entropy

MDPI

Article

A Transformer-Based Channel Estimation Method for OTFS Systems

Teng Sun [1], Jiebiao Lv [2] and Tao Zhou [2,*]

[1] The 54th Research Institute of CETC, Shijiazhuang 050081, China; 15176948810@126.com
[2] School of Electronic and Information Engineering, Beijing Jiaotong University, Beijing 100044, China; 21120105@bjtu.edu.cn
* Correspondence: taozhou@bjtu.edu.cn

Abstract: Orthogonal time frequency space (OTFS) is a novel modulation scheme that enables reliable communication in high-mobility environments. In this paper, we propose a Transformer-based channel estimation method for OTFS systems. Initially, the threshold method is utilized to obtain preliminary channel estimation results. To further enhance the channel estimation, we leverage the inherent temporal correlation between channels, and a new method of channel response prediction is performed. To enhance the accuracy of the preliminary results, we utilize a specialized Transformer neural network designed for processing time series data for refinement. The simulation results demonstrate that our proposed scheme outperforms the threshold method and other deep learning (DL) methods in terms of normalized mean squared error and bit error rate. Additionally, the temporal complexity and spatial complexity of different DL models are compared. The results indicate that our proposed algorithm achieves superior accuracy while maintaining an acceptable computational complexity.

Keywords: OTFS; channel estimation; deep learning; transformer

Citation: Sun, T.; Lv, J.; Zhou, T. A Transformer-Based Channel Estimation Method for OTFS Systems. *Entropy* **2023**, *25*, 1423. https://doi.org/10.3390/e25101423

Academic Editor: Sergio Saponara

Received: 22 August 2023
Revised: 19 September 2023
Accepted: 27 September 2023
Published: 7 October 2023

1. Introduction

Currently, orthogonal frequency division multiplexing (OFDM) is the main communication modulation technique used in long-term evolution (LTE) and fifth-generation (5G) systems because of its excellent spectrum utilization [1]. However, OFDM is highly susceptible to the effects of Doppler spread, as it relies on maintaining orthogonality between subcarriers [2]. Consequently, inter-subcarrier interference becomes prevalent in high-mobility communication scenarios within an OFDM system [3]. To address this issue, a novel modulation method called orthogonal time frequency space (OTFS) was introduced in [4], specifically designed for high-mobility environments. The OTFS method extends the signal into the time–frequency (TF) domain and introduces the delay-Doppler (DD) domain through a two-dimensional transformation, thus modulating the signal in the DD domain [5]. This transformation offers the advantage of converting the double-dispersive channel in the TF domain into a time-invariant channel in the DD domain. As a result, all the symbols within an OTFS frame encounter nearly identical sparse channels, reducing the channel estimation overhead in time-varying channels and eliminating the need for high-density pilots to estimate the channel response [6].

Accurately estimating the channel in the delay-Doppler (DD) domain is crucial for successful signal detection at the receiver. In [7], the authors presented a channel estimation algorithm that relies on a single pilot symbol. To enhance the accuracy of the channel estimation, the authors in [7] incorporated a protection interval around the pilot to mitigate interference between the pilot symbol and data symbols. Within this protection interval, channel estimation was performed. Additionally, a data-aided channel estimation scheme was proposed in [8]. In this particular scheme, the pilot symbols are directly embedded

within the data symbols without the inclusion of a separate protection interval. This approach leads to improved spectrum utilization while maintaining a similar bit error rate (BER) performance compared to the scheme described in [7]. A new pilot pattern was introduced in [9], building upon the single pilot scheme proposed in [7]. The authors optimized the pilot pattern by removing the protection symbols on the right side of the pilot symbol. Since the data symbols on the right side do not influence the pilot in terms of delay domains, eliminating the protection symbols in this region enhances spectral efficiency effectively. Apart from the single pilot channel estimation techniques, another approach was introduced by the authors in [10], which involved a channel estimation method based on compressed channel sensing. This method takes advantage of the sparsity property of the channel in the DD domain and converts the channel estimation task into signal recovery. By doing so, this method achieves more precise channel estimation while reducing the required pilot overhead. A high-performance algorithm with low complexity is presented in [11]. The memory approximate message passing (AMP) detector can exploit the sparsity of the channel matrix and perform only matrix-vector multiplication in each iteration. To alleviate the performance degradation caused by positive reinforcement during the iteration process, the memory AMP detector utilizes all previous information to ensure the orthogonality principle. In [12], the authors propose a scheme for joint channel estimation and data detection in a hybrid reconfigurable intelligent surface-aided millimeter wave OTFS system. The simulation results demonstrate that the proposed method can accurately obtain the channel and the unknown data symbols.

Deep learning (DL) has experienced significant advancements in image and speech recognition, and its application has extended to the field of communication in recent years [13]. In [14], the authors proposed a method that utilizes a deep neural network (DNN) for both channel estimation and signal detection. They trained a neural network model using the received signals and the original transmission data, allowing for online recovery of the transmitted data through the trained model. In [15], the time–frequency response of the fast-fading channel was treated as a two-dimensional image. By leveraging the known values of the pilot, the unknown values of the channel response were reduced, addressing the challenge of estimating the channel in the presence of fast fading. In [16], a multi-layer neural network model based on a recurrent neural network (RNN) was proposed to estimate the DD domain channel with embedded pilots. The simulation results show that this method had better performance than that in [7]. In [17], the authors utilized a deep convolutional neural network (CNN) to address the issue of interference and noise in the channel matrix within the delay-Doppler (DD) domain. By applying the CNN for denoising, they achieved improved channel estimation results compared to existing methods. Additionally, this scheme demonstrated a notable enhancement in spectral efficiency. In [18], a two-dimensional CNN was proposed to accomplish signal detection in OTFS systems. The 2D-CNN can incorporate the imaginary part of the signal during training and the online phase to acquire a three-dimensional array channel. This method has a significant performance improvement over current methods in high-Doppler channels. In [19], deep RNN was used for channel prediction for the MIMO system, the proposed channel predictor contained a long short-term neural network (LSTM) and gated recurrent unit (GRU) to predict the channel state information (CSI). The authors in [20] proposed a 2D-convolutional long short-term memory network (2D-ConvLSTM) to estimate the channel coefficients. In this context, the channel is considered a two-dimensional convolution in the DD domain, which allows for the utilization of the 2D-ConvLSTM network to predict these coefficients. In [21], a deep residual learning network (ResNet) was used to enhance the performance of traditional channel estimation algorithms. The LSTM and implied temporal correlation between channels were used in the channel prediction in [22,23].

