

We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

6,400

Open access books available

174,000

International authors and editors

190M

Downloads

Our authors are among the

154

Countries delivered to

TOP 1%

most cited scientists

12.2%

Contributors from top 500 universities



WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index
in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com



Chapter

Utilized System Model Using Channel State Information Network with Gated Recurrent Units (CsiNet-GRUs)

Hany Helmy, Sherif El Diasty and Hazem Shatila

Abstract

MIMO: multiple-input multiple-output technology uses multiple antennas to use reflected signals to provide channel robustness and throughput gains. It is advantageous in several applications like cellular systems, and users are distributed over a wide coverage area in various applications such as mobile systems, improving channel state information (CSI) processing efficiency in massive MIMO systems. This chapter proposes two channel-based deep learning methods to enhance the performance in a massive MIMO system and compares our proposed technique to the previous methods. The proposed technique is based on the channel state information network combined with the gated recurrent unit's technique CsiNet-GRUs, which increases recovery efficiency. Besides, a fair balance between compression ratio (CR) and complexity is given using correlation time in training samples. The simulation results show that the proposed CsiNet-GRUs technique fulfills performance improvement compared with the existing literature techniques, namely CS-based methods Conv-LSTM CsiNet, LASSO, Tval3, and CsiNet.

Keywords: massive MIMO, FDD, compressed sensing, deep learning, conventional neural network

1. Introduction

For fifth-generation wireless communication systems, the massive multiple-input multiple-output (MIMO) system is recognized as a powerful technology.

Such a system can significantly reduce multi-user interference and offer a multi-fold boost in cell throughput by outfitting a base station (BS) with hundreds or even thousands of antennas in a dispersed or centralized way. Utilizing channel state information (CSI) at base stations is the primary method for obtaining this potential benefit (BSs). The downlink channel state information (CSI) in modern frequency division duplex (FDD) MIMO systems (such as long-term evolution Release-8) is collected at the user equipment throughout the training phase and transmitted back to the BS via feedback links.

To minimize feedback overhead, vector quantization or codeword-based techniques are frequently used. The feedback quantities generated from these methods are not permitted in a massive MIMO system since they must be scaled linearly with the number of transmit antennas. As shown in [1], the difficulty of CSI feedback in massive MIMO systems has inspired several studies. By using the spatial and temporal correlation of channel state information (CSI), which describes how a signal travels from the transmitter to the receiver and represents the combined effect of, for example, scattering, fading, and power decay with distance, these works have primarily concentrated on reducing feedback overhead. To minimize feedback overhead, vector quantization or code-word-based techniques are frequently used. To minimize feedback overhead, vector quantization or codeword-based techniques are frequently used. The feedback quantities generated from these methods are not permitted in a massive MIMO system since they must be scaled linearly with the number of transmit antennas. As shown in [1], the difficulty of CSI feedback in massive MIMO systems has inspired several studies. By using the spatial and temporal correlation of channel state information (CSI), which describes how a signal travels from the transmitter to the receiver and represents the combined effect of, for example, scattering, fading, and power decay with distance, these works have primarily concentrated on reducing feedback overhead. To minimize feedback overhead, vector quantization or code-word-based techniques are frequently used.

A difficult issue in wireless communications systems is channel estimate during auto-encoding. Most of the time sent signals are reflected and scattered as they reach the receiver. The channel moves over time as a result of the mobility of the transmitter, receiver, or scattering objects. Deep learning (DL) trains massive, multilayered neural networks using lots of training data to approximate how the human brain does a particular activity. Channel State Information Networks (CsiNet), which we created as CSI sensing (or encoder) and recovery (or decoder) networks, include the features listed below in the auto-encoder system (Figure 1).

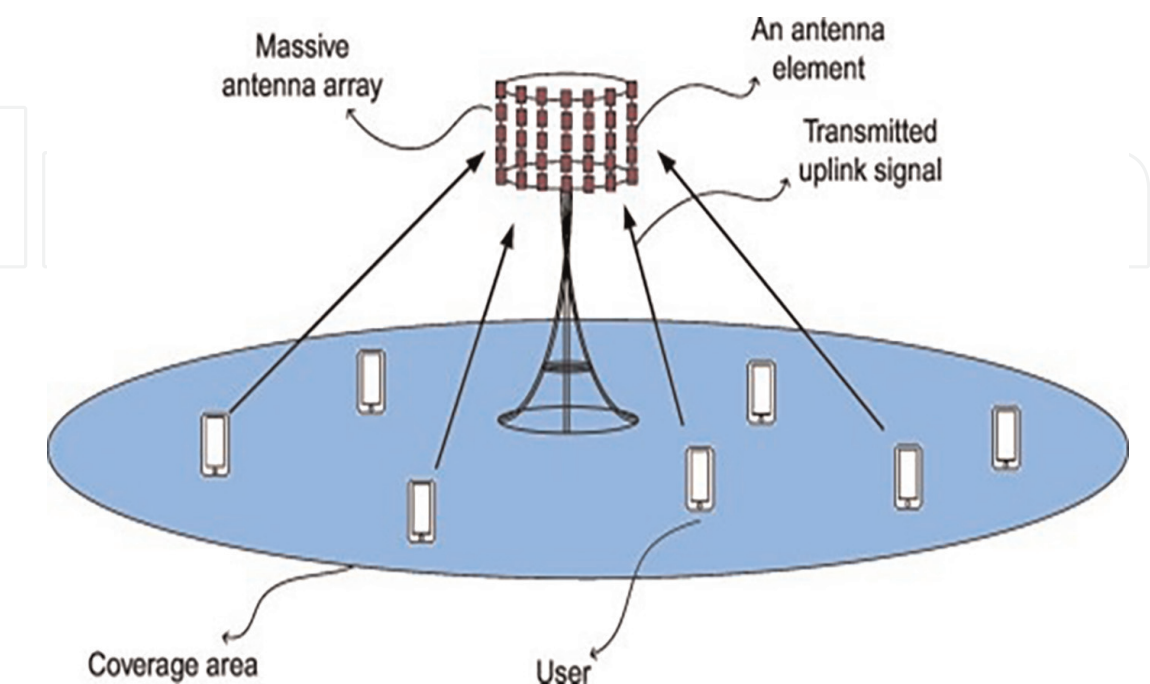


Figure 1.
Enhanced multiple-access for mmWave massive MIMO [2].

- Encoder: CsiNet learns transformation from original channel matrices to compress representations (codewords) through training data.
- Decoder: CsiNet learns inverse transformation from codewords to original channels; The inverse transformation is not iterative and multiple orders of magnitude faster than iterative algorithms. The algorithm is agnostic to human knowledge of channel distribution and instead directly learns to use the channel structure from training data effectively.

User equipment encodes channel matrices into codewords using the encoder; after the codewords are returned to the BS, it uses the decoder to reconstruct the original channel matrices. The technique can be applied as a feedback protocol in FDD MIMO systems. The autoencoder [3] in deep learning, which is used to learn an encoding for a set of data types for dimensionality reduction, and CsiNet are closely related. To recreate accurate models from CS data, several deep learning (DL) architectures have recently been designed and introduced in [4–6].

DL shows state-of-the-art performance in natural-image reconstruction, but because wireless channel reconstruction is more difficult than image reconstruction, it can also demonstrate that this capability is unclear. The DL-based CSI reduction and recovery strategy is introduced in the current work. The most significant research appears to be [7], in which a closed-loop MIMO system implements DL-based CSI encoding. It differs from previous research that did not consider CSI recovery by demonstrating that, as compared to current CS-based methods, CSI can be recovered with a significantly increased reconstruction quality by DL.

2. The structure of channel state information network (CsiNet)

The structure of CsiNet [8] according to Depth wise Separable Convolution in feature recovery reconstruction illustrated in detail, CsiNet remarkably outperforms the CS-based methods. Introducing the CSI network feedback process, which considers a single-cell FDD massive MIMO-OFDM framework, where there is N_t ($\gg 1$) transmit antennas at the BS and a single receiver antenna at the UE, OFDM is with N_c subcarriers the received signal at the n^{th} subcarrier can be communicated as:

$$y_n = \tilde{h}_n^H v_n x_n + z_n \quad (1)$$

where \tilde{h}_n^H and $y_n \in \mathbb{C} N_t \times 1$ is the channel frequency response vector and the pre-coding vector at the n^{th} subcarrier, separately, x_n represents the transmitted information image, z_n is the additive noise or obstruction and $(\cdot)^H$ is a conjugate transpose. In the FDD system, improving feedback links through UE and BS, focus on the feedback scheme which allows autoencoder processing, assume:

$\hat{\mathbf{H}} = [\tilde{\mathbf{h}}_1 \dots \tilde{\mathbf{h}}_{N_c}]^H \in \mathbb{C}^{N_c \times N_t}$ in CSI stacked in the spatial frequency domain, which means the UE should return $\hat{\mathbf{H}}$ to the BS through feedback links, and in the feedback system, the total number parameter is $N_t N_c$, using a 2D (DFT) discrete Fourier transform, which introducing $\tilde{\mathbf{H}}$ can be improved in the angular-delay domain to reduce feedback overhead:

$$\mathbf{H} = \mathbf{F}_d \cdot \tilde{\mathbf{H}} \mathbf{F}_a^H \quad (2)$$

where \mathbf{F}_d and \mathbf{F}_a are $N_c \times N_c$ and $N_t \times N_t$ DFT matrices, respectively. So, considering the COST 2100 as was illustrated in [9] channel model as shown in **Figure 2**. depending on a uniform linear array (ULA), \mathbf{H} has a small fraction of significant components. According to the delay domain, the first N_c rows of \mathbf{H} contain values, retain the first N_c Rows of \mathbf{H} and remove remaining rows. In a massive MIMO system, the total number of feedback parameters can be reduced to $N = N_c N_t$. So, we design the encoder S ,

$$S = f_{en}(\mathbf{H}) \quad (3)$$

We can convert \mathbf{H} into a codeword M vector, where $M < N$, and design the decoder inverse transformation from the codeword to \mathbf{H} original channel.

$$\mathbf{H} = f_{de}(S) \quad (4)$$

The European Cooperation in Science and Technology COST 2100 channel model is a GSCM that can reproduce the stochastic properties of massive MIMO channels

