CSI Prediction Using Diffusion Models

Mehdi Sattari, *Graduate Student Member, IEEE,* Javad Aliakbari, *Graduate Student Member, IEEE,* Alexandre Graell i Amat, *Senior Member, IEEE,* and Tommy Svensson, *Senior Member, IEEE*

***Abstract*—Acquiring accurate channel state information (CSI) is critical for reliable and efficient wireless communication, but challenges such as high pilot overhead and channel aging hinder timely and accurate CSI acquisition. CSI prediction, which forecasts future CSI from historical observations, offers a promising solution. Recent deep learning approaches, including recurrent neural networks and Transformers, have achieved notable success but typically learn deterministic mappings, limiting their ability to capture the stochastic and multimodal nature of wireless channels. In this paper, we introduce a novel probabilistic framework for CSI prediction based on diffusion models, offering a flexible design that supports integration of diverse prediction schemes. We decompose the CSI prediction task into two components: a temporal encoder, which extracts channel dynamics, and a diffusion-based generator, which produces future CSI samples. We investigate two inference schemes—autoregressive and sequence-to-sequence—and explore multiple diffusion backbones, including U-Net and Transformer-based architectures. Furthermore, we examine a diffusion-based approach without an explicit temporal encoder and utilize the DDIM scheduling to reduce model complexity. Extensive simulations demonstrate that our diffusion-based models significantly outperform state-of-the-art baselines.**

***Index Terms*—CSI prediction, Deep learning, Diffusion models, MIMO.**

I. Introduction

Multiple-input multiple-output (MIMO) technology plays a pivotal role in advanced wireless communications. To fully exploit the benefits of MIMO systems, accurate acquisition of channel state information (CSI) is essential for designing efficient precoding and combining algorithms. However, CSI acquisition is challenged by high overhead, channel aging, and significant computational complexity. To address these issues, a variety of algorithms have been proposed to obtain CSI with reduced overhead and complexity [1]–[5].

CSI prediction refers to the task of forecasting future CSI based on historical observations. Various classical approaches have been proposed, including linear extrapolation [6], Kalman filtering [7], sum-of-sinusoids models [8], Gaussian processes [9], predictor antenna [10], and autoregressive (AR) models [11]. However, the underlying statistics of wireless channels are inherently complex and difficult to model accurately, making reliable CSI prediction particularly challenging. As a result, traditional methods often fail to deliver satisfactory performance, especially under highly dynamic channel conditions.

Deep neural networks (DNNs) have been increasingly adopted in wireless communication tasks. Motivated by their significant success in time-series forecasting problems [12]–[16], the use of DNNs for CSI prediction has attracted growing research interest [17]–[31]. Various neural network (NN) architectures have been explored to capture the temporal dynamics of CSI, including recurrent neural networks (RNNs), such as long short-term memory (LSTM) and gated recurrent units (GRUs), as well as Transformer-based models. RNN-based architectures have been successfully employed for CSI prediction in [20], [21], [24], while Transformer-based approaches have been investigated in [25], [26], showing superior performance in modeling long-range temporal dependencies.

Accurately predicting CSI is a fundamentally challenging problem due to the highly stochastic nature of wireless channels and their complex underlying distributions. Future channel states are influenced by a wide range of factors, including multipath fading, user mobility, and environmental dynamics, which together lead to significant uncertainty in CSI evolution. Deep learning models such as RNNs and Transformers have shown promising results for time-series forecasting, but they primarily learn a deterministic mapping from historical CSI to future CSI. This approach often fails to capture the inherent randomness and multimodality of wireless channel evolution. In contrast, recent advances in generative artificial intelligence (AI) have enabled the development of models that learn the full underlying probability distribution of data. Among these, diffusion models [32] have achieved state-of-the-art performance in high-dimensional data generation tasks across various modalities. By modeling CSI prediction as a probabilistic generative process, diffusion models can naturally account for uncertainty in channel evolution while learning the joint spatiotemporal structure of CSI sequences.

Motivated by the success of recent advances of diffusion models in modeling complex data distributions and the limitations of RNN- and Transformer-based approaches in CSI prediction, this paper explores the application of diffusion models to the CSI prediction problem. The main contributions of this work are summarized as follows:

- We propose, for the first time, a diffusion-based framework for CSI prediction. The framework offers a flexible design that enables seamless integration of diverse CSI prediction models. We formulate the CSI prediction task as two components: temporal encoding and generative

This work was supported in part by the SEMANTIC project, funded by the EU's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No. 861165, and in part by the Hexa-X-II project, which has received funding from the Smart Networks and Services Joint Undertaking (SNS JU) under the European Union's Horizon Europe research and innovation programme under grant agreement No. 101095759. This work was also partially supported by the Swedish Research Council (VR) under grants 2020-03687 and 2023-05065, by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation, and by the Swedish Innovation Agency (Vinnova) under grant 2022-03063. The computations were enabled by resources provided by the National Academic Infrastructure for Supercomputing in Sweden (NAISS), partially funded by the Swedish Research Council through grant agreement No. 2022-06725.

sampling. The temporal encoder extracts latent representations of the channel dynamics, while the diffusion generator produces the next CSI frame conditioned on these latent features.

- We explore two inference schemes: AR and sequence-to-sequence (seq2seq). The AR strategy recursively predicts future CSI frames, enabling a single trained model to flexibly adapt to arbitrary context lengths and prediction horizons. In contrast, the seq2seq scheme enables parallel prediction of multiple future CSI frames in a single pass, albeit with reduced flexibility in handling variable context lengths and prediction horizons.
- To reduce complexity for real-time applications, we investigate a simplified design in which the diffusion model directly learns temporal and spatial dependencies without an explicit temporal encoder. Furthermore, to enhance sampling efficiency and stability, we adopt the denoising diffusion implicit model (DDIM) [33] scheduling technique. Our experiments show that as few as three sampling steps are sufficient to achieve competitive performance.
- We evaluate multiple generator backbones, including U-Net [34], 3D U-Net [35], and diffusion Transformer (DiT) [36], in combination with temporal encoders such as ConvLSTM [37] and LinFormer [26]. Extensive simulation results show that our diffusion-based models significantly outperform state-of-the-art baselines and exhibit strong generalization to unseen wireless environments.

The rest of this paper is organized as follows: Section II introduces the channel model and CSI prediction problem, as well as a preliminary for diffusion models. In Section III, we present our diffusion-based CSI prediction framework and training and inference procedures. Section IV provides numerical simulations and compares different CSI prediction schemes in terms of the normalized mean squared error (NMSE). Finally, Section V concludes the paper.

II. Preliminaries and Problem Formulation

A. Channel Model

We focus on the clustered delay line (CDL) channel model based on the 3GPP specification [38]. The CDL channel model is a spatial channel model for wireless channel simulation. This model comprises different scenarios, denoted as `CDL-A` to `CDL-E`, where each scenario models different channel statistics with different delay profiles, angular spreads, etc. For example, `CDL-B` generates an urban macrocell scenario with a wider angular spread. The time-varying MIMO channel impulse response in the CDL model is given by

$$\mathbf{H}(t) = \sum_{l=1}^{L}\sum_{r=1}^{R} \alpha_{l,r}(t)\, \mathbf{a}_{\text{Rx}}\left(\phi_{l,r}^{\text{Rx}}\right)\, \mathbf{a}_{\text{Tx}}^{H}\left(\phi_{l,r}^{\text{Tx}}\right)\, e^{-j2\pi f \tau_l}, \quad (1)$$

or its discrete counterpart,

$$\mathbf{H}_n = \sum_{l=1}^{L}\sum_{r=1}^{R} \alpha_{l,r}[n]\, \mathbf{a}_{\text{Rx}}\left(\phi_{l,r}^{\text{Rx}}\right)\, \mathbf{a}_{\text{Tx}}^{H}\left(\phi_{l,r}^{\text{Tx}}\right)\, e^{-j2\pi \frac{k f_s}{N_c}\tau_l}, \quad (2)$$

where, L is the number of clusters, R the number of rays per cluster, $\alpha_{l,r}(t)$ time-varying complex path gain for ray r in cluster l, $\mathbf{a}_{\text{Rx}}(\cdot), \mathbf{a}_{\text{Tx}}(\cdot)$ denote receive/transmit array response vectors, $(\phi^{\text{Rx}}, \phi^{\text{Tx}})$ represent angles of arrival/departure, f_s the sampling frequency, N_c the number of subcarriers, and τ_l is the delay of the l-th cluster. The time-varying complex path gain is modeled as

$$\alpha_{l,r}[n] = \sqrt{P_{l,r}}\, e^{j(2\pi f_{l,r} n + \phi_{l,r})}, \quad (3)$$

where $P_{l,r}$ is the power of the r-th ray in cluster l, $f_{l,r}$ the Doppler shift, and $\phi_{l,r}$ the random initial phase. For a uniform linear array (ULA), the array response vector is

$$\mathbf{a}(\phi) = \frac{1}{\sqrt{N_{\text{t}}}} \left[1,\, e^{j2\pi \frac{d}{\lambda}\sin(\phi)},\, \ldots,\, e^{j2\pi \frac{d}{\lambda}(N_{\text{t}}-1)\sin(\phi)}\right]^T, \quad (4)$$

where, N_{t} denotes the number of antenna elements and d the antenna spacing.