In this work, we employ a predictive approach to enhance the performance of channel estimation by considering the implied time correlation. By capturing and utilizing this time correlation information, we aim to make predictions about the channel state and improve the accuracy of the estimation process. First, we obtain preliminary channel estimation

results by using the algorithm in the literature [6]. The preliminary results are considered as time series due to the temporal correlation between channels, and then the series are put into the neural network to obtain more accurate channel estimation results. The remainder of this paper is outlined as follows: Section 2 describes the system model and threshold channel estimation algorithm. Then, the proposed channel estimation algorithm based on a transformer neural network is presented in Section 3. Simulation results are provided in Section 4, and the paper concludes in Section 5.

2. System Model

2.1. Basic Concept

The OTFS modulation block diagram is shown in Figure 1. At the transmitter, data symbols, $x[k,l]$, of size $N \times M$ are placed in the DD domain, and the data symbols are mapped to the TF domain symbols, $X[n,m]$, by the inverse symplectic finite Fourier transform (ISFFT):

$$X[n,m] = \frac{1}{\sqrt{MN}} \sum_{k=0}^{N-1} \sum_{l=0}^{M-1} X[k,l] e^{-j2\pi(\frac{ml}{M} - \frac{nk}{N})} \tag{1}$$

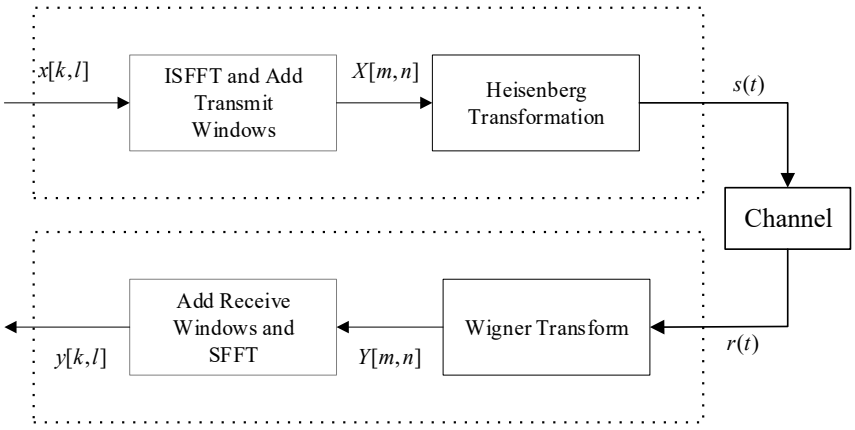


Figure 1. OTFS Modulation.

After that, $X[n,m]$ is transformed from the TF domain to the time domain by the Heisenberg transform:

$$s(t) = \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} X[n,m] e^{j2\pi m\Delta f(t-nT)} p_{tx}(t-nT) \tag{2}$$

where $p_{tx}(t)$ is the pulse shape at the transmitter, and Δf and T are the subcarrier interval and the sampling interval, respectively.

From [3], the channel model in the DD domain can be expressed as

$$h(\tau,\nu) = \sum_{i=1}^{P} h_i \delta(\tau-\tau_i)\delta(\nu-\nu_i) \tag{3}$$

where P is the number of paths, and h_i, τ_i, ν_i stand for channel path gain, delay, and Doppler shift, respectively. The time domain signal, $r(t)$, is obtained at the receiver after passing through the channel, which can be written as

$$r(t) = \int_{\nu} \int_{\tau} h(\tau,\nu) e^{j2\pi v(t-\tau)} s(t-\tau) d\tau d\nu + n(t) \tag{4}$$

where $n(t)$ denotes Gaussian noise.

The Wigner transform converts the received signal, $r(t)$, into the TF domain signal, $Y[n,m]$:

$$A_{p_{rx,r}}(t,f) \triangleq \int p_{rx}^{*}(t'-t)r(t)e^{-j2\pi f(t'-t)}dt' \tag{5}$$

$$Y[n,m] = A_{p_{rx,r}}(t,f)\Big|_{t=nT,\ f=m\Delta f} \tag{6}$$

where $p_{rx}(t)$ represents the receive pulse shape, and $(\bullet)^*$ represents the complex conjugate operation. Finally, the TF domain signal is converted to the DD domain signal by SFFT:

$$y[k,l] = \frac{1}{\sqrt{MN}} \sum_{k=0}^{N-1} \sum_{l=0}^{M-1} Y[n,m] e^{-j2\pi(\frac{nk}{N} - \frac{ml}{M})} \tag{7}$$

From (1)–(7), we can obtain the input–output relationship formula in the DD domain:

$$y[k,l] = \sum_{k'=0}^{N-1} \sum_{l'=0}^{M-1} x[k',l'] h_{\omega}[(k-k')_N, (l-l')_M] + n[k,l] \tag{8}$$

