Bayesian Convolutional Neural Networks for Compressed Sensing Restoration

Xinjie Lan, Student Member, IEEE, Xin Guo, Student Member, IEEE, Kenneth E. Barner, Fellow, IEEE,

Abstract—Compressed Sensing (CS) is a novel paradigm for sampling theory, and signal structures beyond sparsity are commonly exploited to improve CS restoration performance. Conventional, Markov Random Fields (MRFs) are preferred to model signal structure. However, MRFs are hard to describe signal structure precisely and comprehensively due to their inherent drawbacks. Meanwhile, Convolutional Neural Networks (CNNs) have aroused great attention to realize CS restoration. However, CNNs lack convincing theory to clarify its working mechanism and to guide networks designing for CS restoration. In this paper, we prove the output of CNNs can be viewed as a Gibbs distribution and the whole CNNs architecture is equivalent to a Bayesian hierarchical model. That provides a comprehensive interpretation of CNNs in various dimensions and a novel perspective to reflect the existing neural networks for CS restoration. Following our proposed theory, a hybrid CS restoration algorithm is designed through combining CNNs and Bayesian inference method. Finally, extensive simulations demonstrate the validity and applicability of the proposed theory and algorithm.

Index Terms—Compressed Sensing, Markov Random Fields, Gibbs distribution, Inverse Problem, Bayesian hierarchical model, Convolutional Neural Networks.

I. INTRODUCTION

Sampling theory is a cornerstone of signal processing. Shannon/Nyquist sampling theory indicates that a bandlimited signal can be precisely recovered from its uniform samples as long as the sampling rate is at least two times of the bandwidth of this signal, namely the Nyquist rate \(^1\). However, the explosive growth of the Nyquist rate makes the bandwidth of this signal, namely the Nyquist rate \(^1\). However, the explosive growth of the Nyquist rate makes signal processing extremely difficult and complicated in the big data era. In other words, conventional sampling theory plagues our ability to acquire, transmit, and process high dimensional signal, and hence spurs an increasing interest in novel sampling theories and techniques.

Compressive Sensing (CS) provides an alternative paradigm for sampling theory \(^2, 3\). In general, \(K\)-sparse signal \(x \in \mathbb{R}^N\) is to be sampled by a linear measurement matrix \(A \in \mathbb{R}^{M \times N} (M < N)\), and the measurement \(y\) can be written in matrix form as \(y = Ax + n\), where \(n\) is noise. While \(M < N\) renders matrix \(A\) rank deficient, signal still can be accurately restored from \(M = O(K \log(N/K))\) measurements with high probability as long as \(A\) satisfies Restricted Isometry Property (RIP) \(^4, 5\).

Though CS has significantly changed traditional sampling theory, it only leverages sparsity to restore signal but ignores other pervasive signal structures, such as cluster structure \(^6\) and wavelet tree structure \(^7\). Consequently, Structured Compressed Sensing (SCS) was proposed to refine traditional CS by integrating extra signal structures \(^6, 8\). Specifically, SCS first exploits a mathematical model \(g(x; \theta)\) to describe signal structure, and then incorporates \(g(x; \theta)\) into a traditional CS restoration algorithm, where \(\theta\) denotes all parameters of this model \(g(x; \theta)\).

Since signal structure plays a crucial role in SCS restoration, the kernel of SCS is choosing an appropriate model \(g(x; \theta)\) to describe it precisely and comprehensively. Markov Random Fields (MRFs), an essential type of graphical model, has become a ubiquitous method for SCS due to its expressive and flexible way of describing signal structure \(^9\). In practice, lots of MRFs models have been applied to SCS restoration. For example, Total Variation (TV) \(^10, 11\), Ising model \(^12\), Spike and Slab model \(^13, 14\), and Gaussian Scale Mixture (GSM) \(^15, 16\) have been intensely investigated to model different signal structures in the framework of SCS.

While MRFs have been widely applied in SCS restoration, they still have some inherent issues. Above all, an essential problem of MRFs is lack of powerful parameter inference method. In the context of optimization, the inference of MRFs parameters can be viewed as an energy minimization problem, which is known to be NP-hard except for certain simple cases such as pairwise MRFs \(^9\). This essential problem makes two adverse effects on the application of MRFs. First, many MRFs models have to exploit hand-crafted clique potential functions, which are very difficult to express signal structure precisely unless taking lots of trials to modify parameters \(^13\). Second, the signal structure that MRFs can model is commonly restricted into a small neighborhood in order to reduce the complexity of parameter optimization, but most signal structures have abundant high-order dependencies \(^17\). Hence, traditional MRFs can hardly express signal structure comprehensively as well \(^18\). Moreover, certain practical limitations, such as requiring too many hyper-parameters, further hinder MRFs deriving optimal parameter.

As the resurgence of deep neural networks, Convolutional Neural Networks (CNNs) have aroused considerable attention in various applications. In particular, many neural networks were recently proposed for CS restoration based on CNNs \(^19, 20, 21\). In general, most neural networks aim for designing a complex inverse operator to reconstruct \(x\) from \(y\). Notably, some networks presented better restoration than the SCS algorithms mentioned above.

This material is based upon work supported by the National Science Foundation under Grant No. 1319598. The material in this paper was presented in part at IEEE GlobalSIP, Montreal, Canada, November 2017.

Xinjie Lan, Xin Guo, and Kenneth E. Barner are with Department of Electrical and Computer Engineering, University of Delaware, Newark, DE, 19711 USA (email: lxjbit, guoxin, barner@udel.edu)

19711 USA (email: lxjbit, guoxin, barner@udel.edu)
Despite the impressive success of CNNs, there is still no convincing theory to clarify how CNNs outperform traditional methods. Initially, CNNs were regarded as a discriminative model [22], but it merely establishes a statistical relation between the input and output of CNNs, the inside of CNNs is still a black box [23]. Meanwhile, the generative model provides a better perspective for interpreting the internal logic of CNNs [24]. It has been demonstrated that some shallow neural networks, such as Boltzmann machine [25] or recurrent Multilayer Perceptron (MLP) [26], are equivalent to Boltzmann distribution. However, Boltzmann distribution is hard to interpret deep neural networks like CNNs.

Consequently, new generative theories, such as Conditional Random Fields (CRF) [27], [28], Deep Gaussian Mixture Model (DGMM) [29], [30], and Deep Rendering Mixture Model (DRMM) [31], were recently proposed to explain CNNs. Although these theories revealed certain secrets of CNNs, they still had three defects. First, they only established a coarse connection between their proposed generative models and CNNs architecture, which is not enough to clarify the functionality of every module of CNNs, e.g., CRF cannot explain the commonly used non-linear layer, namely ReLU. Second, they cannot fully interpret the working mechanism of CNNs. For instance, DRMM is hard to elucidate what kind of knowledge CNNs learn from the training labels. Third, these theories were only valid for specific applications but not consistent in general situations. Hence, they are difficult to provide general principles to guide network designing.

In this paper, we attempt to shed light on the above limitations and present a novel algorithm for CS restoration. Overall, our contributions to this literature are three folds. First, we demonstrate the output of CNNs can be regarded as a Gibbs distribution and the whole CNNs architecture is equivalent to a Bayesian hierarchical model, which provides a rational explanation of CNNs in various dimensions, such as neuron, layer, and networks. Second, we point out a defect of the architecture of CNNs for CS restoration and propose a novel approach to circumvent this defect. Finally, a hybrid CS restoration algorithm is constituted by combining CNNs and Bayesian inference method based on our proposed theory.

This paper is organized as follows. Section II lays out the necessary background related to this topic. Furthermore, we present a novel explanation of CNNs and reflect the existing neural networks for CS restoration in Section III. Subsequently, Section IV specializes the hybrid CS restoration algorithm. In the end, numerical simulations validate our proposed theory and algorithm in Section VI.

