Cross-shaped Separated Spatial-Temporal UNet Transformer For Accurate Channel Prediction

Hua Kang*¶, Qingyong Hu*, Huangxun Chen‡†, Qianyi Huang‡, Qian Zhang*, Min Cheng¶
*The Hong Kong University of Science and Technology,
†The Hong Kong University of Science and Technology (Guangzhou), ‡Huawei, §Sun Yat-Sen University,
¶Noah's Ark Lab, Huawei
Email: kang.hua1@huawei.com, qhuag@cse.ust.hk huangxunchen@hkust-gz.edu.cn,
huangqy89@mail.sysu.edu.cn, qianzh@cse.ust.hk, min.cheng@huawei.com

Abstract—**Accurate channel estimation is crucial for the performance gains of massive multiple-input multiple-output (mMIMO) technologies. However, it is bandwidth-unfriendly to estimate large channel matrix frequently to combat the time-varying wireless channel. Deep learning-based channel prediction has emerged to exploit the temporal relationships between historical and future channels to address the bandwidth-accuracy trade-off. Existing methods with convolutional or recurrent neural networks suffer from their intrinsic limitations, including restricted receptive fields and propagation errors. Therefore, we propose a Transformer-based model, CS3T-UNet tailored for mMIMO channel prediction. Specifically, we combine the cross-shaped spatial attention with a group-wise temporal attention scheme to capture the dependencies across spatial and temporal domains, respectively, and introduce the shortcut paths to well-aggregate multi-resolution representations. Thus, CS3T-UNet can globally capture the complex spatial-temporal relationship and predict multiple steps in parallel, which can meet the requirement of channel coherence time. Extensive experiments demonstrate that the prediction performance of CS3T-UNet surpasses the best baseline by at most 6.86 dB with a smaller computation cost on two channel conditions.**

I. Introduction

Massive multiple-input multiple-output (mMIMO) has emerged as a promising technology for communication. mMIMO exploits a great number of antennas deployed at the base station (BS) to concurrently serve users for significant performance gains in spectral and energy efficiency [1]. Many critical technologies in mMIMO rely on accurate channel state information (CSI), such as precoding and beamforming. For example, precoding design plays a vital role in mMIMO to reduce interference and path-loss impacts and increase the throughput. However, these gains rely heavily on accurate channel state information (CSI) estimation and could be compromised by channel aging, which is caused by the time-varying nature of the propagation channel as well as the processing delay at the BS. It indicates that the channel varies between the estimated moment and the moment used for precoding or detection [2]. To mitigate the performance loss, an intuitive idea is to increase the frequency of CSI estimation. However, frequently estimating the large CSI matrix of mMIMO will pose a tremendous burden on bandwidth [3].

Corresponding author: Qian Zhang. Hua Kang finished this work when she was a PhD student at HKUST. Now she is a Researcher at Huawei. Huangxun Chen participated in this research when she was a Researcher at Huawei. She now joins HKUST (GZ) as an Assistant Professor.

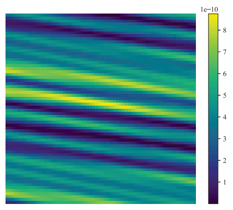
To balance the bandwidth overhead and channel estimation accuracy, channel prediction emerges as a natural solution that intends to exploit the temporal correlation between the historical CSI and the future CSI. It is desired to predict channel multiple time steps ahead accurately to mitigate the channel aging with minimal bandwidth overhead. Existing channel prediction approaches can be roughly divided into two categories: model-based approaches and learning-based ones. The model-based methods include the linear extrapolation model [4, 5], the autoregressive (AR) model [6], and the sum-of-sinusoids model [7]. However, they are insufficient to capture the complicated evolution of wireless channels due to multipath effects and Doppler effects. To this end, deep learning-based methods have been proposed to capture complicated dependencies. The basic methods include multi-layer perceptron (MLP) [8], recurrent neural networks (RNN) [9], and convolutional neural networks (CNN) [10]. Nevertheless, the existing neural network-based methods have some limitations. MLP needs to flatten the input before feeding into the network. Thus, it fails to capture the spatial relationship between adjacent elements. CNN-based methods only have a limited receptive field without global information [11]. Sequential models like RNNs may suffer from propagation loss problems and gradient vanishing issues in multi-step prediction.

To overcome the above-mentioned problems and improve the channel prediction performance, we argue that the model should be able to capture global information, explore the complex spatial-temporal relationship, and be flexible to predict multiple steps. Recent advances [12–14] have demonstrated the great potential of Transformers to address the above issues. However, applying Transformers for spatial-temporal channel prediction still demands an elaborate design. The Transformer-based method in [14] requires transforming each time step's input into a 1D vector, which obviously breaks the spatial relationship and is inefficient for the mMIMO case with large numbers of antennas and subcarriers. To fill this gap, we propose a Cross-Shaped Separated Spatial-Temporal UNet-like Transformer (CS3T-UNet). We first analyze the characteristics of the space/frequency domain channel data and transform them into the angular-delay domain, which demonstrates propagation path distributions with physical prior on spatial sparsity. We adopt the cross-shaped spatial attention [13] to process different spatial dimensions with different physical meanings. This stripe-shape spatial window attention is beneficial to deal with aliasing along different dimensions caused by spectral leakage [15]. We design memory-efficient group-wise temporal attention following spatial attention to capture the temporal relationship. Motivated by the physical prior that the intrinsic spatial and temporal correlations between historical and future CSI should be consistent across different resolutions, we further combine spatial and temporal attention with the UNet architecture [16], which concatenates the encoder-extracted feature maps with the decoder-extracted feature maps at different scales to well aggregate multi-resolution information. Based on the proposed network, we further explore introducing the energy mask-guided sparse attention scheme, guiding the model to focus on the significant elements to boost performance further. With the dedicated design, our system can achieve better performance across different channel conditions with smaller FLOPs than the best baseline. Compared with directly applying Transformer without considering CSI properties, our system can have at most 6.86 dB gain with only around 50% computation cost. Our key contributions are summarized as follows:

- We propose a Transformer-based model, CS3T-UNet, tailored for mMIMO channel prediction. CS3T-UNet can capture the complicated spatial-temporal relationship of the wireless channel to predict the channel multiple time steps ahead accurately and in parallel.
- We design a separated spatial-temporal attention scheme combined with UNet-like structure with consideration of CSI properties such as spectral leakage and feature consistency of different spatial-temporal resolutions.
- We conduct comprehensive experiments on two datasets to demonstrate the superiority of our method on multi-step channel prediction. Our model outperforms the best baseline by at most 6.86dB and over 3dB on average, with a smaller computation cost. Moreover, our evaluation shows that the inference time of our model is within the channel coherence time when predicting multiple steps, which validates the practicality of our approach.