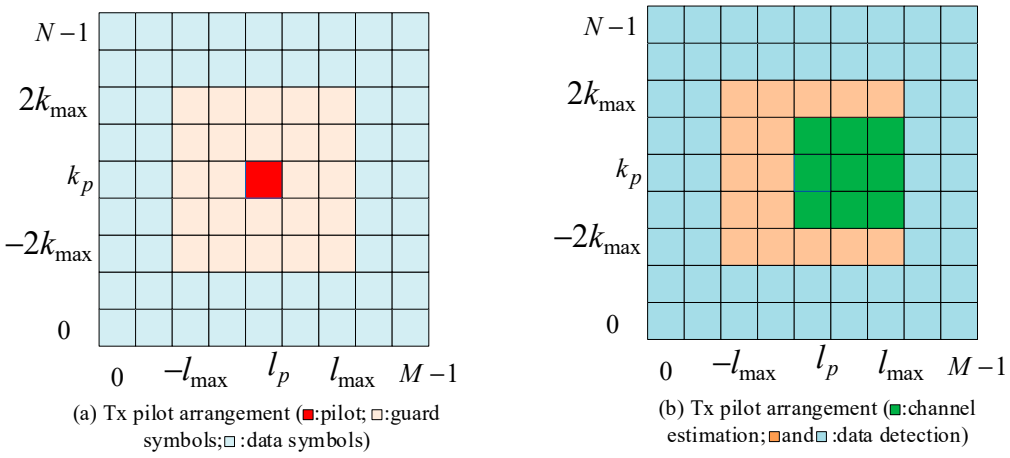
where $n[k,l]$ is the Gaussian noise term with power spectral density, N_o, and $h_{\omega}[k,l]$ denotes the equivalent channel matrix in the DD domain. It is worth mentioning that the pulse used in Equation (4) is the ideal pulse. However, in real systems, non-ideal pulses can lead to cases of fractional Doppler [24]. In this paper, we focus on the ideal pulse case, and Equation (8) is only applicable to that specific scenario.

2.2. The Threshold Scheme

The pilot placement of the threshold scheme in [7] is shown in Figure 2. For a given maximum delay index, $l_{\max}$, and Doppler index, $k_{\max}$, since each symbol in the DD domain is considered to experience similar fading, only one pilot symbol is needed to complete the channel estimation. The pilot symbol at the transmitter in the DD domain can be represented as

$$x[k,l] = \begin{cases} x_p & k = k_p, l = l_p \\ 0 & k \in [k_p - 2k_{\max}, k_p + 2k_{\max}], \\ & l \in [l_p - l_{\max}, l_p + l_{\max}] \\ x_d & otherwise. \end{cases} \tag{9}$$

where x_p denotes the pilot symbol, x_d denotes the data symbols, and we have $N_n = (2l_{\max}+1)(4k_{\max}+1) - 1$ as guard symbols.



(a) Tx pilot arrangement (■:pilot; □:guard symbols;□ :data symbols)

(b) Tx pilot arrangement (■:channel estimation; ■and□ :data detection)

Figure 2. Pilot placement scheme.

We can rewrite the input–output relationship of the pilot according to (8) as

$$y[k,l] = b[k-k_{\max}, l-l_{\max}]\hat{h}[k-k_{\max}, l-l_{\max}]x_p + n[k,l] \tag{10}$$

At the receiver, for the channel estimation part, $k \in [k_p - k_{\max}, k_p + k_{\max}], l \in [l_p, l_p + l_{\max}]$, we set a threshold, $\Gamma = 3\sqrt{N_o}$, where N_o represents the power spectral density of noise. If $|y[k,l]| \geq \Gamma$, we can assume that $b[k-k_{\max}, l-l_{\max}] = 1$ and $\hat{h}[k-k_{\max}, l-l_{\max}] = y[k,l]/x_p$. Otherwise, the result can be written as $\hat{h}[k-k_{\max}, l-l_{\max}] = n[k,l]$. This means that if the path exists, the received signal after this algorithm is a pilot containing Gaussian noise. Otherwise, it is only noise.

2.3. Minimum Mean Square Error Detection Algorithm

The BER is also a measure of channel estimation performance. We assume that in the received signal, $\hat{x} = Wy$, where W denotes the weighted matrix, the error of estimation is $e = \hat{x} - x = Wy - x$, and the mean square error can be written as $e_{MSE} = E\|Wy - x\|^2$. The minimum mean square error (MMSE) detection algorithm minimizes the mean square error between the actual signal and the estimated signal using the minimum mean square error as a criterion. When e_{MSE} is minimized, the weighed matrix can be written as

$$W_{MMSE} = H^H(H^H H + \sigma^2 I)^{-1} \tag{11}$$

where H represents the effective channel matrix of size $MN \times MN$, and σ^2 represents the variance of noise. The recovered signal after using this algorithm can be represented as $\hat{x} = W_{MMSE}y$.

3. Proposed Transformer Estimation Algorithm

3.1. Architecture of the Proposed Channel Estimation Algorithm

Our proposed Transformer estimation structure is shown in Figure 3. The Transformer estimation algorithm is divided into two phases: offline training and online prediction. During the offline training phase, the neural network is trained using a substantial amount of channel data. This training process aims to optimize the parameters of the Transformer neural network for accurate channel estimation. In the subsequent online prediction phase, the preliminary result obtained from the threshold channel estimation algorithm is utilized as input to the trained neural network. By feeding this preliminary result into the network, the algorithm produces a more accurate output for channel estimation.

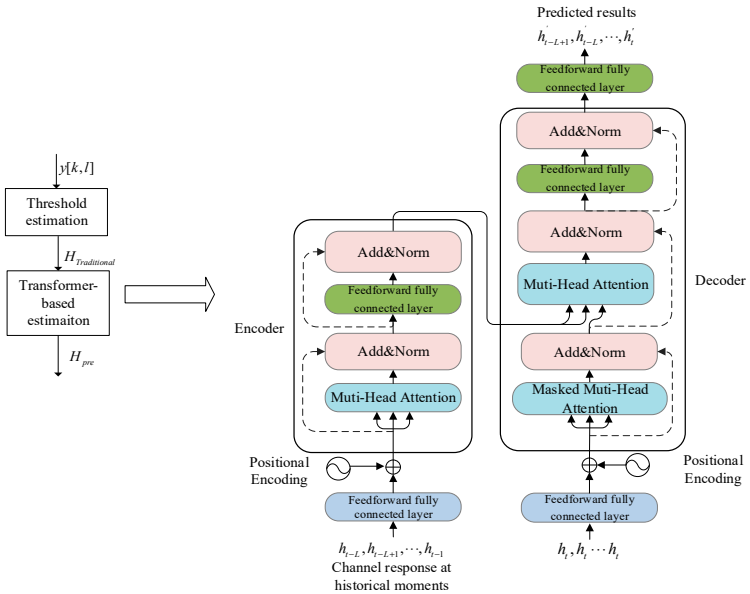


Figure 3. Transformer estimation structure.

