximately calculated without requiring the explicit form of the denoiser. On the other hand, the Denoiser-AMP also has some requirements for denoisers, which are proper and monotone in Definition 2 and Definition 3.

B. Dictionary Learning Denoiser

In this section, we introduce some DL denoisers, such as the K-SVD algorithm, the MOD algorithm, and the SGK algorithm.

1) Sparse Representation of Signals by Dictionary: Using a dictionary matrix \(D \in \mathbb{R}^{n \times K}\), which contains \(K\) prototype signal-atoms for columns, \(\{d_j\}_{j=1}^{K}\), a signal \(y \in \mathbb{R}^n\) can be represented as a sparse linear combination of those atoms, which can be described as \(y = Dx\) or \(y \approx Dx\), where \(x \in \mathbb{R}^K\) contains the representation coefficients of the signal \(y\). Similar to Equation (3), if \(n < K\) and \(D\) is a full-rank matrix, the solution is not unique. The fewest number of non-zero coefficients is the sparsest representation as shown in the following:

\[\min_{x} \|x\|_0, \text{ subject to } y = Dx.\]

If the noise is considered, the form of the sparsest representation is

\[\min_{x} \|x\|_0, \text{ subject to } \|y - Dx\|_2 < \epsilon.\]

Generally speaking, the DL denoising method contains two stages, one is sparse representation, the other is dictionary update. Define \(\mathcal{X}\) is a set of coefficient matrices and \(\mathcal{D}\) is a set of all dictionaries including unit column-norms. The notation \(\|P\|_F\) stands for the Frobenius norm, defined as \(\|P\|_F = \sqrt{\sum_{i,j} P_{ij}^2}\), where \(P_{ij}\) is an element of \(P\). The solution is obtained iteratively by alternating between these two stages as follows:

1) Sparse representation: Define a set of training signals \(Y = [y_1, y_2, \ldots, y_n]\), and obtain \(X^{(l)}\) for each \(y_i\) in \(Y\) as

\[X^{(l)} = \arg \min_{X \in \mathcal{X}} \|Y - D^{(l-1)} X^{(l-1)}\|_F^2,\]

where \(X^{(l)} = [x_1^{(l)}, x_2^{(l)}, \ldots, x_n^{(l)}]\) is the sparse representation in the \(l\)th iteration.

2) Dictionary update: For the obtained \(X^{(l)}\), update \(D^{(l)}\) such that

\[D^{(l)} = \arg \min_{D \in \mathcal{D}} \|Y - DX^{(l)}\|_F^2.\]

Based on this concept, we introduce some DL algorithms in the next subsection.

2) Dictionary Learning (DL) Algorithms:
a) Method of Optimal Directions (MOD) Dictionary Learning Algorithm: The method of optimal directions (MOD) is a DL algorithm, which is presented in [13]. The advantage of the MOD method is the simplicity of its mechanism for updating the dictionary. MOD is a coder independent dictionary training algorithm, which can be used for all sparse representation applications. Assume that the sparse coding for each example is known, then we define the errors \( e_i = y_i - D x_i \), for all \( i \). The overall representation mean square error is given by
\[
\| E \|_F^2 = \| e_1 e_2 \ldots e_n \|_F^2 = \| Y - DX \|_F^2. \tag{7}
\]
Here all the \( y_i \)'s columns of the matrix \( Y \), are concatenated and similarly the representative coefficient vectors \( x_i \) are gathered to build the matrix \( X \). We can update \( D \), such that the above error is minimized, with the assumption of fixed \( X \). One can obtain the relation \((Y - DX) X^T = 0\), by taking the derivative of (7) with respect to \( D \), and then it leads to
\[
D^{(t+1)} = Y X^{(t)T} (X^{(t)} X^{(t)T})^{-1}. \tag{8}
\]
In each iteration, we first obtain \( X^{(t)} \) by a given \( D^{(t)} \), then \( D^{(t+1)} \) can be updated by using the formula in (8).