We consider a MIMO system in which a base station (BS) serves multiple single-antenna users. The BS is equipped with a ULA of N_{t} antennas. Orthogonal frequency division multiplexing (OFDM) with N_{c} subcarriers is employed for downlink transmission. For each subcarrier $m \in \{1, \ldots, N_{\text{c}}\}$ and time index n, $\mathbf{h}_{m,n} \in \mathbb{C}^{N_{\text{t}}}$ denotes the channel vector from the BS with N_{t} antennas to the user. By stacking all subcarriers, the downlink CSI matrix in the spatial–frequency domain at time n is given by

$$\mathbf{H}_n = \left[\mathbf{h}_{1,n}\ \mathbf{h}_{2,n}\ \cdots\ \mathbf{h}_{N_{\text{c}},n}\right] \in \mathbb{C}^{N_{\text{t}} \times N_{\text{c}}}. \quad (5)$$

B. Diffusion Models

a) Denoising diffusion probabilistic models (DDPMs): Diffusion models are a class of generative models that learn to approximate complex data distributions by progressively corrupting the data with noise and then reversing this process. The foundational work on DDPM was introduced in [32]. In DDPMs, the forward process gradually adds Gaussian noise to the data, while the reverse process aims to reconstruct the original data using a NN. This is modeled as a Markov chain with a predefined, time-dependent noise schedule.

The forward process defines a Markov chain that incrementally perturbs the data over T time steps. Starting from a data sample $\mathbf{H}^0 \sim p(\mathbf{H}^0)$, the forward diffusion is expressed as

$$q(\mathbf{H}^{1:T} \mid \mathbf{H}^0) = \prod_{t=1}^{T} q(\mathbf{H}^t \mid \mathbf{H}^{t-1}), \quad (6)$$

where each conditional distribution is Gaussian

$$q(\mathbf{H}^t \mid \mathbf{H}^{t-1}) = \mathcal{N}\left(\mathbf{H}^t; \sqrt{1-\beta_t}\, \mathbf{H}^{t-1}, \beta_t\, \mathbf{I}\right), \quad (7)$$

with $\beta_t \in (0,1)$ denoting the noise variance at step t, and $\mathbf{H}^t$ representing the noisy latent at time t.

The marginal distribution at an arbitrary time step t, conditioned directly on the original sample $\mathbf{H}^0$, is

$$q(\mathbf{H}^t \mid \mathbf{H}^0) = \mathcal{N}\left(\mathbf{H}^t; \sqrt{\bar{\alpha}_t}\, \mathbf{H}^0, (1-\bar{\alpha}_t)\, \mathbf{I}\right), \quad (8)$$

which leads to the reparameterization

$$\mathbf{H}^t = \sqrt{\bar{\alpha}_t}\, \mathbf{H}^0 + \sqrt{1-\bar{\alpha}_t}\, \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \quad (9)$$

where $\boldsymbol{\epsilon}$ denotes standard Gaussian noise, and

$$\bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i, \quad \text{with} \quad \alpha_t \triangleq 1 - \beta_t. \tag{10}$$

The noise schedule is crafted such that $\bar{\alpha}_T \approx 0$, ensuring that the final latent $\mathbf{H}^T$ is nearly standard Gaussian.

The reverse process is a learned Markov chain that denoises $\mathbf{H}^T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ back to the data domain, and each reverse transition is approximated as

$$p_{\boldsymbol{\theta}}(\mathbf{H}^{t-1} \mid \mathbf{H}^t) = \mathcal{N}\left(\mathbf{H}^{t-1}; \boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{H}^t, t), \boldsymbol{\Sigma}_{\boldsymbol{\theta}}(t)\right), \tag{11}$$

with the prior given by

$$p(\mathbf{H}^T) = \mathcal{N}(\mathbf{H}^T; \mathbf{0}, \mathbf{I}). \tag{12}$$

In DDPM, the mean $\boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{H}^t, t)$ is expressed using a predicted noise function $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}$

$$\boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{H}^t, t) = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{H}^t - \frac{1 - \alpha_t}{\sqrt{1 - \bar{\alpha}_t}} \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{H}^t, t) \right), \tag{13}$$

where $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{H}^t, t)$ is an NN trained to predict the forward noise at time t. The variance $\boldsymbol{\Sigma}_{\boldsymbol{\theta}}(t)$ is usually fixed and time-dependent

$$\boldsymbol{\Sigma}_{\boldsymbol{\theta}}(t) = \tilde{\beta}_t \mathbf{I}, \quad \text{with} \quad \tilde{\beta}_t = \frac{1 - \bar{\alpha}_{t-1}}{1 - \bar{\alpha}_t} \beta_t. \tag{14}$$

The model is trained by minimizing a simplified variational lower bound, which reduces to a weighted mean squared error (MSE) between the true noise $\boldsymbol{\epsilon}$ and the predicted noise $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}$

$$\mathcal{L}_{\text{noise}} = \mathbb{E}_{\mathbf{H}^0, t, \boldsymbol{\epsilon}} \left[\left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{H}^t, t) \right\|^2 \right]. \tag{15}$$

An alternative yet equivalent training objective is to have the model directly predict the clean data $\mathbf{H}^0$ from the noisy observation $\mathbf{H}^t$. In this case, the training loss becomes the MSE between the true clean sample and the predicted sample $\mathbf{H}^0_{\boldsymbol{\theta}}$

$$\mathcal{L}_{\text{data}} = \mathbb{E}_{\mathbf{H}^0, t, \boldsymbol{\epsilon}} \left[\left\| \mathbf{H}^0 - \mathbf{H}^0_{\boldsymbol{\theta}}(\mathbf{H}^t, t) \right\|^2 \right]. \tag{16}$$

This formulation is particularly useful in inverse problems, such as denoising or super-resolution, where the objective is to recover a clean signal rather than generate diverse samples. Both loss functions are mathematically connected through the forward diffusion equation, and the predicted clean sample can be converted to noise (or vice versa) using closed-form expressions.

b) DDIM: The DDPM framework involves a stochastic reverse process and often requires a large number of diffusion steps (e.g., $T = 1000$) to achieve high-quality generation. To reduce the number of steps without significant degradation in quality, Song *et al.* [33] proposed DDIM, which reinterpret the reverse process as a non-Markovian mapping that can be made fully deterministic. Starting from the DDPM estimate of the clean sample $\hat{\mathbf{H}}^0$ at time step t

$$\hat{\mathbf{H}}^0 = \frac{1}{\sqrt{\bar{\alpha}_t}} \left(\mathbf{H}^t - \sqrt{1 - \bar{\alpha}_t}\, \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{H}^t, t) \right), \tag{17}$$

and the DDIM update is given by

$$\mathbf{H}^{t-1} = \sqrt{\bar{\alpha}_{t-1}}\, \hat{\mathbf{H}}^0 + \sqrt{1 - \bar{\alpha}_{t-1} - \sigma_t^2}\, \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{H}^t, t) + \sigma_t\, \boldsymbol{\epsilon}, \tag{18}$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$ and

$$\sigma_t = \zeta \cdot \sqrt{\frac{1 - \bar{\alpha}_{t-1}}{1 - \bar{\alpha}_t}} \cdot \sqrt{1 - \frac{\bar{\alpha}_t}{\bar{\alpha}_{t-1}}}, \tag{19}$$

where $\zeta \in [0, 1]$ controls the level of stochasticity, $\zeta = 0$ yields a fully deterministic process, while $\zeta = 1$ recovers the stochastic DDPM update. This formulation eliminates the need to sample fresh Gaussian noise at each step in the deterministic setting ($\zeta = 0$), enabling fast sampling and allowing for arbitrary step schedules. By tuning ζ, DDIM provides a trade-off between sample diversity and generation speed.

C. Problem Formulation

Time-varying wireless channels can be naturally modeled as time-series data, where CSI prediction entails capturing both the spatial distributions and temporal dynamics of the channel. Under the MSE criterion, the optimal predictor corresponds to the minimum mean squared error (MMSE) estimate, which can be formulated as

$$f^*(\mathbf{H}_\text{p}) = \arg\min_{f} \ \mathbb{E}\left[\| \mathbf{H}_\text{f} - f(\mathbf{H}_\text{p}) \|^2 \right], \tag{20}$$

where $\mathbf{H}_\text{f} \triangleq \{\mathbf{H}_{n+1}, \mathbf{H}_{n+2}, \ldots, \mathbf{H}_{n+N_\text{f}}\}$ denotes the set of future CSI over the next N_f time steps, and $\mathbf{H}_\text{p} \triangleq \left\{\mathbf{H}_{n-N_\text{p}+1}, \mathbf{H}_{n-N_\text{p}+2}, \ldots, \mathbf{H}_n\right\}$ represents the historical CSI observations over the previous N_p time steps. The optimal predictor $f^*(\mathbf{H}_\text{p})$ is the conditional mean estimator (CME) [39], given by

$$\begin{aligned} f^*(\mathbf{H}_\text{p}) &= \mathbb{E}[\mathbf{H}_\text{f} \mid \mathbf{H}_\text{p}] \\ &= \int \mathbf{H}_\text{f}\, p(\mathbf{H}_\text{f} \mid \mathbf{H}_\text{p})\, d\mathbf{H}_\text{f}. \end{aligned} \tag{21}$$

Deriving the optimal MMSE predictor requires knowledge of both the conditional distribution of the future CSI given the past (capturing the channel dynamics) and the prior distribution of the wireless channel (capturing the spatial statistics). Deriving this predictor in closed form is intractable due to high-dimensional integrals, nonlinear time-varying dynamics, and the lack of a known prior for practical channels.

