Deep Learning Channel Estimation Based on Edge Intelligence for NR-V2I

Yong Liao, *Member, IEEE*, Zhirong Cai, Guodong Sun, Xiaoyi Tian, Yuanxiao Hua, and Xiaoheng Tan

***Abstract*— The fifth generation New Radio vehicle-to-everything (5G NR-V2X) has higher requirements for delay and reliability, which brings challenges to the physical layer signal processing. Mobile edge computing (MEC) can provide larger capacity data storage and more efficient computation through localization of on-board unit (OBU) and next generation Node B (gNB) services. This paper combines the channel estimation of New Radio vehicle-to-infrastructure (NR-V2I) communication baseband signal processing with MEC and designs an intelligent channel estimation framework. In the MEC server, this paper proposes a channel estimation algorithm based on deep learning. This algorithm uses a one-dimensional convolutional neural network (1D CNN) to complete frequency-domain interpolation and conditional recurrent unit (CRU) for time-domain state prediction. Additional velocity coding vector and multipath coding vector track changes in the environment, and accurately train channel data in different mobile environments. System simulation and analysis show that the proposed algorithm improves the channel estimation accuracy, reduces the bit error rate, and enhances the robustness compared with the representative channel estimation algorithms for NR-V2I.**

***Index Terms*— V2X, mobile edge computing, channel estimation, deep learning, edge intelligence.**

I. Introduction

THE application of 5G cellular mobile communication has promoted tremendous development in many fields. As an important application field supported by 5G, vehicle-to-everything (V2X) communication has its core technical requirements for Ultra-Reliable Low-Latency Communication (URLLC) [1], [2]. Nowadays, due to the development of V2X facilities and mobile terminal applications, data traffic has increased rapidly. In order to ensure that the transportation system is more secure, the requirements for communication delay are also constantly increasing, which demands more for 5G NR-V2X communication [3]–[5]. Mobile edge computing (MEC) can transfer some functions of on-board units (OBUs) and next generation Node B (gNB) to nearby MEC servers, which can increase information storage and increase data processing speed, thereby reducing communication delay and improving the effect of reliable data transmission rate [6]–[8].

Manuscript received May 21, 2020; revised November 1, 2020 and March 16, 2021; accepted October 1, 2021. This work was supported in part by the National Natural Science Foundation of China under Grant 61501066 and in part by the Natural Science Foundation of Chongqing, China, under Grant cstc2019jcyj-msxmX0017. The Associate Editor for this article was C. Guo. *(Corresponding author: Yong Liao.)*

The authors are with the School of Microelectronics and Communication Engineering, Chongqing University, Chongqing 400054, China (e-mail: liaoy@cqu.edu.cn).

Digital Object Identifier 10.1109/TITS.2021.3123339

In order to ensure the regular operation of the intelligent transportation system (ITS), V2X communication must provide efficient and reliable information transmission. In view of the special scenario of V2X communication, users have high requirements on communication delay, reliability, and other indicators in practical applications, which is also an important factor to ensure road traffic safety. Channel estimation has received wide attention as a key process in V2X communication. The current global road network is very complete, and the relative velocity between vehicles can reach up to 400km/h, which means that the process of V2X communication will be affected by severe Doppler shift [9]. The cyclic prefix orthogonal frequency-division multiplexing (CP-OFDM) system is used in NR-V2X physical layer standard may cause severe inter-subcarrier interference (ICI) even at small frequency offsets, which brings great difficulties to the synchronization and channel estimation of wireless communication. Therefore, the performance of channel estimation will greatly affect the effect of V2X communication [10], [11].

In view of the above problems, in our previous research [12], model training is performed on the simulated channel data through deep learning, and the deep learning model is used to solve the channel tracking of time-varying channels. Based on [12], this paper designs an intelligent channel estimation framework based on MEC and propose a deep learning channel estimation algorithm. The specific contributions are as follows.

(1) A framework for intelligent channel estimation based on MEC is designed. For the vehicle-to-infrastructure (V2I) scenario, the MEC server deployed in the gNB performs channel frequency response (CFR) storage and deep learning channel estimation model training and then transmits the trained prediction model to the OBU and roadside unit (RSU) to complete channel estimation. At the same time, OBU and RSU can feedback the estimated CFR of the correctly received signal to the gNB-side MEC server to enrich the CFR data set.

(2) In order to cope with the problem that the varying channel causes the performance of the estimation algorithm to decrease significantly in the high-speed mobile environment, this paper designs the multipath coding vector and the velocity coding vector, and then uses the convolutional neural network (CNN) to complete the frequency-domain interpolation and conditional recurrent unit (CRU) to perform time-domain state prediction. Through offline training, the algorithm can learn a priori multipath coding and velocity coding vector and realize accurate training of channel data in different mobile environments. Finally, the effectiveness of the proposed algorithm is verified through simulation.

The rest of this paper is organized as follows. In Section II, relevant studies are listed. In Section III, we propose the MEC-based intelligent MEC channel estimation framework, signal processing system model, and V2I channel model. In Section IV, the implementation process of the deep learning-based channel estimation algorithm is described. In section V, we analyze and compare the algorithm complexity of the algorithms involved in this article. In Section VI, the performance comparison of each algorithm is completed through system simulation. Finally, we conclude the paper in Section VII.

II. Related Work

Due to the complexity of the V2X wireless channel, channel estimation has higher real-time and accuracy requirements. Therefore, the pilot-based channel estimation method is adopted in V2X communication. Least squares (LS) and linear mean square error estimation (LMMSE) [13] are the most classic channel estimation methods, and the linear interpolation can be used to obtain complete channel matrix information. Reference [14] introduces an enhanced equalization scheme spectral time averaging (STA). This method uses the decision symbols from data subcarriers and averages both in the time and frequency-domains to update the channel estimate, improving the V2X channel estimation performance. Reference [15] proposes a channel estimation scheme based on long training pilot sequences to estimate channel frequency response and construct data pilot (CDP). The scheme uses data symbols to construct pilots and utilizes the channel correlation characteristics in two adjacent symbols to further optimize the accuracy of channel updates. Reference [16] makes improvements to the performance defects of the CDP algorithm under low signal-to-noise ratio (SNR), and proposed a modified CDP algorithm (SAMCDP) assisted by SNR. For different SNR regions, different estimation methods are used to ensure good performance in low SNR and high SNR. In order to further improve the accuracy of channel estimation, some algorithms add the process of channel coding and decoding into the channel estimation, and update it repeatedly. The traditional pilot-assisted channel estimation scheme based on iterative coding and decoding is difficult to be applied in practice due to its high complexity and error propagation. In [17], a state feedback decision (SFD) algorithm is proposed, which can further improve the accuracy of channel estimation by extracting more reliable data pilots from each data symbol without using a conventional decoder, and the computational complexity of the algorithm is also significantly reduced. In [18], pilot symbols based on basis expansion model (BEM) is proposed, and the channel coefficients of data symbols are reconstructed by using the piecewise interpolation method based on Slepian sequence (SS-PWI). The simulation results also show that the method can get smaller mean square error results of channel estimation.

In recent years, channel estimation based on deep learning has developed rapidly. Reference [12] proposes a channel estimation algorithm based on deep learning. In this method, the frequency-domain interpolation and the interpolation coefficient are jointly completed by using the deep neural networks (DNN), and the time-domain state prediction and the time-domain correlation coefficient are jointly completed by using the long short-term memory (LSTM). In [19], to address channel distortion, the deep learning-based approach estimates CSI implicitly and detects the transmitted symbols with performance comparable to the minimum mean-square error estimator. Reference [20] proposes a general pipeline using deep image processing techniques, image super-resolution (SR), and image restoration (IR). This scheme considers the pilot values, altogether, as a low-resolution image and uses an SR network cascaded with a denoising IR network to estimate the channel. Literature [21] regards the channel matrix as a two-dimensional image and proposes a channel estimation method based on deep learning for the first time. The idea of image denoising was used to restore the channel matrix and achieved excellent performance. In [22], a deep learning method based on CNN in the high-mobility C-V2X scene is studied for channel estimation. The training data generated by the vehicle channel model with TDL power delay profile is simulated, and then the model is obtained through training complete channel matrix prediction.