II. Preliminaries

We consider a single-cell massive MIMO system where a single base station (BS) equipped with a N_t-antenna uniform linear array (ULA) serves a single user equipment (UE) with N_r omnidirectional antennas. Orthogonal frequency division multiplexing (OFDM) is adopted, consisting of N_f subcarriers. The signal is propagated through M multipaths, where each path corresponds to a particular delay τ_m, an angle of arrival (AOA) to the BS's antenna θ_m, and a complex amplitude gain α_m. The wireless channel at the l-th subcarrier f_l can be written as


(a) CSI raw data

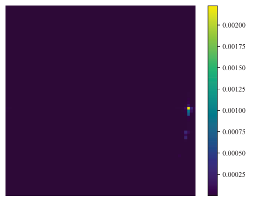

(b) CSI ADP profile

Fig. 1: Comparison between original CSI and ADP profile.

$$\boldsymbol{h}[l] = \sum_{m=1}^{M} \alpha_m \boldsymbol{a}(\theta_m) e^{-j2\pi f_l \tau_m + j\phi(m,l)}, \tag{1}$$

where $\boldsymbol{a}(\theta_m) = [1, e^{-j2\pi\frac{dcos(\theta_m)}{\lambda}}, ..., e^{-j2\pi\frac{(N_t-1)dcos(\theta_m)}{\lambda}}]^T$ represents the array response vector of the ULA (d is the antenna spacing distance). $\phi(m, l)$ is the additive phase due to scattering or reflections during the signal propagation. The overall CSI $\boldsymbol{H}$ between the BS and the UE is

$$\boldsymbol{H} = [\boldsymbol{h}[1], ..., \boldsymbol{h}[N_f]]. \tag{2}$$

To further exploit sparsity inside CSI matrix, the original CSI can be converted to the angular-delay profile (ADP) H' by discrete Fourier transformation (DFT) [17] along the antenna array and the subcarriers:

$$\boldsymbol{H'} = \boldsymbol{F}_f \boldsymbol{H} \boldsymbol{F}_t^H, \tag{3}$$

where $\boldsymbol{F}_f$ and $\boldsymbol{F}_t$ are the DFT matrices with dimension $N_f \times N_f$ and $N_t \times N_t$, respectively. The $(i,j)_{th}$ value of F_f is $F_f(i,k) = exp(-j\frac{2\pi(i-1)(k-1)}{N_f})$ where j is the imaginary unit and F_t is calculated in the same way. Fig. 1 shows an example of amplitude of the original CSI data and the ADP. The x axis of Fig. 1a represents the antenna while the y axis of Fig. 1a represents the subcarrier. The x axis of Fig. 1b represents the angle while the y axis of Fig. 1b represents the delay. We can find that the ADP is much more sparse than the raw original CSI data.

We conduct channel prediction on the ADPs. Specifically, we take the historical T steps' channels to predict future channels in the next L consecutive steps, which can be formulated as:

$$(\hat{\boldsymbol{H}}'_{t+1}, ..., \hat{\boldsymbol{H}}'_{t+L}) = f_\Theta(\boldsymbol{H}'_{t-T+1}, ..., \boldsymbol{H}'_t), \tag{4}$$

where f_Θ represents the channel predictor, *e.g.*, CS3T-UNet.

III. System Design

A. Data Analysis

The channel prediction task uses the historical CSI frames to predict future CSI frames. To design a model tailored for CSI data, we conduct an analysis of the characteristics of CSI data. First, as illustrated in Section II, the data is transformed into an angular-delay profile which has more sparse data and physical

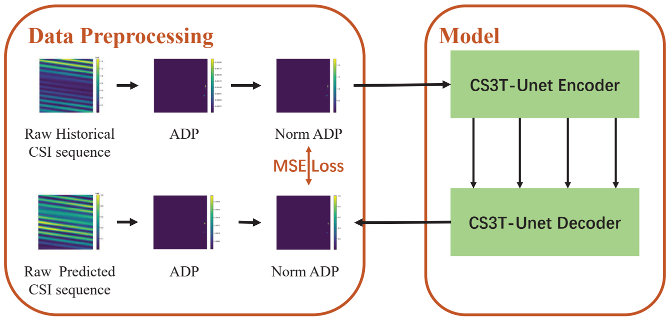


Fig. 2: Overall framework

meaning. However, as the DFT process is conducted with limited antennas and subcarriers, spectral leakage exists in the ADP profile, leading to aliasing between adjacent elements. The finite DFT transformation is equivalent to multiplying a rectangular window in the time domain, which leads to the convolution with the $Sinc(\cdot)$ function in the frequency domain. Thus the elements with the same antenna number or the same frequency have correlations even though they are distant. This calls for a unique feature extraction method for the CSI ADP profile. CSwin Transformer [13] calculates attention through the height and width of the image separately and thus fits our problem. However, the CSwin Transformer is proposed for 2D image classification problem. We need to design a new structure to combine the extra time dimension in channel prediction and satisfy the need for such a regression problem. Another characteristic of the data exists in the sparsity where only several physical paths show significant energy in the ADP profile. We may leverage the sparsity to make the model more focused on the data and reduce computational power.

B. Overview

The overall framework is shown in Fig. 2. The processing procedure includes two main steps, i.e., data preprocessing and model prediction. Both the input and output data are transformed into the angular-delay domain and passed through the data normalization procedure. We use mean squared error (MSE) loss to optimize our model.

The overall architecture of our proposed CS3T-UNet is presented in Fig. 3. For an input channel with the size of $N_f \times N_t \times T \times 2$, we first combine the temporal dimension with the complex dimension, i.e., $N_f \times N_t \times 2T$. The CS3T-UNet is composed of an encoder, a decoder, and skip connections. The encoder consists of shallow feature embedding and deep feature learning, while the decoder consists of deep feature learning and final feature prediction. The shallow feature embedding uses a convolutional layer to tokenize the input. The token embedding dimension is C, and the token size is 2×2. Deep feature learning adopts the Transformer-based architecture. The fundamental component of deep feature learning is our CS3T-UNet layer. Each layer consists of multiple CS3T-UNet blocks followed by a merge block or expand block for the encoder and decoder, respectively. The final feature prediction module utilizes a linear layer to recover

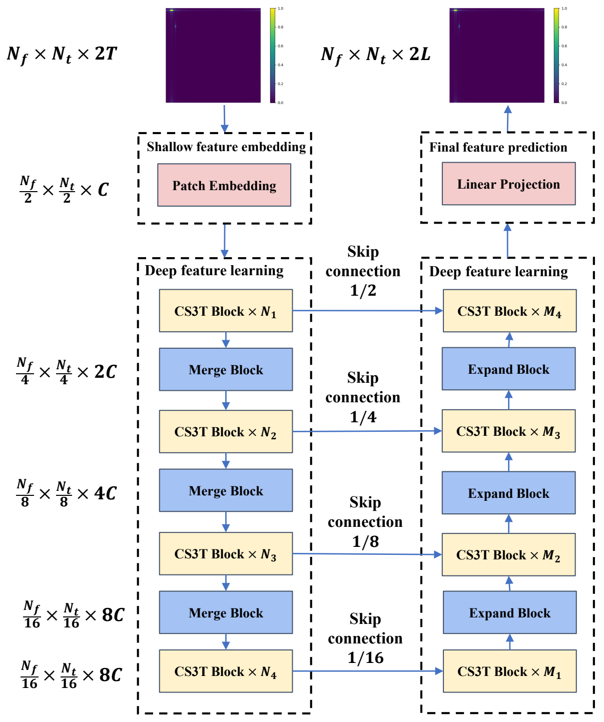


Fig. 3: The overview architecture of CS3T-UNet.

the channel dimension corresponding to the prediction steps, *i.e.*, $2L$. Inspired by the physical prior that features should be consistent across different resolutions, skip connections [16] between the encoder and decoder are deployed to aggregate features.