A practical approach to approximate the optimal predictor is to employ discriminative DNNs, such as RNNs or Transformers. A discriminative DNN with learnable parameters $\boldsymbol{\theta}$ aims to learn the nonlinear mapping from historical CSI observations, $\mathbf{H}_\text{p}$, to the future CSI, $\mathbf{H}_\text{f}$. With sufficient data and model capacity, a DNN trained using the MSE loss provides a data-driven approximation of the MMSE predictor, i.e.,

$$f_{\boldsymbol{\theta}}(\mathbf{H}_\text{p}) \approx f^*(\mathbf{H}_\text{p}). \tag{22}$$

However, such approximations face several limitations. First, jointly capturing the temporal dynamics of the wireless channel together with its spatial distribution is highly challenging. Second, discriminative models provide a deterministic mapping that fails to capture the uncertainty of channel evolution. Third, learning such complex dynamics from finite training data makes these models prone to overfitting, thereby limiting their robustness in practical deployments.

Algorithm 1 Training

```
Input: Dataset $\mathcal{D} = \{(\mathbf{X}, \mathbf{Y})\}$; model parameters $\boldsymbol{\theta} = [\boldsymbol{\theta}_{\mathrm{TE}}, \boldsymbol{\theta}_{\mathrm{G}}]$; diffusion scheduler (noise schedule $\{\beta_t\}_{t=1}^{T}$); diffusion steps $T$; SNR range $[\rho_{\min}, \rho_{\max}]$.
Output: Trained model parameters $\boldsymbol{\theta}^*$.
 1: for epoch $= 1, \ldots, N_{\text{epochs}}$ do
 2:     for each batch $(\mathbf{X}, \mathbf{Y}) \in \mathcal{D}$ do
 3:         $\rho \sim \mathcal{U}[\rho_{\min}, \rho_{\max}]$
 4:         $\tilde{\mathbf{X}} \leftarrow \sqrt{\rho}\mathbf{X} + \mathbf{N}, \quad \mathbf{N} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
 5:         $\mathbf{Z} \leftarrow f_{\boldsymbol{\theta}_{\mathrm{TE}}}(\tilde{\mathbf{X}})$
 6:         $t \sim \mathcal{U}[0, T-1]$
 7:         $\mathbf{X}^t \leftarrow \sqrt{\bar{\alpha}_t}\,\mathbf{Y} + \sqrt{1-\bar{\alpha}_t}\,\boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
 8:         $\mathbf{Y}_{\text{diff}} \leftarrow \text{Concat}(\mathbf{X}^t, \mathbf{Z})$
 9:         $\hat{\mathbf{Y}} \leftarrow f_{\boldsymbol{\theta}_{\mathrm{G}}}(\mathbf{Y}_{\text{diff}}, t)$
10:         $\mathcal{L}(\theta) \leftarrow \text{Loss}(\hat{\mathbf{Y}}, \mathbf{Y})$
11:         Update $\boldsymbol{\theta}$ using gradients $\nabla_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\theta})$
12:     end for
13: end for
14: Return $\boldsymbol{\theta}^*$
```

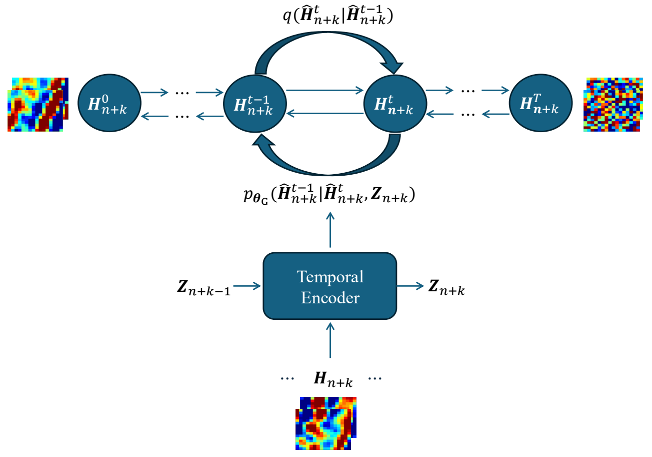
$q(\hat{\boldsymbol{H}}_{n+k}^{t}|\hat{\boldsymbol{H}}_{n+k}^{t-1})$
$\boldsymbol{H}_{n+k}^{0}$
$\boldsymbol{H}_{n+k}^{t-1}$
$\boldsymbol{H}_{n+k}^{t}$
$\boldsymbol{H}_{n+k}^{T}$
$p_{\boldsymbol{\theta}_{\mathrm{G}}}(\hat{\boldsymbol{H}}_{n+k}^{t-1}|\hat{\boldsymbol{H}}_{n+k}^{t}, \boldsymbol{Z}_{n+k})$
$\boldsymbol{Z}_{n+k-1}$
Temporal Encoder
$\boldsymbol{Z}_{n+k}$
$\boldsymbol{H}_{n+k}$

Fig. 1: CSI prediction with a diffusion model conditioned on the temporal encoder's latent representation, using an AR inference strategy.

III. Diffusion CSI Prediction

Inspired by the formulation of the optimal CME in (21), we decompose the prediction task into two components: a temporal encoder and a generator. The temporal encoder extracts latent temporal representations from the time-varying channel, while the generator learns the underlying channel distribution. To capture the probabilistic nature of CSI prediction, we employ diffusion models as the generator. In contrast to discriminative DNNs that produce deterministic point estimates, diffusion models yield probabilistic predictions, thereby accounting for the inherent uncertainty of wireless channels. Furthermore, since diffusion models are trained across the entire signal-to-noise ratio (SNR) range, they are expected to generalize more robustly to diverse channel conditions compared to task-specific discriminative models.

We apply two different inference schemes: the first follows an AR approach, while the second adopts a seq2seq prediction framework. In the following sections, we elaborate on these prediction schemes and summarize the overall methodology in algorithmic formulations. The overall training procedure of our diffusion-based CSI predictor, which leverages latent representations from the temporal encoder, is outlined in Algorithm 1. Note that the training process is identical for both AR and seq2seq schemes, except for the output target: in the AR scenario, the model is trained to generate the next CSI frame, whereas in the seq2seq case, it is trained to produce a sequence of CSI samples over a fixed prediction horizon.

A. AR

AR inference is a sequential prediction strategy in which each future time step is generated conditioned on the previously observed or predicted steps. In this scheme, the model takes a historical sequence, predicts the next step, and then recursively feeds this prediction back to forecast subsequent steps until the desired horizon is reached. The AR inference scheme provides a flexible prediction framework, where a single trained model can operate with arbitrary context lengths and prediction horizons. This flexibility makes it well-suited for sequential modeling tasks, as the model dynamically adapts its predictions based on evolving inputs. The overall scheme of the AR CSI prediction framework is depicted in Fig. 1.

a) Temporal Encoding: Learning the dynamics of CSI data is a crucial component of the CSI prediction problem. To capture the complex nature of a time-varying wireless channel, we employ a learning model to extract latent features along the temporal dimension. Let the CSI sequence input to the temporal encoder at the k^{th} time step be

$$\mathbf{H}_{n+k} = \{\mathbf{H}_{n-N_{\mathrm{p}}+1:n}, \hat{\mathbf{H}}_{n+1:n+k-1}\}, \tag{23}$$

where $\hat{\mathbf{H}}_{n+1:n+k-1}$ denotes the predicted CSI sequence up to step k. The temporal encoder processes this input at the k^{th} step and extracts the corresponding latent temporal representation,

$$\mathbf{Z}_{n+k} = f_{\boldsymbol{\theta}_{\mathbf{TE}}}(\mathbf{H}_{n+k}). \tag{24}$$

The resulting latent representation is then used as a conditioning input to the diffusion model for CSI prediction.

b) Generator: To generate future CSI samples from the latent features extracted by the temporal encoder, we employ a diffusion model. Conditioning on the latent features in the reverse diffusion process steers the generation toward plausible future CSI realizations. The forward process follows the Markov transition in (9), where Gaussian noise is progressively added according to a predefined schedule. To approximate the reverse process, an NN is trained to approximate the conditional distribution

$$f_{\boldsymbol{\theta}_{\mathrm{G}}}(\hat{\mathbf{H}}_{n+k}^{t}, \mathbf{Z}_{n+k}) \approx p_{\boldsymbol{\theta}_{\mathrm{G}}}(\hat{\mathbf{H}}_{n+k}^{t-1} \mid \hat{\mathbf{H}}_{n+k}^{t}, \mathbf{Z}_{n+k}), \tag{25}$$

and after T sampling steps, the predicted CSI sample is

$$\hat{\mathbf{H}}_{n+k} = \hat{\mathbf{H}}_{n+k}^{0}. \tag{26}$$

Inference: In the inference step, after predicting the next CSI frame, the generated sample is appended to the input

Algorithm 2 AR Inference

Input: Historical CSI sequence $\mathbf{H}_\mathrm{p}$; trained parameters $\boldsymbol{\theta}^* = [\boldsymbol{\theta}^*_\mathrm{TE}, \boldsymbol{\theta}^*_\mathrm{G}]$; diffusion scheduler with noise schedule $\{\beta_t\}_{t=1}^T$; diffusion steps T; prediction horizon N_f.