II. BACKGROUND

This section reviews three fundamental topics involved our research: SCS, MRFs, and CNNs. First, a generalized solution to SCS restoration is derived in the framework of the inverse problem. Second, we decompose MRFs into three elements and analyze their respective functionalities. Finally, the third part discusses CNNs and summarizes the existing neural networks for CS restoration.

A. Structured Compressed Sensing

Since SCS can be regarded as a specific case of inverse problem, we first discuss the inverse problem. Suppose a system function \( F \) is to model a signal processing application, the output \( y \in R^M \) can be defined as \( y = F(x) + n \), where \( x \in R^N \) denotes the input and \( n \) is the system noise. Inverse problem aims to restore \( x \) from \( y \). In general, the dominant approach to solve the inverse problem is deriving an estimation \( \hat{x} \) from \( y \) by minimizing a predefined cost function \( e(F(x), y) \), i.e., \( \hat{x} = \arg \min_{x \in R^N} e(F(x), y) \) [22].

However, most inverse problems are ill-posed, such that it is tough to derive the optimal solution without regularization. Hence, the solution \( \hat{x} \) to the inverse problem is commonly relaxed as

\[
\hat{x} = \arg \min_{x \in R^N} e(F(x), y) + \lambda g(x; \theta)
\]

where \( g(x; \theta) \) denotes a regularization term of \( x \), namely the prior knowledge of \( x \), and \( \lambda \) is a constant parameter.

Regarding SCS, \( F \) is commonly defined as a random matrix \( A \) to satisfy RIP and signal structure can be expressed by \( g(x; \theta) \). Baraniuk et al. have proven that integrating signal structures beyond sparsity can improve the performance of CS restoration [6], [8]. Therefore, the most important work of SCS is to design an appropriate \( g(x; \theta) \).

Of all the SCS algorithms, Bayesian Compressed Sensing (BSC) has received much attention, since it has the advantage of naturally including arbitrary signal structure as long as designing a corresponding prior distribution [13], [43].

The prior distribution \( p(x; \theta) \) is typically chosen as the Gibbs distribution, which can be expressed as

\[
p(x; \theta) = \frac{1}{Z(\theta)} e^{-g(x; \theta)/T}
\]

where \( g(x; \theta) \) denotes the energy function conveying all signal structures. \( Z(\theta) \) and \( T \) are defined as the partition function and annealing temperature, respectively [34]. Moreover, since the noise \( n \) in CS is commonly assumed as a Gaussian distribution with unknown variance \( \sigma_n^2 \), the likelihood distribution of CS can be formulated as

\[
p(y|A, \sigma_n^2) = \frac{1}{\sqrt{(2\pi\sigma_n^2)^N}} \exp\left(-\frac{1}{2\sigma_n^2} \|Ax - y\|^2_2\right)
\]

Therefore, the posterior distribution \( p(x|A, y, \sigma_n^2) \) of \( x \) can be derived as follows based on Bayesian rule.

\[
p(x|A, y, \sigma_n^2) \propto p(y|x, A, \sigma_n^2) \cdot p(x; \theta)
\]

If \( e(F(x), y) \) is chosen as Maximum a Posteriori (MAP), SCS restoration can be derived below based on Equation (4).

\[
\hat{x}_{SCS} = \arg \min_{x \in R^N} \|Ax - y\|^2_2 + \frac{2\sigma_n^2}{T} \cdot g(x; \theta)
\]

Equation (5) can be regarded as a generalized SCS solution, since it can include arbitrary signal structure by specifying \( g(x; \theta) \). Moreover, BCS provides a feasible approach to realize SCS restoration through designing a prior distribution.
the clique type determines the dimension of $F$. For instance, clique types $C_2$ and $C_3$ make $\Delta_{C_2}(x)$ and $\Delta_{C_3}(x)$ as $3 \times 3$ convolutional matrices. Second, the clique type confines the entries of $F$, e.g., $\Delta_{C_2}(x)$ and $\Delta_{C_3}(x)$ only have two and three non-zero elements and can be expressed below:

\[ \Delta_{C_2}(x) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & \alpha_1 & 0 \\ 0 & \alpha_2 & 0 \end{bmatrix}, \quad \Delta_{C_3}(x) = \begin{bmatrix} 0 & \beta_2 & 0 \\ 0 & \beta_1 & \beta_3 \\ 0 & 0 & 0 \end{bmatrix} \]

where $\alpha_i, \beta_j$ are non-zero elements. In order to keep the integrity of discussion structure, we will discuss how to derive the optimal values of $\alpha_i, \beta_j$, and $b$ in the third question.

For the second question, the difficulty of designing $f_{NL}^c$ is to simultaneously satisfy two conditions: (i) accurately simulate the non-linearity of signal and (ii) the generated prior distribution \( \mathcal{F} \) should be a conjugate prior for likelihood distribution \( \mathcal{G} \), such that posterior inference is feasible. The most popular solution to this problem is Hidden MRFs (HMRFs), which include a pair of random fields \((X, Z)\), where $X = \{x_i, i \in S\}$ is an observable random field and $Z = \{z_i, i \in S\}$ is a hidden MRF connected to $X$ \[1\]. In order to satisfy the second condition, the distribution of $X$ is commonly chosen as a conjugate prior distribution $p_{con}(x; \theta)$, such that $f_{NL}^c$ can be derived as $f_{NL}^c = -\log(p_{con}(x; \theta))$. For example, spike and slab model is a typical HMRFs, which is defined as follows \[13\].

\[ p(x; \theta) = (1 - \pi)\delta_0 + \pi N(x; 0, \alpha^{-2}) \]

where $\delta_0$ is the Dirac function, $\pi$ denotes the weight of the Gaussian distribution, and precision $\alpha^2$ is the reciprocal of variance. In the context of SCS restoration, $\pi$ is commonly extended to the hidden MRF $Z$ to describe signal structure \[13\], \[14\], \[18\]. For the sake of satisfying the first condition, HMRFs need a powerful learning algorithm to optimize all parameters related to $p_{con}(x; \theta)$.

For the third fundamental question, the parameter set $\theta$ includes all parameters related to $\Delta_c(x)$ and $f_{NL}^c$. Bayesian posterior inference algorithm usually regards $\theta$ as a prior distribution $p(\theta)$ as well. Hence, $p(x|A, y, \theta)$ \[4\] will be extended as follows:

\[ p(x|A, y, \theta) \propto p(y|A, x, \theta) \cdot p(x|\theta) \cdot p(\theta) \]

However, regarding $\theta$ as a prior distribution introduces too many hyper-parameters in the Bayesian posterior inference algorithm, which makes obtaining the optimal parameters difficult. For instance, if the spike and slab model is chosen as $p(x|\theta)$ and signal structure is embedded into $\pi$, then $\theta = \{\alpha, \alpha_n, \pi\}$ and $p(\theta) = p(\alpha) \cdot p(\alpha_n) \cdot p(\pi)$. In order to guarantee $p(x|\theta)$ is still a conjugate prior, the distribution of each parameter is typically chosen as $p(\alpha) = \text{Gamma}(a_0, b_0)$, $p(\alpha_n) = \text{Gamma}(e_0, d_0)$, and $p(\pi) = \text{Beta}(e_0, f_0)$, where $a_0, b_0, \cdots, f_0$ denote the hyper-parameters of $\alpha, \alpha_n$, and $\pi$, respectively. It shows that if we want to infer parameters $\theta$, we need much more hyper-parameters. However there is no optimal solution to choose these hyper-parameters except trial and error. That means traditional Bayesian posterior inference algorithm is difficult to derive the optimal parameters $\theta$. 