C. CS3T Block

1) Cross-Shaped Sparse Spatial-Temporal Self-Attention

The input feature of the first CS3T-UNet block is with size $\mathbb{R}^{N_f \times N_t \times C}$, where the spatial domain contains angular and delay information while the channel domain contains temporal information. We design a cross-shaped separated spatial-temporal self-attention mechanism to capture representations of the angular-delay profile and the temporal-spatial correlation. The calculation process is separated into two steps: the first is to calculate the cross-shaped spatial attention, and the second is to calculate the temporal attention.

Cross-Shaped Spatial Self-Attention. In this work, we adopt the cross-shaped spatial self-attention [13]. The reason is twofold. First, the cross-shaped spatial self-attention performs the self-attention calculation in the horizontal and vertical stripes in parallel. This parallel operation introduces no extra computation burden compared to using a single stripe. We can enlarge the receptive field by adjusting the stripe width. Second, calculating self-attention within each stripe aligns with the spatial characteristics of the wireless channel data. Previous works [15, 18] have demonstrated the impact of DFT leakage due to finite antennas and bandwidths. This leads to aliasing between adjacent subcarriers and antennas. Thus, the stripe attention is suitable to capture the attention within the whole angular stripe or delay stripe.

The input $X \in \mathbb{R}^{H\times W\times C}$ is first linearly projected to K heads, and each head will conduct local attention within either horizontal or vertical stripes. We take the calculation process of horizontal attention as an example, and vertical attention is calculated similarly. X is evenly divided into M horizontal stripes $[X^1, ..., X^M]$, each with stripe width sw, and we have $sw \times M = H$. The output of the kth head is calculated as:

$$\begin{aligned} &X = [X^1, X^2, ..., X^M], \\ &Y_k^i = Attention(X^i W_k^Q, X^i W_k^K, X^i W_k^V), \\ &\text{H-Attn}_k(X) = [Y_k^1, Y_k^2, ..., Y_k^M], \end{aligned} \tag{5}$$

where $X^i \in \mathbb{R}^{(sw\times W)\times C}$ and $i = 1, ..., M$. W_k^Q, W_k^K, and W_k^V are the projected matrices for queries, keys, and values of the kth head, respectively. The vertical stripe self-attention of the kth head is denoted as V-Attn$_k(X)$.

We divide the K heads into two parts, each with $K/2$ heads for either horizontal or vertical stripe attention. Finally, the outputs of these two groups will be concatenated.

$$\begin{aligned} \text{CSWin-Atten}(X) = &\text{CONCAT}[\text{H-Attn}_1(X), ..., \text{H-Attn}_{K/2}(X), \\ &\text{V-Attn}_{K/2+1}(X), ..., \text{V-Attn}_K(X)]. \end{aligned} \tag{6}$$

Group-Wise Temporal Attention. Temporal attention is employed after spatial attention. Before feeding the feature map for attention calculation, we add the positional encoding to the feature map. The positional encoding [19] is added along the channel dimension C' as follows:

$$\begin{aligned} PE_{pos,2i} &= sin(pos/10000^{2i/C'}), \\ PE_{pos,2i+1} &= cos(pos/10000^{2i/C'}). \end{aligned} \tag{7}$$

Where pos is the position of the token within the temporal sequence and i is the index of the channel dimension. Then we tokenize the channel dimension by evenly splitting the channel dimension into N non-overlapping groups, each with sw channels, i.e., $X = [X^1, X^2, ..., X^N]$, and $N \times sw = C$. The multi-head self-attention (MSA) is performed within each temporal group such that the dependencies between different channels can be modeled. This step complements spatial attention for temporal feature extractions. The calculation process is as follows:

$$\begin{aligned} &X = [X^1, X^2, ..., X^N], \\ &Y_k^i = Attention(X^i W_k^Q, X^i W_k^K, X^i W_k^V), \\ &\text{T-Attn}_k(X) = [Y_k^1, Y_k^2, ..., Y_k^N] \end{aligned} \tag{8}$$

where $X^i \in \mathcal{R}^{sw\times(H\times W)}$ and the attention is calculated within X^i where sw serves as the sequence length.

Sparse Attention Calculation with Mask In this section, upon the main network structure illustrated before, we introduce the sparsity masks leveraging the sparsity characteristics of our channel prediction data.

Fig. 4 shows one sample from DeepMIMO [20] dataset, which is normalized by dividing its maximum amplitude. We can observe that most elements of the ADP approach zero, and only a few elements contain the most energy in the sample. Thus, the attention calculation in most patches may waste computation power. We can leverage the sparsity in the wireless channel by dedicating the model to the significant values. This design could have two possible benefits: 1. reduce computational consumption during training. 2. make the model focus more on the significant values, which may bring faster convergence.

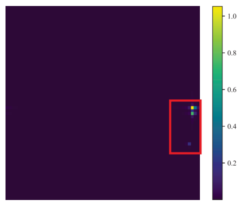


Fig. 4: An angular-delay profile of DeepMIMO dataset (The red box points out the area of significant values).

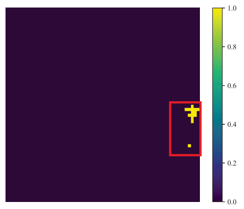


Fig. 5: The mask selects significant values. The red box points out the area where the mask equals 1.

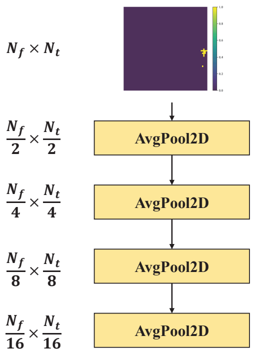
To leverage the channel sparsity characteristic with the mask, we first need to determine how to generate the mask for each data sample. Then we need to propose a method to combine the mask in the proposed network architecture.