b) K-mean Singular Value Decomposition (K-SVD) Dictionary Learning Algorithm: K-SVD algorithm breaks the global minimization problem (6) into \( K \) sequential minimization problems in the dictionary update stage. Every column \( d_k \) in \( D \) and its corresponding row of coefficients \( x_{row,k} \) in \( X \) are evaluated and updated in the algorithm. The error term in (7) can be rewritten as
\[
\| E \|_F^2 = \| Y - \sum_{j \neq k} d_j^{(t)} X_{row,j}^{(t)} - d_k^{(t)} X_{row,k}^{(t)} \|_F^2.
\]
Let's define \( E_k^{(t)} = Y - \sum_{j \neq k} d_j^{(t)} X_{row,j}^{(t)} \), then we have
\[
\left\{ d_k^{(t)}, \hat{X}_{row,k}^{(t)} \right\} = \arg \min_{d_k, X_{row,k}^{(t)}} \| E_k^{(t-1)} - d_k X_{row,k}^{(t-1)} \|_F^2. \tag{9}
\]
In [14], the proposed algorithm employs SVD to find the closest rank-1 matrix (in Frobenius norm) that approximates \( E_k^{(t-1)} \) subject to \( d_k^{(t)} = 1 \). SVD decomposition is performed on \( E_k^{(t)} = U \Lambda V^T \); \( d_k^{(t)} \) is taken as the first column of \( U \), and \( \hat{X}_{row,k}^{(t-1)} \) is obtained as \( \Delta_1 V_1 \), where \( \Delta_1 \) is the first diagonal element of \( \Delta \), and \( V_1 \) is the first column of \( V \).

c) Sequential Generalization of K-Means Dictionary (SGK) learning Algorithm: K-means and sequential algorithms consume lesser resources [15]. In the algorithm, when \( X_{row,k}^{(t)} \) is unchanged, the loss of sparsity and structure of \( X^{(t)} \) will be eliminated. Note that \( X^{(t)} \) is defined in (9). Thus, the sequential update problem is posed as
\[
d_k^{(t)} = \arg \min_{d_k} \| E_k^{(t-1)} - d_k X_{row,k}^{(t-1)} \|_F^2. \tag{10}
\]
The solution to (10) can be obtained in the same way as (8)
\[
d_k^{(t)} = E_k^{(t-1)} X_{row,k}^{(t-1)T} (X_{row,k}^{(t-1)T} X_{row,k}^{(t-1)})^{-1}. \tag{8}
\]
The term \( d_k^{(t)} \) replaces \( d_k^{(t-1)} \) before updating the next atom in the sequence, and it can account for the overlap among \( X_{row,k}^{(t)} \)’s clusters \( R_k \), where \( R_k = \{ i : y_i = D e_k \} \). For terminating the algorithm, we repeat this process for all \( K \) atoms sequentially (procedure similar to K-means).

C. Description of the Combined Framework

In this section, we present the detailed structure of the DL-AMP framework, and further explain some details about it. First, we need to prove that the DL denoising algorithms satisfy requirements for AMP.

**Theorem 1:** The DL denoising method is a proper family of denoisers of level \( \kappa \) (\( \kappa \in (0, 1) \)) for the class of signals \( C \).

**Proof:** Similar to [13], [14], [15], [25], we have the denoised signal as follows:
\[
D_\sigma(X + \sigma e) = \left( \lambda I + \sum_{ij} R_{ij} \right)^{-1} \left( \lambda Y + \sum_{ij} R_{ij} DX_{ij} \right),
\]

where \( \sigma \) is the standard deviation of that noise, \( e \sim N(0, I) \), and \( D \) is the trained dictionary. The matrix \( R_{ij} \) is an \( n \times N \) matrix that extracts the \((ij)\) block from the signal, and each patch of signals is described as \( X_{ij} = R_{ij} X \) of size \( \sqrt{n} \times \sqrt{n} \) in each location. The matrix \( Y \) is the signal with noise, which can be written as
\[
Y = X + \sigma e. \tag{11}
\]