---

**Fig. 1:** MRFs Neighborhood System and Clique Types

**B. Markov Random Fields**

As a probabilistic graphical model, MRFs is designed to analyze the spatial or temporal dependence of signal \[9\], \[35\]. Specifically, random fields $X$ can be regarded as MRFs regarding a neighborhood system $N_d$ on sites $S$ if and only if

\[ p(x_s) > 0, \forall x_s \in X \text{ and } p(x_s|x_{S-\{s\}}) = p(x_s|x_{N_d^s}) \]

where $S = \{1, 2, \cdots, m\}$ is a set of $m$ sites to index a point or region in the Euclidean space, $x_{S-\{s\}}$ means all entries except $x_s$, and $N_d^s = \{s_n \in S | ||s-s_n|| \leq d^2\}$ is a homogeneous neighborhood system to indicate all neighbors surrounding $s$ within radius $d$. In short, MRFs indicates the probability of $x_s$ only depends on its local neighbors in $X$.

Based on the Hammersley-Clifford theorem \[35\], MRFs can be equivalently characterized by the Gibbs distribution expressed in Equation \[2\]. The energy function $g(x; \theta)$ of MRFs consists of multiple clique potential functions $f_c(x; \theta_c)$, i.e., $g(x; \theta) = \sum_{c \in C} f_c(x_c; \theta_c)$, where $f_c(x_c; \theta_c)$ describes the signal structure in the clique $c$, which is a subset of sites. Besides, every pair of sites in the clique $c$ are connected with each other. For example, $N_1$ has two clique types $C_1$ and $C_2$, $N_2$ has two more cliques $C_3$ and $C_4$. Their topological structures are depicted in Fig. [1].

Since $g(x; \theta)$ includes all the information of signal structure, it is the kernel of MRFs. One typical way of designing $g(x; \theta)$ is the linear filter $\Delta_c(x)$, but it can hardly represent non-linear structure, such as object boundary \[35\]. An effective remedy for that is adding a non-linear function $f_{NL}^c$ outside of $\Delta_c(x)$, i.e., $f_c(x_c; \theta_c) = f_{NL}^c(\Delta_c(x); \theta_c)$. Hence, $g(x; \theta)$ can be formulated as

\[ g(x; \theta) = \sum_{c \in C} f_c(x_c; \theta_c) = \sum_{c \in C} f_{NL}^c(\Delta_c(x); \theta_c) \]

(6)

where $\theta$ denotes the union of all $\theta_c$. Consequently, three fundamental questions involved MRFs are how to derive the optimal $\Delta_c(x)$, $f_{NL}^c$, and $\theta$ for a specific application.

For the first question, $\Delta_c(x)$ can be rewritten as the convolutional form $\Delta_c(x) = F \ast x + b$ because it is linear, where $F$ denotes the convolutional matrix and $b$ is the bias. Therefore, this question is equivalent to determine $F$ and $b$. Regarding $F$, it largely depends on the clique type. First,
Inverse connected layer

Refining Networks

Fig. 2: The architecture of the inverse CNNs. ReconNet uses this architecture to implement CS restoration. The inverse networks contain single fully connected layer, and the refining networks include six convolution and ReLU layers.

Since lack of powerful parameter inference algorithm, most MRFs models have no choice but to exploit hand-crafted linear filters to describe signal structure and choose a simple conjugate prior distribution like the spike and slab model to preserve non-linearity. However, that is not enough to establish a powerful prior distribution for SCS restoration. Though other optimization algorithms, e.g., Expectation-Maximization (EM), can improve this situation, it still has certain limitations, such as slow convergence ratio for high dimensional applications, sensitive to the initial values, etc. Overall, the parameter inference is critical to MRFs, but there is no optimal solution now.

C. Convolutional Neural Networks

Artificial Neural Networks (ANNs) are designed to solve complex problems via constituting intelligent networks. By mimicking biological neural networks, the primary element of ANNs is neuron as well, which is defined as

\[ y = f(x) = f^{\text{act}}(\sum_i \omega_i \cdot x_i) \]

where \( x_i \) denotes the input nodes connected to this neuron, and \( \omega_i \) indicates the weight of \( x_i \). In addition, \( f^{\text{act}} \) is an activation function and it is usually non-linear. Similar to biological neural networks, multiple neurons constitute one layer \( f = \{ f_1, ..., f_N \} \) and multiple layers compose the whole ANNs = \{ \( y; f_N; ...; f_1; x \) \}.

In the development of ANNs, the most fundamental ANNs is Multilayer Perceptron (MLP). Fig. 3 shows a simple but comprehensive example of MLP. This MLP has two hidden layers \( L1 \) and \( L2 \), which contains four and three neurons denoted by \( f_i \) and \( s_j \), respectively. Based on the definition of neuron, the output of this MLP can be formulated as

\[ y_m = y^{\text{act}}(\sum_k \gamma_{mk} \cdot s^\text{act}(\sum_j (\beta_{kj} \cdot (f^\text{act}(\sum_i (\alpha_{ji} \cdot x_i))))) \]

where \( \alpha_{ji} \) is the weight of the \( i \)th input node \( x_i \) for the \( j \)th neuron \( f_j \), and \( f^\text{act} \) denotes the activation function in the first layer, \( \beta_{kj}, s^\text{act}, \gamma_{mk}, \) and \( y^{\text{act}} \) have similar definitions.

CNNs have similar architecture as MLP, but they exploit new hidden layers, especially convolutional, Rectified Linear Unit (ReLU), and pooling layers. In contrast to MLP assuming that the neurons in a hidden layer are connected to all neurons in the previous layer, convolutional layer constrains the connections to be local, so it can extract local signal features from the previous layer. ReLU (\( y = \max(0, x) \)) can be viewed as a novel activation function aiming for sparse representation and preserving non-linearity. Moreover, the role of pooling layer is merging similar signal features to avoid overfitting. Leveraging these novel hidden layers, CNNs achieved impressive success in image classification and triggered the renaissance of ANNs in many fields.

Lots of efforts recently were devoted to applying CNNs to CS restoration. Since standard CNNs can be viewed as a function projecting the high dimensional input \( x \) such as an image dataset, to the low dimensional output \( y \) like classification labels, the most intuitive way of designing neural networks for CS restoration is directly reversing the standard CNNs. The derived architecture is denoted as inverse CNNs here (Fig. 2). Moreover, the inverse CNNs can be divided into two sub-networks: the inverse and refining networks. The former is to project the low dimensional measurement \( y \) to a high dimensional output \( x_P \) through fully connected layer(s). The later aims to derive the final restoration \( \hat{x} \) via refining \( x_P \) by convolutional and ReLU layers.

Based on this intuition, some neural networks like ReconNet were proposed for CS restoration. However, ReconNet only exhibited similar restoration as traditional MRFs models, and even worse than the later in some cases. Hence, new neural networks, such as DR2Net, were proposed to improve this situation and its general architecture is depicted in Fig. 4.
Unlike ReconNet training the inverse and refining networks simultaneously, DR2Net first designed a linear map through training the inverse networks only, which can generate a better restoration $\hat{x}$. Subsequently, $x_P$ was fed into the refining networks to derive $\hat{x}$. Furthermore, the residual technique made the refining networks of DR2Net only reconstruct the residual $x_R$ of $\hat{x}$ given $x_P$, namely $\hat{x} = x_P + x_R$ [44]. These refinements helped DR2Net to outperform ReconNet and traditional MRFs models.

Furthermore, it has been shown that replacing the inverse networks with a traditional CS restoration method can improve recovery performance further, e.g., Learning Denoiser Approximate Message Passing (LDAMP) outperformed DR2Net though replacing the inverse networks with AMP method while keeping the same refining networks as DR2Net [21], [37]. Its architecture is shown in Fig. 4.

It is important to note that all existing CNNs theories cannot clarify the different performances of these three neural networks. Fig. 4 shows that their main difference is the inverse networks. Besides, the residual technique is another important factor. But these differences can hardly explain their different recovery performances. In the following sections, we propose a novel theory to explain the working mechanism of CNNs, which provides a new perspective to answer this question. Moreover, we carry out extensive simulations to evaluate the influence of the inverse networks and refining networks on CS restoration, which verifies our proposed theory further.