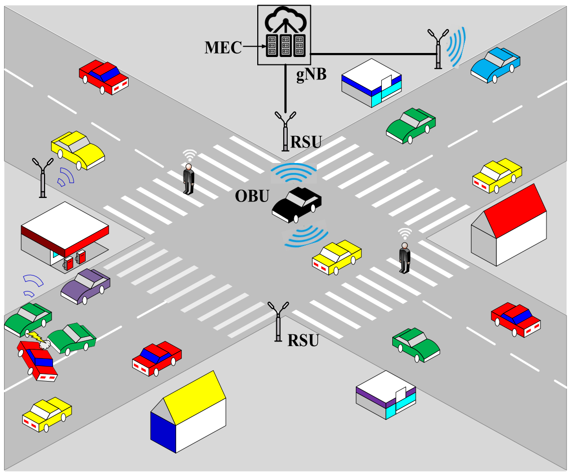


Fig. 1. Schematic diagram of wireless communication in NR-V2X.

III. System Model

A. Intelligent Channel Estimation Framework Based on MEC

The NR-V2X communication scenario is shown in Fig. 1, including V2I, vehicle-to-vehicle (V2V), and vehicle-to-pedestrian (V2P). This paper mainly considers the V2I scenario, completing the communication between the OBU and RSU. gNB is set up within a certain coverage of the road, and the MEC server is deployed at gNB. MEC is used to store the historical V2I channel data, model training, and channel estimation.

The specific steps of the MEC-based intelligent channel estimation framework are as follows.

(1) The MEC server deployed in the gNB stores CFR data, and combines the historical CFR for deep learning training to obtain a channel estimation model. See Section IV for details.

(2) MEC sends the trained prediction model to RSU.

(3) For OBUs which enter the coverage of the gNB, the gNB sends the prediction model trained by the MEC to the OBU.

(4) When the OBU communicates with the RSU, it can use the model trained by MEC to complete the channel estimation.

(5) The OBU and RSU can feedback the estimated CFR of the correct received signal to the gNB-side MEC server to enrich the CFR data set.

B. Signal Processing of CP-OFDM System

The NR-V2X physical layer transmission system is based on CP-OFDM transmission mode [23]. The signal processing process at the signal transmitting end mainly includes channel coding, modulation, pilot inserting, carrier mapping, inverse fast Fourier transform (IFFT), cyclic prefix (CP) insertion, digital to analog conversion, etc., while the receiver is the reverse process of signal processing at the transmitter, and finally recovers the transmitted signal.

According to CP-OFDM transmission frame structure, a transmission frame contains T transmission symbols, and the total number of information subcarriers is M. Note that the frequency-domain symbols of the transmitted data is $\boldsymbol{X}_t^d = \left[X_t^d(0), \ldots, X_t^d(M-1)\right], t = 1, 2, \ldots, T$. In order to complete the N-point IFFT and modulate the frequency-domain symbols to each subcarrier, it is necessary to carry out the zero filling operation for the current frequency-domain symbols, carry out the zero filling operation for the $(N-M)$ subcarrier positions not used as information transmission to obtain the complete frequency-domain transmission symbols $\boldsymbol{X}_t = [X_t(0), \ldots, X_t(N-1)]$ and then complete the IFFT operation to obtain the time-domain transmission symbols $\boldsymbol{x}_t = [x_t(0), \cdots, x_t(N-1)]$. Finally, CP is added and digital to analog conversion is carried out, and then it can be sent to the wireless transmission channel.

After V2X wireless channel is successfully received at the receiver, analog-to-digital conversion and CP removal operations are required, and the frequency-domain receiving symbol $\boldsymbol{Y}_t = [Y_t(0), \ldots, Y_t(N-1)]$ is obtained after FFT transformation. The frequency-domain transmission model of CP-OFDM system can be described as

$$\boldsymbol{Y}_t = \boldsymbol{X}_t \boldsymbol{H}_t + \boldsymbol{W}_t \tag{1}$$

where $\boldsymbol{W}_t$ is the channel noise vector, the covariance matrix can be expressed as $\boldsymbol{Q}_w = \sigma_w^2 \boldsymbol{I}_N$, σ_w^2 is the noise variance. $\boldsymbol{I}_N$ is the N-by-N identity matrix, and $\boldsymbol{H}_t$ represents the CFR matrix of the channel to be estimated.

V2I communication has the characteristics of high-speed movement and limited moving area. Therefore, the scattering characteristics of the V2I communication channel are fast time-varying, with complex and time-varying Doppler characteristics, which also brings faster and deeper multipath fading.

In this paper, the Tapped Delay Line model (TDL) [24] is used as the V2I channel model, which can be expressed as

$$h(t,\tau) = \sum_{l=0}^{L-1} h_l(t) \cdot \delta(\tau - \tau_l) \cdot \exp\left(j2\pi f_{d,l} t\right) \tag{2}$$

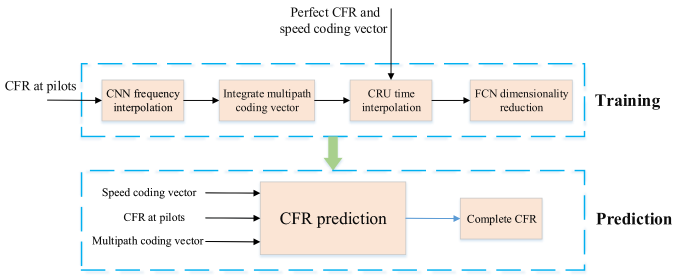


Fig. 2. CNN-CRU processing flow.

where L is the total number of channel taps, $h_l(t)$ is the complex gain of the channel at t-th time of the l-th tap, τ_l is the delay of the l-th tap, and $f_{d,l}$ is the maximum Doppler shift of the l-th tap.

IV. CNN-CRU Channel Estimation

Aiming at the problem of limited performance of traditional channel estimation methods in the V2I environment, combined with MEC, this paper proposes a CNN-CRU channel estimation method that takes advantage of the storage and computing power of the MEC, and places the training of the CNN-CRU algorithm on the MEC server, which improves the computation efficiency of the algorithm. CNN-CRU algorithm uses a one-dimensional convolutional neural network (1D CNN) to complete the frequency-domain interpolation and CRU for time-domain state prediction. Besides, to enable the proposed algorithm to track changes in the V2I environment, we introduce an additional velocity coding vector and multipath coding vector. The proposed algorithm can accurately train for different environments through these two vectors to achieve the purpose of tracking different channel environments.

A. CNN-CRU Channel Estimation Structure

The structure of CNN-CRU channel estimation is divided into two stages: training and prediction, and its structure is shown in Fig. 2. Training stage: (1) Input the acquired channel pilot data into CNN to achieve frequency interpolation. (2) Integrate multipath coding vector into the channel matrix through a fully connected network (FCN). (3) CRU network for time-domain state prediction, In the CRU network, it is necessary to integrate the velocity coding vector information and perfect channel data to train the channel estimation model. (4) Since the CRU network includes the forward and reverse parts, the output dimension is twice the input dimension, and FCN is needed to complete the dimension reduction. Prediction stage: Obtain the CNN-CRU channel estimation model through deep learning training, input the to-be-predicted CFR at pilot symbols, velocity coding vector, and multipath coding vector into the model, and output the channel matrix estimated by CNN-CRU.