In [25], a transformer based on a self-attentive mechanism was first proposed. As shown in Figure 3, the Transformer estimation structure consists of two parts: the encoder and the decoder. To balance performance and computational complexity, our Transformer estimation model retains the encoder–decoder structure of the original model and changes the number of encoder and decoder layers from six layers to two layers.

In the encoder, the time series data $[h_{t-L}, h_{t-L+1}, \cdots, h_{t-1}]$ are mapped into multiple high-dimensional vectors through the feedforward fully connected layer at the input, where L denotes the time step. Subsequently, the temporal position information of the series is encoded in the position encoding area. The role of position encoding is to provide position information to the model through the linear variation of sin and cos functions. The positional encoding can be represented as

$$PE(pos, 2i) = \sin(pos/10,000^{2i/d_{\text{model}}}) \tag{12}$$

$$PE(pos, 2i+1) = \cos(pos/10,000^{2i/d_{\text{model}}}) \tag{13}$$

where pos represents the location index of information for each moment of the series, i represents the dimensional index of high-dimensional vectors obtained by mapping the time series through the input layer, and d_{model} is the dimension of high-dimensional vectors.

Then, the positional encoding information is added to the output of the input layer. Concatenating these vectors together yields a matrix, $H' = [h'_{t-L+1}, h'_{t-L}, \cdots, h'_t]$, as the input, and three vectors are generated from H' at each moment, which are the query vector, q, key vector, k, and value vector, v [26]:

$$\begin{aligned} q_t &= W^q h_t \\ k_t &= W^k h_t \\ v_t &= W^v h_t \end{aligned} \tag{14}$$

where W denotes the weighting matrix. The vectors at each moment are spliced into matrices: Q, K, V.

Matrices Q, K, V are calculated by the self-attentive mechanism to obtain the estimation results:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{15}$$

The multi-headed attention mechanism is the integration of multiple independent attention modules, which is similar to multiple convolutional kernels in convolutional neural networks (CNN) to help the network extract richer features:

$$MultiHead(Q, K, V) = Concat(head_1, \cdots, head_i)W_o \tag{16}$$

where $Concat(\bullet)$ denotes the matrix splicing operation, and W_o denotes the weighting matrix. Finally, the residual network is applied to solve the problems of gradient disappearance and weight matrix degradation.

The results obtained from the multi-attention mechanism are subjected to residual connection layer and layer normalization operations to obtain the final outputs. In [27], ResNet was proposed to solve the problem of the difficult optimization of multilayer neural networks. The input–output relationship of the ResNet can be written as

$$G(x) = F(x) + x \tag{17}$$

where x denotes the input, and $F(x)$ represents the nonlinear variation function. Compared to the non-residual network, when using the ResNet to calculate a partial derivative of x, a constant term is added to the derivation result, which keeps the model from losing gradient during training. Layer normalization serves to normalize all dimensions of each input sequence, thus speeding up model convergence and alleviating the gradient dispersion problem in deep network engineering.

3.2. Training Process

The proposed Transformer estimation algorithm training process can be described as follows: The inputs of the training data are the results of the conventional channel estimation algorithm, and the labels are actual channel responses. Because the channel response is a complex form, our training data splices the real and imaginary parts together. The channel responses from the previous L moments are used to predict the output of the next moment as follows:

$$h_t = f_{Transformer}(h_{t-L+1}, \cdots, h_{t-1}) \tag{18}$$

where the timestep, L, can be determined by using the partial autocorrelation coefficient (PACF). The PACF is the linear correlation of the sequence, $\{h_t\}$, with the sequence $\{h_{t-k}\}$, with lag of order, k, and removes the linear dependence of $\left\{h_{t-1}, h_{t-2}, \cdots, h_{t-(k-1)}\right\}$. In Figure 4, the magnitude of the correlation coefficients of the PACF is compared between the real parts and imaginary parts at different lags of order, k. The black solid line and the red dashed line in the Figure 4 represent the lags of order and the correlation coefficient when the data is approximately uncorrelated, respectively. It is clear to see that the PACF value drops below 0.1 when lags of order $k = 18$, so we can assume that the correlation is almost nonexistent. It is worth mentioning that choosing a lag that is too short may result in incomplete learning of the temporal correlation within the sequence. Conversely, an excessively large lag could introduce extraneous noise during training, thereby impacting its effectiveness.

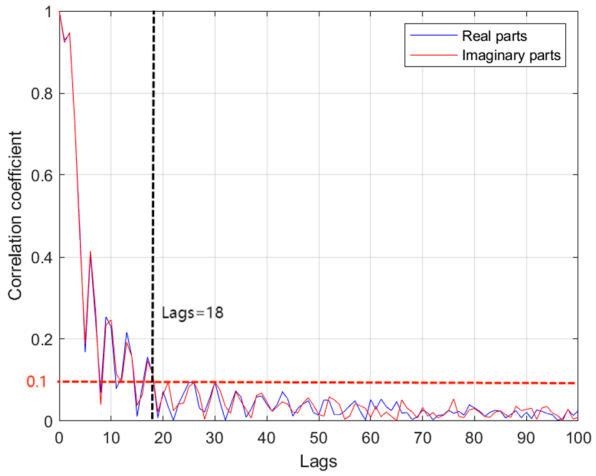


Figure 4. PCAF results for real parts and imaginary parts.