III. MRFs AND CNNS

This section demonstrates our proposed theory of CNNs. First, we demonstrate the statistical relationship between MLP and Bayesian hierarchical model and prove CNNs have the same statistical relationship. Second, we exploit the proposed theory to explain the working mechanism of CNNs. Finally, we reveal a defect of the inverse CNNs for CS restoration and propose a hybrid approach to circumvent this defect through combining CNNs and Bayesian inference algorithm. For simplicity, MLP-Ln is used to denote MLP with $n$ hidden layer(s) in this section.

A. Proposing A New Theory about CNNs

**Lemma 1:** the output of MLP-L1 is equivalent to a Gibbs distribution of the input.

Suppose MLP-L1 = $\{y; f; x\}$, the input $x = \{x_1, ..., x_N\}$, and the hidden layer $f$ has $K$ neurons, which are defined as $f_k = f_{k}^{act}(\sum_{n=1}^{N} \alpha_{kn} \cdot x_n)$, the softmax output $y = \{y_1, ..., y_M\}$ of the MLP-L1 can be expressed as

$$y_m = \frac{1}{Z_{\theta}(\omega)} \exp\{-\sum_{k=1}^{K} \beta_{mk} \cdot f_{k}^{act}(\sum_{n=1}^{N} \alpha_{kn} \cdot x_n)\}$$

where $Z_{\theta}(\omega) = \sum_{m=1}^{M} e^{-\sum_{k=1}^{K} \beta_{mk} f_{k}(\sum_{n=1}^{N} (\alpha_{kn} \cdot x_n))}$ is the partition function, and $\beta_{mk}$ indicates the weight of the neuron $f_k$ for the output $y_m$.

We can find that $y$ represents a Gibbs distribution of $x$ by comparing Equation (11) with (2) and (6) in three aspects. First, the neuron $f_{k}$ can be viewed as a clique potential function, in which the weights $\alpha_{kn}$ represents a linear filter $\Delta_{kn}$ to describe a signal structure, namely $\Delta_{kn}(x) = (\sum_{n=1}^{N} \alpha_{kn} \cdot x_n)$, and $f_{k}^{act}$ corresponds to a non-linear function. Second, multiple neurons composing a single hidden layer is consistent with the definition that the energy function is the summation of multiple clique potential functions in Equation (6). Third, $\beta_{mk}$ can be viewed as the annealing temperature $T$ for a specific output $y_m$.

Meanwhile, arbitrary Gibbs distribution can be converted to a MLP-L1 based on the relationship between neuron and clique potential function. Hence, the output of MLP-L1 is equivalent to a Gibbs distribution of the input. It is important to note that the energy function is a sufficient statistic of Gibbs distribution [45], which means we can obtain all the information of Gibbs distribution given the energy function. Therefore, we can conclude that a single hidden layer defines a Gibbs distribution to some extent.

**Lemma 2:** the distribution of each hidden layer in arbitrary MLP can be viewed as a conditional Gibbs distribution given the previous layers and the input of MLP.

In arbitrary MLP, one hidden layer’s input is typically the output of the previous layer(s). Therefore, the distribution of each hidden layer within MLP can be viewed as a conditional Gibbs distribution given the previous hidden layer(s) and the input data. Specifically, the distribution of the $n$th hidden layer within MLP-Ln = $\{y; f_n; ..., f_1; x\}$ can be formulated as

$$p(f_n | f_{n-1}, ..., f_1, x) = \frac{1}{Z_{\omega_n}(\theta)} e^{-\sum_{k} f_{k}^{act}(\alpha_k \cdot q_{n-1}(x))}$$

(12)

where $q_{n-1}(x)$ denotes the output of the previous $(n-1)$ hidden layers given the input $x$, and $\alpha_k$ is a vector to indicate the weight of each element of $q_{n-1}(x)$ for the neuron $f_k$. In addition, $f_{k}^{act}$ denotes the activation function of the $k$th neuron $f_k$ in the $n$th hidden layer.

**Theorem 1:** the output of MLP can be viewed as a generalized MRFs of the input.

Based on Lemma 1 and Lemma 2, we can prove that the output of MLP with arbitrary layer(s) is equivalent to a Gibbs distribution of the input and all hidden layers of MLP are designed to formulate the corresponding energy function.

Furthermore, the output of MLP can be regarded as a generalized MRFs based on Hammersley-Clifford theorem [35]. Since MLP assumes that the neurons in a hidden layer are connected to all neurons in the previous layer rather than merely considering local dependence, it is rational to regard the output of MLP as a generalized MRFs.
After establishing a statistical explanation for the output of MLP, Theorem 2 attempts to provide a new generative explanation of the whole MLP architecture.

**Theorem 2:** the whole architecture of MLP can be viewed as a Bayesian hierarchical model.

Lemma 2 shows each hidden layer of MLP is equivalent to a conditional Gibbs distribution. Without loss of generality, the first hidden layer can be viewed as a joint Gibbs distribution $p(f_1, x)$ [15], since the parameters of $f_1$ also can be viewed as random variables during the training procedure.

$$p(f_1, x) = \frac{1}{Z_{f_1}(\theta)} e^{-g_{f_1}(x; \theta)}$$

(13)

We can find that multiple conditional Gibbs distributions generate a Bayesian hierarchical model, while multiple layers of MLP are concatenating the whole networks. Specifically, the Bayesian hierarchical model corresponding to MLP-LN = \{y; f_N; \ldots; f_1; x\} can be expressed as

$$p(y, f_N, \ldots, f_1, x) = p(y|f_N, \ldots, f_1) \cdot p(f_N|f_{N-1}, \ldots, f_1, x) \cdots p(f_1, x)$$

(14)

For instance, the MLP-L2 depicted in Fig. 3 can be formulated as a Bayesian hierarchical model with three levels.

$$p(y, s, f, x) = p(y|s, f, x) \cdot p(s|f, x) \cdot p(f, x)$$

(15)

**Lemma 3:** the output of CNNs is equivalent to a Gibbs distribution of the input of CNNs, and the whole architecture of CNNs can be viewed as a Bayesian hierarchical model.

LeCunn et al. demonstrated CNNs as a variation of MLP based on two reasons [46]. First, the combination of convolutional and ReLU layers can be regarded as a single hidden layer of MLP, in which convolutional filter quantifies the weights of different neurons/nodes and ReLU corresponds to an activation function. Second, the fully connected layer is identical to a single hidden layer of MLP for combining all signal features from previous layers and extracting more abstract information.

Based on the equivalence between CNNs and MLP, we can derive the same explanation of CNNs, which provides a new interpretation of CNNs architecture and working mechanism.

**B. Explaining CNNs’ Architecture and Working Mechanism**

In terms of CNNs architecture, above lemmas and theorems establish an exact one-to-one correspondence between CNNs and traditional statistical model in various dimensions. Starting from the microscopic viewpoint, Lemma 1 proves that each neuron can be viewed as a clique potential function of Gibbs distribution. As multiple neurons evolve to a single hidden layer, multiple clique potential functions compose the kernel of Gibbs distribution, namely energy function. As we magnify our perspective, multiple conditional Gibbs distributions form the corresponding Bayesian hierarchical model, while multiple hidden layers concatenate the entire neural networks.

Moreover, our proposed theory provides a novel framework to understand the working mechanism of CNNs. Bayesian hierarchical model suggests that the testing and training procedures of CNNs can be explained as two conditional distributions. For the training procedure, it can be viewed as a conditional distribution of the output $y$ given the testing dataset $x$ and all hidden layers, which is formulated as

$$p(y|f_N, \ldots, f_1, x) = \frac{p(y, f_N, \ldots, f_1, x)}{p(f_N, \ldots, f_1, x)} \propto p(y|f_N, \ldots, f_1, x)$$

(16)

Meanwhile, the training procedure can be formulated as another conditional distribution of all hidden layers given the training dataset $x$ and label $y$.

$$p(f_N, \ldots, f_1|y, x) = \frac{p(y, f_N, \ldots, f_1, x)}{p(y|x)} \propto p(y|f_N, \ldots, f_1, x)$$

(17)

Though $p(f_N, \ldots, f_1|y, x)$ is intractable, we can find that $p(y, f_N, \ldots, f_1, x)$ is still a Gibbs distribution, which can covert the learning procedure of CNNs into the traditional energy minimization problem. That explains why the gradient descent algorithm and its variations like back-propagation [47] can be applied to optimize the parameters of CNNs.