The changes in the delay and angle of the propagation path over time are much slower than the change in the fading coefficients. Thus, the indices of significant values are relatively fixed over time. In this way, referring to [21], we introduce a method to select significant values. We have the previous time steps' data with the format of $\boldsymbol{H} \in \mathcal{R}^{T\times 2\times N_f\times N_t}$. We first calculate the power of each frame and then we take the average along the time dimension. Thus we got the average sample power $|\tilde{\boldsymbol{H}}| \in \mathcal{R}^{N_f\times N_t}$. We calculate the total energy of the sample, and we extract the top-k elements that have a summation of power larger than $\lambda\%$ of the total energy. The calculation process is shown in Alg. 1. M is the generated mask which has the same spatial dimension as the data sample, i.e, $M \in \mathcal{R}^{N_f\times N_t}$. We set λ as an energy threshold, e.g., 0.9. The selected values are set to 1 in the mask. The mask for the sample is shown in Fig. 5.

The second step is considering how to integrate the prior mask into our model design. Every time the data pass through one layer, the width and height are reduced by two. To keep the same shape as the input data, the mask is also reduced by two after each layer. The mask is reduced by the average pooling

Algorithm 1 Mask Generation Algorithm

```
Input: The historical CSI sequence in angular-delay profile $\boldsymbol{H}'_1, ..., \boldsymbol{H}'_T$, the predefined threshold $\lambda$
1: Calculate the average sample power $|\tilde{\boldsymbol{H}}'| = \frac{1}{T}\sum_{t=1}^{T}|\boldsymbol{H}'_t|$.
2: Calculate the total power of the average sample $|\tilde{\boldsymbol{H}}'_t| = \sum_{i=1}^{N_f}\sum_{j=1}^{N_t}|\tilde{\boldsymbol{H}}'|_{i,j}$.
3: Sort the elements of $|\tilde{\boldsymbol{H}}'_t|$ according to their amplitudes. $\boldsymbol{H}_s, \Omega_s = Sort(|\tilde{\boldsymbol{H}}'_t|)$, where $\boldsymbol{H}_s$ is the sorted values and $\Omega_s$ is the set of corresponding indices.
4: $E_{sum} = 0$, $M \in \mathcal{R}^{N_f \times N_t} = 0$.
5: for $(i,j) \in \Omega_s$ do
6:     if $E_{sum} + \boldsymbol{H}_s(i,j) < \lambda \times |\tilde{\boldsymbol{H}}'_t|$ then
7:         $M_{(i,j)} = 1$
8:         $E_{sum} = E_{sum} + \boldsymbol{H}_s(i,j)$
9:     else
10:        break
11:    end if
12: end for
Output: The generated mask $M \in \mathcal{R}^{N_f \times N_t}$
```



$N_f \times N_t$
$\frac{N_f}{2} \times \frac{N_t}{2}$
AvgPool2D
$\frac{N_f}{4} \times \frac{N_t}{4}$
AvgPool2D
$\frac{N_f}{8} \times \frac{N_t}{8}$
AvgPool2D
$\frac{N_f}{16} \times \frac{N_t}{16}$
AvgPool2D

Fig. 6: Architecture to generate masks for each layer.

layer which doesn't introduce extra parameters to the model. The mask calculation layer is illustrated in Fig. 6. Then in each block, the mask goes through the same tokenization process as our data to have the same shape as the data. When calculating the attention, the mask will be treated as a binary mask, and only the windows overlapped with the mask can be selected for further attention calculation. With the introduction of the mask, the attention calculation process is changed as follows.

$$\begin{aligned} M_i &= \text{AVGPOOL2D}(M_{i-1}) \\ Q_i, K_i, V_i &= \text{MLP}(H_i) \\ Q_{sel} &= \text{MASKSELECT}(Q_i, M_i) \\ K_{sel} &= \text{MASKSELECT}(K_i, M_i) \\ V_{sel} &= \text{MASKSELECT}(V_i, M_i) \\ Attention &= \text{SOFTMAX}(\frac{Q_{sel}K_{sel}^T}{\sqrt{V_{sel}}}). \end{aligned} \tag{9}$$

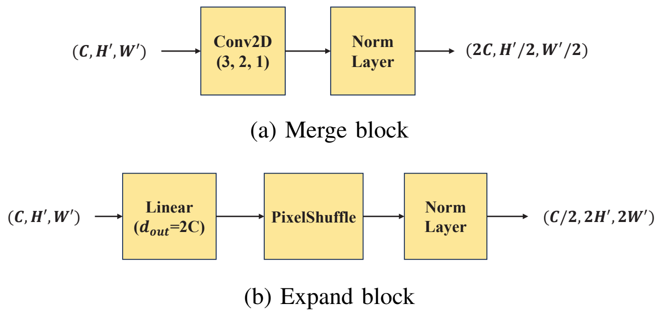


Fig. 7: Architecture of merge block and expand block

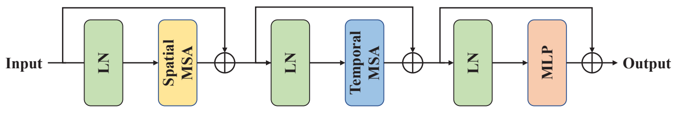


Fig. 8: Illustration of the CS3T-UNet Block.

Note that the above process is conducted in each attention calculation process. For the MaskSelect function, the selection criterion is whether the patch has an overlap with the corresponding mask patch. We show the performance of mask-guided sparse attention in Section IV-D.

Shortcut for Multi-Resolution Aggregation. The channel predictor f_Θ maps historic CSI samples into the future CSI sequence. To balance between the model complexity and training effectiveness, we pose a physical prior to the model training scheme, *i.e.*, the intrinsic spatial and temporal correlations between historical and future CSI should be consistent across different resolutions. Specifically, we resample the extracted features with a Merge and Expand Block as shown in Fig. 7.

The Merge Block at the encoder uses a convolutional kernel with (kernel size=3, stride=2, padding=1) to reduce the number of antennas and time steps, which mimics lower spatial and temporal resolutions. Followed by a CS3T Block, the downsampled features are fed into a shortcut to the corresponding CS3T Block at the decoder for feature aggregation. Then the corresponding Expand Block upsamples the aggregated features for holistic CSI prediction.

2) Block Design

Based on the cross-shaped spatial-temporal attention, we construct the block using spatial self-attention, temporal self-attention, and residual connections. The overall structure of the block is shown in Fig. 8, where LN denotes layer normalization and MSA denotes multi-head self-attention. Note that the linear projection layer for the temporal attention is initialized as zero and then updated during the training process.

D. Network Architecture Design

The CS3T-UNet block maintains the dimensions of the feature. The merge block is responsible for downsampling spatial dimension and upsampling channel dimension, while the expand block has the reverse function. The CS3T-UNet block and the merge block constitute the CS3T-UNet encoder layer, while the CS3T-UNet block and the expand block constitute the CS3T-UNet decoder layer. Skip connections [16] are utilized to fuse the encoder layer features with the decoder layer's upsampled features for accurate high-resolution channel prediction.