In the training phase, the mean square error (MSE) is selected as the loss function [28]:

$$J_{MSE} = \frac{1}{L}\sum_{i=1}^{N}(f_{Transformer}(h_{t-L+1}, \cdots, h_{t-1}) - H_{label})^2 \tag{19}$$

where H_{label} is the actual value and N is the total number of samples in the training set. The adaptive moment estimation (ADAM) algorithm is applied to update our dataset adaptively [29]. The sizes of our training set and test set are 8000 and 2000, respectively. The learning rate, batch size, and epochs are adjusted to 0.001, 128, and 150.

4. Performance Evaluation

4.1. NMSE and BER Performance

In this section, the simulation results of our proposed algorithm are given to verify the better performance than the threshold method and other deep learning methods. The parameters of our simulation are given in Table 1.

Table 1. Simulation parameters.

Parameter	Value
Carrier frequency	3.35 GHz
No. of subcarriers (M)	32
No. of OTFS symbols (N)	32
No. of channel paths	3
Subcarrier spacing (Δf)	15 kHz
Channel model	Rayleigh Channel
Path delay	[0, 0.2, 0.4] μs
Path power	[0, −10, −10] dB
Modulation alphabet	4-QAM

The NMSE of our channel estimation method is computed as

$$NMSE = \frac{\|H' - H\|^2}{\|H\|^2} \tag{20}$$

where H' denotes the predicted value, and H is the true value.

In Figure 5, we compared the NMSE performance at different SNR when the SNR_p is fixed at 30 dB, where $SNR_p = E_{pilot}/E_{data}$. The NMSE curves are close to a horizontal line because the transmitter power is constant, which leads to a reduction in the impact of the SNR. Furthermore, we can observe that the performance of DL methods is better than that of the threshold method. Among these neural networks, the performance of RNN and LSTM exceeds that of DNN. This is because RNN and LSTM have a powerful time series processing capability that DNN does not provide. LSTM slightly outperforms RNN due to the short-term dependency bottleneck inherent in RNN. The NMSE performance of our proposed Transformer estimation method exceeds the threshold method by approximately 18 dB. The Transformer estimation method outperforms LSTM due to its utilization of the multi-headed attention mechanism. This mechanism correlates the input sequences of the encoder with those of the decoder, enabling the model to focus on the encoder input sequences that exhibit a higher correlation with the predicted outputs. As a result, it reduces the impact of irrelevant information and enhances the prediction performance.

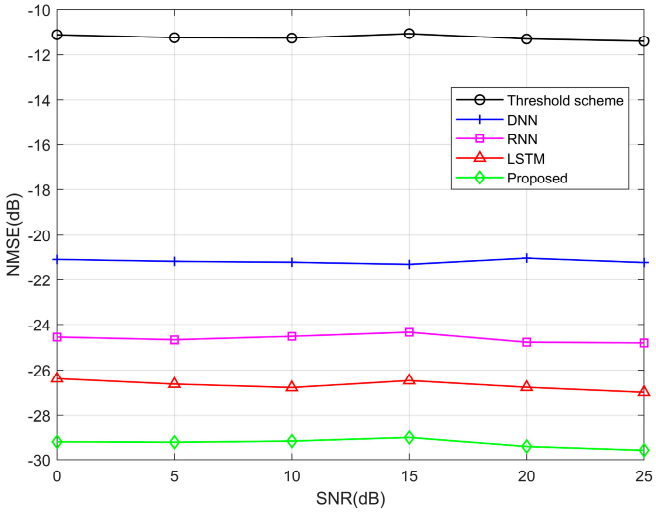


Figure 5. NMSE performance curves between different algorithms.

In Figure 6, the SNR is fixed at 25 dB, and the NMSE performance is compared at different SNR_p. It is evident that the channel estimation performance improves as the SNR_p increases. This is because in the case of the lower SNR_p, the interference of noise is extremely serious, thus causing the threshold method to fail. We have observed that even at lower

SNR_p, our proposed scheme surpasses the performance of the threshold scheme in terms of channel estimation. For example, the NMSE of the threshold scheme is about 12 dB when the SNR_p = 30 dB, while our Transformer estimation model requires a SNR_p of only 10 dB to achieve similar performance, which allows for a reduction in pilot power overhead.

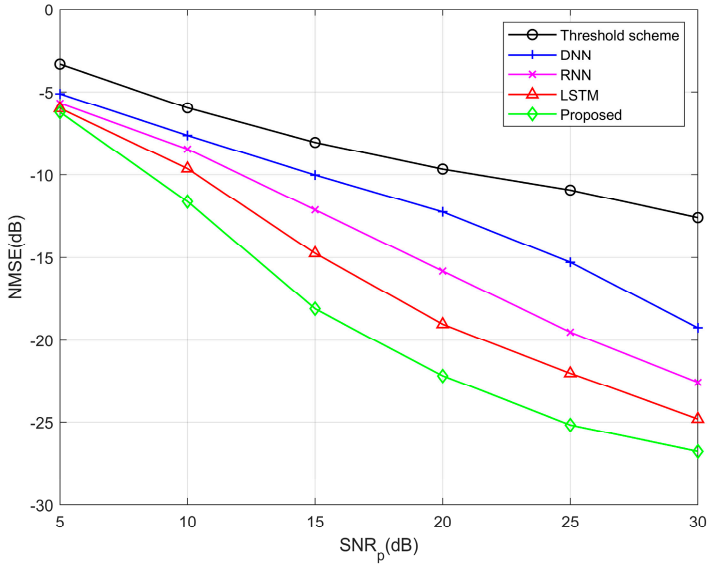


Figure 6. NMSE performance curves between different algorithms under different SNR_p.

Figure 7 illustrates the NMSE capability of the threshold scheme and other neural networks at different speeds. We can observe a decrease in NMSE performance with increasing velocity; this is because the channel we utilize changes at a slower rate in low-speed scenarios, and the channel estimation results are expected to be slightly superior compared to rapidly changing channels in high-speed scenarios. Although the NMSE performance decreases, the variety is not significant, only about 1~2 dB per 100 km/h; this is due to the good robustness of the OTFS system to high Doppler spread.