Then, we have
\[
\mathcal{E}\left( \| D_\sigma(X + \sigma e) - X \|_2 \right) = \mathcal{E}\left( \left\| \left( \lambda I + \sum_{ij} R_{ij} \right)^{-1} \left( \lambda Y + \sum_{ij} R_{ij} DX_{ij} - X \right) \right\|_2 \right). \tag{12}
\]

Assume that \( D \) is well trained, then we have
\[
Y_{ij} \approx DX_{ij}. \tag{13}
\]

Then using (11), (12), and (13), we obtain
\[
\mathcal{E}\left\| D_\sigma(X + \sigma e) - X \right\|_2^2 = \mathcal{E}\left( \left\| \left( \lambda I + \sum_{ij} R_{ij} \right)^{-1} \left( \lambda Y + \sum_{ij} R_{ij} DX_{ij} - X \right) \right\|_2 \right)^2.
\]

resulting in
\[
\mathcal{E}\left\| D_\sigma(X + \sigma e) - X \right\|_2^2 = \frac{1}{\sigma^2}. \tag{14}
\]

According to Definition 1 and Lemma 1, we have that the DL denoising method is a proper family of denoisers of level \( \kappa \) (\( \kappa \in (0, 1) \)) for the class of signals \( C \). This completes the proof. □

**Theorem 2:** The DL denoiser is a monotone denoiser. 

**Proof:** According to the formula in (14), we have
\[
R(\sigma^2, X) = \mathcal{E}\left\| D_\sigma(X + \sigma^2 e) - X \right\|_2^2 = \frac{1}{n} \sigma^2.
\]
Suppose that for \( \sigma_1 < \sigma_2 \), we have
\[
R(\sigma_1^2, X) = \frac{1}{n} \sigma_1^2 < \frac{1}{n} \sigma_2^2 = R(\sigma_2^2, X). \tag{15}
\]

Then, according to (15) and Definition 2, it is easy to prove that the DL denoiser is a monotone denoiser. This completes the proof. □
Algorithm 1 DL-AMP Framework
1: procedure x=DL-AMP(Φ, y, N)
2: set the initial solution \( x_0 = 0 \),
3: set the initial residual \( z_0 = y \),
4: set the initial standard deviation of noise \( \sigma_0 = \frac{1}{m} \| z_0 \|_2^2 \),
5: for \( j = 0, 1, 2, \ldots \), do
6: \( r_j = x_j + \Phi^T z_j \),
7: \( (x_{j+1}, D_j^*) = DL(r_j, \sigma_j, D_j, N) \),
8: \( (\text{div}, D_j^{**}) = DL'(r_j, \sigma_j, D_j, N) \),
9: \( z_{j+1} = y - \Phi x_{j+1} + \frac{1}{m} \text{div} \cdot z_j \),
10: \( \sigma_{j+1} = \frac{1}{m} \| z_{j+1} \|_2^2 \),
11: \( D_{j+1} = D_j^* \),
12: end for
13: Output approximate solution \( x \).
14: end procedure

Theorem 1 and Theorem 2 show that the DL methods satisfy the denoiser properties of AMP Algorithm. Algorithm 1 shows the procedures of DL-AMP algorithm. We present some key remarks about the algorithm in the following:

1) \( (x_{j+1}, D_j^*) = DL(r_j, \sigma_j, D_j, N) \) in Algorithm 1 is the DL denoiser like K-SVD, MOD, or SGK methods, where \( r_j \) is the iteration term, \( \sigma_j \) is the standard deviation of noise, \( D_j \) is the dictionary in current step. \( D_j^* \) is the refreshed trained dictionary in the current step, and \( N \) is the dictionary training iteration time.

2) The \( (\text{div}, D_j^{**}) = DL'(r_j, \sigma_j, D_j^*, N) \) in Algorithm 1 is the derivation of DL algorithm, where

\[
\text{div} = \lim_{\Delta t \to 0} \frac{DL(r_j + \Delta t, \sigma_j, D_j, N) - DL(r_j, \sigma_j, D_j, N)}{\Delta t},
\]

and \( D_j^* \) is the trained dictionary in the previous step, \( D_j^{**} \) is the trained dictionary in the current step.