IV. Evaluation

A. Evaluation Methodology

1) Dataset and Metric

To comprehensively demonstrate the performance of our proposed model, we adopt two simulated datasets in different scenarios, including different frequencies.

The first simulated CSI dataset is generated by DeepMIMO [20] , a widely used framework for generating large-scale MIMO datasets. The dataset is based on the outdoor scenario with a 3.5GHz central frequency. The BS has a uniform linear array (ULA) with $N_t = 64$ antennas, and each UE has $N_r = 1$ antenna. There are $N_f = 64$ subcarriers expanding 10MHz bandwidth. The dataset is composed of 10K samples in total, where each sample contains 20 frames. All the CSI raw data are preprocessed according to the method proposed in Section II. The whole CSI data has a shape of $\boldsymbol{H} \in \mathbb{R}^{10K\times20\times2\times64\times64}$. We randomly split 9K samples for the training and 1K for testing. We fix the number of historical time steps T as 10 and the number of future time steps L as 1 and 5, respectively. For different total time lengths, we use a $T+L$-length sliding window to slide along the 20 frames of each sample with stride 1 for data augmentation.

The second dataset is generated by QuaDriGa [22], a general channel simulator for sequential time-varying wireless channels. We set the scenario as 3GPP urban macro (UMa) NLOS. The carrier frequency f_c is set as 5.0 GHz. The number of sub-carrier N_c is set as 64, and the sub-carrier frequency spacing Δf is 30kHz. The BS is a ULA equipped with 64 antennas. The UE is equipped with an omnidirectional antenna. The moving speed of the UE is 5km/h. We generate the environment 100 times, each containing 100 UEs. Thus we generate 10k samples in total. 9k samples are used for training while the rest are used for testing.

After generating the raw CSI data, we convert the CSI data to the angular-delay domain according to Section II. To make the training smoother and converge faster, data normalization is adopted. The measured wireless channel could vary significantly in absolute values due to different clients' distances, environmental changes, user mobility, *etc*. According to [23], we care more about the relative scaling of MIMO CSI components than the absolute values for MIMO systems. Thus, we divide the previous timesteps and the predictions by their own maximum amplitude to make sure their complex values are in the range of [-1, 1]. The tanh activation function is attached to the outputs of all the models.

We statistic the multipath number of two datasets according to [15]. Fig. 9 shows the CDF of the multipath number of samples in the two datasets, which are mostly in the range of one to five.

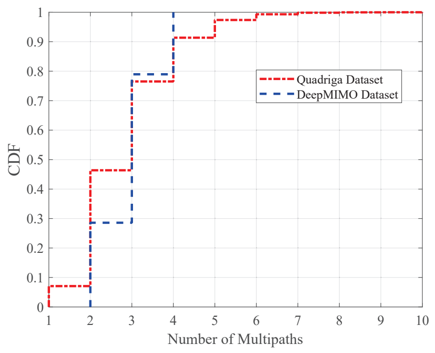


Fig. 9: Multipath statistics of the two datasets

To evaluate the performance of our proposed model, we adopt the commonly used normalized mean squared error (NMSE) as the metric, which is defined as follows:

$$NMSE = 10\log_{10} E\left[\frac{\|H-\hat{H}\|_F^2}{\|H\|_F^2}\right] \tag{10}$$

where $\|\cdot\|_F$ represents the Frobenius norm, H and $\hat{H}$ are the ground truth and predicted wireless channel, respectively.

2) Training Scheme and Hyperparameters

We implement the model with PyTorch on a server with one NVIDIA Tesla V100 card. We optimize the model using the Mean Squared Error (MSE) as the loss function. We adopt AdamW optimizer to train for 400 epochs. We first use ten epochs as the warm-up period where the learning rate gradually increases towards the initial learning rate for AdamW optimizer. We set batch size as 16 and learning rate as 2e-4 for DeepMIMO dataset, and batch size as 32 and learning rate as 2e-3 for QuaDriGa dataset. Without extra illustration, our model adopts an embedding dimension of 64, four encoder layers and four decoder layers, where each layer contains (2, 2, 6, 2) blocks. The patch size in the first layer is two to downsample the original feature at the beginning.

3) Baselines

We adopt typical types of models as baselines as follows:

1) Model-based approach. The autoregressive (AR) model calculates the next frame as a linear combination of the previous frames, and the formula is $H[n] = \sum_{t=1}^{T} a_t H[n-t+1] + a_0$, where a_t is the learnable weight for previous time steps. Thus, to predict multiple steps, AR needs to iteratively predict future frames based on previous predictions.
2) MLP-based approach. We also adopt one linear model to map the previous T steps to the next P steps. In this way, all the elements in different positions share the same parameter. The linear model is proved effective for long-time prediction in [24].
3) Pure CNN-based approach. SimVP [25] is a sequence prediction model built upon CNN. The model has been proven to perform better on video prediction tasks than transformer models. The key part of the model is com-

posed of a translator, which adopts multi-scale convolutional layers to extract different scales' data and uses the group convolutional layers to learn temporal correlation.

4) Convolutional LSTM model. Conv-CLSTM [26] adopts the convolutional long short-term memory (ConvLSTM) to exploit spatial and temporal correlations together. We use two layers of the ConvLSTM model followed by a convolutional layer. The model will iteratively output the predictions based on the previous steps' predictions.

5) DETR [27]. Unlike our model's architecture, DETR demonstrates one intuitive way of using the original Transformer model. The model first uses ResNet50 [28] as the feature extractor to extract the embedding of frames. The embedding dimension is set as 128. The sequential embeddings are fed into the transformer encoder and decoder to extract the temporal relationships. As DETR is designed for object detection while our task needs to output the same spatial shape as the input data, the decoder concatenate several ConvTransposed2D layers to recover the feature map back to the original size.

B. Overall Performance

In this section, we compare the performance and the computation requirements, including FLOPs and parameters of our model with the above-mentioned baselines. The results on DeepMIMO dataset and QuaDriGa dataset are shown in Table I. The results of DeepMIMO and QuaDriGa datasets demonstrate similar performance trends. Although AR and MLP models have extremely low computational requirements, they cannot achieve satisfactory results due to their limited capacity. The models cannot decrease early, especially in the DeepMIMO dataset. Conv-LSTM model performs better than AR and MLP models but suffers from a large computational burden. SimVP model as a non-Transformer model exhibits competitive performance and FLOPs. For the Transformer-based model, DETR proves that directly using the Transformer model cannot easily get good performance. It requires more computational power but easily gets overfitting, resulting in a worse performance than SimVP. With the separated spatial-temporal block design and the shortcut connection, our method demonstrates the best performance among all the baselines on both datasets, with at most 6.86dB improvement. It is worth noting that our method can achieve a lower NMSE for the next 5 timesteps prediction than the best baselines at 1 timestep prediction, which can significantly reduce the channel estimation frequency. In addition to the better performance, the FLOPs of CS3T-UNet is even slightly smaller than the non-Transformer SimVP model, indicating a better trade-off between computation cost and channel prediction accuracy.

