IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 2 January 2024, accepted 14 February 2024, date of publication 19 February 2024, date of current version 25 March 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3366899

TOPICAL REVIEW

Compressed Sensing Vs. Auto-Encoder: On the Perspective of Signal Compression and Restoration

JIN-YOUNG JEONG[1], MUSTAFA OZGER[2], AND WOONG-HEE LEE[3], (Member, IEEE)

[1]Department of Control and Instrumentation Engineering, Korea University, Sejong-si 30019, Republic of Korea
[2]School of Electrical Engineering and Computer Science, KTH Royal Institute of Technology, 11428 Stockholm, Sweden
[3]Division of Electronics and Electrical Engineering, Dongguk University-Seoul, Seoul 04620, Republic of Korea

Corresponding author: Woong-Hee Lee (woongheelee@dongguk.edu)

This work was supported in part by the National Research Foundation of Korea (NRF) grant funded by the Korean Government (MSIT) under Grant 2022R1F1A1068980, and in part by the "Regional Innovation Strategy (RIS)" through NRF funded by the Ministry of Education (MOE) under Grant 2021RIS-004.

ABSTRACT This paper presents a comparison between compressed sensing (CS) and auto-encoder (AE) for compression and restoration of signals. The study used K-sparse vectors and generated an under-determined system, which is a system of linear equations with fewer equations than unknowns. By using CS and AE under various specific conditions, the accuracy of the signal restoration is compared with mean squared error (MSE). The experimental methodology includes comparing and analyzing the signal recovery performance by altering the algorithm and various parameters. The result represents the performance and accuracy of signal compression and restoration obtained using both techniques. It also provides a comprehensive analysis of CS and AE methods. The importance of this research and the possibility of practical application in various fields are discussed. Overall, this study provides insights into the comparison of CS and AE techniques in the field of sparse signal compression and restoration.

INDEX TERMS Compressed sensing, auto-encoder, signal processing, compression, restoration.

I. INTRODUCTION

Signal processing technology allows us to see things that cannot be seen through direct observation and to analyze scientific data. Signal processing encompasses analyzing, converting, and manipulating various signals such as audio, images, video, and waveforms [1]. The signal initially starts in the analog format. This means that the signal is constantly changing. Sampling obtains discrete samples of a continuous-time signal at regular intervals. It is necessary to determine the sampling interval to restore the original signal through sampled data.

The Nyquist-Shannon sampling theory is an essential principle in digital signal processing. It was developed about 100 years ago by Nyquist [2], [3]. According to the theory, the sampling frequency (f_s) should be set at more than twice the maximum frequency (f_m) of the signal [4]. If $f_s < 2f_m$, a distortion occurs since the frequency components overlap. It is called aliasing. When a signal is reconstructed, aliasing can generate a different signal from the original signal. Signal recovery based on the Nyquist-Shannon framework can be performed as a linear process through minimal calculation with sinc interpolation [5].

The associate editor coordinating the review of this manuscript and approving it for publication was Andrea De Marcellis.

A. WHAT IS COMPRESSED SENSING?

Compressed sensing (CS) is a technology that enables signal processing with fewer data than the Nyquist-Shannon sampling theory requires. CS expresses the original signal with a reduced number of measurements and restores the signal through a recovery algorithm. Two conditions must be met for successful signal recovery [6]:

- Sparsity: The sparsity means most elements of the signal must be zero [7]. This signal is called a K-sparse vector, where K is the number of nonzero elements. Transforming the original signal into another specific

domain requires solving the under-determined system. Due to the additional constraint that the initial signal is sparse, solutions can be found even in under-determined systems.

- Incoherence: The incoherence is a characteristic that appears sufficiently in sparse signals and is applied by isometric properties [7], [8]. It can be calculated using the two columns of the sensing matrix and how dissimilar these columns are, indicating a low level of interference. Through the coherence of Φ, the sparsity of the K-sparse vector can be determined, which allows the signal to be fully recovered through measurement [9].

CS technologies are related to the fields of signal processing and data acquisition, and they find applications in a variety of fields such as radar imaging [10], [11], antenna array [12], [13], photography [14], image codec [15], and so on [16].

B. WHAT IS AUTO-ENCODER?

The neural network (NN) is one of the technologies that compresses and restores data. It projects the original data into a low-dimensional latent space and restores the data as output. The NN consists of a structure that follows a neuron [17]. The artificial neuron acts as a node, receives input signals, multiplies by weights, and generates output signals through nonlinear functions. NNs are composed of several layers, each layer consisting of a set of neurons. Weights and biases are learned by using data sets, enabling the model to make predictions or classify the data. The performance and characteristics of NN vary depending on factors such as the number of layers and the presence of a specific structure in NN. For example, deep NN is widely employed in iterative learning and backpropagation techniques for increasing the number of intermediate layers known as hidden layers [18]. Convolutional NN constructs the convolution layer and pooling layer in an integrated way to create algorithms and examines adjacent components of data to identify characteristics [19]. Recurrent NN uses a cyclic structure to reflect past learning into current learning using weights [20].

Auto-encoder (AE) is a type of NN consisting of an encoder and a decoder. The encoder compresses input data and transforms data into low-dimensional data. Conversely, the decoder restores compressed data to the original data. There are two items for AE to operate effectively compared to CS.

- Large-scale datasets: In general, AE requires large datasets for training efficiently. Using small-scale datasets can lead to overfitting to specific examples and makes generalization difficult. Therefore, to learn better features and improve the expressiveness and performance of the model, large-scale datasets are used. Additionally, the AE performs unsupervised learning, which means that the model learns patterns and representations from the input data without explicit labels [21]. Using large-scale datasets allows the model to better learn meaningful representations of the data and better understand the structure of the data.
- Latent space: The encoder compresses the data by representing the input data in a low-dimensional latent space. Nonlinear dimensionality reduction is performed to convert high-dimensional data into a low-dimensional space, and this is called manifold learning [22]. In this process, the main characteristics and patterns of the data are trained. The decoder uses this low-dimensional representation to reconstruct the original input data. These characteristics help learn data effectively.

AEs have variations depending on their learning methods and roles. Sparse AE is a method to solve the overfitting problem and trains the network by creating sparse nodes [23]. Denoising AE removes noises from data and is used to restore the original data [24]. Variational AE has a similar structure to basic AEs, however, its role is entirely different [25]. Variational AE models the data as a distribution when given input data. More recently, a denoising AE algorithm was proposed to remove noise by learning noise, and it was shown that noise learning-based denoising AE is more effective than denoising AE when the distribution of noise is simpler than that of the original data [26].

C. MOTIVATIONS AND CONTRIBUTIONS

CS and AE are able to be the main algorithms used in sparse signal processing, and their functions are similar. Existing papers have successfully integrated algorithms from CS and AE by using their similarities [27], [28]. Despite the similarities between CS and AE, to the best of our knowledge, there is a notable absence of research fundamentally comparing CS and AE. Therefore, this paper analyzes recovery performance, investigating the methodologies employed in CS and AE and drawing comparisons between the two algorithms.

CS and AE have similar capabilities for compressing and restoring data. Fig. 1 shows that CS and AE have similar mechanisms for compressing and restoring input data. In the case of CS, the input data is compressed by multiplying with a sensing matrix and then restored through a recovery process. In the case of AE, input data is compressed into latent space through an encoder, and important features of the data are learned. Similar to the sensing matrix of CS, AE compresses and restores data using a weight matrix for encoding. The AE finds the weight matrices by training. In the process of compressing data, the two techniques have a very similar bottleneck structure, however, there are differences between CS and AE in the process of restoring the signal.

In CS, a solution to an under-determined system is obtained by restoring compressed data through a recovery algorithm such as L1-norm minimization [29]. An under-determined system has fewer equations than unknowns, and since the signal is a K-sparse vector, a solution to the under-determined system needs to be found. The performance of the recovery

algorithm varies based on sparsity, the sensing matrix, and the presence or absence of noise. On the other hand, AE learns more effectively using various nonlinear functions to compress and restore data. Unlike CS, which can customize the sensing matrix, AE optimizes the weight matrix on its own by learning to make the restored signal similar to the original signal. In other words, the characteristics of data can be identified through unsupervised learning. Performance is determined by parameter settings such as epoch, learning rate, and number of data sets.

The objective of this paper is to investigate the fundamental similarities and dissimilarities between CS and AE in terms of signal compression and restoration according to 1) the sparsity, 2) the sensing/weight matrices, 3) the recovery algorithms, 4) the required conditions, and 5) the noisy observations. Based on these intensive comparisons, we further provide inspirations for open challenges and potential applications of CS and AE in wireless communication systems, especially focusing on their potential usefulness, theoretical solidification, and complementary collaborations.

D. ORGANIZATIONS AND NOTATIONS

Section II describes the operating conditions of CS and the algorithm of AE in more detail. In Section III, signal recovery of CS and AE is simulated and analyzed according to various conditions and algorithms. Section IV discusses how to apply it through the analyzed results.

The following notations will be used throughout this paper.