Output: Predicted CSI sequence $\{\hat{\mathbf{H}}_n\}_{n=1}^{N_\mathrm{f}}$.

```
1:  Initialize $\mathbf{H}_c \leftarrow \mathbf{H}_\mathrm{p}$
2:  for $n = 1$ to $N_\mathrm{f}$ do
3:      $\mathbf{Z} \leftarrow f_{\boldsymbol{\theta}^*_\mathrm{TE}}(\mathbf{H}_\mathrm{c})$
4:      Initialize $\mathbf{H}^T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5:      for $t = T$ to $1$ do
6:          $\hat{\mathbf{H}}^0 \leftarrow f_{\boldsymbol{\theta}^*_\mathrm{G}}(\mathrm{Concat}(\mathbf{H}^t, \mathbf{Z}), t)$
7:          $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{H}^t, t) = \frac{\mathbf{H}^t - \sqrt{\bar{\alpha}_t}\,\hat{\mathbf{H}}^0_{\boldsymbol{\theta}}(\mathbf{H}^t, t)}{\sqrt{1-\bar{\alpha}_t}}$
8:          $\mathbf{H}^{t-1} \leftarrow \sqrt{\bar{\alpha}_{t-1}}\,\hat{\mathbf{H}}^0 + \sqrt{1-\bar{\alpha}_{t-1}}\,\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{H}^t, t)$
9:      end for
10:     $\hat{\mathbf{H}}_n \leftarrow \hat{\mathbf{H}}^0$
11:     $\tilde{\mathbf{H}}_n \leftarrow \hat{\mathbf{H}}_n + \mathbf{N}, \quad \mathbf{N} \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$
12:     $\mathbf{H}_\mathrm{c} \leftarrow \mathrm{Concat}(\mathbf{H}_\mathrm{c}, \tilde{\mathbf{H}}_n)$
13: end for
14: Return $\{\hat{\mathbf{H}}_n\}_{n=1}^{N_\mathrm{f}}$
```

sequence and fed back into the model to predict the subsequent time step. This procedure is repeated until the desired prediction horizon is reached, as summarized in Algorithm 2.

B. Seq2seq

Seq2seq inference is a parallel prediction strategy in which the entire sequence of future steps is generated simultaneously rather than recursively. Instead of predicting one step at a time and feeding it back into the model, a seq2seq approach maps the input sequence directly to multiple outputs in a single forward pass.

a) Temporal Encoding: In the seq2seq prediction framework, the temporal encoder processes a fixed-length CSI sequence and extracts a latent representation,

$$\mathbf{Z} = f_{\boldsymbol{\theta}_\mathrm{TE}}(\mathbf{H}_\mathrm{p}). \tag{27}$$

b) Generator: Conditioned on this latent feature, the generator produces the entire sequence of future CSI samples over a fixed prediction horizon in parallel using a diffusion process. The generator then learns to approximate the reverse process by modeling the conditional distribution

$$f_{\boldsymbol{\theta}_\mathrm{G}}(\hat{\mathbf{H}}^t, \mathbf{Z}) \approx p_{\boldsymbol{\theta}_\mathrm{G}}(\hat{\mathbf{H}}^{t-1} \mid \hat{\mathbf{H}}^t, \mathbf{Z}), \tag{28}$$

and after T sampling steps, the predicted CSI sample is

$$\hat{\mathbf{H}}_\mathrm{f} = \hat{\mathbf{H}}^0. \tag{29}$$

This design eliminates the sequential dependency of AR inference, reducing inference time and computational overhead, especially for long prediction horizons. By predicting all future steps in parallel, seq2seq inference also avoids the problem of error accumulation, since early mistakes do not propagate through the sequence. However, this efficiency comes at the cost of reduced flexibility as a seq2seq model can be trained for a fixed context length and prediction horizon, and may struggle to capture fine-grained temporal dependencies compared to AR methods.

Algorithm 3 Seq2seq Inference

Input: Historical CSI sequence $\mathbf{H}_\mathrm{p}$ of length N_p; trained parameters $\boldsymbol{\theta}^* = [\boldsymbol{\theta}^*_\mathrm{TE}, \boldsymbol{\theta}^*_\mathrm{G}]$; diffusion scheduler with noise schedule $\{\beta_t\}_{t=1}^T$; diffusion steps T.

Output: Predicted CSI sequence $\{\hat{\mathbf{H}}_\mathrm{f}\}$.

```
1: $\mathbf{Z} \leftarrow f_{\boldsymbol{\theta}^*_\mathrm{TE}}(\mathbf{H}_\mathrm{p})$
2: Initialize $\mathbf{H}^T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
3: for $t = T$ to $1$ do
4:     $\hat{\mathbf{H}}^0 \leftarrow f_{\boldsymbol{\theta}^*_\mathrm{G}}(\mathrm{Concat}(\mathbf{H}^t, \mathbf{Z}), t)$
5:     $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{H}^t, t) = \frac{\mathbf{H}^t - \sqrt{\bar{\alpha}_t}\,\hat{\mathbf{H}}^0_{\boldsymbol{\theta}}(\mathbf{H}^t, t)}{\sqrt{1-\bar{\alpha}_t}}$
6:     $\mathbf{H}^{t-1} \leftarrow \sqrt{\bar{\alpha}_{t-1}}\,\hat{\mathbf{H}}^0 + \sqrt{1-\bar{\alpha}_{t-1}}\,\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{H}^t, t)$
7: end for
8: $\hat{\mathbf{H}}_\mathrm{f} \leftarrow \hat{\mathbf{H}}^0$
9: Return $\hat{\mathbf{H}}_\mathrm{f}$
```

Inference: During inference, the trained temporal encoder processes the historical CSI samples and encodes them into a temporal latent representation. This representation is concatenated with standard Gaussian noise and passed to the diffusion generator, where the reverse diffusion process is carried out using the DDIM scheduler as outlined in Algorithm 3.

It is worth noting that various temporal encoders (e.g., RNNs, Transformers) and backbone models for diffusion (e.g., U-Net, DiT) can be employed. In Section IV, we examine our diffusion-based CSI prediction framework under different architectural choices.

C. Unified Seq2seq

Employing separate models for temporal encoding and CSI generation provides a robust framework for learning the temporal and spatial characteristics of the dynamic CSI distribution. Training separate models for temporal and spatial dependencies complicates optimization and requires substantial training data. It also increases computational cost, as both networks must be trained and executed for CSI prediction. To address these limitations and enhance practicality in real-world applications, we integrate temporal encoding and generation into a single diffusion model that jointly learns temporal and spatial dependencies from CSI samples. In this formulation, the diffusion model directly approximates the conditional distribution of the CSI dynamics given historical CSI sequences,

$$p_{\boldsymbol{\theta}_\mathrm{G}}\left(\hat{\mathbf{H}}^{t-1} \mid \hat{\mathbf{H}}^t, \mathbf{H}_\mathrm{p}\right). \tag{30}$$

This unified framework reduces model complexity and improves efficiency, though it may be less effective in capturing fine-grained temporal and spatial structures compared to architectures with two dedicated modules.

IV. Experimental Setup

In this section, we present a detailed description of the experimental setup used to evaluate the performance of the proposed diffusion-based CSI prediction models. We first introduce the CDL simulation parameters employed to generate the CSI dataset. We then describe the proposed diffusion models with different backbone architectures, including U-Net, DiT, and 3D U-Net, as well as several state-of-the-art CSI prediction baselines for comparison. The training configuration and the techniques adopted during model training are outlined next. Finally, we provide a comprehensive set of numerical results to assess and compare the performance of these models in terms of NMSE across varying prediction horizons and SNR regimes.

A. Dataset Setup

We generate CSI data according to the CDL channel model introduced in Section II-A. A total of 100,000 independent CSI samples are created. Each sample is stored as a tensor of shape $[100 \times 2 \times N_\mathrm{t} \times N_\mathrm{c}]$, representing 100 time steps, 2 channels for the real and imaginary parts, N_t transmit antennas, and N_c subcarriers. Each sample is then split into past CSI data, $\mathbf{H}_\mathrm{p}$ of length N_p, and future ground-truth CSI data, $\mathbf{H}_\mathrm{f}$ of length N_f, to form training pairs $(\mathbf{X}, \mathbf{Y})$ with $\mathbf{X} = \mathbf{H}_\mathrm{p}$ and $\mathbf{Y} = \mathbf{H}_\mathrm{f}$. Min-max scaling is used to scale the dataset in the range $(0, 1)$. Unless otherwise specified, we set $N_\mathrm{p} = 30$ and $N_\mathrm{f} = 10$ throughout the experiments presented in the subsequent sections.

To generate the CSI dataset, we simulate time-varying channels using 3GPP-compliant CDL channel models, following the parameters summarized below:

- Carrier frequency: $f_\mathrm{c} = 28\,\mathrm{GHz}$.
- Channel model: 3GPP CDL channel model with profiles randomly selected from the set $\{$`CDL-A`, `CDL-B`, `CDL-C`, `CDL-D`, `CDL-E`$\}$.
- Antenna configuration: 16×1 MIMO system, with 16 transmit antennas at the base station and 1 receive antenna at the user terminal.
- Bandwidth: 25 resource blocks (RBs) with a subcarrier spacing of $30\,\mathrm{kHz}$, resulting in a total of 300 subcarriers. CSI is extracted from 16 evenly spaced subcarriers across the band.
- OFDM frame: Each CSI sample consists of 100 OFDM symbols (time steps), with each OFDM symbol having a duration of approximately $T_\mathrm{sym} \approx 33.3\,\mu\mathrm{s}$.
- Velocity: User velocities are sampled uniformly from the range $[30, 120]\,\mathrm{km/h}$ to cover a variety of mobility scenarios.
- Delay spread: The channel delay spread is sampled uniformly from the range $[50, 400]\,\mathrm{ns}$.