3) It is worth noting that there are two steps to refresh the dictionary (Step 7 and Step 8) in every iteration, that’s why we use \( D_j, D_j^* \), and \( D_j^{**} \) to distinguish the trained dictionary in an iteration.

Fig. 2. 256 × 256 natural images for reconstruction.

IV. EXPERIMENTAL RESULTS FOR MULTIMEDIA IMAGES

To demonstrate the efficacy of the DL-AMP framework in IoT, we evaluate its performance in an imaging application and compare the performance of our proposed method with other existing methods. There are many different types of images multimedia IoT applications can generate. For making a broad comparison we chose a broad set of images to compare our framework with other algorithms in the literature. For comparison, we use the MOD, and SGK, DL algorithms for the DL-AMP framework and we compare the results with EM-GM-GAMP [10], l₁-AMP [7], SURE-AMP [11], NLM-AMP [26], Bilateral-AMP [9], Gauss-AMP [9], BM3D-AMP [27], and fast-BM3D-AMP [27] algorithms. We observed the runtime of K-SVD-AMP to be very slow, so have not used it for comparison. Our results show that although the speed of a DL-AMP algorithm is slow, the quality of the reconstructed image is better than most of the other popular algorithms in the literature.

Fig. 2 shows four 256 × 256 pixels nature images (Lena, House, Boat, Cameraman) borrowed from [11] and used to compare with the corresponding results of the SGK-AMP, and l₁-AMP, EM-GM-GAMP, and SURE-AMP. Fig. 3 shows six 128×128 pixels images with texture information (Nebula texture, Brick wall, Wood texture, Carpet, Fingerprint 1, Fingerprintf 2), which are compressed, transferred, and reconstructed to test a majority of the algorithms. For comparing with the state of art results, we introduce the following definition:

Definition 4: Mean squared error (MSE) is defined as follows:

\[
MSE = \frac{1}{mn} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [IM(i, j) - R(i, j)]^2,
\]

where \( n \) and \( m \) are the size of image in pixels, \( IM \) is the original image, \( R \) is the reconstructed image.

Definition 5: Peak signal-to-noise ratio (PSNR) is defined as

\[
PSNR = 10 \cdot \log_{10} \left( \frac{\text{MAX}_IM^2}{\text{MSE}} \right),
\]
where $MAX_{IM}$ is the maximum possible pixel value of the image.

From these definitions, the $MSE$ and $PSNR$ are congruent indices, and thus, we only use $PSNR$ as the evaluation index from here on.

### A. Noiseless Signal Recovery

The performance of our algorithms are compared with results in [7], [10], [11] by testing on the four $256 \times 256$ pixels images in Fig. 2. The peak signal-to-noise ratio ($PSNR$) for reconstruction of the images under various sampling ratios are reported in Table I. We do not have access to the codes of $l_1$-AMP, EM-GM-GAMP, SURE-AMP, so we have compared the PSNR results of these algorithms in [11] to our implementation of SGK-AMP. Our PSNR results are much better than those of $l_1$-AMP, EM-GM-GAMP, SURE-AMP, which highlights the efficacy of the DL-AMP framework over other AMP frameworks.

We use $128 \times 128$ pixels images in Fig. 3 to compare with the results in [9]. Table II lists the results of NLM-AMP, Bilateral-AMP, Gauss-AMP, BM3D-AMP, fast-BM3D-AMP and the results of MOD-AMP and SGK-AMP presented in this paper. The 20% sampling ratio implies that only 20% of the size of the original image was transferred and used for reconstruction of the original image. For 20%, 30% and 40% sampling ratios, the performances of MOD-AMP and SKG-AMP algorithms are better than any other algorithms barring only two cases. The performance of BM3D-AMP and fast-BM3D-AMP are comparable at lower sampling ratios to the DL-AMP algorithms (MOD-AMP and SGK-AMP), but the difference becomes significant with 40% sampling ratios.