C. Performance on Different Scenarios

1) Performance on Different Simulation Environments

To demonstrate the model's generalization performance, we directly test the trained model on different data distributions. We generate data on different conditions by QuaDriGa simulator, including a different center frequency at 3.5GHz, and a different scenario, *i.e.*, LOS scenario. The results shown in Table II have slight differences across different settings. Considering that our model is optimized for the NLOS scenario at 5.0GHz and unseen to other scenarios, the slight differences demonstrate the good generalizability of our model.

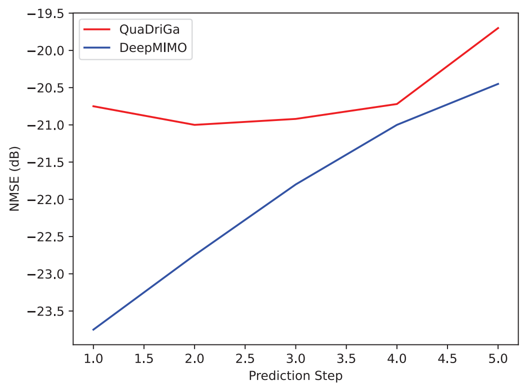


Fig. 10: Results of different timesteps of the two datasets.

2) Performance on Different Prediction Timesteps

We further elaborate on the degradation phenomenon between our and baseline models. We use the models that predict five timesteps ahead and separately calculate the NMSE of each time step. The results are shown in Figure 10. The error increase is predictable due to the longer prediction requirement. We could observe that the errors increase within 3dB with the number of prediction timesteps varying from one to five, which indicates a good trade-off between the accuracy and the number of prediction samples.

D. Impact of Masked Sparse Attention

In this section, we evaluate the performance of mask-guided sparse attention on our proposed model on the Quadriga dataset. We set the energy selection ratio λ as 0.99. Table III compares Masked CS3T and the vanilla CS3T model. The full test data is fed into the network. We can observe that the mask can further boost the performance of the CS3T over 3dB.

E. Ablation and Parameter Study

1) Impact of Different Transformer Block Types

In this section, we substitute the proposed CS3T block with other transformer blocks to study the impact of different Transformer blocks. We show the results to predict the next five steps in the two datasets and compare the FLOPs and model parameters. Note that in this section, we don't apply the mask on all models to demonstrate our proposed block's performance. Specifically, we adopt different types of blocks, including NoTF, which doesn't include any transformer blocks, Swin block [12], which uses the shifted window to calculate attention in the local area, and VideoSwin block [29], which expands the Swin block from 2D to 3D where the window attention includes both temporal and spatial dimension. Compared with VideoSwin-UNet and Swin-UNet, it proves that it is essential to consider the cross-shaped window and temporal correlations inside CSI samples. Compared our CS3T-UNet with VideoSwin-UNet, our method reduces the computation cost by over 3× while achieving a lower

TABLE I: Overall NMSE in dB on DeepMIMO and QuaDriGa datasets.
"**Bold**" and "Underline" format represents the best and second best results, respectively.

Type	Model	FLOPs	Params	NMSE (L=1)		NMSE (L=5)		NMSE (Average)	
				DeepMIMO	QuaDriGa	DeepMIMO	QuaDriGa	DeepMIMO	QuaDriGa
Non-Transformer	AR	90.11K	11	-0.20	-12.15	-0.10	-11.09	-0.15	-11.62
	Linear	406.91K	55	-0.21	-12.14	-0.11	-11.65	-0.16	-11.90
	Conv-LSTM	9.04G	157.65K	-16.32	-9.91	-12.05	-9.74	-14.20	-9.83
	SimVP	1.82G	1.69M	-20.38	-20.61	-15.67	-19.64	-18.03	-20.13
Transformer	DETR	3.04G	42.45M	-17.09	-5.91	-13.76	-6.51	-15.43	-6.21
	CS3T-UNet	1.61G	19.64M	**-26.05**	**-27.47**	**-21.58**	**-20.58**	**-23.82**	**-24.03**

TABLE II: Cross-scenario Test NMSE (L=1) in QuaDriGa.

Scenario	Center Frequency	NMSE (dB)
NLOS	5.0GHz	-27.47
LOS	5.0GHz	-27.99
NLOS	3.5GHz	-23.64

TABLE III: Impact of Masked Sparse Attention in QuaDriGa.

Selection method	NMSE (dB)		
	L=1	L=5	Average
CS3T-UNet (vanilla)	-27.47	-20.58	-24.03
Mask guided attention window selection	**-32.73**	**-21.88**	**-26.68**

NMSE error in both datasets, which validates the effectiveness of our method tailored for CSI prediction.

TABLE IV: NMSE (dB) of different Transformer blocks.

Model	FLOPs	Params	NMSE (L=5)	
			DeepMIMO	QuaDriGa
NoTF-UNet	0.10G	1.58M	-6.20	-12.83
Swin-UNet	0.72G	41.38M	-15.79	-17.71
VideoSwin-UNet	6.00G	55.56M	-19.75	-16.54
CS3T-NoUNet	1.59G	19.64M	-17.15	-16.60
CS3T-UNet	1.61G	19.64M	**-21.58**	**-20.58**

2) *Impact of Shortcuts*

We study the impact of shortcuts by removing the shortcuts between the encoder and decoder of our method, *i.e.*, CS3T-NoUNet in Table IV. It estimates future CSI only from the adjacent block, thus ignoring features from different scales. From the results (CS3T-UNet vs. CS3T-NoUNet), we can find that the performance with the shortcuts can be further improved with a slight FLOPs increase, which implicitly supports the necessity of embedding the physical prior of multi-resolution aggregation into the model design.

3) *Impact of embedding dimension*

In the above experiments, we adopt 64 as the default embedding dimension. In this section, we evaluate the impact of the embedding dimension by conducting experiments on the QuaDriGa dataset with a larger dimension of 128. The results in Table V show that the performance can be further improved with the dimension increase of the proposed method.

TABLE V: Impact of embedding dimensions on CS3T-UNet.

Embedding Dimension	NMSE (L=5)
64	-20.58
128	-22.57

TABLE VI: Inference time I (ms), prediction throughput W (SPS) and NMSE (dB) comparison in DeepMIMO.