- x: the original signal.
- $\hat{x}$: the restored signal.
- K: the sparsity of original signal x.
- Φ: the sensing matrix.
- y: a measurement vector, which is the compressed signal.
- M: the size of the measurement vector.
- N: the size of the original signal vector.
- $f(\cdot)$, $g(\cdot)$: the encoding and decoding function of AE.
- $\mathcal{L}(\cdot)$: the loss function.
- $\mathcal{N}$: Gaussian distribution.

II. PRELIMINARIES

A. A BRIEF DESCRIPTION OF COMPRESSED SENSING

CS is a signal processing technique that can restore sparse signals from a few linear measurements. In CS, an original signal x of length N can be expressed with a sparse vector with K nonzero elements. $M \times 1$ measurement vector y can be obtained with $M \times N$ sensing matrix Φ multiplying x [30], [31]. The linear measurement y in CS can be represented as follows:

$$y = \Phi x. \tag{1}$$

As explained earlier, for CS to function effectively, x must have sparsity, and the recovery of x relies on Φ and y. Since y is determined by Φ, designing Φ is one of the primary objectives of operating CS. The mathematical criteria for the sensing matrix in CS are spark, coherence, and

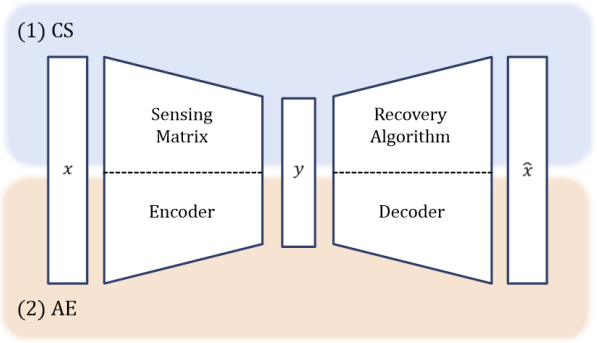


FIGURE 1. Illustration of frameworks of CS and AE.

the restricted isometry property (RIP), which are important concepts that help determine the number of measurements needed to recover signals [30]. Spark is the minimum number of linearly dependent columns in a matrix Φ.

$$spark(\Phi) = \min_{x \neq 0} \|x\|_0 \text{ s.t.} \Phi x = 0 \tag{2}$$

If the $spark(\Phi)$ is larger than $2K$, measurement vector y exists at most one signal x. As $spark(\Phi) \in [2, M+1]$, the requirement $M \geq 2K$ is valid. In general, calculating the spark of the matrix Φ is computationally complex since it is necessary to ensure that all sets of columns are linearly independent. Therefore, coherence, which is easier to compute, is preferred for use. Coherence is defined as the maximum value of the inner product between matrix columns, which is mathematically defined as follows:

$$\mu(\Phi) = \max_{1 \leq i \neq j \leq N} \frac{|\langle \phi_i, \phi_j \rangle|}{\|\phi_i\|_2 \|\phi_j\|_2}. \tag{3}$$

The coherence $\mu(\Phi)$ has a range of $[\sqrt{\frac{N-M}{M(N-1)}}, 1]$ according to the Welch bound [32]. When $N \gg M$, the lower bound is approximately $\mu(\Phi) \leq 1/\sqrt{M}$. Additionally, when $N = M$, the lower bound becomes 0. For any matrix Φ,

$$spark(\Phi) \geq 1 + \frac{1}{\mu(\Phi)}. \tag{4}$$

Since $M \geq 2K$ is required, it is possible to exist measurement vector y at most one signal x such that $y = \Phi x$.

$$K < \frac{1}{2}(1 + \frac{1}{\mu(\Phi)}) \tag{5}$$

The sensing matrix provides guarantees of uniqueness if the measurement vector y is obtained perfectly. However, if $x' = x + n$ and n represent additive noise, the sensing matrix cannot provide guarantees of uniqueness due to a mismatch between the sensing matrix of x' and x. If the distance between x and x' and the distance between the measurement vectors $y = \Phi x$ and $y = \Phi x'$ are proportional, two sparse vectors that are different cannot lead to the same measurement vector, as the noise is small enough. This is called (K, δ)-RIP and is

expressed as an equation as follows:

$$(1-\delta)\|x\|_2^2 \leq \|\Phi x\|_2^2 \leq (1+\delta)\|x\|_2^2. \tag{6}$$

The (K, δ)-RIP means all submatrices of sensing matrix Φ of size $M \times K$ are close to an isometry [30]. This characteristic creates a denoising function that the recovery is stable even if noise is added to x. When M is $\mathcal{O}(K \log(N/K))$, the signal recovery algorithm can achieve a high probability.

B. A BRIEF DESCRIPTION OF AUTO-ENCODER

AE is an algorithm that should obtain output data most similar to input data [33]. The AE must learn to reconstruct the input observations enough. In the most typical form, an AE consists of the encoder and the decoder, which is NN. If the function of the encoder is f for the input data x, y can be represented as follows:

$$y = f(x), \tag{7}$$

where y is the output of the encoder. In this process, the encoder aims to extract the important features of the input data. Then restored data can be represented as follows:

$$\hat{x} = g(y) = g(f(x)). \tag{8}$$

The restored y should be similar to x, therefore, to train an AE means to find $f(\cdot)$ and $g(\cdot)$ that satisfy the equation as follows:

$$\arg\min_{f,g} < [\mathcal{L}(x, g(f(x)))] > \tag{9}$$

where $\mathcal{L}$ is loss function between input data and output data and $< \cdot >$ is the average of all observations. The trained AE can restore the input data through the reconstruction process.

AE training involves optimizing the weight matrix, and there are various parameters that affect this. During training, a substantial dataset is necessary, however, to train the entire dataset all at once can lead to memory limitations or processing constraints. Therefore, it is necessary to split the data. The AE process includes dividing the data by epoch, batch size, and iteration to repeat the learning process several times for optimization. One epoch is completed when all forward or backward passes of training overall the entire dataset. Batch size means the number of data to be used in one batch. Iteration is also called step and represents the total number of batches for the entire data. During training, the optimizer updates the parameters by calculating the error between the predicted value by the AE and the true value by using the data as the batch size.

When updating weights, the performance of AE is adjusted using a learning rate. If the learning rate is large, an overflow occurs, and errors cannot be reduced. On the other hand, if the learning rate is low, the training process will be prohibitively time-consuming, and the error values to be verified increase. Therefore, it is important to find an appropriate learning rate.

TABLE 1. Default parameter set for simulation.

Default parameters	CS	AE
N	128	
M	$[8, 16, 32, 64]$	
# train dataset	–	10000
# (test) dataset	10000	
Distribution of sensing matrix	$\mathcal{N}(0, 1)$	–
Recovery algorithm	Basis pursuit	–
Max epochs	–	1000
Mini-batch size	–	32
Optimization	–	Adam
Activation function	–	Leaky ReLU
Loss function	–	MSE

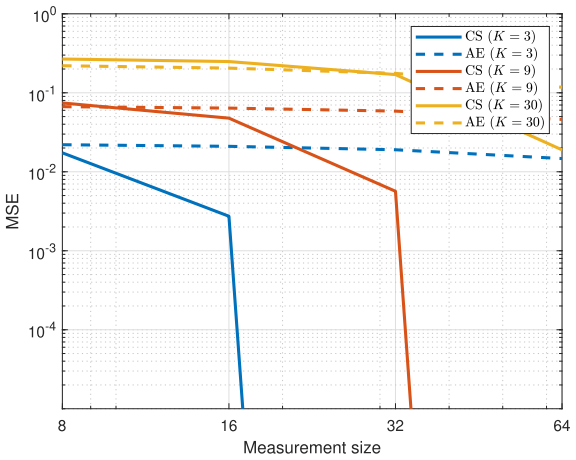


FIGURE 2. MSE depending on sparsity.

III. INTENSIVE COMPARISON OF CS AND AE ON VARIOUS PERSPECTIVES

In this section, we compare the performance of compressing and restoration of K-sparse vectors through CS and AE methods. The experimental default values of CS and AE were set as Table. 1. CS basically uses L1-minimization, while AE also consists of a single hidden layer with a fundamental structure. To generate a K-sparse vector, K indices are randomly chosen from the original vector of size N, and Gaussian distributed values with the mean of 0 and standard deviation of 1 are generated and inserted.

A. ON THE SPARSITY

CS performance is determined by the sparsity and compression size of the original signal. As explained previously, the recovery algorithm is performed when $M = \mathcal{O}(K \log(N/K))$. Fig. 2 shows the comparison of the MSE of CS and AE according to changes in K value. When $K = 3$, CS has a lower MSE for all M values compared to AE. The MSE drops sharply when M increases from 16 to 32, and when M is 32 and 64, the MSE becomes almost 0. Unlike CS, where the MSE changes drastically depending on the M value, the MSE value of AE is relatively constant.