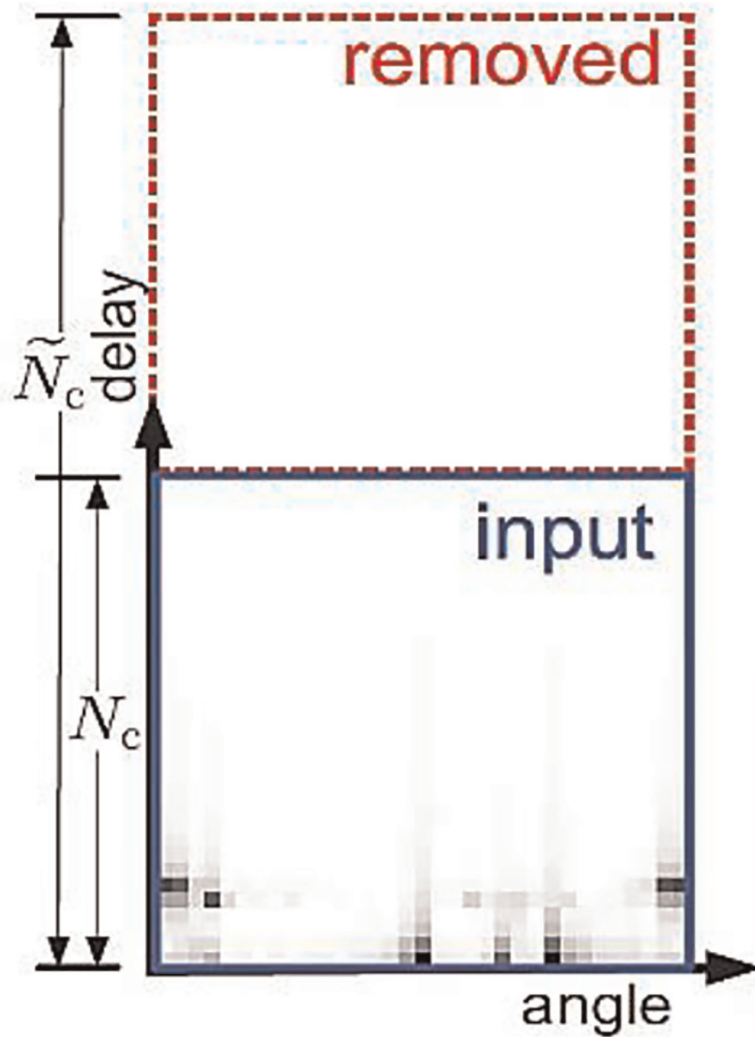


Figure 2.
A plot of the strength of $\mathbf{H} \in \mathbb{C}^{32 \times 32}$ [8].

over time, frequency, and space. A multi-path component MPC is characterized in delay and angular domains by its delay, angle of departure (Azimuth of Departure (AoD), Elevation of Departure (EoD), and angle of arrival (Azimuth of Arrival (AoA), Elevation of Arrival (EoA)). The MPCs with similar delays and angles are grouped into multi-path clusters. The MATLAB implementation of C2CM supports both single-link and multiple-link MIMO channel access indoor (285 MHz) and semi-urban (5.3 GHz) channel scenarios. An overview of the C2CM is presented in a detailed description of the channel model. The parameterization of the C2CM in indoor scenarios is detailed while discussing semi-urban scenarios. On the other hand, it gives the massive multiple-input multiple-output MIMO extensions of the C2CM; The C2CM is implemented in MATLAB, while the semi-urban channel scenario is implemented in [9]. Furthermore, the MATLAB implementation of C2CM with massive MIMO extensions. However, the data generated in these MATLAB implementations are not presented as potential datasets to validate multi-path clustering methods and even in the well-known clustering approaches.

3. Recurrent unit system model

3.1 The structure

It can adaptively capture capable of adaptively capturing dependencies from lengthy data sequences without removing data from previous stages of the sequence due to the gated recurrent unit structure [10]. This is accomplished by its gating units, which are related to those in long short-term memory LSTMs, and which resolve the vanishing/exploding gradient problem of conventional RNNs. These gates control the information that should be retained or discarded at each time step. The GRU operates like an RNN, except for its internal gating mechanisms, where sequential input data is absorbed by the GRU cell at each time step along with memory, also known as the hidden state [11]. The RNN cell and the following input data in the sequence are then fed with the hidden state once more (**Figure 3**).

Fully gated unit

Initially, for $t = 0$, the output vector is $h_0 = 0$.

$$z_t = \rho_g (W_z x_t + U_z h_{t-1} + b_z) \quad (5)$$

$$r_t = \rho_g (W_r x_t + U_r h_{t-1} + b_r) \quad (6)$$

$$h_t = z_t \odot h_{t-1} + (1 - z_t) \odot \phi_h (W_h x_t + U_h (r_t \odot h_{t-1}) + b_h) \quad (7)$$

Were, x_t input vector, h_t output vector, z_t update gate vector, r_t reset gate vector and W , U , and b denote matrices and vectors, respectively.

Activation functions: ρ_g Original sigmoid activation, ϕ_h For the initial hyperbolic tangent, Alternative activation functions can be used, provided the $\rho_g(x) \in [0,1]$.

It is possible to construct alternative forms by modifying z_t and r_t .

GRU's ability to hold on to long-term dependencies or memory stems from the gated recurrent unit cell's computations to produce the hidden state. At the same time, LSTMs have two different states passed between the cell state and hidden state, which carry the long and short-term memory, respectively GRUs only have one hidden state transferred between time steps. This hidden state can hold both long-term and short-

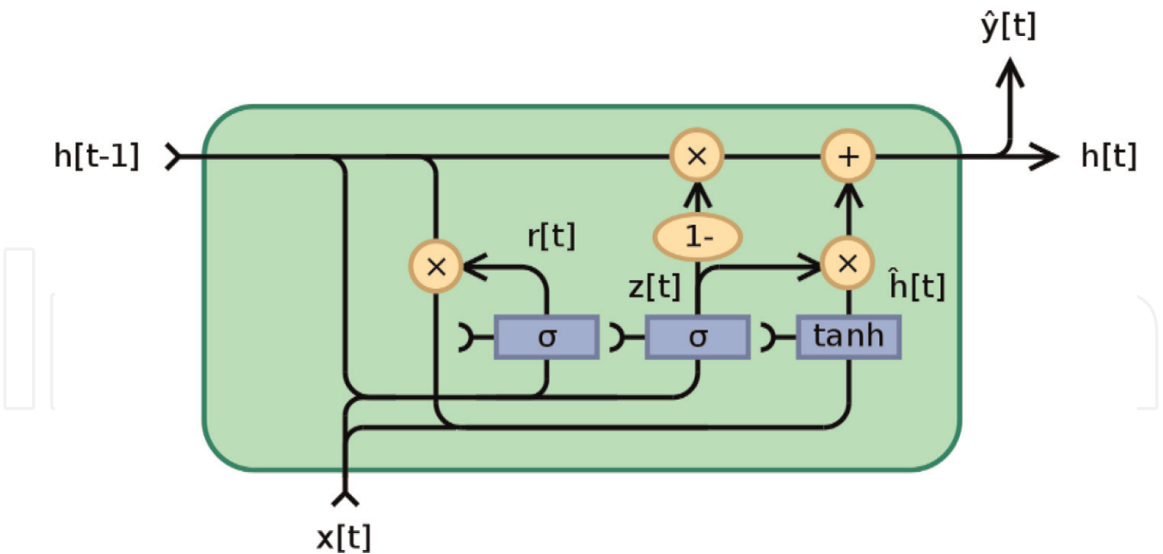


Figure 3.
Gated recurrent unit, fully gated version [12].

term dependencies at the same time due to the gating mechanisms and computations that the hidden state and input data go through.

The GRU cell contains only two gates: The Update gate and the Reset gate; like the gates in LSTMs, the GRU gates are trained to selectively filter out any irrelevant information while keeping what’s useful. These gates are essentially vectors containing values between 0 and 1, multiplying with the input data or hidden state.

A zero (0) value in the gate vectors indicates that the input or hidden state’s corresponding data is unimportant and will, therefore, return as a zero.

On the other hand, a one (1) value in the gate vector means that the corresponding data is essential and will be used. Reset gate: In the first step, we’ll create the Reset gate; this gate is derived and calculated using both the hidden state from the previous time step and the input data at the current time step (**Figure 4**).

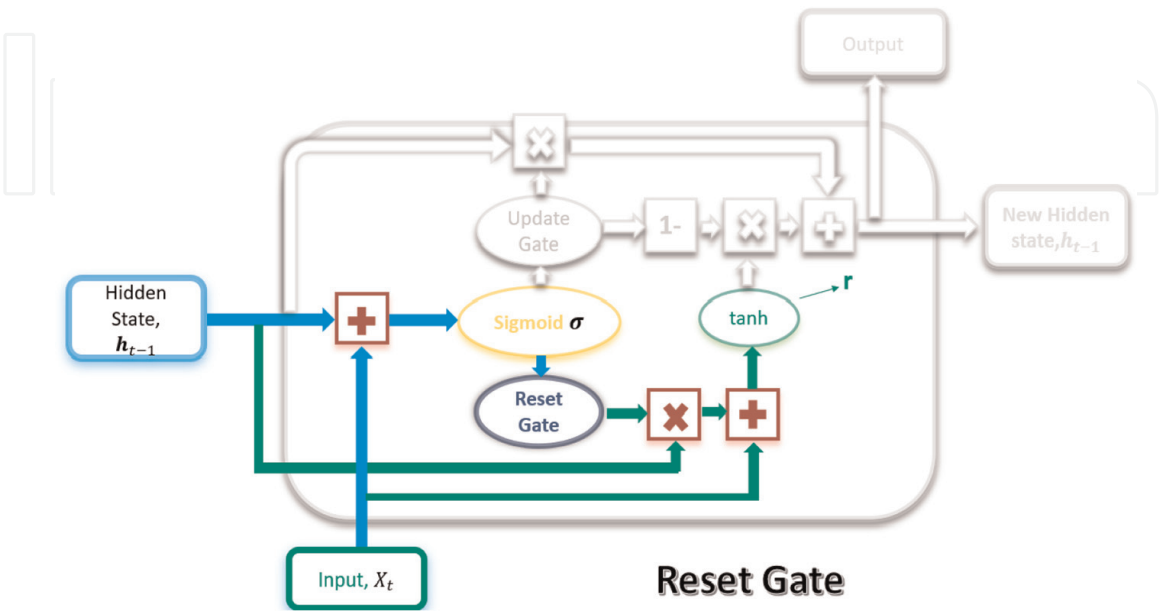


Figure 4.
Reset gate flow [13].

Mathematically, this is achieved by multiplying the previous hidden state and current input with their respective weights, summing before passing the sum through the sigmoid function.

The sigmoid function will transform the values to fall between 0 and 1, allowing the gate to filter between the less-important and more-important information in the subsequent steps.

$$Gate_{reset} = \sigma (W_{inputreset} \cdot x_t + W_{hiddenreset} \cdot h_{t-1}) \tag{8}$$

When the entire network is trained through back-propagation, the weights in the equation will be updated such that the vector will learn to retain only the valuable features. The previous hidden state will first be multiplied by a trainable weight and will then undergo an element-wise multiplication Hadamard product with the reset vector. This operation will decide which information will be kept from the previous time steps and the new inputs.