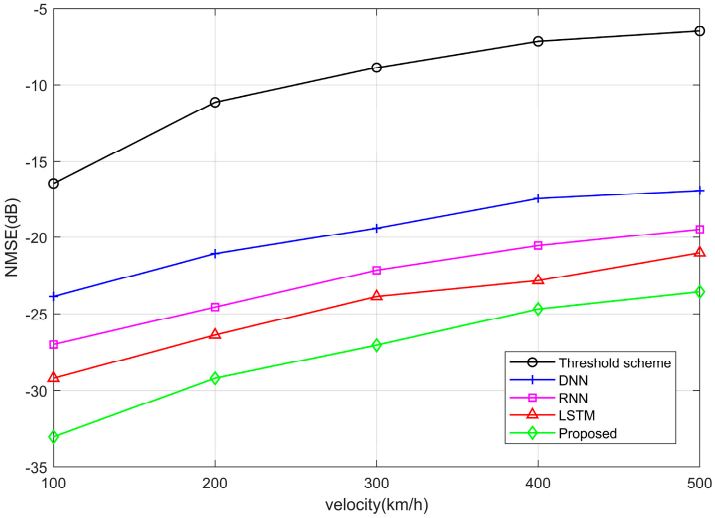


Figure 7. NMSE performance curves between different algorithms under different velocities.

In Figure 8, we show the BER comparison under different SNR, where the case of perfect channel estimation is also taken into account. It is observed that the BER decreases with increasing SNR. In the case of low SNR, the impact of noise can hinder the ability to accurately reflect the difference in channel estimation performance between individual algorithms. As SNR increases, the performance of the Transformer estimation algorithm

acquires acceptable BER results. The threshold method exhibits the worst BER due to its inferior channel estimation performance.

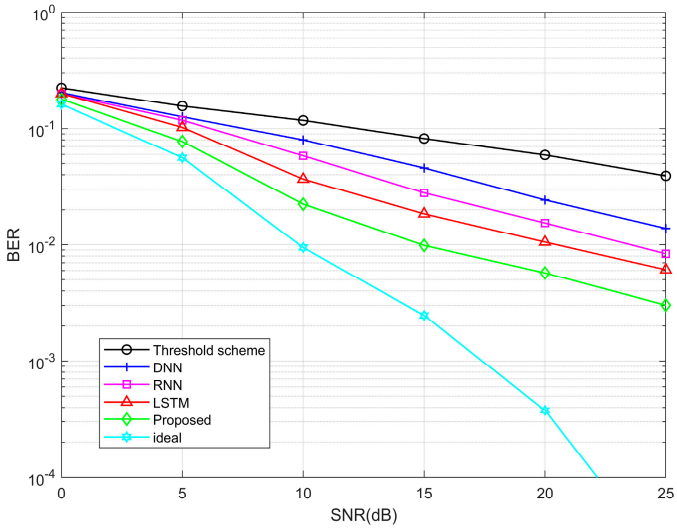


Figure 8. BER performance curves between different algorithms.

4.2. Computational Complexity

Space and time are two important metrics for evaluating the computational complexity of a DL model. In this paper, the quantity of parameters is chosen as the space evaluation metric, and the quantity of floating-point operations (FLOPs) is selected for the time aspect. As the number of parameters in a model increase, it necessitates a larger amount of data to effectively train the model. However, this can lead to a higher risk of overfitting, where the model becomes overly specialized to the training data. Additionally, excessive time complexity can significantly prolong the training and prediction processes.

The quantity of parameters refers to how many parameters the model contains and directly ascertains the size of the model. Usually, the parametric quantities of a model are analyzed to determine the memory usage of a computer. Assuming that the DL models contain multiple hidden layers, the formula for calculating the quantity of parameters for DNN, RNN, LSTM, and Transformer can be written respectively as

$$Q_{DNN} = n_h^1 \times n_i + n_h^1 + \sum_{i=2}^{I} \left(n_h^{i-1} \times n_h^i + n_h^i \right) + n_h^I \times n_o + n_o \tag{21}$$

$$Q_{RNN} = \left(n_h^1 + n_i \right) \times n_h^1 + n_h^1 + \sum_{i=2}^{I} \left[\left(n_h^{i-1} + n_h^i \right) \times n_h^i + n_h^i \right] + n_h^I \times n_o + n_o \tag{22}$$

$$Q_{LSTM} = \left[\left(n_h^1 + n_i \right) \times n_h^1 + n_h^1 \right] \times 4 + 4 \times \sum_{i=2}^{I} \left[\left(n_h^{i-1} + n_h^i \right) \times n_h^i + n_h^i \right] + n_h^I \times n_o + n_o \tag{23}$$

where I denotes the number of hidden layers, and n_i, n_o, n_h^1, n_h^i represent the number of neurons in the input layer, the output layer, the first hidden layer, and the i-th hidden layer, respectively.

The Transformer estimation model constructed in Section 3.1 is then analyzed for spatial complexity. First, a matrix transformation in a fully connected layer that does not contain activation operations can be written as

$$R_o = T_i W + B \tag{24}$$

where $R_o \in^{S\times O}$, $T_i \in^{S\times I}$ denotes the output and the input of the network at that layer, respectively. W represents the weighting matrix, and B represents the bias matrix. Then, the number of parameters for a fully connected layer can be written as

$$Q = (I + S) \times O \tag{25}$$

The quantity of FLOPs can be understood as the amount of computation, which is used to measure the complexity of the model time. Individual FLOPs generally refer to a single addition, subtraction, multiplication, or division operation. In neural networks, since the number of operations of the activation function is very small, it is not considered in FLOPs. Assuming that the DL models contain multiple hidden layers, the formula for calculating the quantity of FLOPs for DNN, RNN, LSTM, and Transformer can be written, respectively, as

$$F_{DNN} = 2n_i \times n_h^1 + \sum_{i=2}^{I} \left(2n_h^{I-1} \times n_h^I\right) + 2n_h^I \times n_o \tag{26}$$

$$F_{RNN} = \left(\left(n_i + n_h^1\right) \times n_h^1\right) \times 2 + 2 \times \sum_{i=2}^{I} \left[\left(n_h^{I-1} + n_h^I\right) \times n_h^I\right] + 2n_h^I \times n_o \tag{27}$$

$$F_{LSTM} = \left(\left(n_i + n_h^1\right) \times n_h^1\right) \times 2 \times 4 + 4 \times 2 \times \sum_{i=2}^{I} \left[\left(n_h^{I-1} + n_h^I\right) \times n_h^I\right] + 2n_h^I \times n_o \tag{28}$$

The number of FLOPs for a fully connected layer in the Transformer estimation model can be written as

$$F = S \times 2IO \tag{29}$$

where the significance of each parameter is consistent with the space complexity analysis, so we do not explain it in detail.