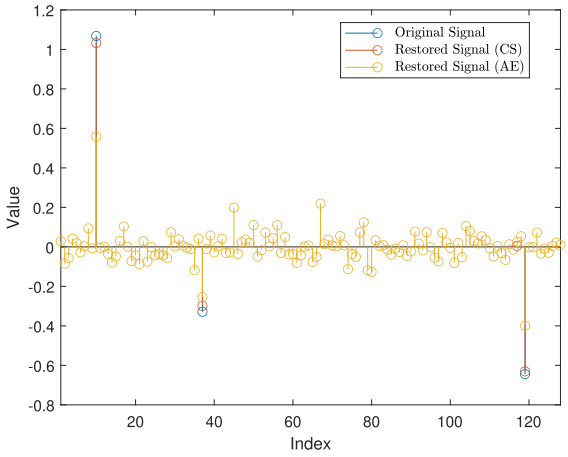


FIGURE 3. An example of signal restoration with CS and AE.

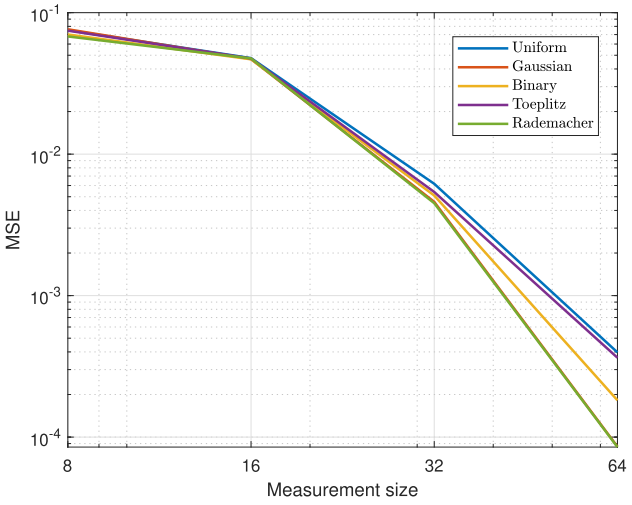


FIGURE 4. MSE depending on sensing matrix.

As K increased to 9, MSE overall increased slightly, and performance changes occurred in CS according to M. Unlike when MSE decreased sharply from $M = 32$ when $K = 3$, MSE became almost 0 from $M = 64$. When M is 8, MSE is higher compared to AE, indicating that recovery performance is gradually decreasing. It was confirmed that AE also increased MSE compared to when $K = 3$, although the shape was similar. This indicates that the performance of AE is relatively insensitive to M.

When $K = 30$ in Fig. 2, the MSE of CS was higher than AE when M was 8 and 16. In CS, as M increases, MSE decreases, however, the reduction in MSE when comparing K values of 3 and 9 is relatively small. It shows that 64 observations are not enough to recover the signal, even though the system is under-determined. The result shows that CS performs better than AE when certain conditions are met, however, the MSE of AE is relatively stable across changes in M compared to CS.

Fig. 3 shows how the original signal is restored to CS and AE when K is 3 and M is 32. In CS, the value for a specific index was restored to a similar value. The recovered signal maintains sparse characteristics. On the other hand, when AE restores a sparse signal, the restored signal loses its sparse characteristics, and a value is added to the index that originally had a value of 0.

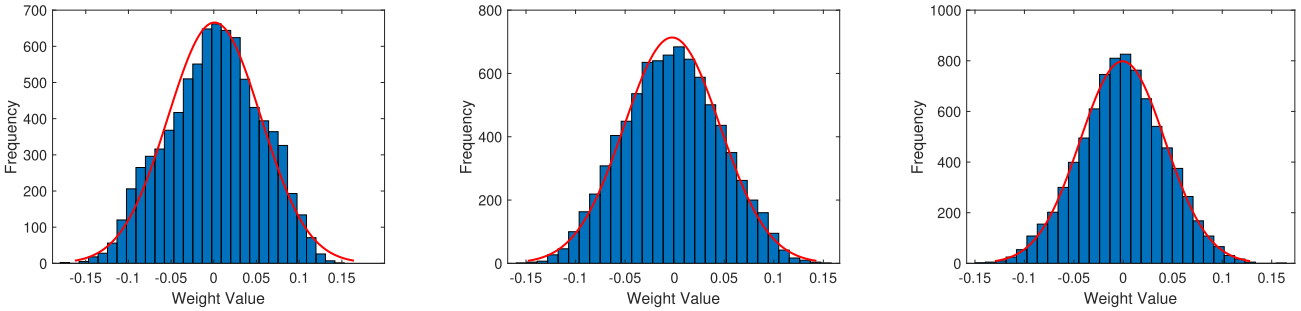
B. ON THE SENSING/WEIGHT MATRICES

Constructing sensing matrices that satisfy the RIP is important in CS. The sensing matrices serve to compress and reconstruct the signals. The sensing matrices include random sensing matrices and structured sensing matrices. The random sensing matrices mean a matrix with a random distribution, offering the advantage of ease to construct [34]. However, it requires a large amount of calculation and memory, above all, there is no efficient algorithm to verify the RIP condition, so it is not suitable for real applications. Examples of random sensing matrices are Gaussian, Bernoulli random matrix, and so forth.

Structured sensing matrices are divided into subsampled incoherent bases, structurally subsampled matrices, subsampled circulant matrices, and separable matrices [30]. Subsampled incoherent bases are famous for random subsampled Fourier and Walsh-Hadamard matrices and random Toeplitz matrices. Structurally subsampled matrices can be used as a discrete Fourier transform and can simplify signal recovery with greedy algorithms. Subsampled circulant matrices use Toeplitz and circulant structures and have fewer degrees of freedom since rows and columns are repeated. Separable matrices are a method for measuring very large signals. These matrices have a special structure that makes them efficient in construction, computation, or hardware implementation.

In this paper, uniform random sampling matrix, Gaussian random matrix, binary random matrix, Toeplitz matrix, and Rademacher matrix were used. Fig. 4 shows how the MSE changes with measurement size and the performance differences between different types of sensing matrices. As M increases, MSE generally decreases in all cases. It shows that when M increases, the Gaussian matrix and Rademacher matrix show better performances. This indicates that when compared to other matrices, these two matrices exhibit relatively strong signal recovery performance and can be considered effective sensing matrices.

In CS, the Gaussian matrix satisfies RIP and compresses data effectively. This has been confirmed through 4, and it has also been observed that the weight matrix in the AE. Fig. 5 represents the distribution of the weight matrix of the encoder, which has a similar role as the sensing matrix. Fig. 5 shows the weight histograms learned by AE according to the K value. As the K value increases, it becomes possible to more precisely check how the weight data in latent space is distributed. As a result, the weight histogram becomes increasingly closer to the Gaussian distribution, which indicates that the Gaussian distribution has an influence on the weight learning of AE.



(a) Distribution of weight matrix ($K = 3$) (b) Distribution of weight matrix ($K = 9$) (c) Distribution of weight matrix ($K = 30$)

FIGURE 5. Histograms of weight matrix of trained AE when $M = 64$. The distribution of weights follows a Gaussian distribution.

C. ON THE RECOVERY ALGORITHMS

As x is a K-sparse vector, CS solves the solution of the underdetermined system. To find the solution to this sparse system, three methods are introduced the characteristics respectively.

The L2-norm is also called the Euclidean distance, and this method minimizes the distance between two vectors, which is mathematically described as:

$$\min ||x||_2 \text{ s.t. } y = \Phi x. \tag{10}$$

It is a common technique in the least squares problems. However, this method is not suitable for finding K-sparse signals solution. Non-sparse solutions can be derived and this may lead to significant differences from the actual sparse solution.

The L0-norm is the total number of nonzero elements in a vector. The L0-minimization method can be effective in restoring K-sparse signals [35].

$$\min ||x||_0 \text{ s.t. } y = \Phi x \tag{11}$$

This process of minimization can result in NP-Hard problems, meaning that the problem is computationally extremely difficult or the algorithm cannot efficiently solve all instances of the problem. In other words, it can be difficult to use for large problems, as it requires checking all the K-sparse vectors.

The L1-norm is the sum of the absolute values of all elements of the vector. L1-minimization, known as basis pursuit [36], is defined as follows:

$$\min ||x||_1 \text{ s.t. } y = \Phi x. \tag{12}$$

Using L1-minimization, K-sparse vector can be restored with only the measured value of $M = \mathcal{O}(K \log(N/K))$ [37].

Through the L0-norm, the exact location of the element can be found, however, the NP-hard problem occurs. It is necessary to make the solutions of the L0-minimization and the L1-minimization the same. To find the L1-minimization solution, we will express the relationship between the matrix Φ and the sparse signal x. The sensing matrix Φ has a lot of influence on signal recovery. To find a unique solution in the linear equation system (1), Donoho and Huo introduced the necessary condition for uniquely obtaining sparse signals through coherence $\mu(\Phi)$ [38]. The condition for the L0-minimization solution to be unique is defined as follows:

$$K < \frac{1}{\mu(\Phi)} \tag{13}$$

As the coherence value of the sensing matrix decreases, the recoverable K value increases. The condition for the L0-minimization solution to match the L1-minimization solution is (5). This is a more detailed equation than (13). Consequently, if the equation (5) is met, it implies that the condition of (13) is also satisfied.