Furthermore, our proposed theory offers a new viewpoint to understand what kind of knowledge CNNs learn from the training dataset. Bayesian theory indicates that the joint distribution associated with CNNs [18] can be decomposed into two components: the prior distribution of $x$ and the likelihood distribution between $x$ and $y$.

$$p(y, f_N, \ldots, f_1, x) = p(y|f_N, \ldots, f_1, x) \cdots p(f_N|f_{N-1}, \ldots, f_1, x) \cdots p(f_1, x)$$

(18)

Hence, the training procedure of CNNs can be interpreted as an action to extract the prior knowledge of the training dataset and formulate a statistical relation between the extracted prior knowledge and the predefined output labels.

Overall, our proposed theory presents a novel viewpoint to clarify the architecture and working mechanism of CNNs. Subsequently, we will analyze the existing CNNs architectures for various applications based on our proposed theory.

**C. Reflecting The Existing Architectures of CNNs**

For image classification, Hinton et al. claimed that the great success of the standard CNNs is because they can fully exploit the hierarchical properties of natural images [42]. Lemma 3 provides a mathematical basis for this claim. In general, each hidden layer of the standard CNNs forms a conditional Gibbs distribution to generate new features based on the features extracted from previous layers. Stacking multiple hidden layers forms a Bayesian hierarchical model, which combines all features in a hierarchical way.

For CS restoration, Equation (18) reveals an inherent defect of the inverse CNNs (Fig. 2). Since the input of the inverse CNNs (i.e., CS measurement $y$) is typically compressed and corrupted, it can hardly provide enough prior knowledge for the inverse CNNs to learn an expressive prior distribution even if they have complex architecture. Regarding CS restoration, this defect significantly constrains the performance of the inverse CNNs. Also note that it is true for most inverse problems in a broad sense.
TABLE I: Four Categories of CS Restoration Methods

| Category | Prior     | Likelihood | Example   |
|----------|-----------|------------|-----------|
| 1        | statistical| statistical| BCS       |
| 2        | statistical| networks   | LDAMP     |
| 3        | networks   | statistical| None      |
| 4        | networks   | networks   | ReconNet  |

This defect answers the question proposed at the end of Section II-C. It suggests that the inverse networks of ReconNet can not extract enough prior knowledge from the measurement $y$, which results in its worse restoration than the traditional MRFs models. In contrast to ReconNet, DR2Net and LDAMP can introduce extra prior knowledge leveraging the pre-trained linear map and AMP [13]. Therefore, they can outperform ReconNet and the traditional MRFs models.

Furthermore, Equation (13) indicates a new approach to realize CS restoration. Based on how to express the prior and likelihood distributions, CS restoration algorithms can be divided into four categories, which are presented in Table I. Traditional BCS methods belong to the first category, since they choose specific prior and likelihood distributions, such as GMM or Gaussian, to realize restoration. LDAMP can be roughly categorized into the second category since it utilizes neural networks to refine the traditional AMP method. Besides, ReconNet and DR2Net can be classified into the fourth one. But until now, nobody attempts to use the third approach.

Notably, we find the third approach can circumvent the defect of the inverse CNNs. Lemma 1 and 3 suggest that the standard CNNs can be trained as a prior distribution if we choose an image dataset as the input, which can provide much more prior knowledge than CS measurement. Therefore, this approach can circumvent the defect of the inverse CNNs. After generating a powerful prior distribution leveraging the standard CNNs, we can derive the posterior distribution, and CS restoration can be realized by Bayesian posterior inference.

IV. BAYESIAN CNNS METHOD FOR CS RESTORATION

This section shows our proposed Bayesian CNNs (BCNs) algorithm for CS restoration. First, a new CNNs architecture is designed to train the prior distribution $p(x; \theta)$. Second, we propose a new Bayesian inference method for CS restoration based on our previous work [15].

A. The Architecture of CNNs for Modeling Prior Distribution

Since CNNs are only used to train $p(x; \theta)$ given a training image set, it may not require very deep neural networks as the inverse CNNs, but it must have closed-form expression. To satisfy this condition, we propose a new architecture of CNNs (Fig. 3), which includes three hidden layers: convolutional, non-linear, and fully connected layers. If the output layer is defined as softmax, it can be formulated as

$$p(x; \theta) = \frac{1}{Z(\theta)} e^{-\sum_{m=1}^{M} (f_{m}^{NL}(f_{m}(x)))}$$  \hspace{1cm} (19)$$

where $f_{m}$ and $f_{m}^{NL}$ denote convolutional and non-linear layer, respectively. The term $-\sum_{m=1}^{M} (f_{m}^{NL}(f_{m}(x)))$ represents the output of the fully-connected layer.

To ensure posterior inference is feasible, $p(x; \theta)$ should be a conjugate prior for the likelihood distribution of CS [3]. Hence, the non-linear layer is chosen as GMM and expressed as

$$f_{m}^{NL} = -\log[\sum_{n=1}^{N} \pi_{mn} \cdot N(f_{m}(x); 0, \sigma_{b}^{2} \delta_{n})],$$

where $\sigma_{b}^{2}$ is a fixed base variance, $\delta_{n}$ is a broad range of constant scales, and $\pi_{mn}$ denotes the weight of each Gaussian distribution. Ultimately, the $p(x; \theta)$ generated from the proposed CNNs can be formulated as

$$p(x; \theta) = \frac{1}{Z(\theta)} \prod_{m=1}^{M} \sum_{n=1}^{N} \pi_{mn} \cdot N(f_{m}(x); 0, \sigma_{b}^{2})$$  \hspace{1cm} (20)$$

After specifying the CNNs architecture, the next problem is to construct the learning algorithm to optimize $\theta = \{f, \pi\}$, where $f = \{f_{1}, ..., f_{m}\}$ and $\pi = \{\pi_{11}, ..., \pi_{NM}\}$. Here we choose Kullback-Leibler divergence (KLD) [48] as the criterion to measure the distance between $p(x^{prior}; \theta)$ and $p(x^{data}; \theta)$. KLD is formulated as

$$\text{KLD}(p(x^{prior}; \theta)||p(x^{data}; \theta)) = \sum_{i} p(x^{data}(i); \theta) \log \frac{p(x^{data}(i); \theta)}{p(x^{prior}(i); \theta)},$$

(21)

where $p(x^{prior}; \theta)$ is the stationary distribution generated by sampling $p(x; \theta)$, and $p(x^{data}; \theta)$ denotes the empirical distribution of the training dataset, namely the output of the proposed CNNs. In order to make $p(x; \theta)$ model the statistical property of the training dataset as precisely as possible, the KLD should be minimized, which can be realized by Maximum-likelihood (ML) learning algorithm [48].

In practice, it is time-consuming to obtain $p(x^{prior}; \theta)$ by sampling $p(x; \theta)$ continuously until stationary. Therefore, Contrastive Divergence (CD) learning method [49] is adopted to quickly optimize $\theta = \{f, \pi\}$, and it can be formulated as

$$\theta_{n+1} = \theta_{n} - \eta \cdot \nabla_{\theta} \text{C}

\text{C} = \text{C}_{\text{prior}} - \text{C}_{\text{data}}$$

where $\theta_{n}$ denote the optimized parameters at the $n$th training epoch. Besides, $\nabla_{\theta} \text{C} = \frac{\delta \text{C}}{\delta \theta}$, where $\eta$ indicates the average over $p$, and $\eta \in (0, 1]$ is the learning rate.