Simultaneously, the current input will also be multiplied by a trainable weight before being summed with the reset vector's product and the previous hidden state above. Finally, a non-linear activation tanh function will be applied to the result to obtain r in the equation below.

$$r = \tanh (gate_{reset} \odot (W_{h1} \cdot h_{t-1}) + W_{x1} \cdot x_t) \tag{9}$$

Update gate: next, we'll create the Update gate, like the Reset gate; the gate is computed using the previous hidden state and current input data (**Figure 5**). Both the Update and Reset gate vectors are created using the same formula, but the weights multiplied with the input and hidden state are unique to each gate, which means that each gate's final vectors are different; This allows the gates to serve their specific purposes.

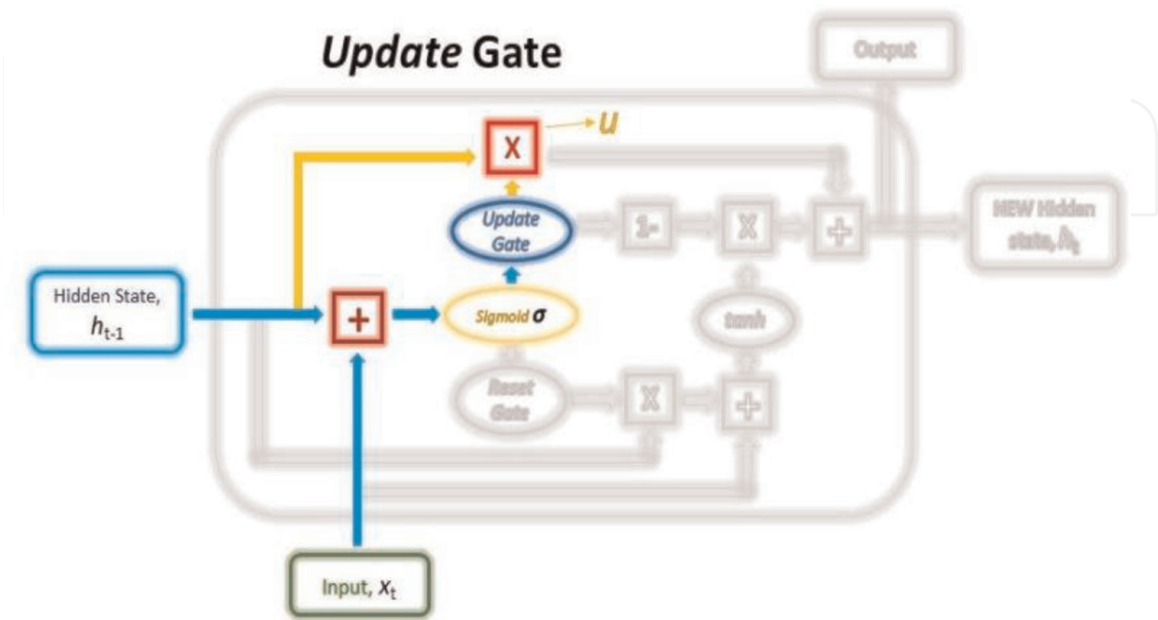


Figure 5.
Update gate flow [13].

$$gate_{update} = \sigma (W_{inputupdate} \cdot x_t + W_{hiddenupdate} \cdot h_{t-1}) \quad (10)$$

The Update vector will undergo element-wise multiplication with the previous hidden state to obtain u in our equation below, which will be used to compute our final output later.

$$u = gate_{update} \odot h_{t-1} \quad (11)$$

The Update vector will also be used in another operation later when obtaining our final output.

The purpose of the Update gate here is to help the model determine how much of the past information stored in the previous hidden state needs to be retained for the future. Combining the outputs: In the last step, we will be reusing the Update gate and obtaining the updated hidden state (**Figure 6**).

This time, we will be taking the element-wise inverse version of the same Update vector ($1 - \text{Update gate}$) and doing an element-wise multiplication with our output from the Reset gate, r . This operation's purpose is for the Update gate to determine what portion of the new information should be stored in the hidden state. Lastly, the result of the above operations will be summed with our output from the Update gate in the previous step, u .

This will give us our new and updated hidden state; We can use this new hidden state as our output for that time step by passing it through a linear activation layer.

$$h_t = r \odot (1 - gate_{update}) + u \quad (12)$$

The Reset gate determines which parts of the previous hidden state are to be combined with the current input to propose a new hidden state, and the Update gate determines how much of the previous hidden state is to be retained and what part of the new proposed hidden state derived from the Reset gate is to be added to the final

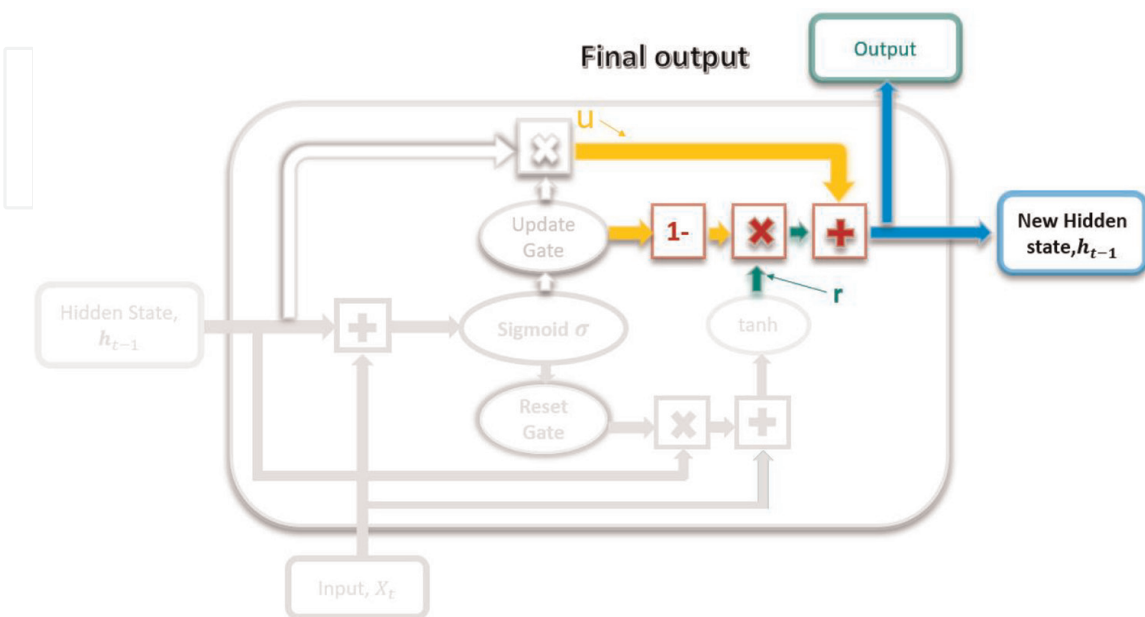


Figure 6.
Final output computations [13].

hidden state. This solves the Vanishing/Exploding Gradient Problem. The network chooses which components of the previous hidden state to keep in memory while discarding the rest when the Update gate is first multiplied with it. When it uses the Reset gate's inverse gate to filter the proposed new hidden state from the Update gate, it then fills in the gaps in the information that were previously missing. The network can maintain long-term dependencies as a result. If the Update vector values are close to 1, the Update gate may decide to keep most of the previous memories in the hidden state rather than recalculating or altering the hidden state entirely.

When training a recurrent neural network RNN, the vanishing or exploding gradient problem can happen, especially if the RNN is processing lengthy sequences or has multiple layers. The network's weight is updated in the right direction and by the right amount using the error gradient that was calculated during training. However, this gradient is determined using the chain rule, beginning at the end of the network. As a result, for long sequences, the gradients will undergo continuous matrix multiplications and either disappear (vanish) or explode (explode) exponentially.

A gradient that is too small will prevent the model from effectively updating its weights, whereas a gradient that is too large will make the model unstable.

Due to the additive nature of the Update gates, the long short-term memory (LSTM) and gated recurrent units (GRUs) can keep most of the existing hidden state while adding new content on top of it, unlike traditional RNNs that always replace the entire hidden state content at each time step.

This prevents the additional operations from causing the error gradients to vanish or explode too quickly during backpropagation. Utilizing alternative activation functions, like ReLU, which does not result in a small derivative, is the simplest solution. Another option is residual networks, which offer residual connections directly to earlier layers. In a feedforward network (FFN), the backpropagated error signal typically decreases (or increases) exponentially as a function of the distance from the final layer. This technique is referred to as the vanishing gradient.

4. Design of the CsiNet-GRUs system model

We enhance the architecture in this chapter by considering time correlation. The recurrent convolutional architecture that has been used to represent and reconstruct images successfully provides references for our work. The basic idea is to extract spatial features and inter-frame correlation using convolutional neural networks (CNN) and recurrent neural networks (RNN), respectively. The following is a summary of our contribution to this chapter. CsiNet is extended with a gated recurrent unit network, a common type of recurrent neural network, and a DL-based CSI feedback protocol for FDD MIMO systems is proposed (RNN). CsiNet-GRUs is a proposed network that modifies the CNN-based convolutional neural network CsiNet for channel state information compression and initial recovery and uses a gated recurrent unit technique to extract time correlation for further resolution improvement.

CsiNet-GRUs exhibit remarkable robustness to compression ratio (CR) reduction and enable real-time and extensible channel state information (CSI) feedback applications without considerably increasing overhead compared with CsiNet, to reduce feedback overhead, we can exploit the following observations: **Observation 1 (angular-delay domain sparsity):** H_t can be transformed into an approximately

sparsified matrix \mathbf{H}'_t in the angular-delay domain via 2D discrete Fourier transform (2D-DFT) by $\mathbf{H}'_t = \mathbf{F}_d \mathbf{H}_t \mathbf{F}_a$, where $\mathbf{F}_d \in \mathbb{C}^{N_c \times N_c}$ and $\mathbf{F}_a \in \mathbb{C}^{N_t \times N_t}$ are two DFT matrices. First, due to limited multipath time delay, performing DFT on frequency domain channel vectors (i.e., column vectors of \mathbf{H}_t) can transform \mathbf{H}_t into a sparse matrix in the delay domain, with only the first N'_c ($< N_c$) rows have distinct non-zero values. Secondly, the channel matrix is sparse in a defined angle domain by performing DFT on spatial domain channel vectors (i.e., row vectors of \mathbf{H}_t). If the number of transmit antennas, $N_t \rightarrow +\infty$, is very large. Usually, \mathbf{H}'_t is only approximately sparse for finite N_t which challenges conventional compressed sensing methods. Therefore, we will propose a DL-based feedback architecture without sparsity prior constraint. We perform sparsity transformation to decrease parameter overhead and training complexity.

We retain the first N'_c non-zero rows and truncate \mathbf{H}'_t to a $N'_c \times N_t$ matrix, \mathbf{H}''_t , which reduces the total number of parameters for feedback to $N = N'_c N_t$.