In general, when solving a system of N linear equations, the complexity becomes $\mathcal{O}(N^3)$. As the length of the signal increases, the problem becomes more difficult to solve. To solve this problem, a greedy algorithm was suggested. There are matching pursuit (MP), orthogonal matching pursuit (OMP), and compressive sampling matching pursuit (CoSaMP) [39], [40], [41]. Additionally, the least absolute shrinkage and selection operator (Lasso) proposed by Tibshirani is also one of the algorithms to find the K-sparse signal [42].

MP is a sparse approximation algorithm proposed by Mallat and Zhang in 1993 [39]. To find the most similar signal, a Fourier transform is conducted. This transforms the signal into a K-sparse vector form to allow finding the optimal matching. However, this has the limitation that it can only find the global features of the signal and cannot apply to analyzed signals. In addition, MP is effective when the sparsity of the signal is high and can be time-consuming to calculate. To overcome this, an enhanced version known as OMP has been proposed [40]. In contrast to MP, OMP computes an orthogonal projection of the signal and updates the coefficients after every step. CoSaMP has an algorithmic structure similar to OMP. CoSaMP selects and learns from multiple supports while OMP selects one support in one iteration [41]. Lasso uses the L1-norm penalty for linear regression. The Lasso problem is defined as follows:

$$\min_x \|y - \Phi x\|_2^2 \text{ s.t. } \|x\|_1 \leq t, \tag{14}$$

where t is a tuning parameter determining the degree of regularization [42]. The purpose of the Lasso is to minimize the sum of the MSE and the penalty L1-norm.

In AE, various optimization processes are used to perform signal processing. First, gradient descent (GD) is the most basic method which was first suggested by Augustin Louis Cauchy in 1847 [43]. The GD updates the parameter as follows:

$$\theta_{l+1} = \theta_l - \alpha \nabla \mathcal{L}(\theta_l) \tag{15}$$

where l is iteration number, $\alpha > 0$ is learning rate, θ is parameter vector, and $\nabla \mathcal{L}(\cdot)$ is the gradient of loss function [44]. GD uses the entire dataset to perform updates at each iteration. This method guarantees steady convergence until the optimal solution of the convex function is found. However, this can be computationally intensive for large datasets, as the gradient must be computed over the entire dataset. Stochastic gradient descent (SGD) is a variant algorithm of GD [44], [45]. The parameter update formula of SGD is the same as (15), however, SGD uses mini-batches which are randomly selected at each step. Another GD algorithm variation, momentum, is characterized by larger weight changes at each step [46]. It accelerates learning by mitigating the oscillation problem, which helps learning rapidly. The parameter of SGD with momentum update equation is defined as follows:

$$\theta_{l+1} = \theta_l - \alpha \nabla \mathcal{L}(\theta_l) + \gamma(\theta_l - \theta_{l-1}), \tag{16}$$

where γ determines the contribution of the previous gradient to the current step. SGD with momentum can determine the direction of progress by accumulating the gradient of the previous step through this parameter. Root mean square propagation (RMSProp) is an algorithm developed to solve the learning rate vanishing problem [47]. It does not simply accumulate gradients, instead, accumulates gradients using an exponential moving average. This approach stabilizes the learning process and reduces the oscillation of the optimization path. The moving average of the RMSProp is defined as follows:

$$v_l = \beta_2 v_{l-1} + (1 - \beta_2)[\nabla \mathcal{L}(\theta_l)]^2, \tag{17}$$

where v_l is the moving average of the squared gradients, β_2 is the decay rate, which is a hyperparameter for controlling the moving average. The parameter update formula of RMSProp is defined as follows:

$$\theta_{l+1} = \theta_l - \frac{\alpha \nabla \mathcal{L}(\theta_l)}{\sqrt{v_l} + \epsilon}, \tag{18}$$

where α is the learning rate, $\nabla \mathcal{L}(\theta_l)$ is the gradient of the loss function, and ϵ is a small constant added to prevent division by zero. Adam is the most common optimization method combining RMSProp and Momentum, which contributes to minimizing the loss function by simultaneously adjusting the weight change rate and step size [48].

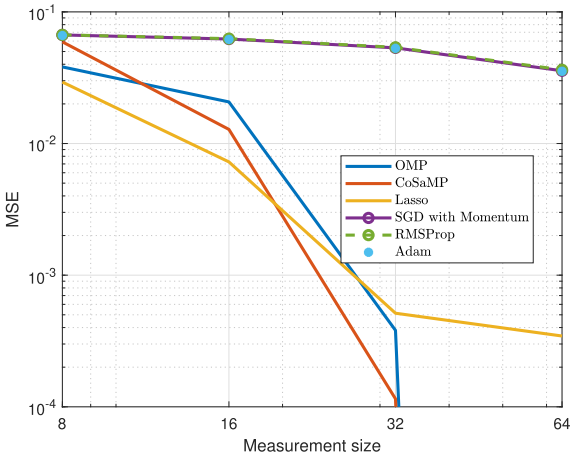


FIGURE 6. MSE depending on CS recovery Algorithm ($K = 4$).

$$m_l = \beta_2 m_{l-1} + (1 - \beta_1)[\nabla \mathcal{L}(\theta_l)], \tag{19}$$

$$v_l = \beta_2 v_{l-1} + (1 - \beta_2)[\nabla \mathcal{L}(\theta_l)]^2, \tag{20}$$

where β_1 is exponential moving average of Momentum, β_2 is exponential moving average of RMSProp. With this moving average, Adam updates the parameters as follows:

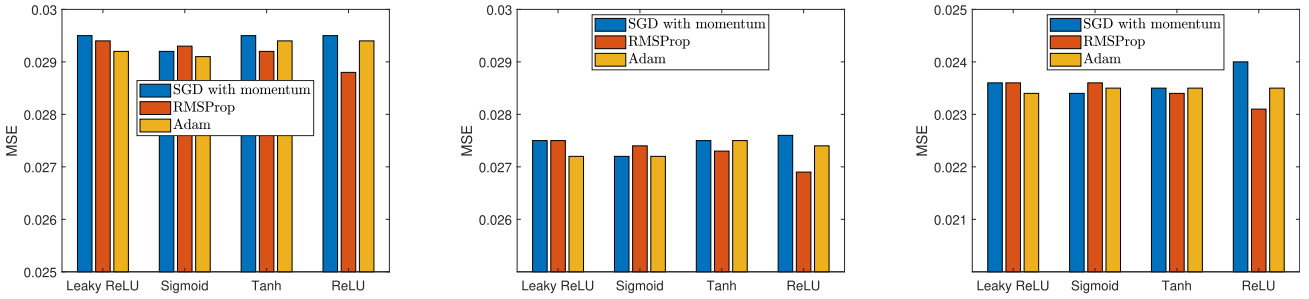
$$\theta_{l+1} = \theta_l - \frac{\alpha m_l}{\sqrt{v_l} + \epsilon} \tag{21}$$

This method allows more efficient and accurate learning. The activation function was simulated by setting the alpha of the leaky ReLU function to 0.01.

In the case of CS, the MSE of OMP, CoSaMP, and Lasso were compared, and the results are shown in Fig. 6. RIP affects the operating conditions of CoSaMP, and M must be greater than $\mathcal{O}(K \log(N/K))$ for operating the recovery algorithm well. Based on Fig. 6, the recovery performance of CoSaMP becomes better than OMP when M ranges from 16 to 64. It was confirmed that CoSaMP performs better than OMP when the operating conditions are met.

In the case of AE, when comparing Adam, RMSProp, and SGD with momentum, the differences may be obscure from a graphical standpoint. Upon closer examination of the results, Adam achieves a lower MSE compared to other optimizations. During the learning RMSProp, a bias problem which is the continuous gradients in a specific direction can occur. This problem can disturb the learning process. Adjusting the learning rate to a more moderate level in Adam provides better performance. As a result, the performance of AE is determined in various ways depending on the algorithm and setting parameters.

Activation function also has an important role in AE learning [49], [50]. It is a nonlinear function that determines what to do with the output by calculating the value obtained by multiplying the input value by a weight. The sigmoid function is one of the most commonly used nonlinear activation functions [51]. As an activation function of early



(a) MSE comparison when $M = 8$ (b) MSE comparison when $M = 16$ (c) MSE comparison when $M = 32$

FIGURE 7. MSE comparison based on changes in optimizer and activation function.