Firstly, we need to train CNN-CRU. The purpose of training is to adjust the parameters in CNN-CRU. All we need to do is to collect enough channel data samples, and then use these channel data samples to train the CNN-CRU algorithm, and solve the optimal parameters under the minimum mean

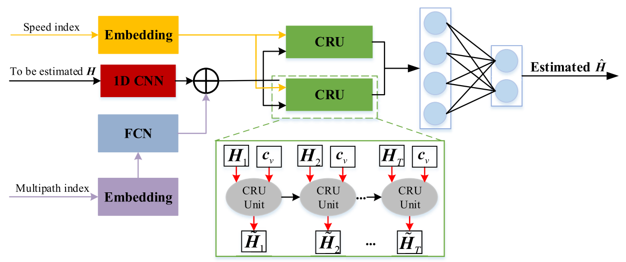


Fig. 3. CNN-CRU network structure.

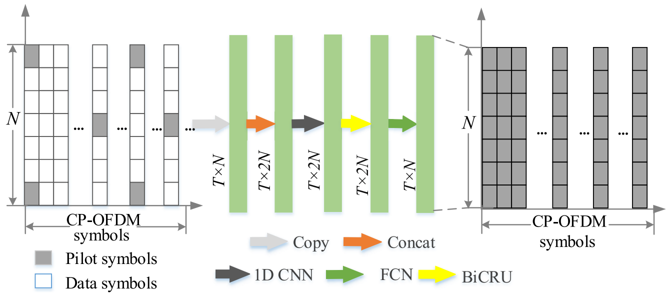


Fig. 4. Data flow in CNN-CRU network.

square error of the algorithm through training iterations, it can learn the feature of channel changes. For prediction, at this time, the parameters in the CNN-CRU algorithm have been trained, so the trained CNN-CRU is directly used to track channel changes to complete online channel estimation. The CNN-CRU is mainly composed of a 1D CNN and a bidirectional CRU (BiCRU), and its structural diagram is shown in Fig. 3.

B. Data Flow in CNN-CRU

Input data passes through the CNN-CRU network that generates the predicted CFR, and the data flow is shown in Fig. 4. In what follows, we describe the input data preprocessing, conditional coding vector acquisition, frequency-domain interpolation, time-domain state prediction, and data dimensionality reduction in detail, respectively.

1) Input Data Preprocessing: The purpose of channel estimation is to estimate the CFR matrix $\boldsymbol{H}$ at the receiver. For the pilot-assisted channel estimation method, the purpose is to predict the channel response at the data symbol through the channel response at the pilot symbol. For the CNN-CRU channel estimation algorithm, the input of the model is a subframe-sized CFR matrix, the channel response at the pilot symbol position in the matrix is initialized by the LS method, and the value of the data symbols is set to 0. The input data representation is

$$\boldsymbol{H} = [\boldsymbol{H}_1, \cdots, \boldsymbol{H}_t, \cdots, \boldsymbol{H}_T] \tag{3}$$

where $\boldsymbol{H}_t = \left[\boldsymbol{H}_{t,1}, \boldsymbol{H}_{t,2}, \ldots, \boldsymbol{H}_{t,N}\right]^{\mathrm{T}} \in \mathbb{C}^N$, the position of the data symbols $\boldsymbol{H}_{t,n} = 0$. Since the channel data is a complex signal, it is necessary to preprocess the data before entering the proposed CNN-CRU. By extracting the real part and the imaginary part of the input data and concatenating the real part and the imaginary part, the input data becomes $\boldsymbol{H}' = \mathbb{R}^{T\times 2N}$, $\boldsymbol{H}'_t \in \mathbb{R}^{2N}$.

2) Conditional Encoding Vector Acquisition: In order to adapt the CNN-CRU algorithm to a specific mobile environment, we introduce an additional multipath encoding vector and velocity encoding vector as additional inputs in frequency-domain interpolation and time-domain state prediction, respectively, and track the change of the channel through these two encoding vectors. In this paper, these two encoding vectors are obtained through the Embedding network in deep learning. The Embedding layer can convert the input index value into a vector of a certain dimension. More specifically, the Embedding layer is essentially a fully connected network, but its emphasis is different from that of the fully connected network. The output of the Embedding layer is equivalent to the weight in the fully connected network, that is, the Embedding layer obtains the weight in the network. In this paper, suppose that we need to simulate different multipath conditions and different velocity numbers are N_d and N_v, the dimension of each coding vector is $2N$, so the Embedding matrix can be expressed as

$$\boldsymbol{E}_d = \left[\boldsymbol{e}_1^d, \boldsymbol{e}_2^d, \ldots, \boldsymbol{e}_{N_d}^d\right]^{\mathrm{T}} \tag{4}$$

$$\boldsymbol{E}_v = \left[\boldsymbol{e}_1^v, \boldsymbol{e}_2^v, \ldots, \boldsymbol{e}_{N_v}^v\right]^{\mathrm{T}} \tag{5}$$

where $\boldsymbol{E}_d \in \mathbb{R}^{N_d\times 2N}$, $\boldsymbol{E}_v \in \mathbb{R}^{N_v\times 2N}$. Before training, the data in the embedding matrix is initialized randomly. During training, we only need to give the embedding layer an index of each velocity v_{index} and multipath d_{index}. The embedding layer obtains the specified row vector in the embedding matrix as the encoding vector according to the given index, its expression is

$$\boldsymbol{c}_d = \text{select\_row}\left(\boldsymbol{E}_d, d_{\text{index}}\right) \tag{6}$$

$$\boldsymbol{c}_v = \text{select\_row}\left(\boldsymbol{E}_v, v_{\text{index}}\right) \tag{7}$$

where $\boldsymbol{c}_d$ and $\boldsymbol{c}_v$ are the coding vectors given the multipath and velocity conditions, respectively.

3) 1D CNN Frequency-Domain Interpolation: In this paper, compared with the use of DNN for frequency-domain interpolation in [12], we use 1D CNN for frequency-domain interpolation. The reason is channel response at a certain data location has only a strong correlation with the channel response of the pilot symbol near that symbol, while the channel response farther away from the data symbol has a weaker correlation. However, DNN-based frequency-domain interpolation estimates the channel response at each data symbol utilizes all pilot symbols, which increases the redundancy of the model. Moreover, the interpolation of each CP-OFDM symbol is independent, so the characteristic of time-domain change cannot be used for frequency-domain interpolation. Therefore, this paper uses 1D CNN for frequency-domain interpolation. The output of each neuron in 1D CNN only uses the data of convolution kernel size in each channel, and it can make full use of the relationship between the time-domain characteristics and frequency-domain characteristics of the channel.