Observation 2 (correlation within coherence time): The user equipment motion during communication results in a Doppler spread, time-varying characteristics of wireless channels with the maximum movement speed denoted as \mathbf{v} , coherence time can be calculated as:

$$\Delta t = \frac{c}{2 v f_0} \quad (13)$$

where f_0 is the carrier frequency, and c is the speed of light. The CSI within Δt is considered correlated with one other. Therefore, instead of independently recovering CSI, the BS can combine the feedback and previous channel information to enhance the subsequent reconstruction.

We set the feedback time interval as δt and place \mathbf{T} adjacent instantaneous angular-delay domain channel matrices into a channel group, i.e.,

$$\{\mathbf{H}''_t\}_{t=1}^T = \{\mathbf{H}''_1, \dots, \mathbf{H}''_t, \dots, \mathbf{H}''_T\} \quad (14)$$

The group exhibits correlation property if \mathbf{T} satisfies $\mathbf{0} \leq \delta t \cdot \mathbf{T} \leq \Delta t$.

In this chapter, we design an encoder, $\mathbf{S}_t = f_{en}(\mathbf{H}''_t)$, at the UE to compress each complex-valued \mathbf{H}''_t of $\{\mathbf{H}''_t\}_{t=1}^T$ into an M-dimensional real-valued codeword vector \mathbf{S}_t ($M < N$). If two real number matrices are used to represent the real and imaginary parts of \mathbf{H}''_t , then CR will be $M/2N$.

We also design a decoder with a memory that can extract time correlation from the previously recovered channel matrices, $\hat{\mathbf{H}}''_1, \dots, \hat{\mathbf{H}}''_{t-1}$ and combine them with the received \mathbf{S}_t for the current reconstruction channel, $\hat{\mathbf{H}}''_t = f_{de}(\mathbf{S}_t; \hat{\mathbf{H}}''_1, \dots, \hat{\mathbf{H}}''_{t-1})$, where $1 \leq t \leq T$. Then, inverse 2D-DFT is performed to obtain the original spatial frequency channel matrix; the Channel state information network demonstrates remarkable performance in CSI sensing and reconstruction. However, the resolution degrades at low CR because it only focuses on angular-delay domain sparsity (Observation 1) and ignores the time correlation (Observation 2) of time-varying massive MIMO channels, the two observations are like the spatial and inter-frame correlations of videos.

Motivated by RCNN, which excels in extracting spatial-temporal features for video representation, we will extend CsiNet with GRU to improve CR and recovery quality

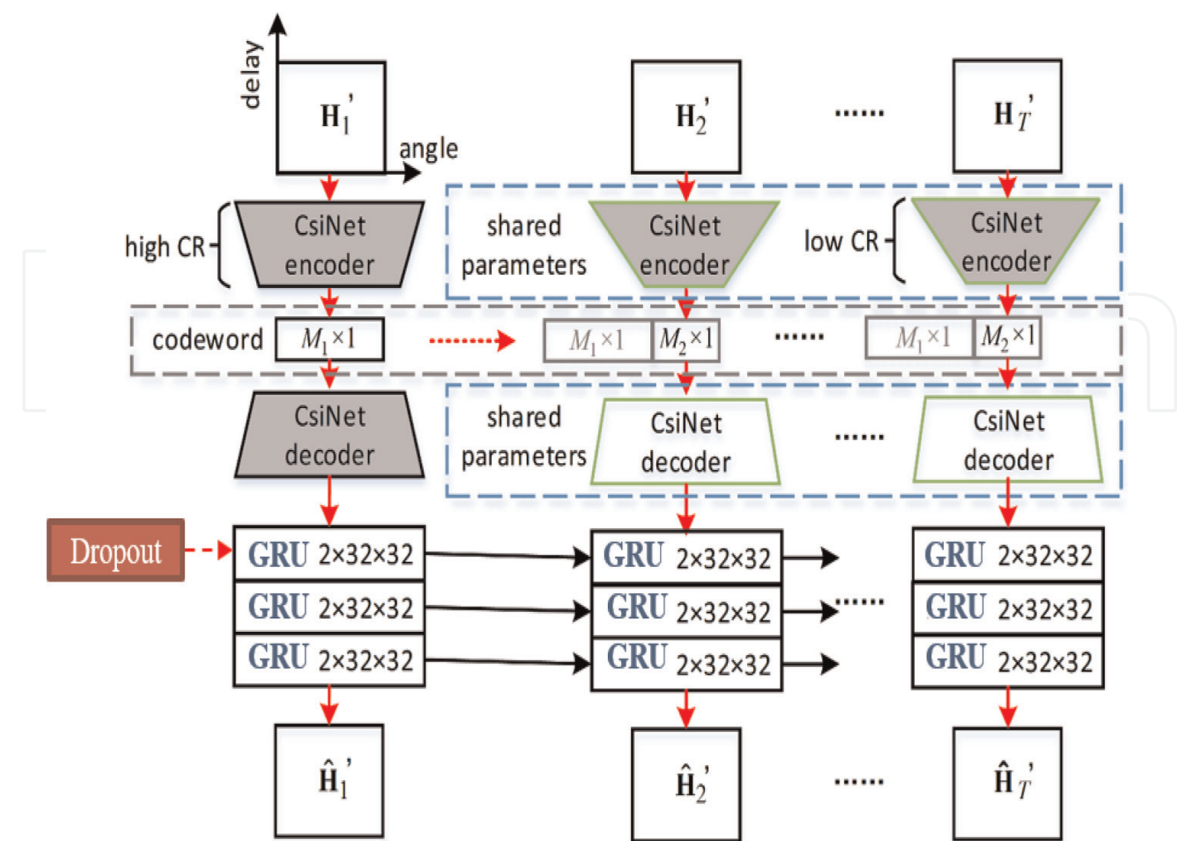


Figure 7.
The structure of the proposed CsiNet-GRUs using dropout technique [14].

trade-off. We will also introduce the multi-CR strategy to implement variable CRs on different channel matrices; The proposed CsiNet-GRU is illustrated in **Figure 7**. with CsiNet. Our model includes the following two steps: angular-delay domain feature extraction, correlation representation, and final reconstruction. Each GRU has an inherent memory unit that, for future prediction, can hold previously extracted information for a long time. A 3×3 convolutional layers and an M-unit dense layer for sensing, and a dense layer with $2N'_c N_t$ units should be considered to facilitate comparison with the results of the CsiNet structure given in [8] and two decoders from RefineNet for reconstruction as shown in **Figure 7**, each RefineNet comprises channel into four 3×3 convolutional layers with different channel sizes.

The CsiNet decoder's output generates a sequence, and the length of every sequence is T , which is then fed into a three-layer GRU. All low-CR CsiNet's shown in **Figure 7**. share the same network parameters, i.e., weights and bias, because they perform the same work, which dramatically reduces parameter overhead. Furthermore, the architecture can be easily rescaled to perform on channel groups with different T if the value of T changes to adapt to the channel-changing speed and feedback frequency; A low-CR CsiNet will be reused $(T - 1)$ time instead of making $(T - 1)$ copies in practice. The gray blocks in **Figure 7** load parameters from the original CsiNet's as pre-training before end-to-end training with the entire architecture. This method can alleviate vanishing gradient problems due to long paths from CsiNet's to GRUs. We use GRUs to extend the CsiNet decoders for time correlation extraction and final reconstruction. Gated recurrent units have inherent memory cells and can keep the previously extracted information for a long period for later prediction. In particular, the CsiNet decoders' outputs form a sequence of length T before

being fed into three-layer GRUs. Each GRU has a $2N'_cN_t$; The hidden unit is the same as the size of the output. Then the final output is reshaped into two matrices as the final recovered \hat{H}_t'' ; This allows the CR-CsiNet encoder to send to the rest $T - 1$.

Because less information is required due to channel correlation, the channel matrix performs operations, $M_2 \times 1$ codewords ($M_1 > M_2$), are generated. The $(T - 1)$ codewords are all concatenated with the first codeword $M_1 \times 1$ before being fed into the low-CR CsiNet decoder to utilize feedback information fully. Each CsiNet outputs two matrices with size $(N'_c \times N_t)$ as extracted features from the angular delay domain as the final recovered \hat{H}_t'' . The spatial frequency domain CSI can then be obtained via inverse 2D-DFT. At each time step, the GRUs implicitly learns time correlation from the previous inputs and then merge them with the current inputs to increase low CR recovery quality.

4.1 The dropout technique

Dropout: During training, randomly selected neurons are ignored and “dropped out.” This means that their contribution to downstream neuron activation is removed temporally on the forward pass, and no weight updates are applied to the neuron on the backward pass. Dropout can be implemented on any hidden layer in the network; the visible or input layer, as well as the term “dropout,” refers to dropping out units (hidden and visible) in a neural network. Dropout is a regularization method used when training the network, as illustrated in **Figure 8**. It is possible that the input and loop connections to the gated recurrent unit (GRU) in **Figure 7** are not excluded from activation and weight updates. Depending on the framework, the dropout regularization Training Phase: Ignore (zero out) a random fraction, p , of nodes for each hidden layer, training sample, and iteration (and corresponding activations). A phase of testing: Use all activations but reduce them by a factor of p . (to account for the missing activations during training). Dropout is a regularization method used when training the network, as shown in **Figure 8**. However, it does not always exclude the input and loop connections to the gated recurrent unit (GRU) from activation and weight updates, as shown in **Figure 7**. To reduce overfitting and improve the efficiency of the CsiNet-GRU structure, a neural network approach is used. We stated

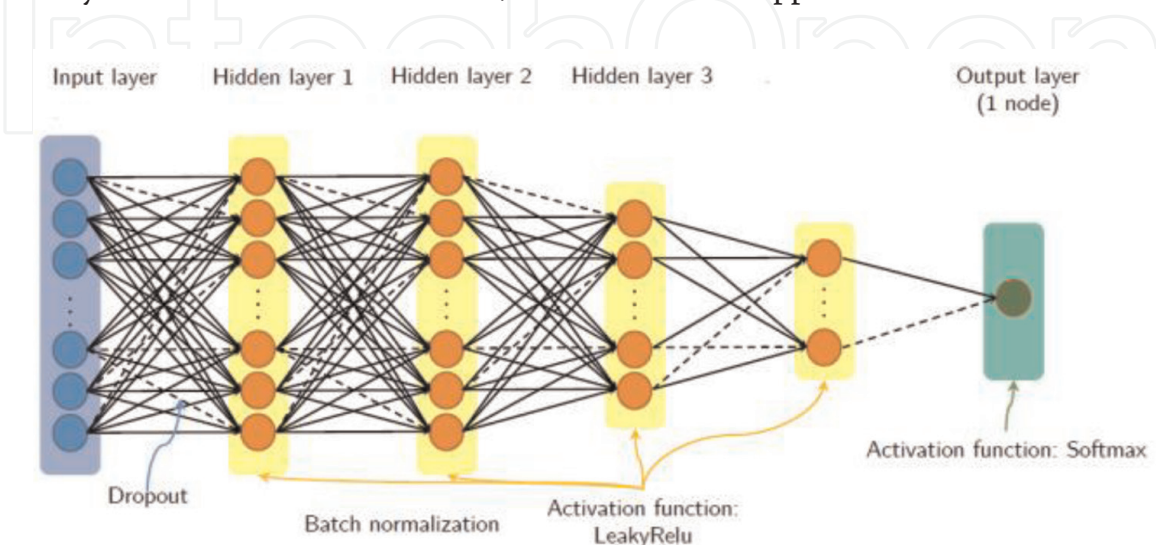


Figure 8.
Neural network with dropout architecture [15].

that the effect on “downstream neurons” activation during the forward process would be temporarily removed and that no weight update for “backward propagation to neurons” would be applied [15].