Model	L=1			L=5		
	I	W	NMSE	I	W	NMSE
AR	0.19	5333	-0.20	0.39	12919.90	-0.10
SimVP	1.75	571.43	-20.38	1.83	2732.24	-15.67
Conv-CLSTM	2.12	471.70	-16.32	2.36	2118.64	-12.05
CS3T-UNet	1.86	537.63	**-26.05**	2.29	2183.41	**-21.58**

F. *Inference Time*

In this section, we compare the inference time of our model with the other baselines to show the practical usage of our model in real scenarios. The inference time starts from the time the BS obtain CSI to the time the GPU output the predicted CSI. We define the prediction throughput W as the steps predicted per second (SPS), and it is calculated as $W = \frac{L}{I}$, where L is the prediction steps, and I is the inference time. For practical usage, the prediction model must complete prediction within the channel coherence time (2.6 ms $\sim$ 43 ms [30] for velocity ranging from 3.6km/h to 60km/h) Thus, the throughput required by the coherence time varies from 23 to 388. We evaluate all our models on a Tesla V100 GPU. Table VI presents the inference time (I), prediction throughput (W), and the NMSE. We can see that the CS3T-UNet achieves the lowest error compared with the baselines in Table VI and can satisfy all the throughput requirements for one step prediction and five step prediction.

V. Discussion

A. Computational Cost

Currently, CS3T-UNet makes a better trade-off on accuracy and computational cost compared with other baselines as shown in Table I. Though CS3T-UNet can achieve a high prediction throughput on the NVIDIA Tesla V100 GPU platform in Table VI, reducing its requirement for ubiquitous deployment is always desirable. We envision that this can be further explored with lightweight design techniques such as model compression, pruning, distillation [31], *etc*.

B. General Adaptation

As the data distribution varies from different simulation methods, current deep CSI prediction systems including ours train a specialized model for each channel model, which may introduce performance differences. Though we have evaluated the generalizability by directly applying the model optimized under the 5.0GHz and NLoS scenario into the unseen data distributions, such as other frequencies and LoS scenarios, it is worth achieving a general model that is aware of and fits different channel conditions. In our future work, we will combine advanced domain generalization techniques [32] to enhance our system's generalizability and robustness.

C. Real World Deployment

CS3T-UNet is evaluated in multiple channel simulations, simulators, frequencies, and scenarios to validate its effectiveness and robustness. The current model design is compatible to different number of antennas. In the future, we plan to implement it with a prototype platform for further investigation on more realistic factors such as hardware imperfections, BS-UE asynchrony, and real-world CSI distributions. Signal distortion caused by the carrier frequency offset (CFO) and the hardware detection delay can be mitigated referring to [23]. In addition, the proposed method can be easily extended to uniform rectangular array (URA) where we only need to sequentially process the rows and columns of antenna array.

D. Knowledge Transfer across Tasks

In this work, we focus on dedicated design for the multi-step channel prediction problem with embedding CSI properties. To build a robust and intelligent wireless communication system, it is indispensable to combat other CSI-related challenges like channel compression [15, 17] and efficient beamforming [1]. From the intrinsic formulation of CSI data, we expect that mutual knowledge can be shared across CSI-related tasks. Thus we will explore the feasibility to transfer knowledge learned from one task to other CSI-related tasks with transfer learning and multi-task learning framework [33].

VI. Related Works

To alleviate the loss caused by the channel aging problem for massive MIMO communication, wireless channel prediction is promising to improve the trade-off between the accuracy and the frequency of CSI estimation. Channel prediction exploits the correlations inside the historically estimated CSI and intends to predict future channels ahead accurately.

Existing channel prediction approaches can be generally divided into two categories: model-based approaches and deep learning (DL) -based approaches. The model-based solutions mainly leverage physical characteristics to model the temporal correlations explicitly with a set of solvable parameters, including the linear extrapolation model [5, 34], the autoregressive (AR) model [35], and the sum-of-sinusoids model [7]. However, they are insufficient to characterize the evolution of wireless channels due to complicated realistic factors such as environmental noises, multipath effects and Doppler effects.

To enhance the capability of channel prediction methods, deep learning has been introduced for representation extraction from the past CSI samples with non-linear transformations. The basic DL-based methods include multi-layer perception (MLP) [36], convolutional neural networks (CNN) [21], recurrent neural networks (RNN) [9] and combinations of CNN-RNN [37, 38]. Though more powerful than pure-model based solutions, they all suffer from their intrinsic shortcomings: MLP lacks spatial sensitivity, CNN has a limited receptive field and RNN has propagation loss problems with gradient vanishing issues in multi-step prediction. With the advances of DL techniques, the Transformer [19] architecture leverages the attention mechanism with a global receptive field for better spatial-temporal feature extractions. The previous work [39] have demonstrated its effectiveness on CSI-related tasks. However, they directly use original Transformer models and need to transform each time step's input as a one-dimensional vector, thus breaking the spatial relationship of CSI samples. Different from them, our CS3T-UNet is tailored for channel prediction with the dedicated considerations on CSI properties such as spectral leakage due to the limited antennas and bandwidth, and feature consistency across spatial-temporal resolutions. Thus CS3T-UNet can achieve much better performance without introducing extra computation burdens in different channel conditions.

VII. Concluding Remarks

In this paper, we develop the cross-shaped separated spatial-temporal UNet-like Transformer (CS3T-UNet) for wireless channel prediction. The proposed method possesses an effective cross-shaped receptive field featuring wireless channel characteristics, shortcut connections to aggregate different scales' features, sufficient capability to capture complicated spatial-temporal relationships, and scalability for multi-step prediction. Extensive experiments on two simulated datasets demonstrate that our proposed method achieves better accuracy with a smaller computation requirement than the common baseline models.

VIII. Acknowledgement

This research is supported in part by RGC under Contract CERG 16204820, 16206122, R6021-20, AoE/E-601/22-R, Contract R8015, and 3030_006. We would like to thank the Turing AI Computing Cloud (TACC) [40] and HKUST iSING Lab for providing us computation resources on their platform.

References

[1] X. Li, S. Jin, H. A. Suraweera, J. Hou, and X. Gao, "Statistical 3-d beamforming for large-scale MIMO downlink systems over rician fading channels," *IEEE Trans. Commun.*, vol. 64, no. 4, pp. 1529–1543, 2016.

[2] C. Kong, C. Zhong, A. K. Papazafeiropoulos, M. Matthaiou, and Z. Zhang, "Sum-rate and power scaling of massive mimo systems with channel aging," *IEEE Transactions on Communications*, vol. 63, no. 12, pp. 4879–4893, 2015.

[3] A. Bakshi, Y. Mao, K. Srinivasan, and S. Parthasarathy, "Fast and efficient cross band channel prediction using machine learning," in *MobiCom*. ACM, 2019, pp. 37:1–37:16.

[4] H. Yin, H. Wang, Y. Liu, and D. Gesbert, "Addressing the curse of mobility in massive mimo with prony-based angular-delay domain channel predictions," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 12, pp. 2903–2917, 2020.

[5] H. P. Bui, Y. Ogawa, T. Nishimura, and T. Ohgane, "Performance evaluation of a multi-user mimo system with prediction of time-varying indoor channels," *IEEE Trans. Antennas Propag.*, vol. 61, no. 1, pp. 371–379, 2012.

[6] K. E. Baddour and N. C. Beaulieu, "Autoregressive modeling for fading channel simulation," *IEEE Transactions on Wireless Communications*, vol. 4, no. 4, pp. 1650–1662, 2005.