The structure of 1D CNN network is shown in Fig. 5. The 1D CNN is mainly composed of an input layer, some hidden layer, and an output layer. For the k-th layer, the

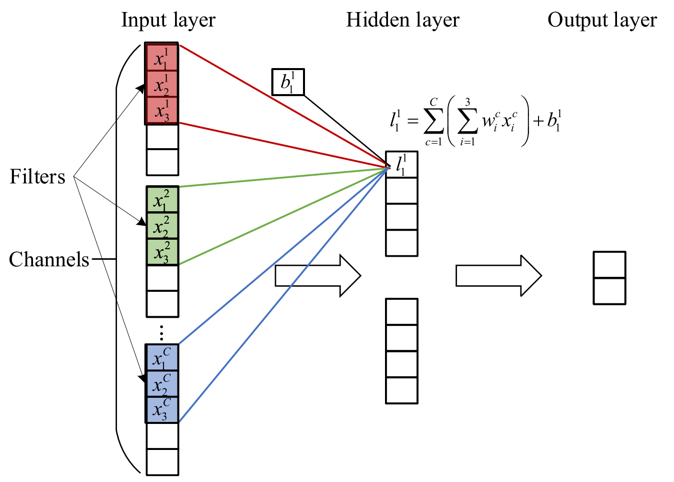


Fig. 5. Structure of 1D CNN.

output is a locally weighted sum of the output elements of the $k-1$ layer. Unlike the two-dimensional CNN, the input transformation of the 1D CNN is a one-dimensional form. Let $\boldsymbol{L}_{k-1}=\left[\boldsymbol{l}_{k-1}^1,\boldsymbol{l}_{k-1}^2,\ldots,\boldsymbol{l}_{k-1}^{C_{k-1}}\right]\in\mathbb{R}^{H_{k-1}\times C_{k-1}}$ be the output of the k-1 layer of the 1D CNN, where $\boldsymbol{l}_{k-1}^c\in\mathbb{R}^{H_{k-1}}$ represents the data vector on the c-th channel. So, the ith output on the $c-$th channel at the k-th layer is

$$l_{k,i}^c=\sum_{t=1}^{C_{k-1}}\left\{\sum_{u=1}^{U}\left(\mathcal{W}_k^{(c,t)}(u)\,\boldsymbol{l}_{k-1}^t(u+(i-1)S)\right)\right\}+\boldsymbol{b}_k^{(c)},\quad c=1,2,\ldots,C_k \tag{8}$$

where $\mathcal{W}_k\in\mathbb{R}^{C_{k-1}\times C_k\times U}$ and $\boldsymbol{b}_k\in\mathbb{R}^{C_k}$ are denote the weight tensor and offset vector of the k-th layer, S is the step size of the convolution filter sliding on the input data, and U is the width of the convolution filter. $\boldsymbol{C}_k$ represents the number of output channels of the k-th layer. Therefore, we can get the output vector on the c-th channel for the 1D CNN of the k-th layer as

$$\boldsymbol{l}_k^c=\left[\boldsymbol{l}_{k,1}^c\ \boldsymbol{l}_{k,2}^c\cdots\boldsymbol{l}_{k,\frac{H_{k-1}-U+1}{S}}^c\right]^{\mathrm{T}} \tag{9}$$

Therefore, the output data of the k-th layer is

$$\boldsymbol{L}_k=\left[\boldsymbol{l}_k^1,\boldsymbol{l}_k^2,\ldots,\boldsymbol{l}_k^{C_k}\right]\in\mathbb{R}^{\frac{H_{k-1}-U+1}{S}\times C_k} \tag{10}$$

Similarly, in order to make the data non-linear transformation, the activation function needs to be connected behind each layer of 1D CNN network, so the transformation formula of each layer of 1D CNN can be abbreviated as follows

$$\boldsymbol{L}_k=f\left(\mathcal{W}_k*\boldsymbol{L}_{k-1}+\boldsymbol{b}_k\right) \tag{11}$$

In a network based on 1D CNN frequency-domain interpolation, the width of the convolution kernel used in this paper is $U=9$, the step size is $S=1$, the number of input and output channels of 1D CNN is T. So, the channel response on the nth subcarrier at the t-th CP-OFDM symbol is

$$\boldsymbol{H}_{t,n}''=\sum_{i=1}^{T}\left(\sum_{u=1}^{9}\mathcal{W}^{(t,i)}(u)\,\boldsymbol{H}'(u+n-1,t)\right)+(\boldsymbol{W}_d(n))^{\mathrm{T}}\boldsymbol{c}_d,\quad t=1,\ldots,T;\ n=1,\ldots,N \tag{12}$$

where $\boldsymbol{W}\in\mathbb{R}^{2N\times 2N}$ is the weight matrix of the multipath coding vector. Because the transformation process of the convolutional neural network will reduce the dimensionality of the original data, we will fill the input with multiple zeros to ensure that the output has the same dimensions as the original input. As can be seen from equation (12), on the one hand, when estimating the CFR on the nth subcarrier at the t-th CP-OFDM symbol, not only the CFR at the pilot symbols near the nth subcarrier at the CP-OFDM symbol is used, but also the CFR at the pilot symbol near the n-th subcarrier at other CP-OFDM symbols is used. So, the frequency-domain interpolation based on 1D CNN jointly uses the time-frequency domain change relationship of the channel, while tracking the channel time-frequency-domain change. On the other hand, the channel response at each data symbol is determined jointly by the channel frequency-domain response at the pilot symbol and the multipath coding vector. The multipath vector tracks the channel changes due to changes in the multipath conditions. The output after interpolation for each CP-OFDM symbol can be expressed as

$$\boldsymbol{H}_t''=f_{\mathrm{T}}\left(\mathcal{W}^{(t)}*\boldsymbol{H}'+\boldsymbol{W}_d\boldsymbol{c}_d\right),t=1,2,\ldots,T \tag{13}$$

4) CRU Time Domain State Prediction: In the time-domain state prediction, in order to adapt to the specific mobile environment, we designed the CRU network, which encodes different speeds into a vector and inputs it into the time-domain state network, so the network at this time has two inputs: specific velocity coding vector and channel data after frequency-domain interpolation. The CRU network is a new recurrent neural network, this network is essentially an RNN network. We changed the inside of each recurrent unit and used the form of the gate in the LSTM network to control the input and output of the network using two gates. The structure of CRU is shown in Fig. 6.

Similarly, the CRU network is mainly composed of several CRU units. Each unit has two gate units: input gate and update gate. The input gate and update gate control the inflow of data in various parts of the network. So, we can get the mathematical transformation of the time-domain state prediction for the t-th CP-OFDM symbol as

$$\boldsymbol{i}_t=\sigma\left(\boldsymbol{U}_i\boldsymbol{H}_t''+\boldsymbol{W}_i\boldsymbol{H}_{t-1}'''\right) \tag{14}$$

$$\boldsymbol{u}_t=\sigma\left(\boldsymbol{U}_u\boldsymbol{H}_t''+\boldsymbol{W}_u\boldsymbol{H}_{t-1}'''\right) \tag{15}$$

$$\boldsymbol{c}_t=\tanh\left(\boldsymbol{U}_c\boldsymbol{H}_t''+\boldsymbol{W}_c\boldsymbol{H}_{t-1}'''+\boldsymbol{i}_t\odot\boldsymbol{V}_c\boldsymbol{c}_v\right) \tag{16}$$

$$\boldsymbol{H}_t'''=\boldsymbol{u}_t\odot\boldsymbol{c}_t+(1-\boldsymbol{u}_t)\odot\boldsymbol{H}_t'' \tag{17}$$

where $\boldsymbol{i}_t$ and $\boldsymbol{u}_t$ are the input gate and update gate of the CRU unit, respectively, and $\boldsymbol{c}_v$ is the coding vector at a specific velocity. It can be seen from equation (16) that the input gate can introduce the velocity coding vector. In this way, only part of the information in the velocity coding vector can be controlled by the door to flow into the memory unit. The advantage is that the information in the memory unit will not be completely controlled by the velocity coding vector, but by the integration of various information. Finally, we directly control the memory unit and the input through an update gate to obtain the final output. Its transformation is shown in equation (17), that is, part of the final output comes from