Training Phase: Ignore (zero out) a random fraction, p , of nodes for each hidden layer, training sample, and iteration (and corresponding activations). A phase of testing: Use all activations but reduce them by a factor of p . (to account for the missing activations during training). Dropout is a regularization method used when training the network, as shown in **Figure 8**. However, it does not always exclude the input and loop connections to the gated recurrent unit (GRU) from activation and weight updates, as shown in **Figure 7**. Depending on the framework, the dropout regularization approach used in neural networks is used to reduce overfitting and improve the efficiency of the CsiNet-GRU structure. We stated that the effect on “downstream neurons” activation during the forward process would be temporarily removed and that no “backward propagation to neurons” weight update would be applied [15]. Training Phase: Ignore (zero out) a random fraction, p , of nodes for each hidden layer, training sample, and iteration (and corresponding activations). A phase of testing: Use all activations but reduce them by a factor of p . (to account for the missing activations during training).

Some observations: Dropout forces a neural network to learn more robust features that can be used in conjunction with the random subsets of many other neurons. Dropout roughly doubles the number of iterations needed to converge; however, each epoch’s training time is less, and during the testing phase, the entire network is considered, and each activation is reduced by a factor of p . When training the network in the proposed structure, the input and recurrent connections to the GRU unit may not be excluded from activation and weight updates.

There are two dropout parameters in RNN layers: *dropout*, applied to the first operation on the inputs, and *recurrent dropout* applied to the other operation on the recurrent inputs. It is worth mentioning that interested in designing the encoder which can transform the channel matrix into an M -dimensional vector (codeword), where $M < N$. Thus, define the data compression ratio γ as ($\gamma = M/2N_tN_c$).

The encoder first extracts CSI features via a convolutional layer with two 3×3 filters, followed by an activation layer. A fully connected (FC) layer with M neurons is then used to compress the CSI features to lower dimensions. The compression ratio (CR) of this encoder can be expressed as $CR = 1/\gamma$. The final reconstruction of the CSI is performed by three $2N_cN_t$ unit GRUs with dropout techniques.

Moreover, adopting depth-wise separable convolutions in feature recovery reduces the model’s size and interacts with information between channels and introducing the delay θ as a parameter used in the encoder and decoder, i.e., $\theta = \{\theta_{en}, \theta_{de}\}$. It is worth mentioning that H_t'' are standardized with all components scaled into the $[0; 1]$, and this standardization is required for CsiNet.

5. COST 2100 data sets and parameters

The COST 2100 channel model is a geometry-based stochastic channel model (GSCM) capable of reproducing the stochastic properties of multi-link multiple-input multiple-output channels across time, frequency, and space. As a result, there is no doubt that more advanced channel estimation methods and good measurement campaigns for parameterization and validation are required for the successful development and long-term use of the COST 2100 channel model. Multiple-input

multiple-output (MIMO) is a technology that enables faster and more reliable transmissions over wireless channels.

The COST 2100 model simulates MIMO channels and generates training samples; we set the MIMO-OFDM system to work on a 20 MHz bandwidth using a uniform linear array (ULA). The parameters utilized in indoor and outdoor channel scenarios are given in **Table 1**; Data sets are generated by randomly setting different start places for indoor and outdoor scenarios and performing the simulations at CR values with the first channel H_1' they were compressed under 1/4. **Table 1** shows the training, validation, and testing sets; some parameters are preloaded from the CsiNet for initialization (epochs from 500 to 1000, learning rate of 0.001, and batch size of 200), as shown in **Table 1**.

We compare the proposed architecture’s performance with previous similar modeling approaches of channel state information (CSI) with different deep learning approaches, namely Conv-LSTM CsiNet, LASSO, TVAL3 [16], and CsiNet, utilizing the default setup in the open-source codes of the previously mentioned techniques for reproduction.

TVAL3 uses a minimum total variation method that provides remarkable recovery quality and high computing efficiency, while LASSO uses simple sparse priors to achieve good performance. In the feature extraction and recovery modules of Convolutional-LSTM CsiNet, RNN, and depth-wise separable convolution were used.

The term “training” refers to the process of determining which parameters to use in a given dataset. We run the modeling CsiNet-GRUs on Collaboratory (python) according to zero configuration required, free access to GPUs, and easy sharing training and testing of the CsiNet, Conv-LSTM CsiNet, and CsiNet-GRUs on python colab editor.

MIMO OFDM		20 MHz bandwidth
H		32×32
N_t		32 Antennas
N_C		1024 Subcarriers
Training samples		100,000
Validation samples		30,000
Testing samples		20,000
Epochs		500–1000
Learning rate		0.001
Batch size		200
∂t		0.04 s
T		10 s–100 s
CR		4, 16, 32, 64
Channel model	Indoor scenario	Outdoor scenario
Frequency, f	Pico, 5.3 GHz,	Rural, 300 MHz
Speed, v	0.0036 km/h	4.24 km/h
Δt	30 s	0.56 s

Table 1.
COST 2100 model DATA-SETS and system, parameters.

Comparisons are made using the normalized mean square error, cosine similarity, accuracy, and run-time in the indoor and outdoor channels, as well as the complexity factored in. The Normalized Mean Square Error measures and reflects the mean relative scatters.

The normalization of the MSE assures that the metric will not be biased when the model overestimates or underestimates the predictions. So, the normalized mean square error (NMSE) utilized for comparisons quantifies the difference between the input $\{\mathbf{H}_t\}_{t=1}^T$ and the output $\{\hat{\mathbf{H}}_t\}_{t=1}^T$ in both proposed techniques CsiNet-GRUs are given by:

$$NMSE = \mathbb{E} \left\{ \frac{1}{T} \sum_{t=1}^T \frac{\|\mathbf{H}_t'' - \hat{\mathbf{H}}_t''\|_2^2}{\|\mathbf{H}_t''\|_2^2} \right\} \quad (15)$$

The correlation coefficient is a statistical measure of the strength of the relationship between the relative movements of two variables. The values range between -1.0 and 1.0 . A correlation of -1.0 shows a perfect negative correlation, while a correlation of 1.0 shows a perfect positive correlation.

To measure the degree of similarity between the actual channel $\mathbf{h}_{n,t}$ and the estimated channel value $\hat{\mathbf{h}}_{n,t}$ of the n^{th} subcarrier at time t , using the cosine similarity coefficient ρ , in CsiNet-GRUs which is given as:

$$\rho = \mathbb{E} \left\{ \frac{1}{T} \frac{1}{N_c} \sum_{t=1}^T \sum_{n=1}^{N_c} \frac{|\hat{\mathbf{h}}_{n,t}^H \mathbf{h}_{n,t}|}{\|\hat{\mathbf{h}}_{n,t}\|_2 \|\mathbf{h}_{n,t}\|_2} \right\} \quad (16)$$

Where $\hat{\mathbf{h}}_{n,t}$ denotes the reconstructed channel vector of the n th subcarrier at time t . ρ can measure the quality of the beamforming vector when the vector is set as $\mathbf{v}_{n,t} = \hat{\mathbf{h}}_{n,t} / \|\hat{\mathbf{h}}_{n,t}\|_2$ since the UE will achieve the equivalent channel $\hat{\mathbf{h}}_{n,t}^H \mathbf{h}_{n,t} / \|\hat{\mathbf{h}}_{n,t}\|_2$.

Introducing a new parameter for comparison, which calling accuracy defining it as the ratio of the number of correct predictions to the total number of input samples, that means accuracy is the ratio of the recovered channel vector to the original channel vector $\{\mathbf{H}_t''\}_{t=1}^T / \mathbf{H}_1''$ so the accuracy in CsiNet-GRUs is defined as:

$$Accuracy = \mathbb{E} \left\{ \frac{1}{T} \frac{1}{N_c} \sum_{t=1}^T \sum_{n=1}^{N_c} \frac{|\hat{\mathbf{h}}_{n,t}^H|}{\|\mathbf{h}_{n,t}\|_2} \right\} \quad (17)$$

6. Comparison of results with different techniques

Figures 9 and **10** show the relationship between CR and NMSE for all structures in indoor and outdoor scenarios. **Figure 9** shows that the proposed CsiNet-GRUs have the lowest NMSE, whereas **Figure 10** shows that it has the lowest NMSE among others except for Conv-LSTM CsiNet at $CR > 20$. **Figures 11** and **12** show the relationship between the CR and accuracy for all structures in indoor and outdoor scenarios.

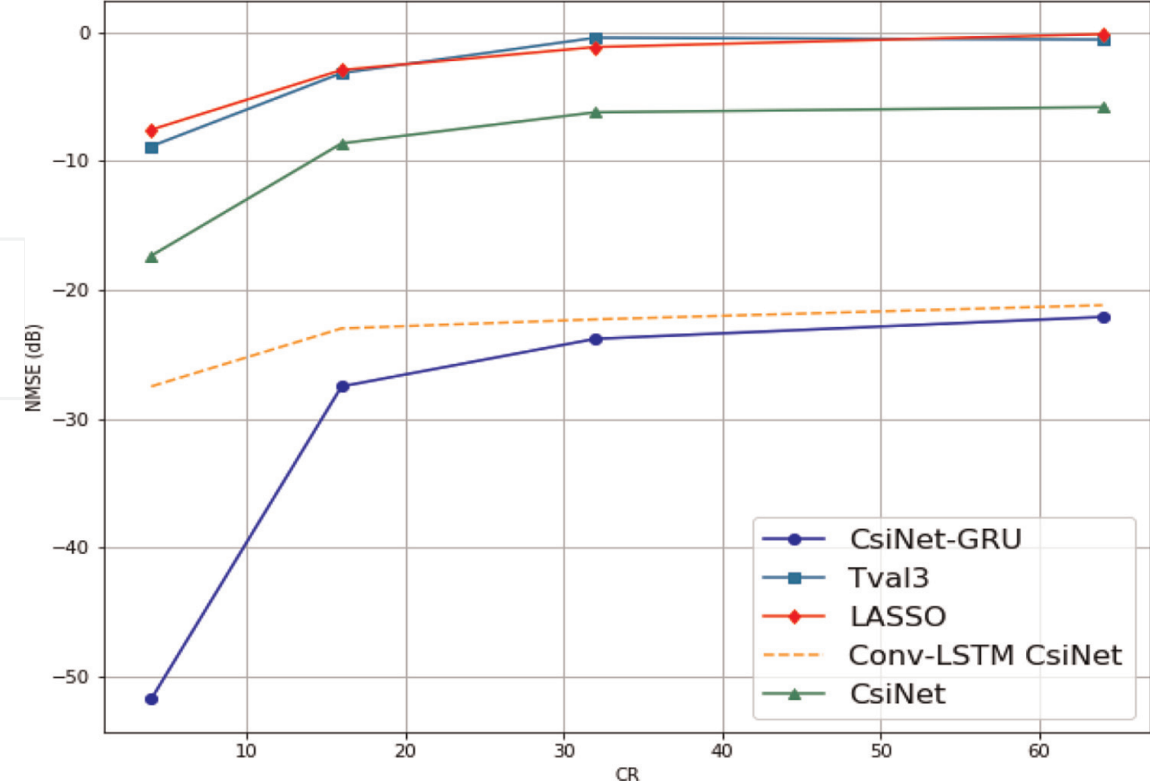


Figure 9.
NMSE (dB) performance comparison between CS methods INDOOR scenario.