NNs, it was mainly used for classification problems. The function outputs the values between 0 and 1. The sigmoid function is defined as follows:

$$\texttt{sigmoid}(x) = \frac{1}{1+e^{-x}}, \tag{22}$$

where x is the input value, and e is the Euler's number. The shape of the sigmoid function is S and approaches 0 as the input approaches negative infinity and approaches 1 as the input approaches positive infinity. This has the limitation of a vanishing gradient problem in which the output value of the function approaches 1 or 0 when the input value is very large or small. The shape of the tanh function is similar to the sigmoid function [52]. The tanh function is defined as follows:

$$\texttt{tanh}(x) = \frac{e^{2x}-1}{e^{2x}+1}. \tag{23}$$

The average of the output values becomes 0 since the output range is -1 to 1, unlike the sigmoid function. However, there is still a vanishing gradient problem, so other activation functions have been proposed. The rectified linear unit (ReLU) function, found in 2011, is one of the functions currently used in many NNs [53]. If the input value is negative, the output is 0, and if the input is positive, the input values remain unchanged. ReLU function is defined as follows:

$$\texttt{ReLU}(x) = \max(x, 0). \tag{24}$$

This function solves the vanishing gradient problem and has the advantage of being simple to operate. However, it causes the dying ReLU problem, which is a phenomenon in which a neuron outputs 0 when the input value is biased toward a negative number. This means that weight updates no more. The leaky ReLU function was proposed in 2013 [54]. Leaky ReLU applies some output to negative inputs using a linear function with a slight slope when the input is less than zero. It can prevent the output from always zero and solve the gradient vanishing problem. The leaky ReLU function is defined as follows:

$$\texttt{leaky-ReLU}(x) = \begin{cases} x, & \text{if } x > 0 \\ \alpha \cdot x, & \text{otherwise,} \end{cases} \tag{25}$$

where α is the small gradient applied when the input value is less than zero. The performance of AE can be adjusted by the α value. Fig. 7 indicates MSE according to the optimizer and activation function. The optimizer used SGD with momentum, RMSProp, and Adam, and when these are changed, the trend of MSE according to the activation function is different. In general, the MSE of sigmoid was low when using SGD with momentum, and the MSE of ReLU was the lowest when using RMSProp. In Adam, leaky ReLU generally had a relatively low MSE.

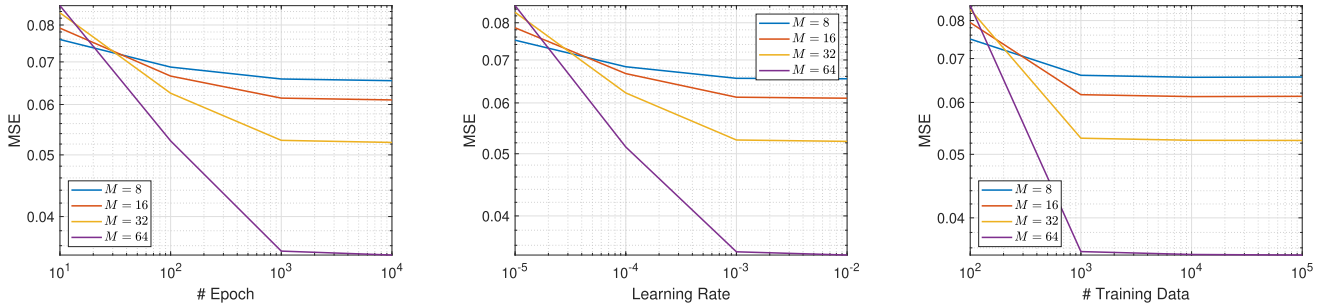
This paper used leaky ReLU functions, where $\alpha = 0.01$, and the simulation was performed using Adam as the optimizer. Since the data is generated according to the Gaussian distribution in this experiment environment, to prevent the vanishing of the hidden layer, it is good to use functions that can activate real values in all ranges, including negative values, in the activation function. In the case of ReLU, it can be useful when the input data has many positive values, and in the case of sigmoid, it is good to use when the value of data is in (0, 1) [33].

D. ON THE SIGNAL RECOVERY GUARANTEES

The CS recovery algorithm has the required conditions to guarantee performance. The Coherence and RIP are two representative conditions that affect the performance of algorithms [30]. The first case is about coherence. In base pursuit or OMP, it is defined that signal recovery is possible if the sensing matrix Φ satisfies (5) in a noise-free environment. The upper bound of sparsity K can be obtained according to the Welch bound, which is described as:

$$K = \mathcal{O}(\sqrt{M}). \tag{26}$$

Since OMP is a greedy algorithm, it needs the minimum absolute value entry of x to detect support correctly. RIP also affects OMP. In condition (6), If $(K+1, \delta)$-RIP and $\delta \leq \frac{1}{3\sqrt{K}}$, OMP can restore the K-sparse vector in K iteration [30], [55].



(a) MSE depending on epoch (b) MSE depending on learning rate (c) MSE depending on # of training data

FIGURE 8. **Comparing MSE with respect to AE parameters.**

Similarly, in the case of CoSaMP, if (cK, δ)-RIP, where the c is a parameter of RIP, x can be restored from y.

Next, as a guarantee by metrics by coherence and RIP, the signal can be restored if the coherence and spectral, which is the L2-norm of the sensing matrix, are sufficiently small. The sparsity level required is defined as follows:

$$K = \mathcal{O}(M). \tag{27}$$

This scaling is called the square root bottleneck and is why random distributions and sparse signal models can be used [30], [56]. AEs also use a bottleneck structure to reduce the number of dimensions [33]. The algorithm is performed by compressing to a dimension smaller than the input size. According to the universal approximation theorem, a single hidden layer neural network can be a universal approximator for real-valued continuous functions when utilizing a nonpolynomial activation function [57], [58]. This theorem makes the ability of the network to possess approximation properties. Additionally, if the number of neurons in the hidden layer is constrained, the performance of AE can have a limit to reconstructing the data. Signal recovery performance varies depending on the type of algorithm, architecture, dataset, parameters, and so forth.

Fig. 8a visualizes MSE changes according to epoch. As epochs increase, MSE generally decreases, indicating that AE is doing better reconstructions. As the degree of compression increased, the MSE increased, however, it was confirmed that the smaller the M size, the less sensitive the performance was to epoch. When the epoch is small, there is no significant difference in MSE. In contrast, the difference gradually increases, as the epoch increases.

Fig. 8b shows the change in MSE according to the learning rate. As the learning rate decreased, MSE generally increased. The MSE value generally decreases as the degree of compression increases. However, the MSE at $M = 64$ was found to be the highest at 10^{-5}. This indicates that selecting appropriate hyper-parameters helps performance.

Fig. 8c shows the MSE variation with respect to the number of training data. It shows a similar pattern to Fig. 8b. As the value of M increases and becomes sufficiently large, a performance gap is large when the number of training data is limited. However, as the amount of data increases, the AE model reconstructs data effectively. Additionally, the performance of AE is maintained when the training data exceeds a certain value. This shows that when the amount of training data grows beyond a certain level, the model can generalize the data better.

The larger the M value, the less data is lost during compression. As in the example above, It was confirmed that better restoration performance was achieved under conditions where the parameters were the same. This indicates that the model can learn more accurately with the same parameters. In the case of AEs, when trying to improve model performance, better generalization performance can be obtained by selecting appropriate parameter values and using sufficient training data above a specific value.

E. ON THE NOISY OBSERVATIONS

In this section, we will discuss the performance of CS and AE when noise is added to x. If noise is added to x in CS, it can be expressed with the formula $y = \Phi x + n$ and the optimization (12) can be changed to allow for noise as follows:

$$\min ||x||_1 \text{ s.t. } \|y - \Phi x\|_2 \leq \epsilon, \tag{28}$$

where $\epsilon \leq \|n\|_2$ is parameter about noise magnitude. This is called basis pursuit with inequality constraints (BPIC) and limits y and Φx through ϵ [30], [59]. If we redefine this optimization problem using Lagrange multipliers, we get the formula as follows:

$$\min ||x||_1 + \lambda \|y - \Phi x\|_2, \tag{29}$$

which is called basis pursuit denoising (BPDN) [30], [60]. When additive white Gaussian noise (AWGN) is added as noise according to $\mathcal{N}(0, \sigma^2)$, where $\gamma = \|n\|_2$ and $\hat{x}$ is the output, the guarantee of BPIC can be defined as follows [30] and [61]:

$$\|x - \hat{x}\|_2 \leq \frac{\gamma + \epsilon}{\sqrt{1 - \mu(\Phi)(4K - 1)}}, \tag{30}$$

where $K \leq \frac{1}{4}(\frac{1}{\mu(\Phi)} + 1)$ and $\epsilon = \sigma\sqrt{M + \eta\sqrt{2M}}$ when $\gamma \leq \epsilon$ and a parameter η prevents $\|n\|_2$ being too large. These parameters ensure that the algorithm performs with a high probability. In this section, data was restored by using

TABLE 2. MSE performance according to the K value and SNR in the CS and AE.

(a) MSE performance (Nonzero values follow a Gaussian distribution.)