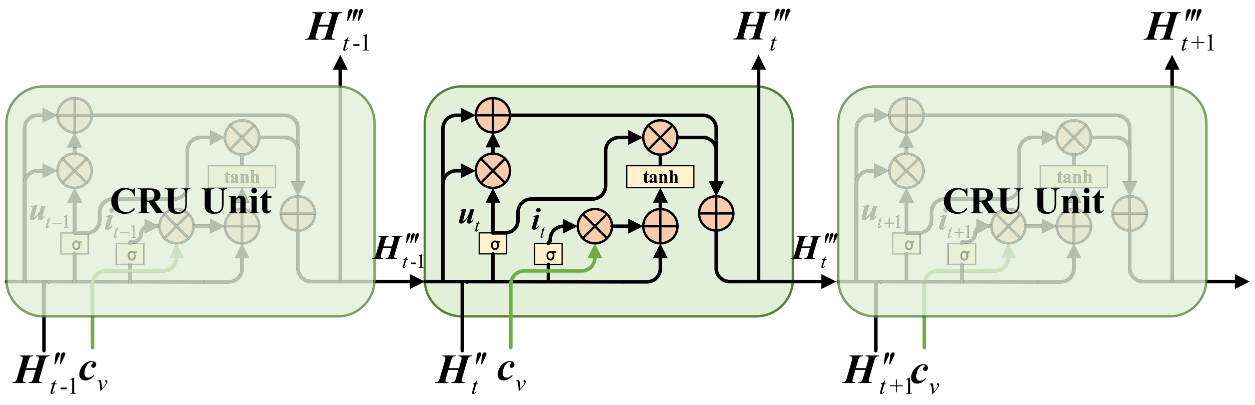


Fig. 6. CRU structure.

the memory unit, and the other part comes directly from the input. The inflow degree of each part is determined by update gate $\boldsymbol{u}_t$. For example, under low-speed conditions, the time-domain of channel response changes slowly, so the value $\boldsymbol{u}_t$ is small, and the final output is mainly determined by the input. On the contrary, for high-speed mobile conditions, the time-domain of channel response presents a fast time-varying characteristic, the $\boldsymbol{u}_t$ is larger, so the final output is determined by the memory unit containing the influence of Doppler. This paper uses the BiCRU network for time-domain state prediction. The BiCRU network is a combination of two CRU networks, one of which performs forward prediction and the other performs backward prediction. On the one hand, using the BiCRU network can make full use of all information before and after the current time for the prediction of the current time; on the other hand, bidirectional prediction can suppress the error propagation caused by the unidirectional prediction. For time-domain state prediction, each CRU network has T CRU units. The outputs of the two CRUs at time t of the BiCRU network are

$$\boldsymbol{H}_{\mathrm{FW},t} = \mathrm{CRU}\left(\boldsymbol{H}_{\mathrm{FW},t-1}, \boldsymbol{H}''_t, \boldsymbol{c}_v, \Theta_{\mathrm{FW}}\right) \tag{18}$$

$$\boldsymbol{H}_{\mathrm{BW},t} = \mathrm{CRU}\left(\boldsymbol{H}_{\mathrm{BW},t-1}, \boldsymbol{H}''_{T-t+1}, \boldsymbol{c}_v, \Theta_{\mathrm{BW}}\right) \tag{19}$$

where Θ_{FW} and Θ_{BW} are all parameters in the forward CRU and backward CRU network, respectively. The output of the BiCRU network at time t is

$$\boldsymbol{H}'''_t = \left[\boldsymbol{H}_{\mathrm{FW},t}; \boldsymbol{H}_{\mathrm{BW},t}\right] \in \mathbb{R}^{4N\times 1} \tag{20}$$

5) FCN Dimension Reduction: Through 1D CNN frequency-domain interpolation and BiCRU time-domain state prediction, we could obtain the complete channel data of the resource grid. Due to the use of bidirectional prediction, the output data is increased to twice the original channel data, so we need to correct output data for dimensionality reduction. We use FCN for dimensionality reduction. The first advantage is that the output after FCN dimensionality reduction is a weighted combination of inputs, which can make full use of the input information to ensure that the original data information is as little as possible. Another advantage is that the FCN network can select better data from the forward prediction and backward prediction data as the output to compensate for the non-stationarity of the channel estimation and improve the accuracy of the final estimation. The final channel estimation result of the t-th CP-OFDM symbol is

$$\hat{\boldsymbol{H}}_t = f_{\mathrm{T}}\left(\boldsymbol{W}_{\mathrm{DR}}\boldsymbol{H}'''_t\right) \tag{21}$$

where $\boldsymbol{W}_{\mathrm{DR}}$ represents the weight parameter to be trained in FCN. Then, we combine the real and imaginary parts of the output to get the complex channel response.

C. Model Training

There are many well-known channel models proposed by scholars or standards, and these channel models can describe channels well in the field of statistics. We can get the training data from the simulation by these channel models. In this paper, we generate training data based on the TDL channel model. For offline training, the acquisition of training data is to collect channel data supervision data of all resource grids in each frame through system simulation, and then set the channel response of the data to 0 according to certain pilot insertion rules and the corresponding velocity and multiple of each sample. The index value of the path is also used as training data. We train CNN-CRU in order to minimize the error between the output of the learning network and the label data so update all the parameters in the network. The parameters in the CNN-CRU network can be divided into four categories: frequency-domain interpolation coefficients, conditional code embedding matrix, time-domain correlation coefficients, and dimensionality reduction coefficients. After training, the CNN-CRU network can learn the optimal solutions of these coefficients, so that the parameters in the CNN-CRU network can better track the channel changes. For online prediction, we can use the trained CNN-CRU network directly for channel estimation due to the various parameters in CNN-CRU that have been trained.

We use an end-to-end manner to obtain all the weights of the network in order to train the CNN-CRU network. Supposed that the transformation formula and all parameters of the entire CNN-CRU network are $f_{\mathrm{est}}(\cdot)$ and Θ_{est}, respectively, the CFR estimated by the CNN-CRU network can be

TABLE I
COMPUTATIONAL COMPLEXITY

Algorithms	Number of multiplication operations	Complexity
CNN-CRU (offline)	$[(2MT+2N_d)N]+[64N^2+(2N_v+12)N]$	$O(N^2)$
DNN-LSTM (offline)	$[4N^2]+[72N^2+12)N]$	$O(N^2)$
LMMSE with Linear Interpolation	$[3N^3+N^2]+[N]$	$O(N^3)$
LS with Linear Interpolation	$[N]+[N]$	$O(N)$
STA	$[N]+[2N]$	$O(N)$
CDP	$[N]+[4N]$	$O(N)$

expressed as $\hat{\boldsymbol{H}} = f_{\text{est}}(\boldsymbol{H}, d_{\text{index}}, v_{\text{index}}, \Theta_{\text{est}})$. The adaptive moment estimation (ADAM) algorithm and the stochastic gradient descent (SGD) algorithm are used for updating the parameter set of the DNN-LSTM network. Different from the traditional gradient descent algorithm with a fixed learning rate, the learning rate of ADAM algorithm can be updated by training adaptively. The loss function of the network is MSE. Therefore, the predicted loss of our model is

$$L(\Theta_{est}) = \frac{1}{M_s}\sum_{i=1}^{M_s}\left(f_{est}\left(\boldsymbol{H}_i, d_{\text{index}}^i, v_{\text{index}}^i, \Theta_{\text{est}}\right) - \boldsymbol{H}_i^*\right)^2 \tag{22}$$

where $\boldsymbol{H}^*$ is the supervision data and M_s is the total number of samples in the training sample set. In this paper, the training set, check set, and test set we used are 300,000, 40000 and 20000 respectively. Each dataset contains the channel data to be estimated, the velocity index value v_{index} and multipath index value d_{index} corresponding to each data sample. Among them, the moving speed parameter set of channel data contained in each dataset is $\left[50\text{km/h}, 100\text{km/h}, 150\text{km/h}, 200\text{km/h}, 250\text{km/h}, 300\text{km/h}\right]$, i.e., $v_{\text{index}} = \left[0, 1, \ldots, 5\right]$, and the multipath number set is $[3, 6, 9, 12, 15, 18]$, i.e., $v_{\text{index}} = \left[0, 1, \ldots, 5\right]$. We use MATLAB platform to generate training data, and the training device is GeForce GTX 1060 graphics card, Intel Xeon® E3-1231 V3 processor. The total training epoch is 200, in which the first 100 epochs are trained with ADAM algorithm, and the last 100 epochs are trained with SGD algorithm. The learning rate of SGD algorithm is 0.01, and the training time is about 130 minutes.