B. Neural Prediction Models

We evaluate the performance of several baseline models and our proposed diffusion-based scheme for CSI prediction. These models are selected to cover a diverse range of architectural paradigms, including RNNs such as GRU and ConvLSTM, efficient Transformer variants such as LinFormer [26], and proposed generative diffusion-based predictors. A brief description of each model is provided below:

- **GRU:** We use a GRU model with two hidden layers, each comprising 128 channels and employing the $\tanh$ activation function. This serves as a standard sequential baseline for capturing temporal dependencies in CSI prediction.
- **ConvLSTM:** We adopt a single-layer ConvLSTM with a hidden state of 128 channels and a spatial kernel size of 3×3, producing latent feature maps in $\mathbb{R}^{128 \times N_\mathrm{t} \times N_\mathrm{c}}$. The output is followed by Group Normalization (single group), Dropout with a rate of 0.2, a 3×3 convolutional projection layer with 4 output channels, and a final $\tanh$ activation. A brief review of ConvLSTM fundamentals is given in Appendix A.
- **LinFormer:** LinFormer, introduced in [26], is a Transformer variant tailored for CSI prediction. It replaces the computationally expensive self-attention mechanism with a time-aware multi-layer perceptron (TMLP), significantly reducing complexity. The model comprises six encoder blocks, each with a model dimension of 512 and a feed-forward hidden size of 512.
- **DiU:** DiU represents a diffusion-based CSI prediction framework employing a U-Net backbone with AR inference. A ConvLSTM acts as the temporal encoder to capture channel dynamics. The U-Net design is detailed in Appendix B.
- **DiU-seq2seq:** This is a seq2seq variant of the diffusion framework using the same U-Net backbone as DiU but predicts multiple future CSI frames in a single forward pass, without an explicit temporal encoder.
- **LinFusion:** LinFusion combines a LinFormer-based temporal encoder with a U-Net diffusion generator under a seq2seq inference scheme. The LinFormer and U-Net components follow the same architecture configurations as in the LinFormer and DiU models, respectively.
- **DiT:** In this variant, we employ a DiT as the denoising backbone, while a ConvLSTM serves as the temporal encoder. Architectural details and parameter settings for DiT are provided in Appendix C.
- **DiU3:** We employ a 3D U-Net as the diffusion backbone with seq2seq inference. The 3D U-Net extends the standard U-Net into three dimensions, jointly modeling spatio-temporal features. The details for the 3D U-Net design used in our simulation are provided in Appendix D.

C. Training Parameters

For training the diffusion model, we employ the DDIM scheduling technique to accelerate the sampling process. Diffusion timesteps t are drawn uniformly from $\{0, \ldots, T-1\}$. The input to the diffusion generator is formed by concatenating the noisy CSI at timestep t with the latent features extracted from the temporal encoder. To enhance robustness against varying SNR conditions and to account for channel estimation errors during pilot transmission, we corrupt the training data with additive noise.

We adopt the Huber loss [40] to train both the diffusion model and the temporal encoder, and use the diffusion loss formulated in (16) to predict clean samples. The Huber loss is a robust regression loss that behaves quadratically for small errors and linearly for large errors, combining the benefits

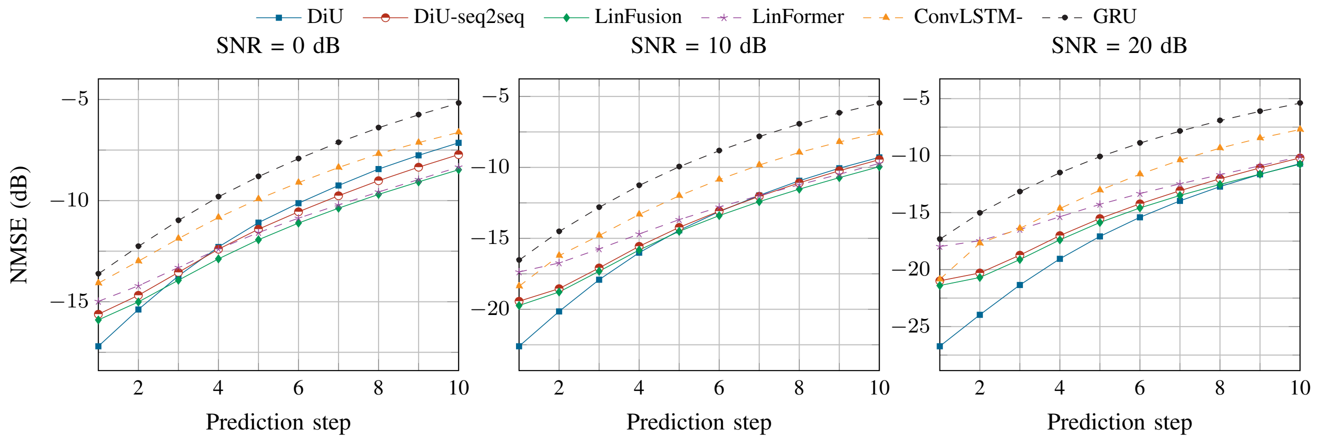

Fig. 2: NMSE of different CSI prediction models over varying prediction steps at inference SNRs of 0, 10, and 20 dB.

of MSE and mean absolute error (MAE). With a threshold parameter $\delta > 0$, it is defined as

$$\mathcal{L}_\delta(y, \hat{y}) = \begin{cases} \frac{1}{2}(y - \hat{y})^2, & \text{if } |y - \hat{y}| \le \delta, \\ \delta\left(|y - \hat{y}| - \frac{\delta}{2}\right), & \text{otherwise.} \end{cases} \tag{31}$$

We employ the Adam optimizer to jointly update the parameters of the temporal encoder and the diffusion generator, using learning rates of 1×10^{-3} and 1×10^{-3}, respectively. To enhance training stability, we apply an exponential moving average (EMA) with a decay factor of 0.995 and an update interval of 10 steps, together with gradient clipping at a maximum norm of 1.0. The model is trained for 500 epochs with a batch size of 512.

In the implementation, the Huber loss threshold is set to $\delta = 0.016$, while the SNR values are uniformly sampled from the range $[-20, 20]$ dB. For the diffusion noise schedule, we adopt the scaled and clipped squared–cosine design proposed in [41], defined as

$$\bar{\alpha}_t = \cos^2\left(\frac{\frac{t}{T} + 0.008}{1.008} \cdot \frac{\pi}{2}\right), \tag{32}$$

which leads to the variance schedule

$$\beta_t = 1 - \frac{\bar{\alpha}_t}{\bar{\alpha}_{t-1}}, \quad t \ge 1. \tag{33}$$

To prevent degenerate values, the coefficients are further clipped as

$$\beta_t = \min(\beta_{\max},\ \max(\beta_{\min},\ \beta_t)), \tag{34}$$

with $\beta_{\min} = 10^{-4}$, $\beta_{\max} = 2 \times 10^{-2}$, and $T = 2000$ in our experiments. Compared to a linear schedule, the squared–cosine design produces smaller β_t values in the early steps, resulting in a slower corruption rate at the start of the forward process, and larger β_t near the end, leading to stronger noise injection in later steps.

D. Numerical Results

We evaluate the performance of the proposed diffusion-based CSI prediction models and baseline approaches introduced in Section IV-B under various experimental settings, including different SNR levels, prediction horizons, and user mobility scenarios. The performance of all methods is quantified using the NMSE, defined as

$$\text{NMSE} \triangleq \mathbb{E}\left[\frac{\|\mathbf{H} - \hat{\mathbf{H}}\|^2}{\|\mathbf{H}\|^2}\right], \tag{35}$$

where $\mathbf{H}$ and $\hat{\mathbf{H}}$ denote the ground-truth and predicted channel matrices, respectively.

Fig. 2 illustrates the prediction accuracy of different models across varying prediction steps under three inference SNR regimes: 0, 10, and 20 dB. The results show that the proposed diffusion models consistently outperform state-of-the-art benchmarks such as GRU, ConvLSTM, and LinFormer. The performance gain is most pronounced at higher SNRs and shorter prediction horizons. For example, at an inference SNR of 20 dB, the diffusion model with AR inference achieves improvements of more than 5 dB and 8 dB in NMSE compared to ConvLSTM and LinFormer, respectively. When comparing the three diffusion-based predictors, the AR variant noticeably outperforms the seq2seq variants. In contrast, the seq2seq diffusion models provide slightly better NMSE performance at lower SNRs and for longer prediction horizons, primarily because of error propagation over time in AR inference. Furthermore, incorporating diffusion into the Linformer architecture yields an NMSE improvement of approximately 4 dB compared to the Linformer baseline. While LinFusion outperforms Linformer alone, the diffusion model using a ConvLSTM temporal encoder and a U-Net backbone consistently achieves the lowest NMSE, particularly for short prediction horizons.