Since CD only takes few $k$ sampling iterations to estimate $p(x; \theta)$, it can increase training speed greatly while obtain similar training result. The proposed CNNs learning algorithm is summarized in Algorithm [1]
Algorithm 1 CNNs learning algorithm for prior distribution

input: training data \(TD\), CNNs model
1: initialize
2: GSM scales \(\delta_n\), base variance \(\sigma_b^2\)
3: CNNs parameters \(\theta = \{f, \pi\}\)
4: sampling iteration \(k\)
5: training iteration count \(n = 0\)
6: training parameters, e.g., learning rate \(\eta\), batch size.
7: repeat
8: prepare training batch \(TB_n\) from \(TD\)
9: compute \(<V\text{CNN}>)_{\theta_j}(x_{data}, \theta_n)\) based on \(TB_n\)
10: obtain training label \(TL_n\) via sampling \(p(x_{prior}; \theta_n)\)
11: compute \(<V\text{CNN}>)_{\theta_j}(x_{prior}; \theta_n)\) based on \(TL_n\)
12: update \(\theta_n\) based on gradient descent method \(22\)
13: \(n \leftarrow n + 1\)
14: until \((|\theta_{n+1} - \theta_n| < \xi \text{ or } n > N)\)
output: optimal parameter \(\theta^* = \theta_n\)

B. Bayesian CNNs Inference for CS restoration

While the proposed CNNs has much simpler architecture than the inverse CNNs, the corresponding \(p(x; \theta)\) \(20\) is still intractable for most conventional inference algorithms \(34\). Therefore, auxiliary variable Gibbs sampler \(15\) is exploited to simplify \(p(x; \theta)\) for applicable posterior inference.

Since \(p(x; \theta)\) \(20\) can be viewed as a discrete case of GSM model \(50\), which is defined as
\[
p(x; z) = \int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi}\sigma_n} \exp\left(-\frac{(z - x)^2}{2\sigma_n^2}\right) p(z) dz \tag{23}
\]
where \(z\) denotes the auxiliary random vector \(z \in \{1, \ldots, N\}\) to represent the scale \(\delta_n\) in Equation \(20\), \(p(x; \theta)\) thus can be augmented into a new joint distribution \(p(x, z; \theta)\).

Given \(p(x, z; \theta)\), two conditional distributions \(p(z|x; \theta)\) and \(p(z|x; \theta)\) can be derived as follows.
\[
p(z_m|x; \theta) \propto \pi_{m} \cdot \mathcal{N}(f_m(x); 0, \frac{\sigma_b^2}{\sigma_m^2}) \tag{24}
\]
\[
p(x|z; \theta) \propto \prod_{m=1}^{M'} \mathcal{N}(f_m(x); 0, \frac{\sigma_b^2}{\sigma_m^2}) \tag{25}
\]
Subsequently, the posterior distribution \(\mathcal{N}\) of CS restoration can be formulated below based on \(25\) and \(2\).
\[
p(x|A, y, z; \sigma_n^2; \theta) \propto \prod_{m=1}^{M'} \mathcal{N}(f_m(x); 0, \frac{\sigma_b^2}{\sigma_m^2}) p(y|x, A, \sigma_n^2) \tag{26}
\]
Since all \(f_m\) and \(A\) are linear, \(p(x|A, y, z; \sigma_n^2; \theta)\) can be rewritten in matrix form to get more intuitive expression.
\[
p(x|A, y, z; \sigma_n^2; \theta) \propto \exp\left(-\hat{F}(x-\hat{\mu})^T \hat{\Sigma}^{-1}(x-\hat{\mu})\right) \tag{27}
\]
where
\[
\hat{F} = \begin{bmatrix} F_1 \\ \vdots \\ F_m \\ A \end{bmatrix}, \quad \hat{\Sigma} = \begin{bmatrix} \sigma^2_1 & 0 & \cdots & 0 \\ 0 & \ddots & \vdots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & \sigma^2_m & \sigma^2_n I \end{bmatrix}, \quad \hat{\mu} = \begin{bmatrix} \hat{\mu}_1 \\ \vdots \\ \hat{\mu}_m \end{bmatrix}
\]

We can find that Equation \(27\) is a Gaussian distribution, such that CS restoration can be inferred through alternately sampling \(p(x|A, y, z; \sigma_n^2; \theta)\) and \(p(z|x; \theta)\) in Algorithm 2

Algorithm 2 Bayesian CNNs algorithm for CS restoration

input: \(y, A, p(x; \theta)\), and \(p(y|x, A, \sigma_n^2)\)
1: initialize
2: randomly initialize \(x, z, \sigma_n^2\)
3: sampling iteration \(n\)
4: repeat
5: sampling \(p(z_m|x; \theta)\)
6: sampling \(p(x|A, y, z; \sigma_n^2)\)
7: sampling \(p(\sigma_n^2|x, y, A) = \mathcal{G}(\frac{M}{2} + 1, \frac{2}{\|y - Ax\|^2})\)
8: \(n \leftarrow n - 1\)
9: until \((n = 0)\)
output: recovery signal \(x^* = x\)

V. EXPERIMENTAL RESULTS

This section presents our experimental results. First, BCNNs are evaluated in CS image restoration field given various hyper-parameters. Second, BCNNs are compared to the state-of-the-art CS restoration algorithms in various situations. Besides, BCNNs are realized by MATLAB, and all related simulation code, training image set are available online\(1\)

A. Experimental Setup

There are three hyper-parameters of BCNNs: convolutional depth \(F_n\) (i.e., the number of \(f_m\)), filter dimension \(N_d\), and GSM scales \(\delta\) need to initialize ahead of implementation.

In order to evaluate the influence of different hyper-parameters on BCNNs, five models of BCNNs are instantiated and summarized in Table II. Regarding BCNN2, it has four convolutional filters, namely \(F_n = 4\), and each filter dimension is defined by \(N_d\) as shown in Fig. 1. Also the GSM scales \(\delta\) of BCNN2 are initialized as \(\delta_1 = \{\exp(-7, -3, 0, 3, 7)\}\). Therefore, BCNN2 can be expressed as follows.
\[
p(x; \theta) = \frac{1}{z(\theta)} \prod_{m=1}^{N} \sum_{n=1}^{5} \pi_{mn} \cdot \mathcal{N}(f_m(x); 0, \frac{\sigma_b^2}{\sigma_m^2}) \tag{28}
\]
In addition, BCNN2 has 56 parameters need to learn, i.e., \(\theta = F_n \times ((f_m) + |\pi|) = 56\), where \(|f_m|\) and \(|\pi|\) denote the number of parameters of \(f_m\) and \(\pi_{mn}\), respectively.

Similarly, other models can be instantiated based on Table II in which \(N_d\) denotes the filter dimension is \(5 \times 5\), and \(\delta_2 = \{\exp(\pm7, \pm5, \pm3, \pm1)\}\), such that \(|\pi| = 8\). We can find that larger \(N_d\), \(F_n\), and \(\delta\) require more parameters, so \(|\theta|\) can indicate the complexity of each model to some extent.

TABLE II: BCNNs Models with Different Hyper-parameters

| Model   | \(N_d\) | \(F_n\) | \(\delta\) | \(|\theta|\) |
|---------|---------|---------|-----------|-----------|
| BCNN1   | \(N_1\) | 4       | \(\delta_1\) | 40        |
| BCNN2   | \(N_2\) | 4       | \(\delta_1\) | 56        |
| BCNN3   | \(N_2\) | 8       | \(\delta_1\) | 113       |
| BCNN4   | \(N_2\) | 8       | \(\delta_2\) | 136       |
| BCNN5   | \(N_3\) | 24      | \(\delta_2\) | 792       |

Besides, six recently developed CS restoration methods: LASSO \(51\), TV \(52\), BCS \(13\), ReconNet \(19\), DR2Net \(20\), and LDAMP \(21\) are chosen as references to evaluate BCNNs, and these methods can be roughly divided into two categories.