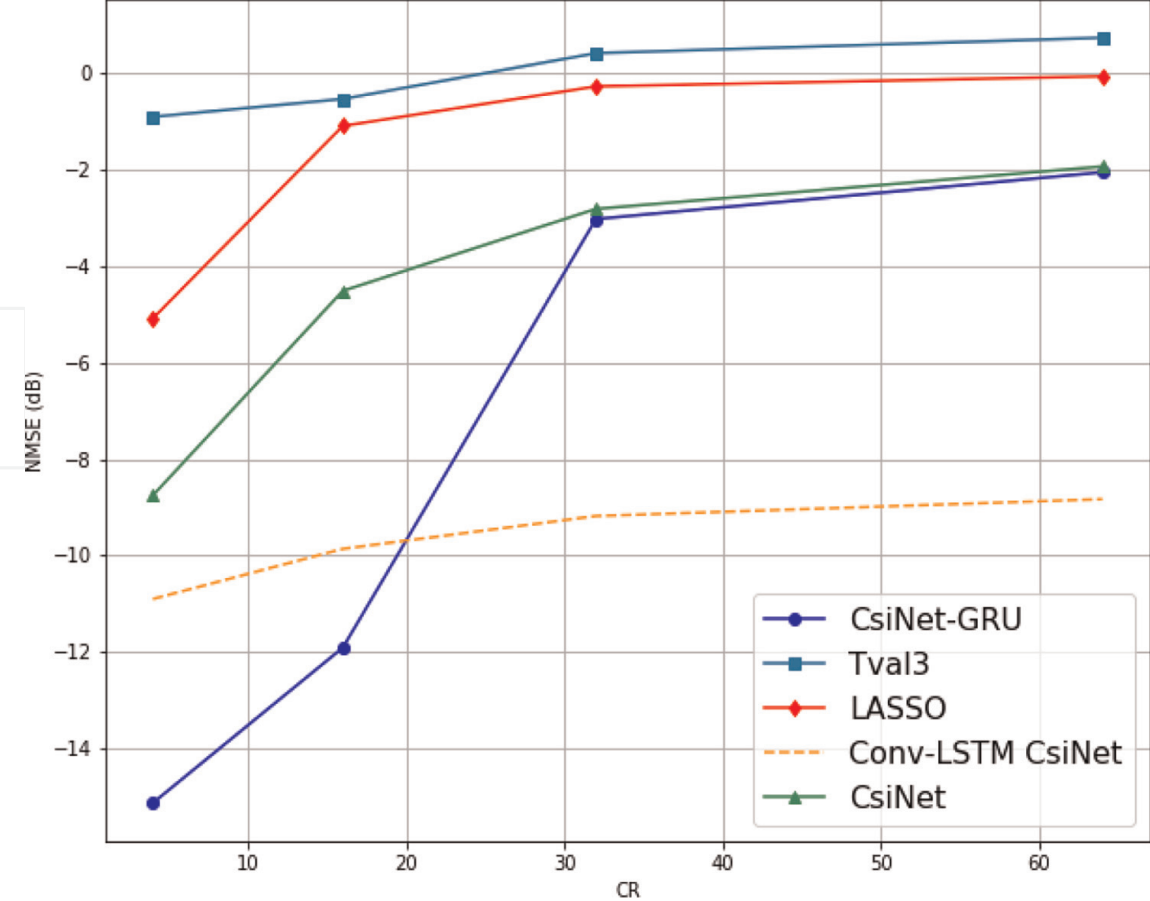


Figure 10.
NMSE (dB) performance comparison between CS methods OUTDOOR scenario.

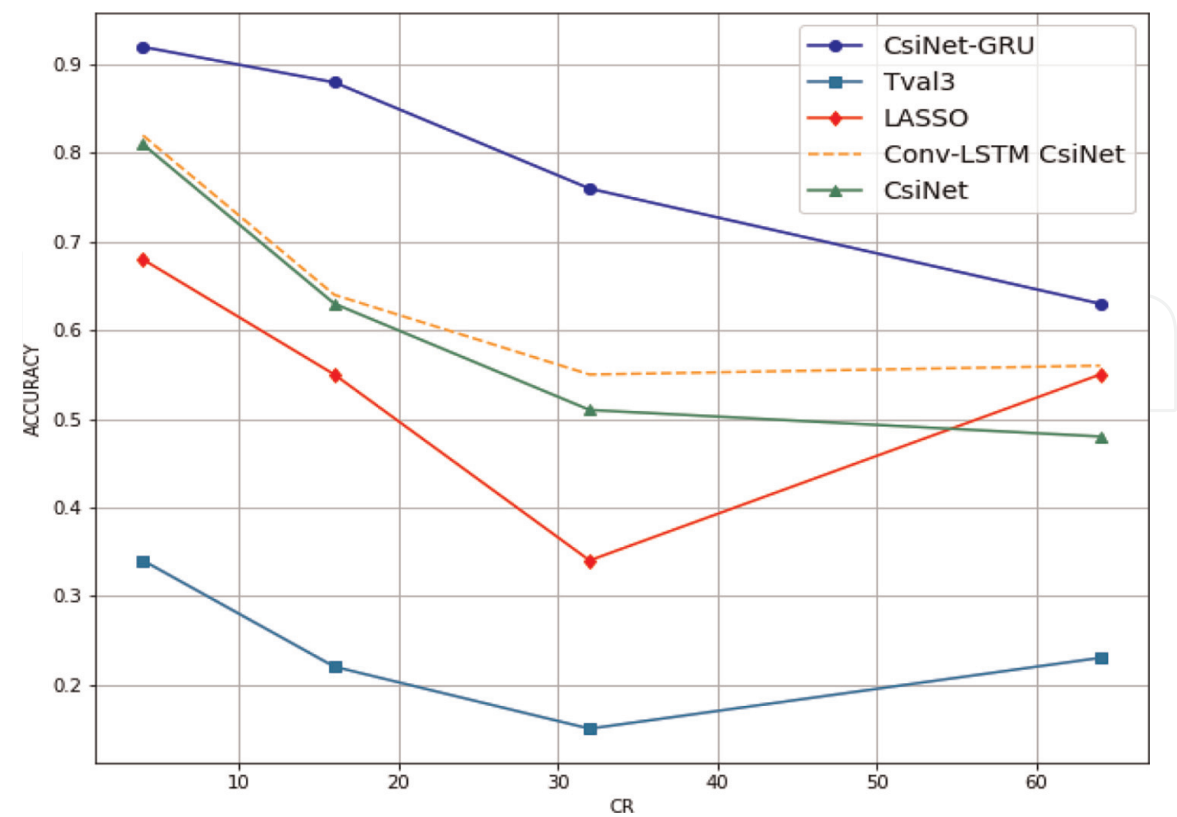


Figure 11.
Accuracy performance comparison between CS methods INDOOR scenario.

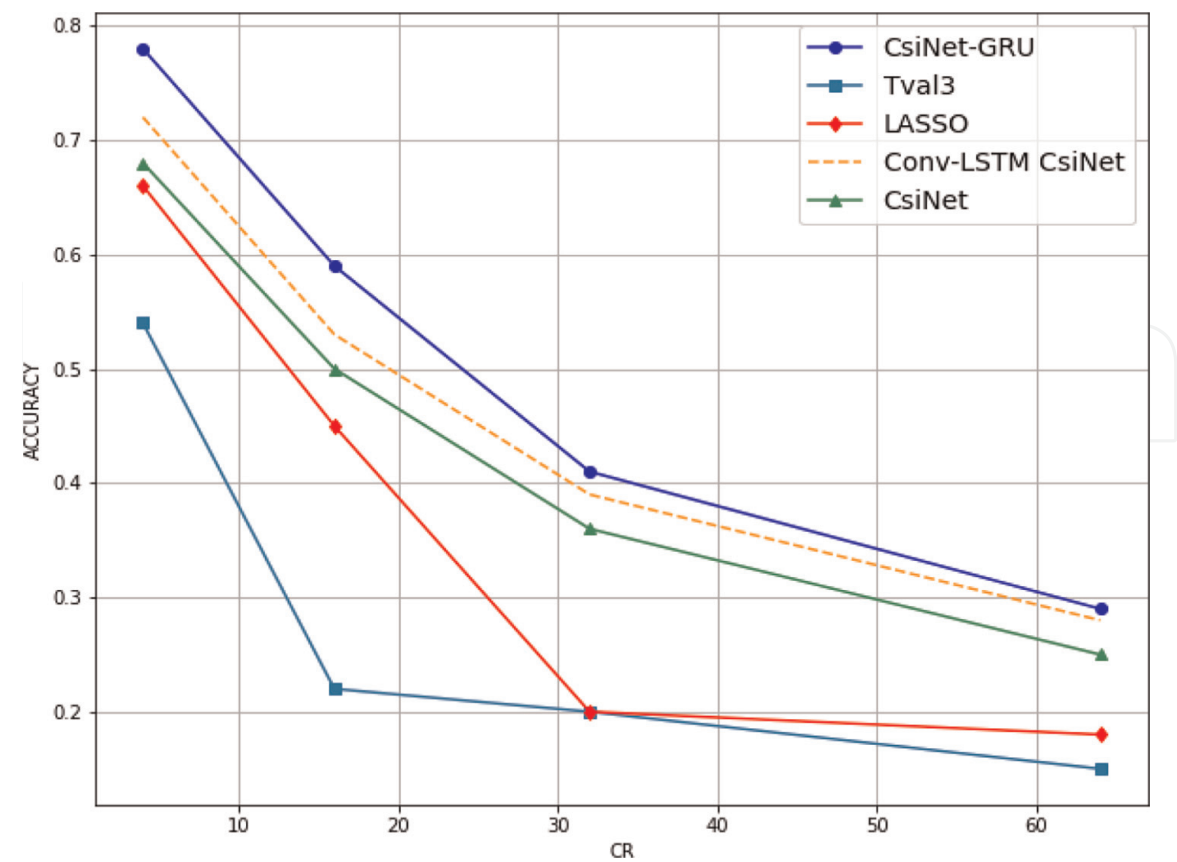


Figure 12.
Accuracy performance comparison between CS methods OUTDOOR scenario.