Fig. 3 presents the NMSE performance of different CSI prediction models as a function of inference SNR, evaluated across three prediction horizons: 1st-step prediction, 10th-step prediction, and the average performance over 10 prediction steps. As anticipated, higher SNR levels enhance the performance of all models, with our diffusion-based schemes consistently surpassing the baselines. Specifically, for the average NMSE across all steps, the diffusion model with a U-Net backbone and AR inference achieves the best overall performance. For first-step prediction, DiU with AR inference

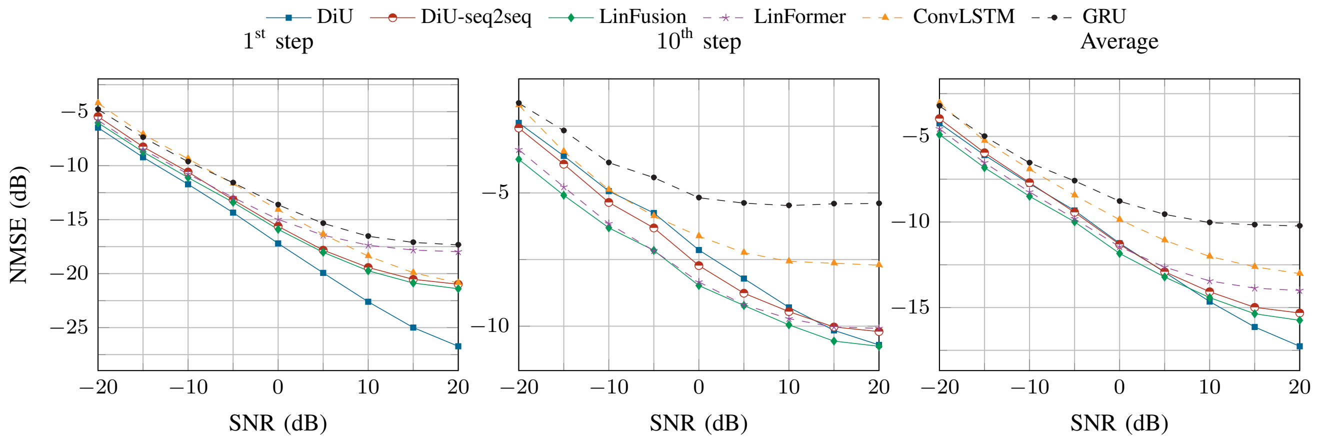

Fig. 3: NMSE performance of CSI prediction models versus inference SNR. Results are reported for the first prediction step, the last prediction step, and the average NMSE across all prediction steps.

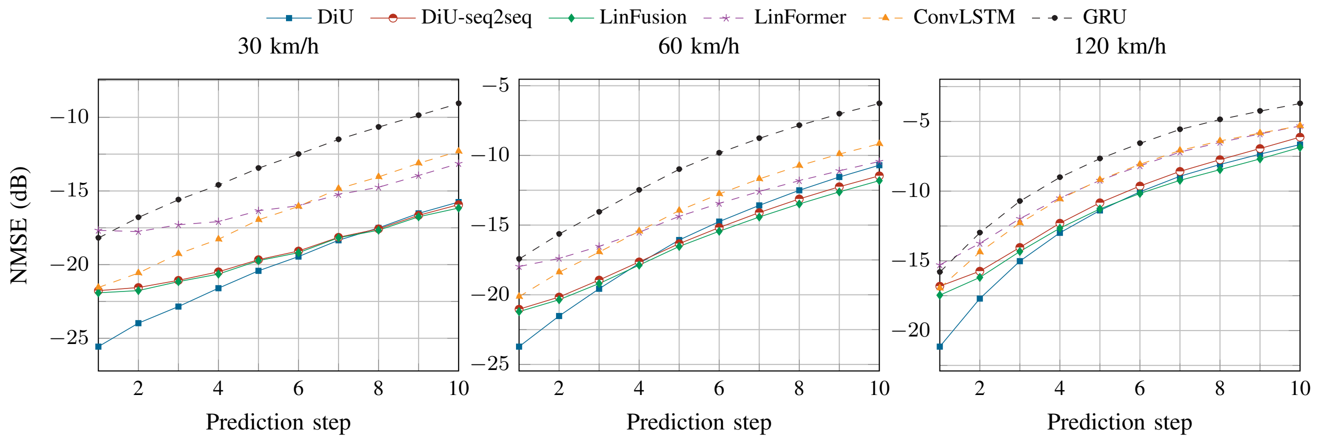

Fig. 4: NMSE performance of CSI prediction models as a function of prediction step, evaluated on CSI samples with user mobilities of 30 km/h, 60 km/h, and 120 km/h.

yields up to 5–8 dB NMSE gains compared to GRU, ConvLSTM, and LinFormer, particularly at high SNRs. At longer prediction horizons, the seq2seq diffusion models surpass AR inference, though at the expense of reduced accuracy for short horizons and lower flexibility in handling variable context lengths and prediction horizons.

We evaluate the prediction accuracy of different models under fixed user mobility levels of 30 km/h, 60 km/h, and 120 km/h, as shown in Fig. 4. As expected, the prediction performance degrades with increasing mobility due to faster channel dynamics and weaker temporal correlations. Nevertheless, the proposed diffusion-based predictors consistently outperform all baseline models across all velocity regimes. The NMSE gains of AR-based diffusion are particularly pronounced at lower user speeds (e.g., 30 km/h), where stronger temporal correlations can be exploited. In this regime and for short prediction horizons, DiU with AR inference achieves up to a 5 dB improvement over both DiU-seq2seq and LinFusion. However, this performance gap gradually diminishes as the prediction horizon extends and the user velocity increases.

Fig. 5 compares the CSI prediction performance of different backbone architectures and context lengths across varying prediction horizons. The left plot evaluates our proposed diffusion-based predictors with U-Net, DiT, and 3D U-Net backbones. These results demonstrate the flexibility of the diffusion framework, as different architectures can be seamlessly integrated into the generator. While all models achieve comparable performance, the U-Net backbone consistently achieves the lowest NMSE. Moreover, although the 3D U-Net variant applies convolutional operations over the 3D volume of CSI data and has significantly higher model capacity, it does not outperform the 2D U-Net variant combined with an explicit temporal encoder and AR inference. This underscores the effectiveness of our proposed diffusion-based CSI prediction framework with an explicit temporal encoder.

The middle plot compares different variants of the seq2seq inference strategy for diffusion-based CSI prediction for DiU. We evaluate multiple configurations, including 1:1, 2:1, 5:1, 5:2, 10:5, and 30:10, where an $N{:}M$ setup denotes predicting M future CSI frames from N historical observations. Configurations with fewer than 10 prediction outputs still require recurrence to reach the full prediction horizon, indicating

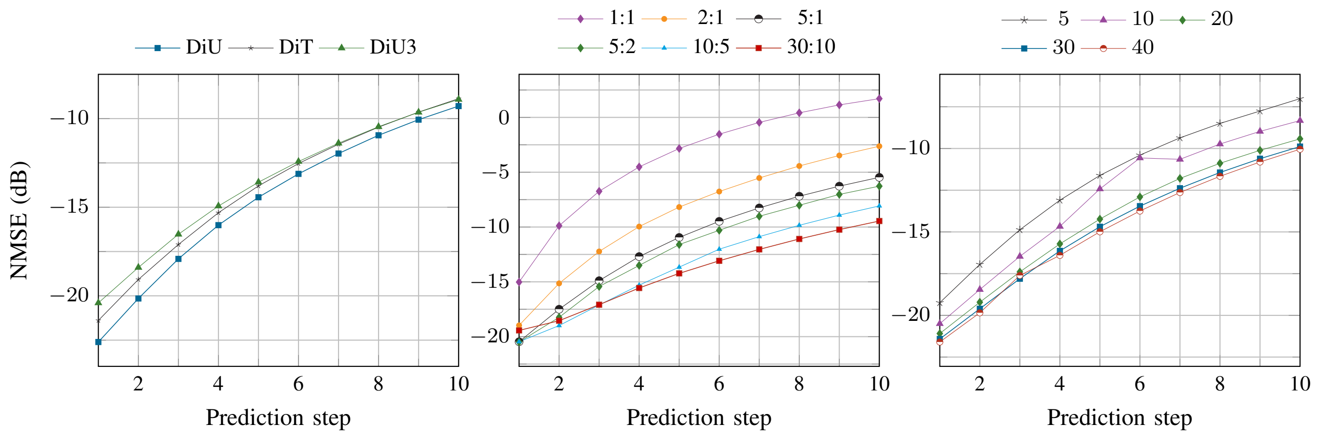

Fig. 5: NMSE performance versus prediction step at SNR = 10 dB. The left panel compares DiU, DiT, and DiU3; the middle panel shows different seq2seq variants; and the right panel presents DiU with varying context lengths.

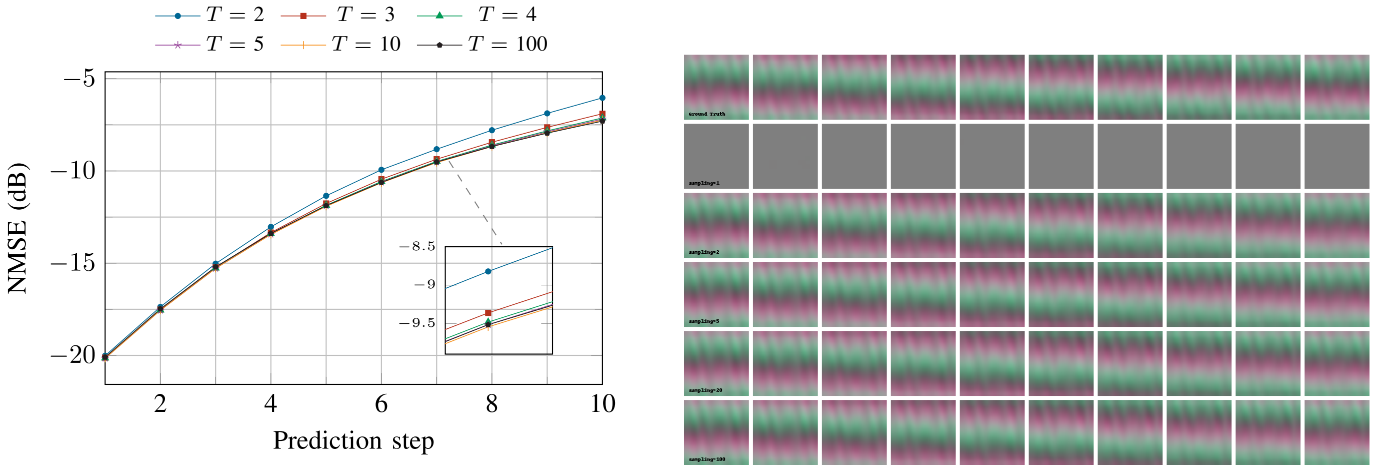

Fig. 6: Left: NMSE performance versus prediction step at 10 dB SNR for different sampling steps. Right: qualitative channel visualization for a single sequence, where rows correspond to the ground truth and predictions for each sampling step, and columns indicate the prediction steps.

that autoregression is unavoidable for many trained seq2seq models. In this setup, 30:10 corresponds to the DiU-seq2seq model. The results show that increasing the context length in seq2seq prediction provides more temporal information and consistently improves NMSE performance.