\(1\)Available at https://github.com/EthanLan/BCNN



The first category includes three traditional CS restoration algorithms: LASSO\(^1\) [3], TV\(^2\) [4], and BCS\(^3\) [15]. LASSO is a canonical CS restoration algorithm merely considering sparsity. TV and BCS are two traditional SCS methods taking the cluster structure into account. In detail, BCS is a statistical SCS method integrating the cluster structure by spike and slab model, which is formulated as

\[
p(x) = (1 - \pi)\delta_0 + \pi N(x; \mu, \sigma^2)
\]

where \(\delta_0\) is the Dirac function centered at zero, and \(\pi\) is commonly extended to a MRFs model to describe the cluster structure. As a counterpart, TV is a deterministic SCS method to include the same structure.

The second category comprises three neural networks for CS restoration, namely ReconNet\(^4\), DR2Net\(^5\), and LDAMP\(^6\), which are discussed in Section II-C. Their architectures are depicted in Fig. 2 and 4. For a fair comparison, the refining networks of DR2Net and LDAMP are redesigned to keep the same as that of ReconNet except for some necessary layers for the residual technique. Specifically, the refining networks of the three algorithms have six convolutional and ReLU layers depicted in Fig. 2. Hence, the difference of these networks are only the inverse networks and residual technique.

Moreover, we use the same training set of ReconNet and DR2Net to train BCNNs and LDAMP. This training set consists of 91 natural images, and we uniformly extract 21,668 \(20 \times 20\) image patches from this set for training. In addition, Peak Signal-to-Noise Rate (PSNR), Structural Similarity Index (SSIM), and KLD are chosen as the quantitative criteria.

**B. Evaluation of BCNNs for CS restoration**

Since the prior distribution \(p(x; \theta)\) plays a vital role in CS restoration, we first evaluate the ability of BCNNs to model \(p(x; \theta)\) given different hyper-parameters under the KLD criterion. We use an empirical distribution and the samples of five BCNN models to calculate KLD value. The empirical distribution (the red curve in Fig. 5) is generated from a testing set, which includes 20 image patches randomly chosen from the training set. The samples of five BCNN models are generated by sampling \(p(x; \theta)\) given different hyper-parameters, which are presented in Fig. 6 as well.

Fig. 6(a) shows that BCNN1 has the highest KLD (0.245). That means it has the worst ability to model the empirical distribution. BCNN2 has smaller KLD (0.156) than BCNN1, since it enlarges \(N_1\) to \(N_2\) while remaining the value of \(F_n\) and \(\delta\) the same as BCNN1. Besides, we can see that increasing \(F_n\) from 4 to 8 makes BCNN3 get smaller KLD (0.126) than BCNN2. Fig. 6(c) shows that the KLD (0.108) of BCNN4 is less than BCNN3 because the former has more GSM scales than the latter. BCNN5 achieves the smallest KLD (0.089), since we increase \(N_4\) and \(F_n\) both. Overall, the model with larger hyper-parameters has smaller KLD, which means it can simulate the empirical distribution better.

Subsequently, the above models are evaluated by grayscale image restoration, and BCS is selected as baseline here. Gaussian random matrix is used to obtain the measurement \(y\) from three standard test images (starfish, butterfly, and parrot), and image dimension is 128 \(\times\) 128. Measurement ratio \((\text{MR} = |y|/|x|)\) is set to 0.25, so the dimension of \(y\) is 4096 \(\times\) 1, where \(|x| = 128^2\) is the cardinality of \(x\). Their performances are posted in Table III, which shows that all models of BCNNs outperform BCS and the model with lower KLD has better restoration in most cases.

We attribute the superiority of BCNNs to three aspects. First, BCNNs can derive optimal parameters by the learning algorithm. For example, though BCNN1 employs the same \(N_1\) filters as BCS to simulate the cluster structure, BCNN1 can learn optimal filters \(f_m\) via CD learning algorithm but BCS merely uses hand-crafted filters. Moreover, BCNN1 can learn optimal weights \(\pi_{mn}\) of GSM to preserve non-linearity accurately, which is also difficult to realize for the spike and slab model [13].

Second, BCNNs can accommodate arbitrary convolutional filters in the architecture of CNNs. Based on MRFs theory, more convolutional filters can describe more signal structures, and thus achieve better restoration. But most MRFs models can only adopt few filters because of the inherent limitations of MRFs. For instance, BCS merely incorporates four linear filters to describe the cluster structure, which is not enough to achieve good restoration. In contrast, the number of linear filters in BCNN3 is twice as many as that in BCS and BCNN2. Hence, BCNN3 can achieve better restoration.

---

\(^1\)Available at http://sparsetlab.stanford.edu/
\(^2\)Available at http://www.caam.rice.edu/~optimization/L1/TVAL3/
\(^3\)Available at https://sites.google.com/site/link2ylie/publications
\(^4\)Available at https://github.com/KuldeepKulkarni/ReconNet
\(^5\)Available at https://github.com/coldrainyht/caffe_dr2/tree/master/DR2
\(^6\)Available at https://github.com/ricedsp/D-AMP_Toolbox
Third, BCNNs can accommodate complicated convolution filters. As discussed before, MRFs can only model signal structure in a small neighborhood due to the lack of efficient parameter inference. For instance, BCS merely describes the cluster structure in $N_1$, which is hard to model high-order cluster structure. Compared to BCS, the architecture of CNNs enable BCNNs to use much more complicated convolutional filters to describe high-order structure. Fig. 7 presents all convolutional filters from BCNN4 and BCNN5. It shows that they can learn more complicated signal structures than the commonly used pairwise clique potential functions in MRFs. That helps them to generate a powerful prior distribution and achieve better restoration.

Notably, the superiority of BCNN5 over BCNN4 is not very striking, even if the former has more complex networks. That is because BCNN4 has already captured most signal structures for CS restoration and more complicated networks could make BCNN5 hard to converge to the optimal structures. This results also support our hypothesis that there is no need of very deep neural networks to model prior distribution only.

C. Comparison with other CS restoration algorithms

In the following experiments, BCNNs are compared to more CS restoration algorithms under different situations. We choose BCNN4 model to represent the BCNNs algorithm for the subsequent experiments, since it uses much fewer hyper-parameters to achieve similar performance as BCNN5.

The aforementioned CS restoration methods are first tested by four standard grayscale testing images (plane, peppers, mandrill, and boat) under the same experimental conditions as before. Their performance are presented in Table V which shows that BCNNs are still superior over two traditional SCS algorithms, namely TV and BCS, due to the advantages of BCNNs discussed above. Also note that BCNNs outperform three CNNs algorithms even though the former has much simpler architecture than the later.
Also note that DR2Net and LDAMP outperform TV method, but ReconNet not. Since the main difference of the three CNNs methods is their inverse networks (Fig. 4), above simulations prove that the inverse networks of ReconNet cannot extract enough prior knowledge from the input (i.e., the measurement y) for CS restoration. After evaluating the influence of the inverse networks on CS restoration, we will test the effect of the refining networks below.

We design two new networks DR2Net-12 and LDAMP-12. Compared to DR2Net, DR2Net-12 remains the same inverse networks, but has much deeper refining networks, which have twelve convolutional and ReLU layers. The difference between LDAMP-12 and LDAMP is the same. Table VI summarizes their performances under the same experimental conditions as Table V. It shows that LDAMP-12 achieves significant improvement over LDAMP and outperforms BCNNs. However, the improvement of DR2Net-12 is not obvious, especially when MR is small. That means the refining networks cannot improve CS restoration too much unless the inverse networks extract enough prior knowledge from the input. Overall, the above experiments verify the defect of the inverse CNNs.