M	Method	$K=3$				$K=9$				$K=30$			
		SNR [dB]											
		0	10	20	30	0	10	20	30	0	10	20	30
8	CS	0.0340	**0.0218**	**0.0166**	**0.0177**	0.4147	0.1112	0.0766	0.0721	–	–	–	–
	AE	**0.0229**	0.0223	0.0221	0.0221	**0.0680**	**0.0666**	**0.0659**	**0.0658**	**0.2274**	**0.2232**	**0.2201**	**0.2200**
16	CS	0.0265	**0.0082**	**0.0045**	**0.0033**	0.3221	0.1918	**0.0511**	**0.0488**	–	–	–	8.6172
	AE	**0.0207**	0.0184	0.0176	0.0175	**0.0659**	**0.0630**	0.0616	0.0615	**0.2203**	**0.2116**	**0.2057**	**0.2054**
32	CS	0.0356	**0.0030**	**0.0003**	**0.0000**	0.3380	**0.0193**	**0.0030**	**0.0003**	–	–	5.6537	5.7681
	AE	**0.0207**	0.0184	0.0176	0.0175	**0.0616**	0.0555	0.0530	0.0527	**0.2062**	**0.1884**	**0.1769**	**0.1764**
64	CS	0.0285	**0.0034**	**0.0003**	**0.0000**	0.3252	**0.0248**	**0.0031**	**0.0003**	–	–	2.7611	2.2802
	AE	**0.0179**	0.0131	0.0118	0.0117	**0.0533**	0.0405	0.0357	0.0353	**0.1785**	**0.1401**	**0.1192**	**0.1180**

(b) MSE performance (Nonzero values are fixed at 1.)

M	Method	$K=3$				$K=9$				$K=30$			
		SNR [dB]											
		0	10	20	30	0	10	20	30	0	10	20	30
8	CS	0.0308	0.0243	0.0231	0.0232	0.1267	0.0849	0.0808	0.0797	–	–	–	–
	AE	**0.0222**	**0.0215**	**0.0215**	**0.0215**	**0.0639**	**0.0617**	**0.0613**	**0.0613**	**0.1794**	**0.1718**	**0.1695**	**0.1695**
16	CS	0.0280	**0.0126**	**0.0081**	**0.0076**	0.1146	0.0762	0.0707	0.0694	–	–	3.2098	0.9938
	AE	**0.0216**	0.0202	0.0200	0.0200	**0.0626**	**0.0580**	**0.0572**	**0.0572**	**0.1787**	**0.1634**	**0.1583**	**0.1582**
32	CS	0.0213	**0.0024**	**0.0002**	**0.0000**	0.1092	**0.0143**	**0.0014**	**0.0001**	–	2.3945	0.4458	0.1537
	AE	**0.0202**	0.0175	0.0172	0.0172	**0.0595**	0.0505	0.0490	0.0489	**0.1713**	**0.1425**	**0.1358**	**0.1359**
64	CS	0.0253	**0.0024**	**0.0002**	**0.0000**	0.1267	**0.0149**	**0.0014**	**0.0001**	–	2.6602	0.3830	0.1520
	AE	**0.0177**	0.0121	0.0114	0.0114	**0.0538**	0.0359	0.0327	0.0325	**0.1597**	**0.1072**	**0.0911**	**0.0907**

BPIC with $\epsilon = 0.1$. In the case of AE, denoising AE is used for signal reconstruction. It has the same structure as the AE used previously. However, the difference is that noise is added when inputting input data.

Table 2 shows the performance of CS and AE according to SNR for each M value. In case (a), the nonzero value of x follows a Gaussian distribution, while fixed at 1 in case (b). We generally observed lower MSE under conditions where the value was fixed at 1. In both cases, AE shows relatively consistent performances regardless of signal-to-noise ratio (SNR), while CS shows sensitivity to SNR. As M decreases, the MSE of CS becomes higher than AE. Additionally, as K increases, the performance of CS starts to become severely unstable. As the SNR approaches zero, the performance of the CS lags behind AE. This indicates that CS tends to be vulnerable to noise and that AE can identify and remove noise during the data reconstruction. When noise management is crucial, utilizing AE can be suitable.

IV. OPEN CHALLENGES AND POTENTIAL APPLICATIONS

A. AVAILABILITY OF CS AND AE IN WIRELESS COMMUNICATIONS

Analog to digital converter (ADC) refers to converting a continuous signal into a discontinuous signal based on the time axis. It goes through filtering, sampling, quantization, and encoding. After the filtering and sampling steps, the y-axis value is digitized and quantized, and the degree of delicacy of the signal varies depending on the resolution [62]. Finally, the signal is converted to binary through encoding.

A one-bit ADC is used in the receiver antenna to reduce implementation complexity and power consumption in the multiple-input and multiple-output (MIMO) system. Information loss occurs during quantization using a one-bit ADC, resulting in performance degradation. If a few-bit ADC is used to increase resolution, the maximum sampling rate will inevitably decrease as the number of bits per measurement increases. The CS method can be used to alleviate the challenges with ADC design. CS minimizes the number of bits per measurement with higher sampling rates and is helpful for sparse signal recovery. This advantage can be used to improve battery-related issues in non-terrestrial networks (NTN) with efficiently handled data.

Both CS and AE are effective signal processing methods and can be used for noise processing and signal improvement in communication systems such as orthogonal frequency division multiplexing (OFDM). Computational cost and performance must be considered, and how it can be used in a real environment is important. If the K value of the sparse signal is known, the CS method may be more advantageous. In cases where CS does not work, overall recovery performance is expected to improve if AE is used as an alternative method. CS is an effective method when the signal is sparse, so if the signal is not sparse, it is necessary to consider other reconstruction methods, such as AE.

Noise signals are generated in the process of transmitting and receiving signals. Denoising the AE method to remove noise, in addition to directly recovering the original signal, learning noise or learning in another dimension through a linear transformation of the signal is also a new method. For example, in denoising AE, an algorithm learns noise rather than learning the original signal, then the algorithm can recover the signal or can be used for specific applications in

wireless communications such as symbol demodulation and precise localization [26].

B. ESSENTIAL UTILIZATION OF CS AND AE IN WAY FORWARD OF 6G AND BEYOND

Using mmWave/THz frequency bands in 6G communications enhances high-speed data transmission and spectral efficiency. Massive MIMO and extremely large-scale massive MIMO (XL-MIMO) have emerged to improve data transmission rates and support inter-device communication in these high-frequency bands [63], [64]. As the frequency bandwidth widens, the center frequency rises and the straightness of the wave is strengthened, leading to phenomena such as diffraction, scattering, and reflection. This causes the channel to have sparse characteristics, and algorithms such as CS and AE can be used to estimate this efficiently.

Furthermore, applications of CS and AE include the essential technologies in wireless communications such as channel estimation and beamformer design [65]. The CS mechanism was used to estimate channel state information in the mmWave large-scale MIMO system [66], enabling high performance of channel estimation even in low SNR environments. Optimizing the beamforming patterns using AEs in mmWave massive MIMO vehicular networks [67]. To alleviate the delay overhead issues encountered in high-rate mmWave applications, multiple base stations jointly serve one mobile station and learn an optimal beam selection policy.

In 6G networks, integrating communication and sensing functions optimizes network utilization, supporting various applications and services. Integrated sensing and communication (ISAC) combines signal detection and communication, enabling the entire communication network to function as a sensor. This integration allows for precise channel modeling through sensing, leading to increased accuracy. Beyond 6G, scenarios demand both advanced sensing and wireless communication capabilities, with the potential to enhance spectral efficiency by sharing the same frequency band and hardware [68]. Furthermore, the incorporation of artificial intelligence presents advantages like improved RF detection performance and the potential for faster operation through parallel calculation in 6G applications [69]. Various sensors, including radar, lidar, ultrasonic waves, and cameras, play a crucial role in sensing environments. Specifically, radars that can be utilized in the automotive field operate at frequencies between 76 - 81 GHz [70], leading to a sparse signal characteristic. This sparsity facilitates efficiently utilizing technologies such as CS and AE.

C. THEORETICAL SOLIDIFICATION OF AE BY TAKING A HINT FROM CS

This subsection elucidates specific application cases that provide the intersection between CS and AE, which might be a chance for AE to take inspiration from the diversified mathematical analysis of CS. A deep learning-based CS framework is proposed in [28] and [71]. In these frameworks, the encoder employs a fully connected layer that serves as an adaptive sensing matrix. This implies that the layer of the encoder is able to compress the data efficiently and create a more effective sensing matrix for compressing data. On the other hand, the decoder uses a deconvolution network for data restoration.

The data collection process is performed more efficiently through sparsity, which is one of the benefits of CS [72]. Additionally, the network was trained to restore the original data using a pattern of the CNN recognition function. This shows the synergy between CS techniques and deep learning for effective data restoration.

The research in the preceding papers shows how AE technology leverages CS methods to collect and reconstruct data efficiently and improve overall performance. This fusion of AE and CS techniques demonstrates the potential for a variety of applications, such as signal processing, where data compression and reconstruction are important.

D. POSSIBILITY OF COLLABORATIONS OF CS AND AE

In the previous subsection, we provide the potential theoretical specification of AE through the concepts of CS. The relevant literature references suggest significant potential for collaboration between CS and AE, indicating operating together effectively [28], [71], [72]. Through experimental results, it has been verified that if certain conditions are met, CS shows remarkable performance. Conversely, when the conditions are not met, the performance of AE remains stable. This implies that CS and AE complement each other, raising the expectation for enhanced overall signal compression and restoration performance.