The CsiNet-GRUs outperform the other structures, with higher accuracy observed at lower CR values. **Figures 13** and **14** illustrate the relation between the cosine similarity (ρ) and CR in indoor and outdoor scenarios for all structures. Again, the proposed CsiNet-GRUs outperform the other structures, and besides, it exhibits a near-linear performance with the lowest slope.

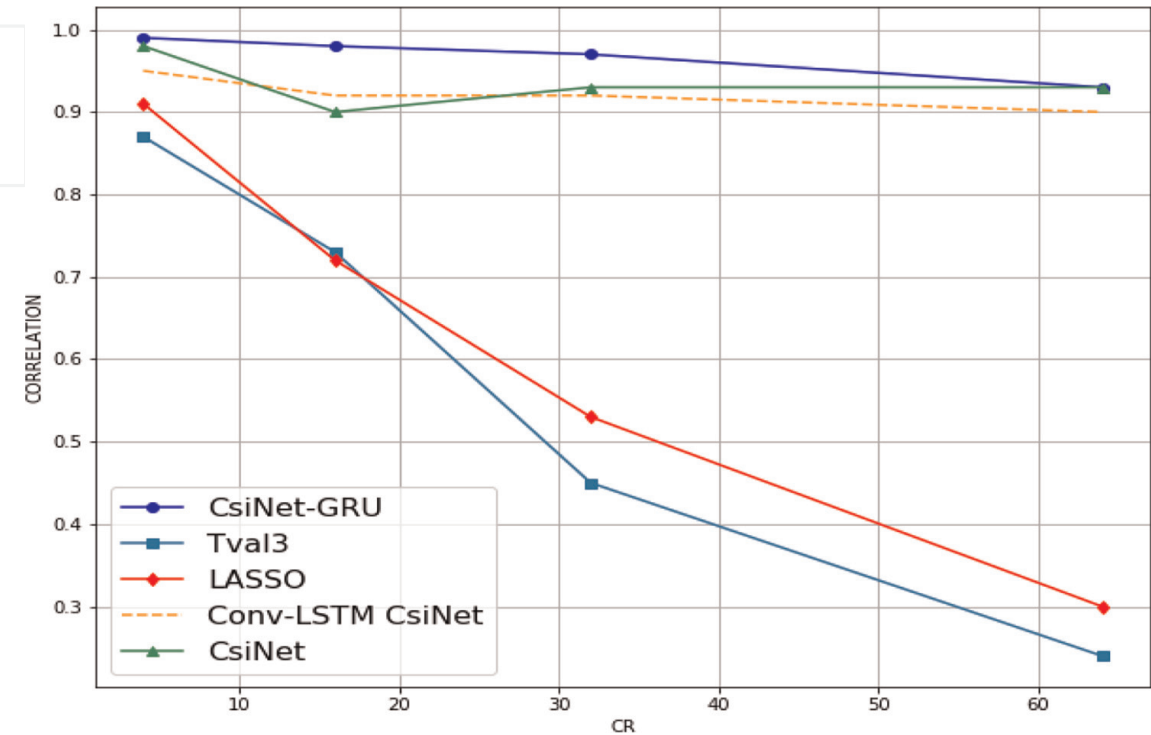


Figure 13.
 ρ Performance comparison between CS methods INDOOR scenario.

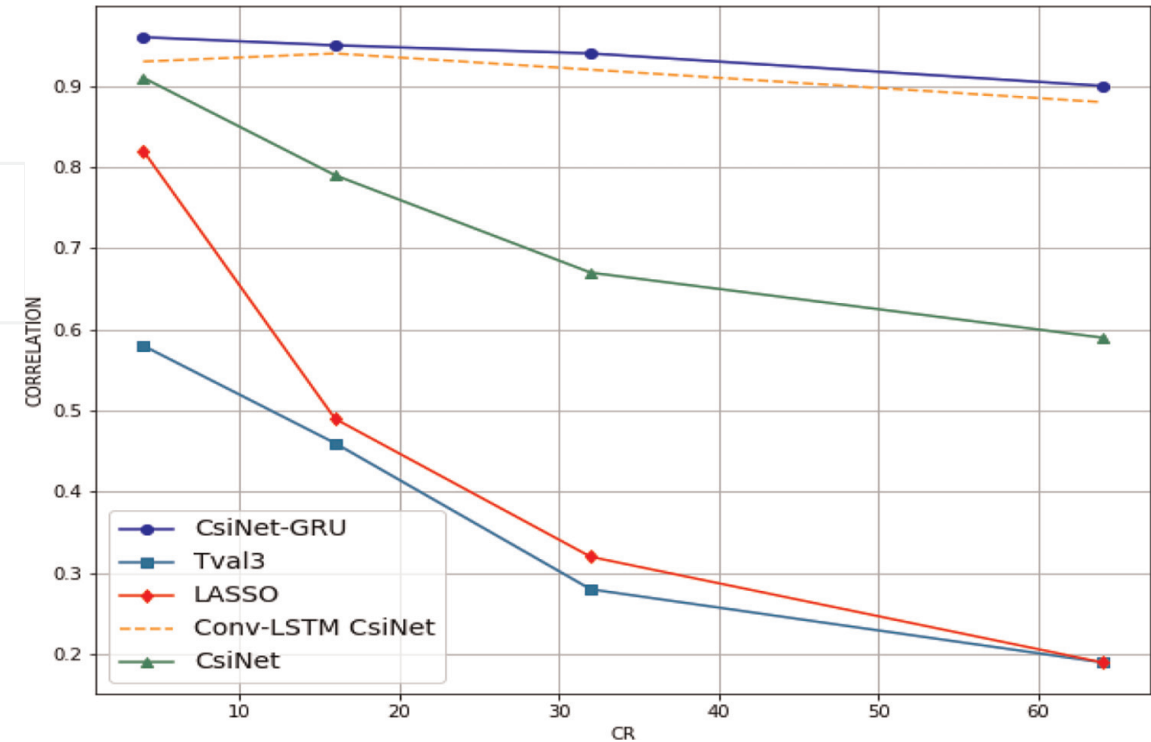


Figure 14.
 ρ Performance comparison between CS methods OUTDOOR scenario.

The performance comparison between the proposed CsiNet-GRUs to other available techniques. Where corresponding values of normalized mean square error (NMSE), ρ , accuracy, and run-time are calculated for different values of γ for indoor and outdoor scenarios, all techniques have better performances in the indoor scenario than the outdoor one.

It is worth noting that channel state information network (CsiNet) techniques significantly outperform the other CS-based methods. LASSO, TVAL3, CsiNet, and CsiNet-GRUs continue to provide the highest cosine similarity values at low CRs, where other CS-based methods fail. However, the proposed CsiNet-GRUs outperform the channel state information network (CsiNet) in terms of correlation and accuracy, as shown in **Table 2**. The same comparison is simulated again in terms of

		γ	LASSO [17]	TVAL3 [16]	CsiNet [8]	Conv-LSTM CsiNet [18]	CsiNet-GRUs Dropout Epoch/1000 [14]
Indoor	NMSE Epoch = 1000	1/4	-7.59	-8.87	-17.36	-27.5	-51.73
		1/16	-2.96	-3.2	-8.65	-23	-27.3
		1/32	-1.18	-0.46	-6.24	-22.3	-23.81
		1/64	-0.18	-0.6	-5.84	-21.2	-22.11
	ρ Epoch = 1000	1/4	0.91	0.87	0.98	0.95	0.99
		1/16	0.72	0.73	0.90	0.93	0.94
		1/32	0.53	0.45	0.83	0.85	0.89
		1/64	0.30	0.24	0.83	0.84	0.87
	Accuracy Epoch = 1000	1/4	0.68	0.34	0.81	0.82	0.84
		1/16	0.55	0.22	0.60	0.60	0.62
		1/32	0.34	0.15	0.51	0.51	0.52
		1/64	0.55	0.23	0.48	0.53	0.54
	Run time Epoch = 1000	1/4	0.345	0.314	0.0001	0.0001	0.0003
		1/16	0.555	0.314	0.0001	0.0001	0.0003
		1/32	0.604	0.286	0.0001	0.0001	0.0003
		1/64	0.708	0.285	0.0001	0.0001	0.0003
Outdoor	NMSE Epoch = 1000	1/4	-5.08	-0.9	-8.75	-10.9	-15.13
		1/16	-1.09	-0.53	-4.51	-9.86	-11.91
		1/32	-0.27	0.42	-2.81	-9.18	-3.02
		1/64	-0.06	0.74	-1.93	-8.83	-2.05
	ρ Epoch = 1000	1/4	0.82	0.58	0.87	0.90	0.92
		1/16	0.49	0.46	0.79	0.81	0.81
		1/32	0.32	0.28	0.67	0.68	0.70
		1/64	0.19	0.19	0.60	0.62	0.68
	Accuracy Epoch = 1000	1/4	0.66	0.54	0.68	0.70	0.71
		1/16	0.45	0.22	0.49	0.49	0.51
		1/32	0.20	0.20	0.36	0.36	0.38
		1/64	0.18	0.15	0.26	0.26	0.27

	γ	LASSO [17]	TVAL3 [16]	CsiNet [8]	Conv-LSTM CsiNet [18]	CsiNet-GRUs Dropout Epoch/1000 [14]
Run time Epoch = 1000	1/4	0.2122	0.15	0.0001	0.0001	0.0003
	1/16	0.2409	0.3145	0.0001	0.0001	0.0003
	1/32	0.598	0.2985	0.0001	0.0001	0.0003
	1/64	0.6758	0.285	0.0001	0.0001	0.0003

Table 2.
Comparison of results between the proposed framework and other available Ones (Epoch = 1000 iterations in the proposed techniques and others previous techniques).

epoch = 1000 (1000 iterations) in terms of correlation and accuracy in the proposed technique CsiNet-GRUs. In terms of the NMSE, the CsiNet-GRUs achieve the lowest values of all compressed ratios (CRs), particularly when CR is low.

CsiNet-GRUs have very short run periods when compared to LASSO and TVAL3 techniques. However, when compared to the other CsiNet technique and the proposed modeling technique, CsiNet-GRUs lose time efficiency slightly. It is worth noting that, despite the addition of significant complexity as a result of the GRU layers, the run time is still comparable to that of the CsiNet.

Figure 15 depicts in comparison to the other modeling techniques, the reconstruction results of the proposed technique, namely LASSO, TVAL3, CsiNet, and Conv-LSTM CsiNet in an indoor Picocellular scenario, the figure represents the average performance at different CRs, reflecting on the reconstructed images to use the other techniques.