V. Complexity Analysis

We compared the computational complexity of the four traditional channel estimation algorithms of LS, LMMSE, STA, and CDP, as well as the DNN-LSTM and CNN-CRU mentioned in this paper. This paper compares the multiplication times and computational complexity required by the algorithm to estimate two CP-OFDM symbols. One CP-OFDM symbol contains pilots, and the other CP-OFDM symbol does not include pilots. The results are shown in Table I.

In Table I, the computational complexity of LS, STA, and CDP is in the same order of magnitude. LMMSE has the highest computational complexity due to matrix inversion. The complexity of the online estimation of CNN-CRU is comparable to the DNN-LSTM algorithm, which is lower than LMMSE with linear interpolation algorithm, and slightly higher than LS with linear interpolation algorithm. For the CNN-CRU algorithm, the algorithm is just some matrix multiplication and addition operations, so its complexity is lower than the LMMSE algorithm. Moreover, parallel processing can help us to save time in neural network training. For the online estimation stage, the trained network is directly used for channel estimation, so it has low complexity. Although the computational complexity of the CNN-CRU algorithm is comparable to the DNN-LSTM algorithm, it can be seen from the required number of multiplications that the complexity of the CNN-CRU algorithm is lower than that of the DNN-LSTM algorithm. In general, the complexity of the CNN-CRU algorithm is between the LS and LMMSE algorithms, and the time complexity is $O\left(n^2\right)$. In terms of theoretical research and engineering implementation, it is an effective channel estimation algorithm.

TABLE II
PARAMETERS OF SIMULATION SYSTEM

Parameter	Value
Carrier Frequency	5.9GHz
Bandwidth	10MHz
Number of Subcarriers	1024
Subcarrier spacing	15KHz
CP length	$4.7us$
Length of symbol	$66.7us$
Modulation	QPSK

VI. Simulation Analysis

The simulation parameters and values of the V2I point-to-point CP-OFDM transmission system are presented. The frame structure of the physical layer of the transmission system is set according to the NR-V2X protocol [25]. The system parameters are shown in Table II and the channel parameters are shown in Table III.

Fig. 7 and Fig. 8 show the normalized mean square error (NMSE) performance and BER performance of each algorithm with 6 taps, and the velocity is 100km/h. From the figure, we can see that LS with linear interpolation, CDP, and the STA algorithms have poor NMSE performance because these algorithms use LS to estimate the channel responses at the pilot. However, the LS algorithm cannot eliminate the influence of noise, making the channel response at the pilot inaccurate, which in turn affects the accuracy of the channel response at the data symbol. LMMSE has better NMSE performance, because the algorithm can use channel

TABLE III
CHANNEL PARAMETERS

Parameter	Value
Channel mode	TDL
Channel taps	$[3, 6, 9, 12, 15, 18]$
Maximum tap delay	$[400, 800, 1200, 1600, 2000, 2400]ns$
Maximum relative velocity	$[50, 100, 150, 200, 250, 300]$km/h
Maximum Doppler shift	$[273, 546, 819, 1093, 1366, 1638]$Hz

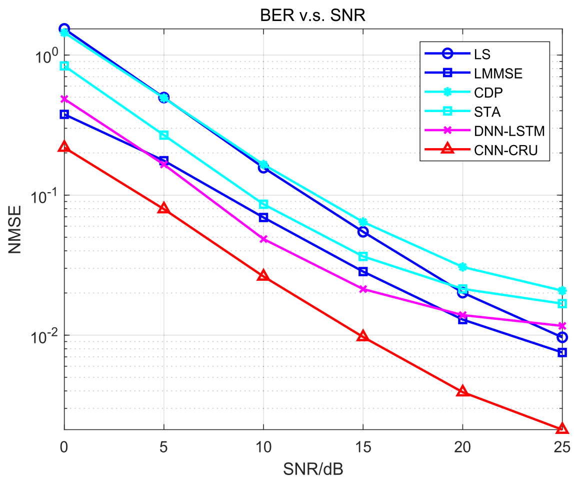


Fig. 7. NMSE versus SNR at 6 taps and 100km/h.

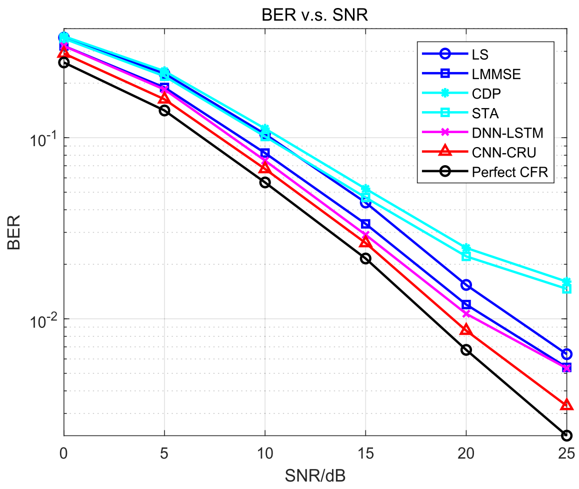


Fig. 8. BER versus SNR at 6 taps and 100km/h.

statistics and noise information to improve the accuracy of channel response estimation at the pilot, and then improve the accuracy of channel response estimation at the data symbol. The CNN-CRU algorithm has the best NMSE performance. Compared to the LMMSE algorithm, the SNR gain of the CNN-CRU algorithm improves 5dB averagely, indicating that the CNN-CRU algorithm can not only effectively reduce the influence of pilot noise but also track the change of channel.

For SNR higher than 15dB, the STA and CDP algorithm have the worst BER performance due to decision feedback is prone to error propagation, resulting in poor

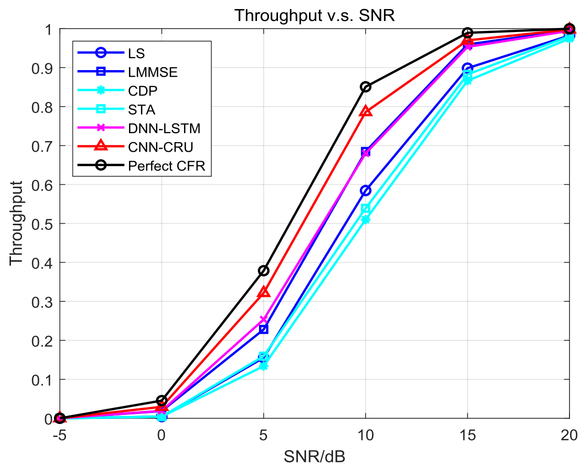


Fig. 9. Throughput versus SNR at 6 taps and 100km/h.