V. CONCLUSION

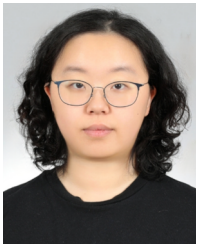
In this paper, we compared CS and AE in various aspects, considering that they have similar structures and approaches in terms of signal recovery. Using the K-sparse vector, numerical experiments were performed depending on various algorithms and linear or nonlinear mathematical properties. Signal recovery performance is evaluated by considering various algorithms and mathematical characteristics of the signal. As described in the theory, if the required conditions for CS, i.e., sparsity, coherence, and RIP, are satisfied, CS shows almost perfect performance in signal recovery. If these conditions are not met, the performance of signal recovery using CS is significantly lower, indicating that the effectiveness of CS is highly affected by the level of sparsity and the type of sensing matrix. Additionally, AE requires overhead for an NN model optimization to learn the distribution of signals with big data, while the recovery performance is relatively stable compared to CS. Both CS and AE can be effective methods in signal processing, and the performance guarantees of the two techniques appear to be different depending on the sparsity of the signal and specific conditions. Both schemes use different approaches and have their advantages and applicability, and future

research methods that combine the frameworks of CS and AE can improve performance to achieve more robust and flexible solutions in signal processing.

REFERENCES

[1] S. Sarangi, M. Sahidullah, and G. Saha, "Optimization of data-driven filterbank for automatic speaker verification," *Digit. Signal Process.*, vol. 104, Sep. 2020, Art. no. 102795.
[2] H. Nyquist, "Certain topics in telegraph transmission theory," *Trans. Amer. Inst. Electr. Eng.*, vol. 47, no. 2, pp. 617–644, Apr. 1928.
[3] C. E. Shannon, "Communication in the presence of noise," *Proc. IRE*, vol. 37, no. 1, pp. 10–21, Jan. 1949.
[4] E. Meijering, "A chronology of interpolation: From ancient astronomy to modern signal and image processing," *Proc. IEEE*, vol. 90, no. 3, pp. 319–342, Mar. 2002.
[5] M. A. Davenport, P. T. Boufounos, M. B. Wakin, and R. G. Baraniuk, "Signal processing with compressive measurements," *IEEE J. Sel. Topics Signal Process.*, vol. 4, no. 2, pp. 445–460, Apr. 2010.
[6] B. Li, M. Salucci, P. Rocca, W. Ke, and W. Tang, "The sparsity and incoherence in compressive sensing as applied to field reconstruction," in *Proc. 14th Eur. Conf. Antennas Propag. (EuCAP)*, Mar. 2020, pp. 1–3.
[7] M. A. Davenport, M. F. Duarte, Y. C. Eldar, and G. Kutyniok, *Introduction to Compressed Sensing*. Cambridge, U.K.: Cambridge Univ. Press, 2012, pp. 1–64.
[8] D. L. Donoho, "For most large underdetermined systems of linear equations the minimal ℓ_1-norm solution is also the sparsest solution," *Commun. Pure Appl. Math.*, vol. 59, no. 6, pp. 797–829, Jun. 2006.
[9] L. Carin, D. Liu, and B. Guo, "Coherence, compressive sensing, and random sensor arrays," *IEEE Antennas Propag. Mag.*, vol. 53, no. 4, pp. 28–39, Aug. 2011.
[10] R. Baraniuk and P. Steeghs, "Compressive radar imaging," in *Proc. IEEE Radar Conf.*, Apr. 2007, pp. 128–133.
[11] J. Yang, T. Jin, C. Xiao, and X. Huang, "Compressed sensing radar imaging: Fundamentals, challenges, and advances," *Sensors*, vol. 19, no. 14, p. 3100, Jul. 2019.
[12] A. Massa, M. Bertolli, G. Gottardi, A. Hannan, D. Marcantonio, G. Oliveri, A. Polo, F. Robol, P. Rocca, and F. Viani, "Compressive sensing as applied to antenna arrays: Synthesis, diagnosis, and processing," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2018, pp. 1–5.
[13] B. Zhang, W. Liu, and X. Gou, "Compressive sensing based sparse antenna array design for directional modulation," *IET Microw., Antennas Propag.*, vol. 11, no. 5, pp. 634–641, Apr. 2017.
[14] K. Marwah, G. Wetzstein, Y. Bando, and R. Raskar, "Compressive light field photography using overcomplete dictionaries and optimized projections," *ACM Trans. Graph.*, vol. 32, no. 4, pp. 1–12, Jul. 2013.
[15] J. Xu, J. Yang, F. Kimishima, I. Taniguchi, and J. Zhou, "Compressive sensing based image codec with partial pre-calculation," *IEEE Trans. Multimedia*, early access, Oct. 26, 2023, doi: 10.1109/TMM.2023.3327534.
[16] A. Massa, P. Rocca, and G. Oliveri, "Compressive sensing in electromagnetics—A review," *IEEE Antennas Propag. Mag.*, vol. 57, no. 1, pp. 224–238, Feb. 2015.
[17] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proc. Nat. Acad. Sci. USA*, vol. 79, no. 8, pp. 2554–2558, Apr. 1982.
[18] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Netw.*, vol. 61, pp. 85–117, Jan. 2015.
[19] R. Venkatesan and B. Li, *Convolutional Neural Networks in Visual Computing: A Concise Guide*. Boca Raton, FL, USA: CRC Press, 2017.
[20] O. I. Abiodun, A. Jantan, A. E. Omolara, K. V. Dada, N. A. Mohamed, and H. Arshad, "State-of-the-art in artificial neural network applications: A survey," *Heliyon*, vol. 4, no. 11, Nov. 2018, Art. no. e00938.
[21] M. A. Kramer, "Nonlinear principal component analysis using autoassociative neural networks," *AIChE J.*, vol. 37, no. 2, pp. 233–243, Feb. 1991.
[22] J. B. Tenenbaum, V. D. Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, no. 5500, pp. 2319–2323, Dec. 2000.
[23] A. Makhzani and B. Frey, "K-sparse autoencoders," 2013, *arXiv:1312.5663*.
[24] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, P.-A. Manzagol, and L. Bottou, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *J. Mach. Learn. Res.*, vol. 11, no. 12, pp. 1–38, 2010.
[25] D. P Kingma and M. Welling, "Auto-encoding variational Bayes," 2013, *arXiv:1312.6114*.
[26] W.-H. Lee, M. Ozger, U. Challita, and K. W. Sung, "Noise learning-based denoising autoencoder," *IEEE Commun. Lett.*, vol. 25, no. 9, pp. 2983–2987, Sep. 2021.
[27] L. Tian, G. Li, and C. Wang, "A data reconstruction algorithm based on neural network for compressed sensing," in *Proc. 5th Int. Conf. Adv. Cloud Big Data (CBD)*, Aug. 2017, pp. 291–295.
[28] H. Wu, Z. Zheng, Y. Li, W. Dai, and H. Xiong, "Compressed sensing via a deep convolutional auto-encoder," in *Proc. IEEE Vis. Commun. Image Process. (VCIP)*, Dec. 2018, pp. 1–4.
[29] D. L. Donoho, "Compressed sensing," *IEEE Trans. Inf. Theory*, vol. 52, no. 4, pp. 1289–1306, Apr. 2006.
[30] M. F. Duarte and Y. C. Eldar, "Structured compressed sensing: From theory to applications," *IEEE Trans. Signal Process.*, vol. 59, no. 9, pp. 4053–4085, Sep. 2011.
[31] D. L. Donoho and M. Elad, "Optimally sparse representation in general (nonorthogonal) dictionaries via ℓ_1 minimization," *Proc. Nat. Acad. Sci. USA*, vol. 100, no. 5, pp. 2197–2202, Mar. 2003.
[32] L. Welch, "Lower bounds on the maximum cross correlation of signals (Corresp.)," *IEEE Trans. Inf. Theory*, vol. IT-20, no. 3, pp. 397–399, May 1974.
[33] U. Michelucci, "An introduction to autoencoders," 2022, *arXiv:2201.03898*.
[34] H.-J. Jo, H.-I. Chun, I.-M. Ban, and W.-K. Lee, "Random hybrid chirp sensing matrix implementation for fast compressive sensing SAR processing," *J. Korean Inst. Electromagn. Eng. Sci.*, vol. 32, no. 12, pp. 1079–1090, Dec. 2021.
[35] D. Wipf and B. Rao, "ℓ_0-norm minimization for basis selection," in *Proc. 17th Int. Conf. Neural Inf. Process. Syst.*, 2004, pp. 1513–1520.
[36] P. Yin, Y. Lou, Q. He, and J. Xin, "Minimization of ℓ_{1-2} for compressed sensing," *SIAM J. Sci. Comput.*, vol. 37, no. 1, pp. A536–A563, 2015.
[37] E. J. Candes, J. Romberg, and T. Tao, "Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information," *IEEE Trans. Inf. Theory*, vol. 52, no. 2, pp. 489–509, Feb. 2006.
[38] D. L. Donoho and X. Huo, "Uncertainty principles and ideal atomic decomposition," *IEEE Trans. Inf. Theory*, vol. 47, no. 7, pp. 2845–2862, Nov. 2001.
[39] S. G. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Trans. Signal Process.*, vol. 41, no. 12, pp. 3397–3415, Dec. 1993.
[40] Y. C. Pati, R. Rezaiifar, and P. S. Krishnaprasad, "Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition," in *Proc. 27th Asilomar Conf. Signals, Syst. Comput.*, 1993, pp. 40–44.
[41] D. Needell and J. A. Tropp, "CoSaMP: Iterative signal recovery from incomplete and inaccurate samples," *Appl. Comput. Harmon. Anal.*, vol. 26, no. 3, pp. 301–321, May 2009.
[42] R. Tibshirani, "Regression shrinkage and selection via the lasso," *J. Roy. Stat. Soc., Ser. B, Methodolog.*, vol. 58, no. 1, pp. 267–288, Jan. 1996.
[43] C. Lemaréchal, "Cauchy and the gradient method," *Doc Math Extra*, vol. 251, no. 254, p. 10, Dec. 2012.
[44] S. Ruder, "An overview of gradient descent optimization algorithms," 2016, *arXiv:1609.04747*.
[45] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proc. COMPSTAT*, Y. Lechevallier and G. Saporta, Eds. Heidelberg, Germany: Physica-Verlag, 2010, pp. 177–186.
[46] I. Sutskever, J. Martens, G. E. Dahl, and G. E. Hinton, "On the importance of initialization and momentum in deep learning," in *Proc. ICML*, vol. 28, Apr. 2013, pp. 1139–1147.
[47] G. Hinton, N. Srivastava, and K. Swersky, "Neural networks for machine learning lecture 6a overview of mini-batch gradient descent," *Cited*, vol. 14, no. 8, p. 2, 2012.
[48] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.
[49] S. Sharma, S. Sharma, and A. Athaiya, "Activation functions in neural networks," *Towards Data Sci.*, vol. 6, no. 12, pp. 310–316, 2017.
[50] J. M. Ede, "Deep learning in electron microscopy," *Mach. Learn., Sci. Technol.*, vol. 2, no. 1, Mar. 2021, Art. no. 011004.
[51] J. Han and C. Moraga, "The influence of the sigmoid function parameters on the speed of backpropagation learning," in *Proc. Int. Workshop Artif. Neural Netw., From Natural Artif. Neural Comput.* Cham, Switzerland: Springer, 1995, pp. 195–201.