Table 2 shows the complexity analysis of different neural network models. The 32-bit system and NVIDIA V100 with 14.13 TFLOPs, manufactured by the American company NVIDIA Corporation headquartered in Santa Clara, California, United States, are used as examples. As observed from the table, the number of parameters required by the Transformer estimation model is fewer than that of the LSTM and RNN. The higher parameter requirement in RNN and LSTM can be attributed to the inclusion of self-connecting feedback loops. These loops necessitate processing information from previous time steps before computing the hidden layer state for the next time step. Consequently, the historical hidden layer state information needs to be preserved until all time step inputs are processed, resulting in a substantial memory overhead. On the other hand, the Transformer estimation model eliminates the need for such sequential dependencies, leading to a smaller parameter count. The Transformer estimation model enables parallel processing of the input time series by using a multi-head attention mechanism, which greatly reduces the number of parameters in the model, and thus takes up only a small amount of memory space. A comparison of the FLOPs of the different models shows that the Transformer estimation model has the highest FLOPs. The higher FLOPs in the Transformer estimation model can be attributed to its extensive use of matrix transformations. These transformations involve a larger number of parameters and consequently result in longer computational time. However, this increased computational complexity allows the Transformer estimation model to effectively capture complex patterns and dependencies in the data, leading to superior performance in various tasks. Despite the additional computational requirements, the benefits of the Transformer estimation model make it a compelling choice.

Table 2. Complexity analysis of different DL models.

DL Model	Space Complexity		Time Complexity	
	Number of Parameters	Memory Usage Size (MB)	Number of FLOPs	Computational Time (μs)
DNN	11,392	0.043	30,720	2.17×10^{-9}
RNN	37,154	0.141	70,989	5.02×10^{-9}
LSTM	148,609	0.567	354,596	2.51×10^{-8}
Transformer estimation	14,077	0.0537	935,680	6.62×10^{-8}

5. Conclusions

In this paper, we focus on exploring the channel estimation algorithm for OTFS systems and propose a novel approach that incorporates the Transformer model. The inclusion of Transformer in our work stems from its exceptional capability to handle time series data more effectively when compared to other neural network architectures. Initially, we employ the threshold method to obtain preliminary channel estimation results. However, in order to achieve superior channel estimation performance, we consider the inherent temporal correlation present in channel responses. To leverage this correlation, we treat the data as time series and utilize them for predicting channel responses. The proposed Transformer estimation algorithm demonstrates better NMSE and BER performance than the threshold method proposed in the literature and conventional neural networks under the same SNR, SNR_p, and velocity. Moreover, through simulations, we validate the robustness of OTFS in scenarios with high Doppler spread. Additionally, our work highlights the potential of employing time series-based channel response prediction methods for accurate channel estimation. We also take into account the time complexity and spatial complexity of different models, where our proposed algorithm achieves the highest accuracy while maintaining acceptable computational complexity. Overall, our findings contribute to advancing the field of channel estimation in OTFS systems, demonstrating the effectiveness of incorporating Transformer models and highlighting their benefits in handling time series data.

Author Contributions: Conceptualization, T.Z.; methodology, J.L.; software, T.S.; validation, T.S.; investigation, T.S. and J.L.; data curation, J.L.; writing—original draft, T.S. and J.L.; writing—review and editing, T.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Beijing Natural Science Foundation under Grants 4212006 and L212030, and the National Natural Science Foundation of China under Grants 62071031 and 62341102.

Institutional Review Board Statement: Not applicable.

Data Availability Statement: Data sharing is not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Mahmoud, H.A.; Yucek, T.; Arslan, H. OFDM for cognitive radio: Merits and challenges. *IEEE Wirel. Commun.* **2009**, *16*, 6–15. [CrossRef]
2. Wang, T.; Proakis, J.G.; Masry, E.; Zeidler, J.R. Performance degradation of OFDM systems due to Doppler spreading. *IEEE Trans. Wirel. Commun.* **2006**, *5*, 1422–1432. [CrossRef]
3. Raviteja, P.; Phan, K.T.; Hong, Y.; Viterbo, E. Interference Cancellation and Iterative Detection for Orthogonal Time Frequency Space Modulation. *IEEE Trans. Wirel. Commun.* **2018**, *17*, 6501–6515. [CrossRef]
4. Hadani, R.; Rakib, S.; Tsatsanis, M.; Monk, A.; Goldsmith, A.J.; Molisch, A.F.; Calderbank, R. Orthogonal Time Frequency Space Modulation. In Proceedings of the 2017 IEEE Wireless Communications and Networking Conference (WCNC), San Francisco, CA, USA, 19–22 March 2017; pp. 1–6.
5. Hadani, R.; Rakib, S.; Molisch, A.F.; Ibars, C.; Monk, A.; Tsatsanis, M.; Delfeld, J.; Goldsmith, A.; Calderbank, R. Orthogonal Time Frequency Space (OTFS) modulation for millimeter-wave communications systems. In Proceedings of the 2017 IEEE MTT-S International Microwave Symposium (IMS), Honolulu, HI, USA, 4–9 June 2017; pp. 681–683.