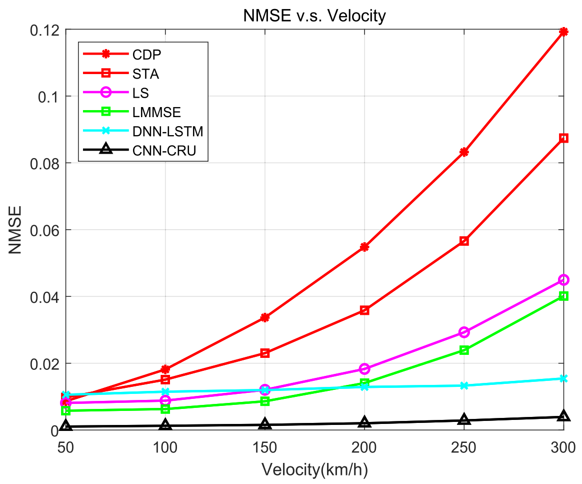


Fig. 10. NMSE versus Velocity at 6 taps.

BER performance. Notice that the BER performance of the DNN-LSTM algorithm is comparable to that of the LMMSE, which indicates that the neural network-based method has excellent denoising performance and robustness. In addition, the CNN-CRU algorithm has reached the upper limit of the estimated performance, indicating that it can perfectly track the time-frequency domain changes of the channel through the multipath coding vector and the velocity coding vector. At this time, the BER performance of CNN-CRU is mainly limited by noise.

Fig. 9 shows the throughput curve under different SNR conditions, and the throughput is the ratio of correctly received packets to transmitted packets. Because different channel estimation algorithms will affect the receiving performance and the BER performance, the throughput will also be affected. It can be seen that the proposed algorithm still has a great advantage in throughput. The throughput of 15dB SNR is close to 1, and its performance is closest to the throughput of ideal CFR.

Fig. 10 and Fig. 11 show the NMSE and BER performance comparison of each algorithm with the change of velocity when the signal-to-noise ratio is 25dB with 6 taps.

From the figures, we can see that the NMSE performance and BER performance of each algorithm show a downward

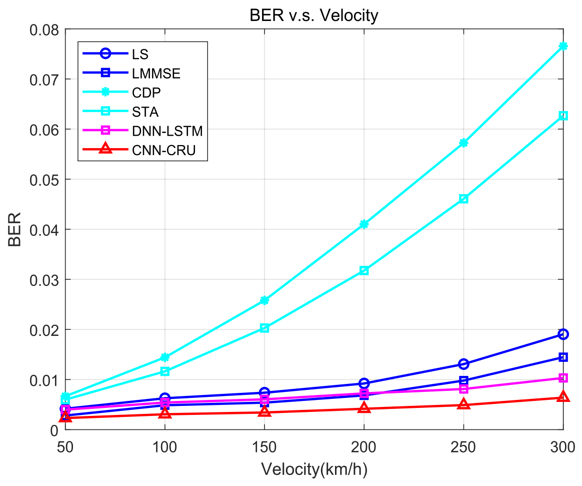


Fig. 11. BER versus Velocity at 6 taps.

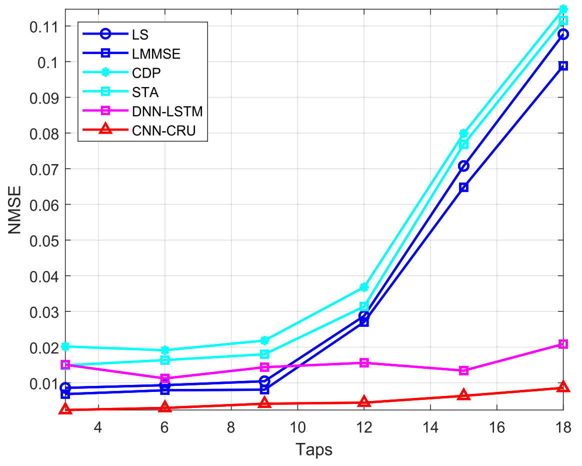


Fig. 12. NMSE versus Taps at 100 km/h.

trend as the velocity increases. When the velocity of the terminal increases from 50km/h to 300km/h, the NMSE performance of the CDP algorithm decreases the most, with a decrease of 0.101, and STA algorithm is also not robust. Both of them are decision feedback algorithms, which show that decision feedback is less robust to velocity. Compared with decision feedback estimation methods, traditional interpolation algorithms can better adapt to changes in speed. The NMSE performance of the CNN-CRU algorithm and the DNN-LSTM algorithm mentioned in this paper tends to be stable with the change of speed, indicating that the deep learning-based algorithms can well track the varying channel. The CNN-CRU algorithm further improves the robustness of the algorithm by introducing a velocity encoding vector. At this time, the NMSE of the CNN-CRU algorithm drops from 0.001 to about 0.0038, and its decrease is only 0.0027, which is about 0.0285 of the CDP algorithm's decrease.

Fig. 12 and Fig. 13 show the comparison of NMSE and BER performance of each algorithm with taps changes when the signal-to-noise ratio is 25dB and velocity is 100km/h.

As we can see from the figures, the NMSE performance and BER performance of various algorithms show a downward trend as the velocity increases. When the taps increase

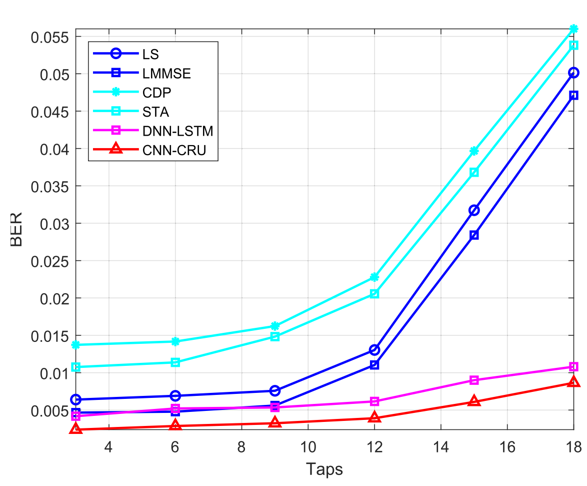


Fig. 13. BER versus Taps at 100 km/h.

from 9 to 18, the NMSE performance and BER performance of the STA, CDP, LS, and LMMSE algorithm degrade rapidly, indicating that the traditional algorithm is less robust to multipath. In contrast, the CNN-CRU algorithm's downward trend is relatively stable, indicating that the proposed algorithm can well adapt to the changing taps. However, the NMSE performance of the DNN-LSTM algorithm jumps with the change of the taps, which means that the DNN-LSTM algorithm cannot effectively track the changing taps. Therefore, introducing the multipath encoding vectors makes the CNN-CRU algorithm more robust to multipath.

VII. Conclusion

Aiming at the problem of 5G NR-V2I communication, this paper uses MEC's data storage and processing capabilities, combined with deep learning-based channel estimation, to design a MEC-based intelligent channel estimation framework. Moreover, for the inadequacy of the traditional algorithm estimation performance in the V2I, a channel estimation network based on CNN-CRU is proposed. CNN is used to complete frequency-domain interpolation and CRU to perform time state prediction, which improves the accuracy of channel estimation. In addition, we introduce velocity coding vector and multipath coding vector as additional inputs to control the output of the entire channel estimation network to achieve precise training. Analysis and system simulation show that the CNN-CRU algorithm can effectively reduce the computational complexity while ensuring the accuracy and robustness of channel estimation.

References

[1] R. Molina-Masegosa and J. Gozalvez, "LTE-V for sidelink 5G V2X vehicular communications: A new 5G technology for short-range vehicle-to-everything communications," *IEEE Veh. Technol. Mag.*, vol. 12, no. 4, pp. 30–39, Dec. 2017.
[2] P. Wang, B. Di, H. Zhang, K. Bian, and L. Song, "Cellular V2X communications in unlicensed spectrum: Harmonious coexistence with VANET in 5G systems," *IEEE Trans. Wireless Commun.*, vol. 17, no. 8, pp. 5212–5224, Aug. 2018.
[3] X. Cheng, C. Chen, W. Zhang, and Y. Yang, "5G-enabled cooperative intelligent vehicular (5GenCIV) framework: When Benz meets Marconi," *IEEE Intell. Syst.*, vol. 32, no. 3, pp. 53–59, May/Jun. 2017.