Finally, the right plot highlights the impact of varying the input context length N_p on prediction accuracy of DiU with AR inference. Increasing the context length from $N_\mathrm{p} = 5$ to $N_\mathrm{p} = 40$ steadily improves NMSE performance, demonstrating that the diffusion model benefits from larger temporal contexts. However, the performance gain begins to saturate beyond $N_\mathrm{p} = 30$, indicating diminishing returns when excessively long histories are used.

Fig. 6 illustrates the impact of different sampling steps on the performance of the proposed diffusion-based CSI prediction model at an inference SNR = 10 dB. The left plot shows the NMSE as a function of prediction step for different sampling schedules, while the right panel provides a qualitative visualization of reconstructed CSI frames for one representative sequence. Rows correspond to different sampling step settings, and columns represent future prediction steps.

The results demonstrate that the proposed model achieves competitive performance even with a small number of sampling steps. In particular, using as few as 3 sampling steps achieves nearly the same NMSE as the 100-step schedule, while significantly reducing inference time. Beyond 5 steps, the performance improvements become marginal, indicating that the proposed method can operate efficiently with very few diffusion iterations. The qualitative reconstructions on the right further confirm that high-fidelity predictions can be achieved even under low sampling budgets.

Fig. 7 illustrates the generalization performance of different CSI prediction models when evaluated on a test dataset generated at a carrier frequency of $f_\mathrm{c} = 3$ GHz, while all models were trained exclusively on data generated at $f_\mathrm{c} = 28$ GHz. This setup induces a noticeable distribution shift in the channel statistics, since lower carrier frequencies exhibit different

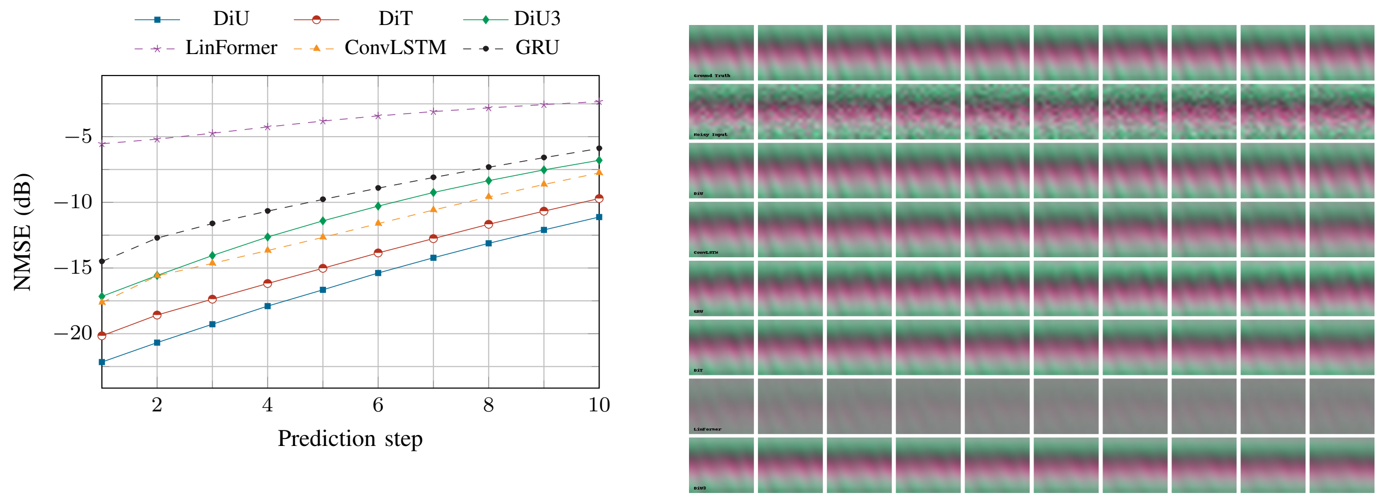

Fig. 7: Left: NMSE performance versus prediction step, evaluated at $f_c = 3$ GHz. Right: qualitative channel visualization for a single sequence, where rows correspond to the ground truth and different prediction schemes, and columns represent the prediction steps.

propagation characteristics and scattering behaviors compared to mmWave frequencies. The results show that the proposed diffusion-based models achieve the lowest generalization error and maintain robust performance under unseen channel statistics. This advantage stems from modeling the full conditional distribution $p(\mathbf{H}_f \mid \mathbf{H}_p)$ rather than learning a single deterministic mapping from past to future CSI. By capturing the underlying spatiotemporal structure of the channel—rather than memorizing training-specific patterns—diffusion-based predictors remain resilient under substantial domain shifts.

The baseline discriminative models suffer from noticeable performance degradation under domain shift. LinFormer in particular exhibits a sharp NMSE increase, indicating strong overfitting to the training distribution at $f_c = 28$ GHz when evaluated at $f_c = 3$ GHz. The 3D U-Net diffusion model also exhibits some performance loss due to its relatively high parameter count, which makes it more susceptible to overfitting. By comparison, the ConvLSTM model generalizes better than LinFormer, largely owing to its simpler architecture and lower capacity, which reduce its sensitivity to domain shifts. Overall, these results highlight that diffusion-based CSI predictors—particularly those with lightweight backbone architectures—are substantially more robust to domain shifts, making them especially promising for deployment in real-world wireless systems.

The right plot presents a qualitative comparison of predicted CSI sequences at $\mathrm{SNR} = 10$ dB across six forecasting models: DiU, ConvLSTM, GRU, DiT, LinFormer, and DiU3. The first row shows the ground-truth CSI frames, followed by the noisy input frames in the second row. The subsequent rows present the reconstructed CSI sequences for each prediction model, where each column corresponds to a future prediction step. As observed, LinFormer struggles to recover the spatial structure of the wireless channel due to its poor generalization capability, while the other models are able to reconstruct CSI samples more faithfully across prediction steps. Furthermore, DiU3 produces slightly blurred reconstructions, reflecting its performance loss when evaluated under different channel statistics.

TABLE I: Comparison of model complexity measured by FLOPs and the number of trainable parameters. Abbreviations: G = billions, M = millions.

Model	FLOPs (G)	Parameters (M)
DiU	10.54	1.60
DiU-seq2seq	0.49	1.03
DiT	6.11	0.70
DiU3	48.32	10.19
LinFusion	1.01	4.60
LinFormer	0.11	3.70
ConvLSTM	1.52	0.15
GRU	0.31	2.10

Table I reports the computational complexity and parameter counts of the evaluated CSI prediction models. The number of floating-point operations (FLOPs) per forward pass and the total number of learnable parameters are provided for a typical configuration with $N_p = 30$ input frames and $N_f = 10$ predicted CSIs. Among the baseline methods, GRU and LinFormer exhibit the lowest computational cost; however, they require a relatively large number of parameters to maintain representational capacity, making them more susceptible to overfitting. Models based on AR prediction generally incur higher computational costs due to temporal recurrence but offer greater flexibility across different context lengths and prediction horizons. Seq2seq diffusion models—and DiU-seq2seq in particular—enable faster inference and lower per-pass computation, making them well suited for latency-critical applications. In contrast, the diffusion-based predictor with a 3D U-Net backbone incurs substantially higher FLOPs owing to the joint spatio-temporal processing performed over the full 3D CSI volume.

V. Conclusion

In this paper, we presented a class of diffusion-based models for CSI prediction, designed to flexibly integrate different time-series forecasting architectures. By progressively learning the underlying distribution of spatial CSI patterns and their temporal evolution, diffusion models capture the inherent uncertainty of wireless channels rather than memorizing a fixed input–output mapping. We considered both designs with and without an explicit temporal encoder: in the former, the temporal encoder extracts latent temporal dynamics from historical CSI, while in the latter, the diffusion generator directly leverages historical CSI samples to predict future frames. Furthermore, we investigated AR and seq2seq inference schemes, explored multiple generator backbones including U-Net, DiT, and 3D U-Net, and incorporated advanced techniques such as DDIM for efficient sampling. Extensive simulations with the CDL channel model in a mmWave MIMO setup demonstrate that the proposed diffusion-based models consistently outperform state-of-the-art benchmarks such as GRU, ConvLSTM, and LinFormer across diverse SNR levels, prediction horizons, and user mobility scenarios. In particular, the diffusion model with a U-Net backbone delivers the best overall performance, achieving up to 5–8 dB NMSE gains for short-term prediction horizons at high SNRs. Moreover, we show that as few as three diffusion sampling steps are sufficient to attain competitive prediction accuracy, and that generative diffusion models exhibit stronger generalization capabilities than discriminative baselines.