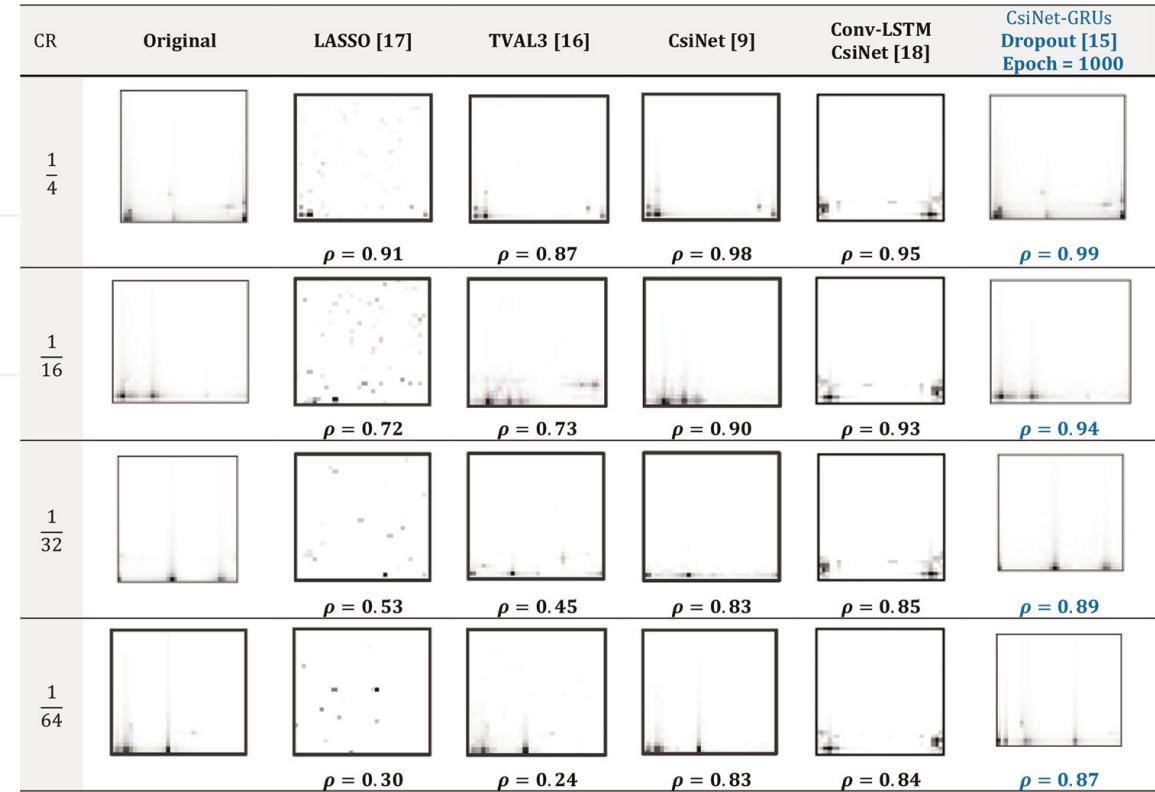


Figure 15.
Reconstruction images for CR in CS algorithms in an indoor scenario.

Conv-LSTM, CsiNet, and CsiNet-GRUs continue to provide acceptable correlation coefficients (ρ) at low CRs, where compressed sensing-based methods fail; it is noteworthy that the proposed CsiNet-GRUs technique outperforms the other methods in an indoor scenario. CsiNet-GRUs achieve the best performance among CR with indicators parameters to improve accuracy, decrease NMSE, and increase correlation (ρ) with dropout to reduce modeling system overfitting in massive multiple-input multiple-output channels. As a result, CsiNet-GRUs outperform both CsiNet and CS-based methods. With advanced deep learning technology, this chapter has the potential to deploy real MIMO systems.

7. Conclusion

We developed and tested a prediction model to evaluate a real-time and end-to-end channel state information (CSI) feedback framework in this chapter by extending the DL-based CsiNet with GRU. By utilizing the time correlation and structure properties of time-varying massive MIMO channels, CsiNet-GRUs achieve an acceptable trade between compressed ratio, recovery quality, accuracy, and complexity. This chapter proposed a channel state information (CSI) feedback network by extending the deep learning DL-based channel state information network (CsiNet) technique to incorporate gated recurrent units (GRUs) and use the dropout method to incorporate gated recurrent units (GRUs) and use the dropout method. The gated recurrent unit (GRU) layers were used to extend the channel state information network CsiNet decoders for time correlation extraction and the final reconstruction of channel state information, whereas the dropout method was used to reduce overfitting in channel modeling. In terms of complexity, the experiment results show that CsiNet-GRUs achieve the best recovery quality and outperform state-of-the-art CS methods.

Appendices and nomenclature

1G	The first generation
2G	The second generation
3G	The third generation
4G	The fourth generation
5G	The fifth generation
AI	Artificial Intelligence
AMP	Approximate Message-Passing
AoA	Analysis of Alternatives
AE	Autoencoder
BER	Bit Error Rate
CE	Channel Estimation
CNN	Convolutional Neural Network
CS	Compressive Sensing
CSI	Channel State Information
CsiNet	Channel State Information Network
CsiNet-GRU	Channel State Information Network-Gated Recurrent Unit
DL	Deep Learning
DNN	Deep Neural Network
FDD	Frequency Division Duplex

GRU	Gated Recurrent Unit
LASSO	Least Absolute Shrinkage and Selection Operator
LSTM	Long Short-Term Memory
MIMO	Multiple-Input Multiple-Output
mmWave	Millimeter Wave
MSE	Mean Squared Error
NMSE	Normalized Mean Square Error
RELU	Rectified Linear Unit
RNN	Recurrent Neural Network
SNR	Signal-to-Noise-Ratio
$(\cdot)^H$	Conjugate transpose
ρ_g	The original is a sigmoid function
ϕ_h	The original is a hyperbolic tangent
γ	Compression Ratio
θ	The delay
∂t	Feedback internal
$\ \cdot\ _2$	The Euclidean norm
\mathbb{E}	Exponent
\mathbb{C}	Complex numbers
\sum	Summation
ϵ	Element
\tilde{h}_n^H	The channel frequency response vector at the n^{th} subcarriers
h_t	Output vector
$L(\theta)$	Loss Function
N_c	Receiver antenna at the user equipment
N_t	Transmit antenna at the base station
r_t	Reset gate vector
T	Time sequence of the channel model
x_n	The transmitted information image
x_t	Input vector
y_n	The pre-coding vector at the n^{th} subcarriers
z_n	The additive noise or obstruction
z_t	Update gate vector

IntechOpen

Author details

Hany Helmy^{1*}, Sherif El Diasty² and Hazem Shatila³


1 Cairo Airport Company (CAC), Cairo, Egypt

2 Department of Electronics, Arab Academy for Science, Technology and Maritime Transport (AASTMT), Cairo, Egypt

3 Virginia Tech, Artificial Intelligence and Markovdata, Cairo, Egypt

*Address all correspondence to: hany.nabil@cairo-airport.com; hnabil110@gmail.com

IntechOpen

© 2023 The Author(s). Licensee IntechOpen. This chapter is distributed under the terms of the Creative Commons Attribution License (<http://creativecommons.org/licenses/by/3.0>), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited. 

References

- [1] Zhang T, Ge A, Beaulieu NC, Hu Z, Loo J. A limited feedback scheme for massive MIMO systems based on principal component analysis. *EURASIP Journal on Advances in Signal Processing*. 2016;2016. DOI: 10.1186/s13634-016-0364-9
- [2] Busari A, Huq KMS, Mumtaz S, Dai L, Rodriguez J. Millimeter-wave massive MIMO communication for future wireless systems: A survey. *IEEE Communications Surveys & Tutorials*. 2018;20(2):836-869
- [3] Tao J, Chen J, Xing J, Fu S, Xie J. Autoencoder neural network based intelligent hybrid beamforming design for mmWave massive MIMO systems. *IEEE Transactions on Cognitive Communications and Networking*. 2020. DOI: 10.1109/TCCN.2020.2991878
- [4] Zhai J, Zhang S, Chen J, He Q. Autoencoder and Its Various Variants. In: 2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Miyazaki, Japan. 2018. pp. 415-419. DOI: 10.1109/SMC.2018.00080
- [5] Karanov B, Lavery D, Bayvel P, Schmalen L. End-to-end optimized transmission over dispersive intensity-modulated channels using bidirectional recurrent neural networks. *Optics Express*. 2019;27:19650-19663
- [6] Sohrabi F, Cheng HV, Yu W. Robust Symbol-Level Precoding Via Autoencoder-Based Deep Learning. 2020. pp. 8951-8955. DOI: 10.1109/ICASSP40776.2020.9054488
- [7] Liu Z, del Rosario M, Liang X, Zhang L, Ding Z. Spherical Normalization for Learned Compressive Feedback in Massive MIMO CSI Acquisition. 2020. pp. 1-6. DOI: 10.1109/ICCWorkshops49005.2020.9145171
- [8] Wen C, Shih W, Jin S. Deep learning for massive MIMO CSI feedback. *IEEE Wireless Communications Letters*. 2018;7(5):748-751
- [9] Liu L, Oestges C, Poutanen J, Haneda K. The COST 2100 MIMO channel model. *IEEE Wireless Communications*. 2012;19(6):92-99
- [10] Hochreiter S. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*. 1998;6(2):107-116
- [11] Aleem S, Huda N, Amin R, Khalid S, Alshamrani SS, Alshehri A. Machine Learning Algorithms for Depression: Diagnosis, Insights, and Research Directions. *Electronics*. 2022;11(7):1111. DOI: 10.3390/electronics11071111
- [12] Cho K, van Merriënboer B, Gulcehre C, Bahdanau D, Bougares F, Schwenk H, et al. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. 2014. DOI: 10.3115/v1/D14-1179
- [13] Dey R, Salem FM. Gate-variants of Gated Recurrent Unit (GRU) neural networks. 2017. pp. 1597-1600. DOI: 10.1109/MWSCAS.2017.8053243
- [14] Helmy HMN, Daysti SE, Shatila H, Aboul-Dahab M. Performance enhancement of massive MIMO using deep learning-based channel estimation. *IOP Conference Series: Materials Science and Engineering*. 2021;1051(1):012029
- [15] Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout: A simple way to prevent neural networks from overfitting. *Journal of*

Machine Learning Research. 2014;**15**(1):
1929-1958

[16] Li C, Yin W, Zhang Y. User's guide
for TVAL3: TV minimization by
augmented Lagrangian and alternating
direction algorithms. CAAM Report.
2009;**20**(4):46-47

[17] Daubechies I, Defrise M, Mol CD. An
iterative thresholding algorithm for
linear inverse problems with a sparsity
constraint. Communications on Pure and
Applied Mathematics. 2004;**75**:1412-
1457

[18] Li X, Huaming W. Spatio-temporal
representation with a deep neural
recurrent network in MIMO CSI
feedback. IEEE Wireless
Communications Letters. 2020;
9(5):653-657