[4] S. Chen *et al.*, "Vehicle-to-everything (V2X) services supported by LTE-based systems and 5G," *IEEE Commun. Standards Mag.*, vol. 1, no. 2, pp. 70–76, Jun. 2017.

[5] Y. Yang, D. Fei, and S. Dang, "Inter-vehicle cooperation channel estimation for IEEE 802.11p V2I communications," *J. Commun. Netw.*, vol. 19, no. 3, pp. 227–238, Jun. 2017.

[6] K. Zhang, Y. Mao, S. Leng, Y. He, and Y. Zhang, "Mobile-edge computing for vehicular networks: A promising network paradigm with predictive off-loading," *IEEE Veh. Technol. Mag.*, vol. 12, no. 2, pp. 36–44, Jun. 2017.

[7] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 2322–2358, 4th Quart., 2017.

[8] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie, "Mobile edge computing: A survey," *IEEE Internet Things J.*, vol. 5, no. 1, pp. 450–465, Feb. 2018.

[9] Y. Wang, G. Liu, F. Han, H. Qu, and Q. Chen, "Channel estimation and equalization for SS-OOFDM system with high mobility," *IEEE Commun. Lett.*, vol. 23, no. 1, pp. 92–95, Jan. 2019.

[10] X. Lai, Z. Chen, and Y. Zhao, "Basis expansion model for fast time-varying channel estimation in high mobility scenarios," in *Proc. 11th EAI Int. Conf. Commun. Netw. China (ChinaCom)*, Chongqing, China, Sep. 2016.

[11] X. Mao, S. Hashemizadeh, R.-R. Chen, and B. Farhang-Boroujeny, "Soft decision directed dual-layer channel estimation for time-varying MIMO channels," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, San Francisco, CA, USA, Mar. 2017, pp. 1–6.

[12] Y. Liao, Y. Hua, X. Dai, H. Yao, and X. Yang, "ChanEstNet: A deep learning based channel estimation for high-speed scenarios," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Shanghai, China, May 2019, pp. 1–6.

[13] P. Aggarwal, A. Gupta, and V. A. Bohara, "A guard interval assisted OFDM symbol-based channel estimation for rapid time-varying scenarios in IEEE 802.Lip," in *Proc. IEEE 26th Annu. Int. Symp. Pers., Indoor, Mobile Radio Commun. (PIMRC)*, Hong Kong, Aug. 2015, pp. 100–104.

[14] J. A. Fernandez, K. Borries, L. Cheng, B. V. K. V. Kumar, D. D. Stancil, and F. Bai, "Performance of the 802.11p physical layer in vehicle-to-vehicle environments," *IEEE Trans. Veh. Technol.*, vol. 61, no. 1, pp. 3–14, Jan. 2012.

[15] Z. Zhao, M. Wen, C.-X. Wang, X. Cheng, and B. Jiao, "Channel estimation schemes for IEEE 802.11p standard," *IEEE Intel. Transport. Syst. Mag.*, vol. 5, no. 4, pp. 38–49, Oct. 2013.

[16] Z. Zhao, X. Cheng, M. Wen, L. Yang, and B. Jiao, "Constructed data pilot-assisted channel estimators for mobile environments," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 2, pp. 947–957, Apr. 2015.

[17] S. Baek, I. Lee, and C. Song, "A new data pilot-aided channel estimation scheme for fast time-varying channels in IEEE 802.11p systems," *IEEE Trans. Veh. Technol.*, vol. 68, no. 5, pp. 5169–5172, May 2019.

[18] H. Qu, G. Liu, Y. Wang, S. Wen, and Q. Chen, "Time-domain channel estimation for the LTE-V system over high-speed mobile channels," in *Proc. IEEE Int. Symp. Broadband Multimedia Syst. Broadcast. (BMSB)*, Valencia, Spain, Jun. 2018, pp. 1–5.

[19] H. Ye, G. Y. Li, and B.-H. Juang, "Power of deep learning for channel estimation and signal detection in OFDM systems," *IEEE Wireless Commun. Lett.*, vol. 7, no. 1, pp. 114–117, Feb. 2018.

[20] M. Soltani, V. Pourahmadi, A. Mirzaei, and H. Sheikhzadeh, "Deep learning-based channel estimation," *IEEE Commun. Lett.*, vol. 23, no. 4, pp. 652–655, Apr. 2019.

[21] H. He, C.-K. Wen, S. Jin, and G. Y. Li, "Deep learning-based channel estimation for beamspace mmWave massive MIMO systems," *IEEE Wireless Commun. Lett.*, vol. 7, no. 5, pp. 852–855, Oct. 2018.

[22] R. Sattiraju, A. Weinand, and H. D. Schotten, "Channel estimation in C-V2X using deep learning," in *Proc. IEEE Int. Conf. Adv. Netw. Telecommun. Syst. (ANTS)*, Goa, India, Dec. 2019, pp. 1–5.

[23] N. Odabasioglu, M. M. Leblebici, B. Karakaya, and H. Dogan, "Cyclic prefix based time synchronization and comb type channel estimation for SC-FDMA systems over time-varying channels," in *Proc. 9th Int. Conf. Appl. Inf. Commun. Technol. (AICT)*, Oct. 2015, pp. 359–362.

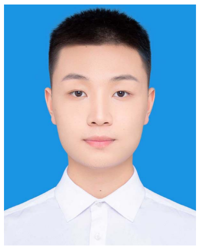
[24] *Study on Channel Model for Frequencies From 0.5 to 100 GHz, v16.1.0*, document TR 38.901, 3GPP, Sophia Antipolis, France, Dec. 2019.

[25] *Study on NR Vehicle-to-Everything (V2X), v16.0.0*, document TR 38.885, 3GPP, Sophia Antipolis, France, Rep. Mar. 2019.

Yong Liao (Member, IEEE) received the Ph.D. degree from Chongqing University, Chongqing, China, in 2014. He is currently a Research Associate with Chongqing University. His research interests include high-speed mobile communications, and 5G and future communications.

Zhirong Cai received the B.S. degree from China West Normal University, Sichuan, China, in 2019. He is currently pursuing the M.S. degree with Chongqing University, Chongqing, China. His current research interests include channel estimation and equalization in OFDM systems.

Guodong Sun received the B.S. and M.S. degrees from Chongqing University, Chongqing, China, in 2018 and 2021, respectively. His research interests include wireless communications channel estimation and channel interpolation algorithms.



Xiaoyi Tian received the B.S. degree from Southwest Minzu University, Sichuan, China, in 2019. He is currently pursuing the M.S. degree with Chongqing University, Chongqing, China. His current research interests include deep learning and its application for channel estimation.

Yuanxiao Hua received the B.S. degree from the North University of China, Taiyuan, China, in 2016, and the M.S. degree from Chongqing University, Chongqing, China, in 2020. His research interests include wireless communications channel estimation and deep learning in communication applications.

Xiaoheng Tan received the B.E. and Ph.D. degrees in electrical engineering from Chongqing University, Chongqing, China, in 1998 and 2003, respectively.

From 2008 to 2009, he was a Visiting Scholar with The University of Queensland, Brisbane, QLD, Australia. He is currently a Professor with the School of Microelectronics and Communication Engineering, Chongqing University. His current research interests include modern communications technologies and systems, communications signal processing, pattern recognition, and machine learning.