[52] B. Karlik and A. V. Olgac, "Performance analysis of various activation functions in generalized MLP architectures of neural networks," *Int. J. Artif. Intell. Expert Syst.*, vol. 1, no. 4, pp. 111–122, 2011.
[53] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proc. 14th Int. Conf. Artif. Intell. Statist.*, 2011, pp. 315–323.
[54] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. 30th Int. Conf. Mach. Learn.*, vol. 30, no. 1, 2013, p. 3.
[55] M. A. Davenport and M. B. Wakin, "Analysis of orthogonal matching pursuit using the restricted isometry property," *IEEE Trans. Inf. Theory*, vol. 56, no. 9, pp. 4395–4401, Sep. 2010.
[56] P. Kuppinger, G. Durisi, and H. Bölcskei, "Where is randomness needed to break the square-root bottleneck?" in *Proc. IEEE Int. Symp. Inf. Theory*, Jun. 2010, pp. 1578–1582.
[57] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Netw.*, vol. 2, no. 5, pp. 359–366, Jan. 1989.
[58] S. Sonoda and N. Murata, "Neural network with unbounded activation functions is universal approximator," *Appl. Comput. Harmon. Anal.*, vol. 43, no. 2, pp. 233–268, Sep. 2017.
[59] S. P. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.
[60] J. A. Tropp and S. J. Wright, "Computational methods for sparse solution of linear inverse problems," *Proc. IEEE*, vol. 98, no. 6, pp. 948–958, Jun. 2010.
[61] D. L. Donoho, M. Elad, and V. N. Temlyakov, "Stable recovery of sparse overcomplete representations in the presence of noise," *IEEE Trans. Inf. Theory*, vol. 52, no. 1, pp. 6–18, Jan. 2006.
[62] A. V. Oppenheim, *Discrete-Time Signal Processing*. London, U.K.: Pearson Education India, 1999.
[63] M. W. Akhtar, S. A. Hassan, R. Ghaffar, H. Jung, S. Garg, and M. S. Hossain, "The shift to 6G communications: Vision and requirements," *Human-Centric Comput. Inf. Sci.*, vol. 10, no. 1, pp. 1–27, Dec. 2020.
[64] L. Mucchi, S. Shahabuddin, M. A. M. Albreem, S. Abdallah, S. Caputo, E. Panayirci, and M. Juntti, "Signal processing techniques for 6G," *J. Signal Process. Syst.*, vol. 95, no. 4, pp. 435–457, Apr. 2023.
[65] A. Salh, L. Audah, N. S. M. Shah, A. Alhammadi, Q. Abdullah, Y. H. Kim, S. A. Al-Gailani, S. A. Hamzah, B. A. F. Esmail, and A. A. Almohammedi, "A survey on deep learning for ultra-reliable and low-latency communications challenges on 6G wireless systems," *IEEE Access*, vol. 9, pp. 55098–55131, 2021.
[66] W. Tong, W. Xu, F. Wang, J. Shang, M. Pan, and J. Lin, "Deep learning compressed sensing-based beamspace channel estimation in mmWave massive MIMO systems," *IEEE Wireless Commun. Lett.*, vol. 11, no. 9, pp. 1935–1939, Sep. 2022.
[67] P. Tarafder and W. Choi, "Deep reinforcement learning-based coordinated beamforming for mmWave massive MIMO vehicular networks," *Sensors*, vol. 23, no. 5, p. 2772, Mar. 2023.
[68] A. Liu, Z. Huang, M. Li, Y. Wan, W. Li, T. X. Han, C. Liu, R. Du, D. K. P. Tan, J. Lu, Y. Shen, F. Colone, and K. Chetty, "A survey on fundamental limits of integrated sensing and communication," *IEEE Commun. Surveys Tuts.*, vol. 24, no. 2, pp. 994–1034, 2nd Quart., 2022.
[69] D. K. Pin Tan, J. He, Y. Li, A. Bayesteh, Y. Chen, P. Zhu, and W. Tong, "Integrated sensing and communication in 6G: Motivations, use cases, requirements, challenges and future directions," in *Proc. 1st IEEE Int. Online Symp. Joint Commun. Sens. (JC&S)*, Feb. 2021, pp. 1–6.
[70] T. Fujibayashi, Y. Takeda, W. Wang, Y.-S. Yeh, W. Stapelbroek, S. Takeuchi, and B. Floyd, "A 76- to 81-GHz multi-channel radar transceiver," *IEEE J. Solid-State Circuits*, vol. 52, no. 9, pp. 2226–2241, Sep. 2017.
[71] A. Grover and S. Ermon, "Uncertainty autoencoders: Learning compressed representations via variational information maximization," in *Proc. 22nd Int. Conf. Artif. Intell. Statist.*, vol. 89, K. Chaudhuri and M. Sugiyama, Eds. Apr. 2019, pp. 2514–2524.
[72] Q. Cheng, A. A. Ihalage, Y. Liu, and Y. Hao, "Compressive sensing radar imaging with convolutional neural networks," *IEEE Access*, vol. 8, pp. 212917–212926, 2020.



JIN-YOUNG JEONG received the B.S. degree in electro-mechanical engineering from Korea University, Sejong-si, Republic of Korea, in 2021, where she is currently pursuing the combined master's and Ph.D. degree. Her research interests include signal processing and machine learning in wireless communications.

MUSTAFA OZGER received the B.Sc. degree in electrical and electronics engineering from Middle East Technical University, Ankara, Turkey, in 2011, and the M.Sc. and Ph.D. degrees in electrical and electronics engineering from Koc University, Istanbul, Turkey, in 2013 and 2017, respectively. He is currently a Senior Researcher with the KTH Royal Institute of Technology, Stockholm, Sweden. He was one of the technical coordinators of the EU Celtic Next 6G-SKY Project. His research interests include 3D wireless networks and the Internet of Things.

WOONG-HEE LEE (Member, IEEE) received the B.S. degree in electrical engineering from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea, in 2009, and the Ph.D. degree in electrical engineering from Seoul National University, Seoul, Republic of Korea, in 2017. From 2017 to 2019, he was an Experienced Researcher with Advanced Standard Research and Development Laboratories, LG Electronics Inc., Seoul. From 2019 to 2020, he was a Postdoctoral Researcher with the Department of Communication Systems, KTH Royal Institute of Technology, Stockholm, Sweden. From 2020 to 2021, he was an Experienced Researcher with Ericsson Research, Stockholm, Sweden. From 2021 to 2024, he was an Assistant Professor at the Department of Control and Instrumentation Engineering, Korea University, Sejong-si, Republic of Korea. Since March 2024, he has been an Assistant Professor with the Division of Electronics and Electrical Engineering, Dongguk University, Seoul. His research interests include signal processing, machine learning, and game theory in wireless communications.