Also, Fig. 4 and Fig. 5 show the PSNR of the images in Fig. 2 and Fig. 3 respectively for different sampling ratios, using the SGK-AMP algorithm. In Fig. 4, the PSNR of all images increase consistently with the increase in sampling ratio, except for house where the rise is slow. The interesting observation is that for most of the images SGK-AMP has an acceptable PSNR ($\geq 25$ dB) when the sampling ratio is 40% or less. The PSNR of the house image rises slowly on account of the higher inherent noise of the image. In Fig. 5, the brick and carpet pictures are the ones where the PSNR is low and climbs slowly. These images also have a lot of inherent noise in them and are low quality to start with.

### B. Noisy Signal Recovery

In the last subsection, we identified that the performance of SGK-AMP and MOD-AMP is better than the other AMP approaches. The two DL-AMP approaches perform similarly, but MOD-AMP has higher running time, so in this subsection we use only SGK-AMP to compare with the other non-DL algorithms.

In this section, we study the reconstruction of noisy signals. As representative images from the nature and the texture groups, we chose Boat and Fingerprint 1. For creating measurement noise in the signal we added an additive white Gaussian noise (AWGN) to the images. Fig. 6 and Fig. 7 show the $PSNR$ of different methods with different sampling ratios for AWGN of $10$ dB. Fig. 8 and Fig. 9 show the same comparison with AWGN of strength $SNR = 20$ dB. These figures imply that if the sampling ratio is under 20%, the BM3D-AMP is the best method. But if the sampling ratio is more than 20%, the SGK-AMP algorithm has the best performance. The gap in performance with just a small increase in sampling ratio is significant in favor of our algorithms.

### C. Runtime comparison

It is known that the DL algorithm is a slow. We did running time performance measurements of the algorithms on a Dell
## TABLE II

$PSNR$ of 128 $\times$ 128 Images Reconstructions with No Measurement Noise.

<table>
<thead>
<tr>
<th>20% Sampling</th>
<th>Nebula</th>
<th>Brick wall</th>
<th>Wood</th>
<th>Carpet</th>
<th>Fingerprint 1</th>
<th>Fingerprint 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>SGK-AMP</td>
<td>27.800</td>
<td>21.703</td>
<td>25.599</td>
<td>16.975</td>
<td>20.841</td>
<td>22.858</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>30% Sampling</th>
<th>Nebula</th>
<th>Brick wall</th>
<th>Wood</th>
<th>Carpet</th>
<th>Fingerprint 1</th>
<th>Fingerprint 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>BM3D-AMP [27]</td>
<td>29.322</td>
<td>22.902</td>
<td>27.177</td>
<td>18.872</td>
<td>22.688</td>
<td>24.617</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>40% Sampling</th>
<th>Nebula</th>
<th>Brick wall</th>
<th>Wood</th>
<th>Carpet</th>
<th>Fingerprint 1</th>
<th>Fingerprint 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOD-AMP</td>
<td>31.229</td>
<td>24.807</td>
<td>29.510</td>
<td>20.637</td>
<td>26.039</td>
<td>27.915</td>
</tr>
</tbody>
</table>

---

Precision T1500 running an Intel(R) Core™ i7-870 with 4 GB RAM, and the Matlab R2015a environment. Table III shows the runtimes of NLM-AMP, Bilateral-GAMP, Gauss-AMP, BM3D-AMP, fast-BM3D-AMP, and our methods’ results. As stated before, MOD-AMP is slow. SGK-AMP algorithm has much lower runtime (faster), which grows slowly with higher sampling ratios. From Table III, SGK-AMP takes longer for reconstruction than the non-DL AMP algorithms. However, if reconstruction quality is of importance and higher runtime can be tolerated, then the DL-AMP algorithms should be preferred. This is especially true for data coming from IoT devices. With the potential for packet losses and low bandwidth, reconstruction quality is very important. Running time is secondary—the algorithms run on a server that has no computation or energy constraints.

![Fig. 7. Boat for reconstruction with AWGN (SNR = 10 dB).](image1)

![Fig. 8. Fingerprint 1 for reconstruction with AWGN (SNR = 20 dB).](image2)

## TABLE III

Average runtime in seconds of 128x128 pixels images reconstruction with no measurement noise.