### Table V: Restoration Comparison [PSNR/SSIM] between BCNNs and Other Algorithms

| Method     | MR = 0.25 | MR = 0.10 | MR = 0.04 |
|------------|-----------|-----------|-----------|
| Original   | 28.58     | 24.67     | 21.27     |
| Plane      | 22.06dB/0.59 | 23.19dB/0.62 | 22.23dB/0.65 |
| Peppers    | 21.79dB/0.61 | 23.45dB/0.71 | 23.00dB/0.73 |
| Mandrill   | 23.96dB/0.57 | 25.28dB/0.66 | 24.15dB/0.62 |
| Boat       | 23.10dB/0.58 | 24.78dB/0.68 | 23.50dB/0.68 |
| LASSO      | 11.43 dB/0.02 | 12.85 dB/0.01 | 13.98 dB/0.01 |
| BCS        | 22.79     | 20.99     | 19.34     |
| TV         | 22.86     | 19.89     | 17.02     |
| ReconNet   | 21.39     | 20.26     | 18.62     |
| DR2Net     | 21.93     | 20.77     | 18.91     |
| LDAMP      | 22.81     | 20.66     | 18.45     |
| BCNNs      | 23.54     | 21.96     | 19.95     |

Moreover, all algorithms are evaluated in noisy conditions (SNR = 16dB and 8dB) by adding Gaussian noise into y. Their average performance in each case is presented in Table VII and Table VIII respectively. We can find that BCNNs are still the best one and BCS outperforms three CNNs algorithms. It is known that CNNs are vulnerable to perturbation, which could result in the bad performance of three CNNs methods. In contrast, Algorithm 2 indicates BCNNs can automatically adjust the posterior distribution based on the variance of noise that can explain its superiority in noisy conditions.

### Table VII: CS Image Restoration with SNR = 16dB

| Method | MR = 0.25 | MR = 0.10 | MR = 0.04 |
|--------|-----------|-----------|-----------|
| LASSO  | 11.43 dB/0.02 | 12.85 dB/0.01 | 13.98 dB/0.01 |
| BCS    | 20.61 dB/0.38 | 19.20 dB/0.29 | 17.80 dB/0.16 |
| TV     | 17.45 dB/0.26 | 17.47 dB/0.25 | 16.03 dB/0.18 |
| ReconNet | 16.08 dB/0.29 | 16.35 dB/0.28 | 15.18 dB/0.23 |
| DR2Net | 16.19 dB/0.28 | 16.67 dB/0.27 | 15.23 dB/0.23 |
| LDAMP  | 18.89 dB/0.33 | 17.07 dB/0.30 | 15.77 dB/0.22 |
| BCNNs  | 21.11 dB/0.46 | 19.32 dB/0.37 | 17.79 dB/0.31 |

### Table VIII: CS Image Restoration with SNR = 8dB

| Method | MR = 0.25 | MR = 0.10 | MR = 0.04 |
|--------|-----------|-----------|-----------|
| LASSO  | 11.43 dB/0.02 | 12.85 dB/0.01 | 13.98 dB/0.01 |
| BCS    | 20.61 dB/0.38 | 19.20 dB/0.29 | 17.80 dB/0.16 |
| TV     | 17.45 dB/0.26 | 17.47 dB/0.25 | 16.03 dB/0.18 |
| ReconNet | 16.08 dB/0.29 | 16.35 dB/0.28 | 15.18 dB/0.23 |
| DR2Net | 16.19 dB/0.28 | 16.67 dB/0.27 | 15.23 dB/0.23 |
| LDAMP  | 18.89 dB/0.33 | 17.07 dB/0.30 | 15.77 dB/0.22 |
| BCNNs  | 21.11 dB/0.46 | 19.32 dB/0.37 | 17.79 dB/0.31 |
VI. Conclusion

In this paper, we attempted to exploit CNNs to propose a novel algorithm BCNNs for CS restoration. After reviewing MRFs and CNNs, we proposed a new theory to clarify CNNs, which demonstrated the output of CNNs can be regarded as a Gibbs distribution and the whole CNNs architecture is equivalent to a Bayesian hierarchical model. More importantly, we revealed a defect of the existing CNNs architecture for CS restoration, namely the inverse CNNs architecture cannot extract enough prior knowledge for CS restoration.

In order to make up this defect, BCNNs were proposed. Unlike the existing CNNs methods directly using the inverse CNNs architecture for CS restoration, BCNNs can be viewed as a hybrid of CNNs and traditional statistical algorithm. Leveraging standard CNNs architecture, BCNNs first trained a powerful prior distribution from the training image set. Subsequently, BCNNs exploited the auxiliary variable Gibbs sampler to implement posterior inference for CS restoration. This strategy enables BCNNs to circumvent the defect of the inverse CNNs architecture.

Extensive simulations validated the superiority of BCNNs over the existing CS restoration methods. First, we showed that BCNNs can train a much better prior distribution than traditional MRFs models, which is essential for CS restoration. Second, BCNNs were compared to six recently developed CS restoration methods in various conditions. The experiment results verified the superiority of BCNNs over the existing CS restoration methods.

Overall, BCNNs provide a novel approach to realize CS restoration. Compared to the traditional SCS methods, BCNNs can describe much more complicated signal structures and thus obtain much better restoration. In contrast to the existing CNNs methods, BCNNs circumvent the defect of the inverse CNNs architecture, such that it can achieve better restoration using much simpler neural networks.

However, BCNNs are still not perfect. First, the speed of BCNNs is much slower than the existing CNNs methods, since it exploits the traditional sampling technique to realize posterior inference for CS restoration. Second, our experiment demonstrates that LDAMP with much deep refining networks outperforms BCNNs. That suggests that deep neural networks have the potential to improve BCNNs further.

There are numerous directions for future work. In the context of deep learning theory, we can explore the robustness of deep learning from the perspective of Bayesian theory, which could improve the performance of neural networks in noisy conditions. Regarding the application of BCNNs, it is an exciting topic to exploit the deep refining networks to further improve the performance of BCNNs. Besides, we can apply BCNNs to other applications, such as image denoising and image super-resolution, since these problems also can be viewed as specific cases of the inverse problem with different system functions.

Acknowledgment

The authors would like to thank Matiz Romero Sergio, Bayram Samet, and anonymous reviewers for their helpful comments and suggestions.
Kenneth E. Barner (S’84-M’92-SM’00-F’16) received the B.S.E.E. degree from Lehigh University, Bethlehem, Pennsylvania, in 1987. He received the M.S.E.E. and Ph.D. degrees from the University of Delaware, Newark, Delaware, in 1989 and 1992, respectively. He was the duPont Teaching Fellow and Visiting Lecturer with the University of Delaware in 1991 and 1992, respectively. From 1993 to 1997, he was an Assistant Research Professor with the Department of Electrical and Computer Engineering, University of Delaware, and a Research Engineer with the duPont Hospital for Children. He is currently Charles B. Evans Professor and Chairman with the Department of Electrical and Computer Engineering, University of Delaware.

His research interests include signal and image processing, robust signal processing, nonlinear systems, sensor networks and consensus systems, compressive sensing, human-computer interaction. Dr. Barner is the recipient of a 1999 NSF CAREER award. He was the Co-Chair of the 2001 IEEE-EURASIP Nonlinear Signal and Image Processing (NSIP) Workshop and a Guest Editor for a Special Issue of the EURASIP Journal of Applied Signal Processing on Nonlinear Signal and Image Processing. He was a member of the Nonlinear Signal and Image Processing Board. He was the Technical Program Co-Chair for ICASSP 2005 and and previously served on the IEEE Signal Processing Theory and Methods (SPTM) Technical Committee and the IEEE Bio-Imaging and Signal Processing (BISP) Technical Committee. He is currently a member of the IEEE Delaware Bay Section Executive Committee. He has served as an Associate Editor of the IEEE Transactions on Signal Processing, the IEEE Transaction on Neural Systems and Rehabilitation Engineering, the IEEE Signal Processing Magazine, the IEEE Signal Processing Letters, and the EURASIP Journal of Applied Signal Processing. He was the Founding Editor-in-Chief of the journal Advances in Human-Computer Interaction.