6. Surabhi, G.D.; Augustine, R.M.; Chockalingam, A. On the Diversity of Uncoded OTFS Modulation in Doubly-Dispersive Channels. *IEEE Trans. Wirel. Commun.* **2019**, *18*, 3049–3063. [CrossRef]
7. Raviteja, P.; Phan, K.T.; Hong, Y. Embedded Pilot-Aided Channel Estimation for OTFS in Delay–Doppler Channels. *IEEE Trans. Veh. Technol.* **2019**, *68*, 4906–4917. [CrossRef]
8. Yuan, W.; Li, S.; Wei, Z.; Yuan, J.; Ng, D.W.K. Data-Aided Channel Estimation for OTFS Systems with a Superimposed Pilot and Data Transmission Scheme. *IEEE Wirel. Commun. Lett.* **2021**, *10*, 1954–1958. [CrossRef]
9. Wang, S.; Guo, J.; Wang, X.; Yuan, W.; Fei, Z. Pilot Design and Optimization for OTFS Modulation. *IEEE Wirel. Commun. Lett.* **2021**, *10*, 1742–1746. [CrossRef]
10. Shen, W.; Dai, L.; An, J.; Fan, P.; Heath, R.W. Channel Estimation for Orthogonal Time Frequency Space (OTFS) Massive MIMO. *IEEE Trans. Signal Process.* **2019**, *67*, 4204–4217. [CrossRef]
11. Ge, Y.; Liu, L.; Huang, S.; González, G.D.; Guan, Y.L.; Ding, Z. Low complexity memory AMP detector for high-mobility MIMO-OTFS SCMA systems. In Proceedings of the IEEE ICC Workshop OTFS-DDMC-6G, Rome, Italy, 28 May–1 June 2023.
12. Li, M.; Zhang, S.; Ge, Y.; Gao, F.; Fan, P. Joint channel estimation and data detection for hybrid RIS aided millimeter wave OTFS systems. *IEEE Trans. Commun.* **2022**, *70*, 6832–6848. [CrossRef]
13. Zhou, T.; Wang, Y.; Wang, C.; Salous, S.; Liu, L.; Tao, C. Multi-feature fusion based recognition and relevance analysis of propagation scenes for high-speed railway channels. *IEEE Trans. Veh. Technol.* **2020**, *69*, 8107–8118. [CrossRef]
14. Ye, H.; Li, G.Y.; Juang, B.-H. Power of Deep Learning for Channel Estimation and Signal Detection in OFDM Systems. *IEEE Wirel. Commun. Lett.* **2018**, *7*, 114–117. [CrossRef]
15. Soltani, M.; Pourahmadi, V.; Mirzaei, A.; Sheikhzadeh, H. Deep Learning-Based Channel Estimation. *IEEE Commun. Lett.* **2019**, *23*, 652–655. [CrossRef]
16. Mattu, S.R.; Chockalingam, A. An RNN based DD Channel Estimator for OTFS with Embedded Pilots. In Proceedings of the 2022 IEEE 33rd Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC), Kyoto, Japan, 12–15 September 2022; pp. 457–462.
17. Yang, C.; Wang, J.; Pan, Z.; Shimamoto, S. Delay-Doppler Frequency Domain-Aided Superimposing Pilot OTFS Channel Estimation Based on Deep Learning. In Proceedings of the 2022 IEEE 96th Vehicular Technology Conference (VTC2022-Fall), London, UK, 26–29 September 2022; pp. 1–6.
18. Enku, Y.K.; Bai, B.; Wan, F.; Guyo, C.U.; Tiba, I.N.; Zhang, C.; Li, S. Two-Dimensional Convolutional Neural Network-Based Signal Detection for OTFS Systems. *IEEE Wirel. Commun. Lett.* **2021**, *10*, 2514–2518. [CrossRef]
19. Jiang, W.; Schotten, H.D. Deep Learning for Fading Channel Prediction. *IEEE Open J. Commun. Soc.* **2020**, *1*, 320–332. [CrossRef]
20. Pfadler, A.; Jung, P.; Shala, V.; Kasparick, M.; Adrat, M.; Stanczak, S. Short-Term Prediction of Doubly-Dispersive Channels for Pulse-Shaped OTFS using 2D-ConvLSTM. In Proceedings of the 2022 IEEE International Conference on Communications Workshops (ICC Workshops), Seoul, Republic of Korea, 16–20 May 2022; pp. 939–944.
21. Li, Q.; Gong, Y.; Meng, F.; Li, Z.; Miao, L.; Xu, Z. Residual Learning based Channel Estimation for OTFS system. In Proceedings of the 2022 IEEE/CIC International Conference on Communications in China (ICCC Workshops), Sanshui, Foshan, China, 11–13 August 2022; pp. 275–280.
22. Zhou, T.; Zhang, H.; Ai, B.; Liu, L. Weighted score fusion based LSTM model for high-speed railway propagation scenario identification. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 23668–23679. [CrossRef]
23. Zhou, T.; Zhang, H.; Ai, B.; Liu, L. Deep-learning-based spatial-temporal channel prediction for smart high-speed railway communication networks. *IEEE Trans. Wirel. Commun.* **2022**, *21*, 5333–5345. [CrossRef]
24. Ge, Y.; Deng, Q.; Ching, P.C.; Ding, Z. Receiver design for OTFS with a fractionally spaced sampling approach. *IEEE Trans. Wirel. Commun.* **2021**, *20*, 4072–4086. [CrossRef]
25. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention is all you need. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 5998–6008.
26. Chen, Z.; Gu, F.; Jiang, R. Channel Estimation Method Based on Transformer in High Dynamic Environment. In Proceedings of the 2020 International Conference on Wireless Communications and Signal Processing (WCSP), Nanjing, China, 21–23 October 2020; pp. 817–822.
27. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
28. Liao, Y.; Hua, Y.; Dai, X.; Yao, H.; Yang, X. ChanEstNet: A Deep Learning Based Channel Estimation for High-Speed Scenarios. In Proceedings of the ICC 2019—2019 IEEE International Conference on Communications (ICC), Shanghai, China, 20–24 May 2019; pp. 1–6.
29. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.