<table>
<thead>
<tr>
<th>Average Runtimes</th>
<th>10%</th>
<th>20%</th>
<th>30%</th>
<th>40%</th>
<th>50%</th>
</tr>
</thead>
<tbody>
<tr>
<td>NLM-AMP [26]</td>
<td>52.9</td>
<td>47.9</td>
<td>39.8</td>
<td>36.0</td>
<td>32.5</td>
</tr>
<tr>
<td>Bilateral-GAMP [9]</td>
<td>16.1</td>
<td>17.6</td>
<td>18.2</td>
<td>19.1</td>
<td>20.0</td>
</tr>
<tr>
<td>Gauss-AMP [9]</td>
<td>7.9</td>
<td>8.4</td>
<td>10.3</td>
<td>12.0</td>
<td>14.2</td>
</tr>
<tr>
<td>BM3D-AMP [27]</td>
<td>26.4</td>
<td>25.1</td>
<td>25.1</td>
<td>27.0</td>
<td>27.3</td>
</tr>
<tr>
<td>fast-BM3D-AMP [27]</td>
<td>16.3</td>
<td>16.3</td>
<td>14.9</td>
<td>15.7</td>
<td>17.2</td>
</tr>
<tr>
<td>SGK-AMP</td>
<td>136.0</td>
<td>113.3</td>
<td>124.2</td>
<td>144.9</td>
<td>172.8</td>
</tr>
<tr>
<td>MOD-AMP</td>
<td>217.5</td>
<td>679.6</td>
<td>2396.0</td>
<td>2880.3</td>
<td>4072.2</td>
</tr>
</tbody>
</table>
In this paper, we proposed a novel DL-AMP based CS framework for multimedia IoTs. First, we introduced the basic framework of AMP algorithms, and some dictionary learning algorithms. Then, we proved that DL algorithms satisfy the requirement of AMP. Finally, we presented new DL-AMP algorithms based on the proposed DL-AMP framework. With experiments we showed that the quality of the reconstructed signals are better with the DL-AMP framework (SGK-AMP) than those obtained with other algorithms in the literature. The DL denoisers’ runtime is usually longer than other denoisers, however, the resultant improvement in reconstruction makes the DL-AMP framework suitable for multimedia IoT devices. In the future, we will explore DL methods with lower runtimes and also collaborative CS with the DL-AMP framework.

REFERENCES


Zhicheng Li received his PhD and MS degrees in Control Science and Engineering from Harbin Institute of Technology, China in 2013 and 2009 respectively. He was a faculty member in Harbin Institute of Technology and now is the faculty member in Sun Yat-sen University. He is a research assistant professor in the Klipsch School of Electrical and Computer Engineering at the New Mexico State University (NMSU). His research interests involve compressed sensing, smart grid, robust control for Markovian jump systems, switched systems, and time-delay systems.

Hong Huang received his B.E. degree from Tsinghua University, Beijing, China, and M.S. and Ph.D. degrees from Georgia Institute of Technology in 2000 and 2002, respectively, all in electrical engineering. He is currently an associate professor with the Klipsch School of Electrical and Computer Engineering at NMSU. His current research interests include wireless sensor networks, mobile ad hoc networks, network security, and optical networks. He is a member of the IEEE.

Satyajayant Misra (SM’05, M’09) is an associate professor in computer science at New Mexico State University. He completed his M.Sc. in Physics and Information Systems from BITs, Pilani, India in 2003 and his Ph.D. in Computer Science from Arizona State University, Tempe, AZ, USA, in 2009. His research interests include wireless networks and the Internet, supercomputing, and smart grid architectures and protocols. He has served on several IEEE journal editorial boards and conference executive committees (Communications Magazine, SECON 2010, INFOCOM 2012). He has authored more than 45 peer-reviewed IEEE/ACM journal articles and conference proceedings. More information can be obtained at www.cs.nmsu.edu/~msira.