[7] I. C. Wong and B. L. Evans, "Joint channel estimation and prediction for OFDM systems," in *GLOBECOM*. IEEE, 2005.

[8] Z. Wen, R. He, B. Ai, C. Huang, M. Yang, and Z. Zhong, "A scheme of channel prediction based on artificial neural network," *arXiv preprint arXiv:2111.15476*, 2021.

[9] M. K. Shehzad, L. Rose, S. Wesemann, and M. Assaad, "Ml-based massive MIMO channel prediction: Does it work on real-world data?" *IEEE Wirel. Commun. Lett.*, vol. 11, no. 4, pp. 811–815, 2022.

[10] T. Zhou, H. Zhang, B. Ai, C. Xue, and L. Liu, "Deep-learning-based spatial–temporal channel prediction for smart high-speed railway communication networks," *IEEE Transactions on Wireless Communications*, vol. 21, no. 7, pp. 5333–5345, 2022.

[11] X. Ding, X. Zhang, J. Han, and G. Ding, "Scaling up your kernels to 31x31: Revisiting large kernel design in cnns," in *CVPR*. IEEE, 2022, pp. 11 963–11 975.

[12] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *ICCV*, 2021.

[13] X. Dong, J. Bao, D. Chen, W. Zhang, N. Yu, L. Yuan, D. Chen, and B. Guo, "Cswin transformer: A general vision transformer backbone with cross-shaped windows," in *CVPR*. IEEE, 2022, pp. 12 114–12 124.

[14] H. Jiang, M. Cui, D. W. K. Ng, and L. Dai, "Accurate channel prediction based on transformer: Making mobility negligible," *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 9, pp. 2717–2732, 2022.

[15] Q. Hu, H. Kang, H. Chen, Q. Huang, Q. Zhang, and M. Cheng, "Csi-stripeformer: Exploiting stripe features for csi compression in massive mimo system," in *INFOCOM*. IEEE, 2023.

[16] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *MICCAI (3)*, ser. Lecture Notes in Computer Science, vol. 9351. Springer, 2015, pp. 234–241.

[17] Z. Lu, J. Wang, and J. Song, "Multi-resolution CSI feedback with deep learning in massive MIMO system," in *ICC*. IEEE, 2020, pp. 1–6.

[18] D. Vasisht, S. Kumar, H. Rahul, and D. Katabi, "Eliminating channel feedback in next-generation cellular networks," in *SIGCOMM*. ACM, 2016, pp. 398–411.

[19] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *NIPS*, 2017, pp. 5998–6008.

[20] A. Alkhateeb, "DeepMIMO: A generic deep learning dataset for millimeter wave and massive MIMO applications," in *Proc. of ITA*, San Diego, CA, Feb 2019, pp. 1–8.

[21] C. Wu, X. Yi, Y. Zhu, W. Wang, L. You, and X. Gao, "Channel prediction in high-mobility massive MIMO: from spatio-temporal autoregression to deep learning," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 7, pp. 1915–1930, 2021.

[22] S. Jaeckel, L. Raschkowski, K. Börner, and L. Thiele, "Quadriga: A 3-d multi-cell channel model with time evolution for enabling virtual field trials," *IEEE Trans. Antennas Propag.*, vol. 62, no. 6, pp. 3242–3256, 2014.

[23] Z. Liu, G. Singh, C. Xu, and D. Vasisht, "FIRE: enabling reciprocity for FDD MIMO systems," in *MobiCom*. ACM, 2021, pp. 628–641.

[24] A. Zeng, M. Chen, L. Zhang, and Q. Xu, "Are transformers effective for time series forecasting?" in *AAAI*, vol. 37, no. 9, 2023, pp. 11 121–11 128.

[25] Z. Gao, C. Tan, L. Wu, and S. Z. Li, "Simvp: Simpler yet better video prediction," in *CVPR*. IEEE, 2022, pp. 3160–3170.

[26] X. Shi, Z. Chen, H. Wang, D. Yeung, W. Wong, and W. Woo, "Convolutional LSTM network: A machine learning approach for precipitation nowcasting," in *NIPS*, 2015, pp. 802–810.

[27] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *ECCV*. Springer, 2020, pp. 213–229.

[28] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*. IEEE, 2016, pp. 770–778.

[29] Z. Liu, J. Ning, Y. Cao, Y. Wei, Z. Zhang, S. Lin, and H. Hu, "Video swin transformer," in *CVPR*. IEEE, 2022, pp. 3192–3201.

[30] E. G. Larsson, "Fundamentals of massive mimo," in *International Workshop on Signal Processing Advances in Wireless Communications*, 2016.

[31] K. T. Chitty-Venkata, S. Mittal, M. Emani, V. Vishwanath, and A. K. Somani, "A survey of techniques for optimizing transformer inference," *CoRR*, vol. abs/2307.07982, 2023.

[32] K. Zhou, Z. Liu, Y. Qiao, T. Xiang, and C. C. Loy, "Domain generalization: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 4, pp. 4396–4415, 2023.

[33] Y. Zhang and Q. Yang, "A survey on multi-task learning," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 12, pp. 5586–5609, 2022.

[34] H. Yin, H. Wang, Y. Liu, and D. Gesbert, "Addressing the curse of mobility in massive MIMO with prony-based angular-delay domain channel predictions," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 12, pp. 2903–2917, 2020.

[35] K. E. Baddour and N. C. Beaulieu, "Autoregressive modeling for fading channel simulation," *IEEE Trans. Wirel. Commun.*, vol. 4, no. 4, pp. 1650–1662, 2005.

[36] H. Kim, S. Kim, H. Lee, C. Jang, Y. Choi, and J. Choi, "Massive MIMO channel prediction: Kalman filtering vs. machine learning," *IEEE Trans. Commun.*, vol. 69, no. 1, pp. 518–528, 2021.

[37] T. Zhou, H. Zhang, B. Ai, C. Xue, and L. Liu, "Deep-learning-based spatial-temporal channel prediction for smart high-speed railway communication networks," *IEEE Trans. Wirel. Commun.*, vol. 21, no. 7, pp. 5333–5345, 2022.

[38] J. Yuan, H. Q. Ngo, and M. Matthaiou, "Machine learning-based channel prediction in massive MIMO with channel aging," *IEEE Trans. Wirel. Commun.*, vol. 19, no. 5, pp. 2960–2973, 2020.

[39] H. Jiang, M. Cui, D. W. K. Ng, and L. Dai, "Accurate channel prediction based on transformer: Making mobility negligible," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 9, pp. 2717–2732, 2022.

[40] K. Xu, X. Wan, H. Wang, Z. Ren, X. Liao, D. Sun, C. Zeng, and K. Chen, "Tacc: A full-stack cloud computing infrastructure for machine learning tasks," *arXiv preprint arXiv:2110.01556*, 2021. [Online]. Available: https://arxiv.org/abs/2110.01556