Appendix A
ConvLSTM

ConvLSTM extends the standard LSTM architecture by replacing fully connected operations with 2D convolutional operations, enabling it to better preserve local spatial correlations in structured data such as images, videos, or CSI. By sliding convolutional filters across spatial dimensions, ConvLSTM is able to jointly capture spatial features and temporal dependencies more effectively than traditional LSTMs.

Similar to the standard LSTM, ConvLSTM consists of three gates: the input gate, forget gate, and output gate. However, unlike fully connected LSTMs, the computations are performed using convolutions instead of matrix multiplications. The cell state and hidden state at time step n are updated as follows:

$$\mathbf{Y}_n = [\mathbf{X}_n, \mathbf{Z}_{n-1}], \tag{36}$$

$$[\mathbf{i}_n, \mathbf{f}_n, \mathbf{o}_n, \mathbf{g}_n] = \mathrm{Conv2D}(\mathbf{Y}_n), \tag{37}$$

$$\mathbf{i}_n = \sigma(\mathbf{i}_n), \quad \mathbf{f}_n = \sigma(\mathbf{f}_n), \quad \mathbf{o}_n = \sigma(\mathbf{o}_n), \tag{38}$$

$$\mathbf{g}_n = \tanh(\mathbf{g}_n), \tag{39}$$

$$\mathbf{S}_n = \mathbf{f}_n \odot \mathbf{S}_{n-1} + \mathbf{i}_n \odot \mathbf{g}_n, \tag{40}$$

$$\mathbf{Z}_n = \mathbf{o}_n \odot \tanh(\mathbf{S}_n), \tag{41}$$

where $\sigma(\cdot)$ denotes the sigmoid activation function, $\tanh(\cdot)$ is the hyperbolic tangent function, and $\odot$ represents element-wise multiplication. The operator $\mathrm{Conv2D}(\cdot)$ applies a 2D convolution over the spatial dimensions. Here, $\mathbf{X}_n$ is the input at time step n, $\mathbf{Z}_{n-1}$ is the previous hidden state, and $\mathbf{S}_n$ denotes the current cell state.

Appendix B
U-Net

U-Nets were originally introduced for biomedical image segmentation in [34] and have since become a standard denoising backbone in diffusion-based generative models due to their strong ability to capture hierarchical spatial features. The U-Net architecture follows an encoder–decoder design, where the encoder progressively downsamples the input to extract latent representations, while the decoder upsamples and reconstructs the output using both learned features and skip connections from the encoder.

The network begins with an initial 3×3 convolution that projects the two input channels (real and imaginary CSI components) into 32 feature channels. These features then enter the encoder path. To incorporate temporal conditioning on the diffusion step t, we embed the scalar timestep using sinusoidal positional encodings, followed by a two-layer multilayer perceptron (MLP) that maps the embedding to a 256-dimensional vector. This timestep embedding is injected into every residual block via adaptive conditioning, allowing the denoising process to depend on the current stage of the diffusion process.

The encoder consists of two resolution stages. The first down block is a standard convolutional block operating on 32 channels. It applies two residual blocks, each conditioned on the timestep embedding, followed by a 2×2 spatial downsampling that reduces the resolution from $N_\mathrm{t} \times N_\mathrm{c}$ to $N_\mathrm{t}/2 \times N_\mathrm{c}/2$. The second down block is an attention-augmented convolutional block that increases the channel width to 64. It applies two residual blocks, each followed by a self-attention layer with a single-head attention mechanism. The outputs of these down blocks are stored and later used as skip connections to improve reconstruction fidelity.

At the bottleneck, the network operates on 64 channels at a spatial resolution of $N_\mathrm{t}/2 \times N_\mathrm{c}/2$. This mid block consists of a residual block, followed by a self-attention layer, and another residual block. These layers are also conditioned on the timestep embedding. Including attention at the bottleneck enables the model to capture global dependencies between antennas and subcarriers.

The first up block is an attention-augmented up block that merges bottleneck features with the skip connection from the second encoder stage, applying three sequences of residual and attention layers. The second up block is a standard up block that fuses features with the first encoder skip connection, applies three residual blocks without attention, and upsamples back to the original $N_\mathrm{t} \times N_\mathrm{c}$ spatial resolution.

Throughout the network, the residual blocks use the Sigmoid Linear Unit (SiLU) activation function. Specifically, every residual block inside the encoder, bottleneck, and decoder applies the sequence: Group Normalization $\rightarrow$ SiLU activation $\rightarrow$ Convolution. Additionally, after the final decoder stage, the features pass through a group normalization layer and another

SiLU activation before the concluding 3×3 convolution, where no activation function is applied after the final convolution.

Appendix C
DiT

DiT adopts a Vision Transformer (ViT)-style architecture and incorporates timestep conditioning via adaptive LayerNorm-Zero (adaLN-Zero). The overall architecture includes patch embedding, positional embedding, timestep embedding, and a Transformer block with specialized adaptive layer norm modulation. In particular DiT is similar to a ViT design that processes 2D images as a sequence of patch embedding with learnable positional encoding and stacking Transformer encoder blocks. DiT further applies embedding to diffusion timesteps using sinusoidal functions and an MLP. It also considers the conditioning input to every Transformer block using adaLN-Zero.

Let the noisy input at diffusion step t be denoted as $\tilde{\mathbf{H}}_t \in \mathbb{R}^{2\times N_\mathrm{t}\times N_\mathrm{c}}$. The model operates on non-overlapping patches of size $P \times P$ with $P = 4$. A strided convolution with kernel and stride equal to P performs the patchification and per-patch projection, yielding a sequence of tokens $\mathbf{Z}_0 \in \mathbb{R}^{T\times D}$, where $T = (N_\mathrm{t}/P)\cdot(N_\mathrm{c}/P)$ is the number of patches and $D = 128$ is the hidden size. To preserve spatial arrangement after flattening, a learnable positional embedding $\mathbf{P} \in \mathbb{R}^{1\times T\times D}$ is added to the patch embeddings, giving $\mathbf{X}_0 = \mathbf{Z}_0 + \mathbf{P}$.

The scalar diffusion index t is mapped into a high-dimensional representation using sinusoidal functions $\boldsymbol{\phi}(t) \in \mathbb{R}^F$ with $F = 256$. This is subsequently projected through a two-layer MLP with SiLU activation to the model width, producing the timestep embedding $\mathbf{c} \in \mathbb{R}^{B\times D}$. This embedding conditions every block through adaLN-Zero.

The Transformer stack is composed of $L = 8$ DiT blocks. Each block first processes the conditioning vector $\mathbf{c}$ using a modulation with a single-layer MLP and SiLU activation to produce six sets of vectors: scale, shift, and gate parameters for both the attention and MLP sublayers. Specifically, we obtain $\left(\boldsymbol{\delta}^{\mathrm{msa}}, \boldsymbol{\gamma}^{\mathrm{msa}}, \mathbf{g}^{\mathrm{msa}}, \boldsymbol{\delta}^{\mathrm{mlp}}, \boldsymbol{\gamma}^{\mathrm{mlp}}, \mathbf{g}^{\mathrm{mlp}}\right)$, all in $\mathbb{R}^{B\times D}$. Given input tokens $\mathbf{X}$, the modulation is applied to LayerNorm outputs as

$$\mathrm{AdaLN}(\mathbf{X};\boldsymbol{\delta},\boldsymbol{\gamma}) = \mathrm{LN}(\mathbf{X})\odot(1+\boldsymbol{\gamma}[:,\mathrm{None},:])+\boldsymbol{\delta}[:,\mathrm{None},:],$$

where broadcasting is over the token dimension. The block then computes

$$\mathbf{U} = \mathrm{MHSA}(\mathrm{AdaLN}(\mathbf{X};\boldsymbol{\delta}^{\mathrm{msa}},\boldsymbol{\gamma}^{\mathrm{msa}}))\,, \tag{42}$$

$$\mathbf{X} \leftarrow \mathbf{X} + \mathbf{g}^{\mathrm{msa}}[:,\mathrm{None},:]\odot\mathbf{U}, \tag{43}$$

$$\mathbf{V} = \mathrm{MLP}\Big(\mathrm{AdaLN}(\mathbf{X};\boldsymbol{\delta}^{\mathrm{mlp}},\boldsymbol{\gamma}^{\mathrm{mlp}})\Big)\,, \tag{44}$$

$$\mathbf{X} \leftarrow \mathbf{X} + \mathbf{g}^{\mathrm{mlp}}[:,\mathrm{None},:]\odot\mathbf{V}. \tag{45}$$

The multi-head self-attention employs $H_a = 8$ heads, so each head has width $D/H_a = 16$. The feed-forward MLP expands the hidden dimension to $\rho D = 256$ with Gaussian Error Linear Units (GELU) nonlinearity before projecting back to D.

After the final block, another adaLN modulation and a linear projection with a single-layer MLP and SiLU activation map each token to $2P^2$ output values, corresponding to a patch of size $P \times P$ with C channels. An inverse patching step reconstructs the spatial tensor $\hat{\mathbf{H}} \in \mathbb{R}^{2\times N_\mathrm{t}\times N_\mathrm{c}}$.

To ensure stable training under high-noise conditions, the weights producing the modulation vectors and residual gates, as well as the final linear projection, are initialized to zero. Consequently, the model behaves as an approximate identity function at initialization. The adopted configuration uses $P = 4$, $D = 128$, $L = 8$, $H_a = 8$, and an MLP expansion ratio of $\rho = 2.0$, which balances global modeling capacity with computational efficiency for CSI prediction tasks.

Appendix D
3D U-Net

We extend the U-Net architecture to a 3D variant in order to model the spatio-temporal structure of CSI volumes. The network takes as input a stack of frames consisting of both the observed past and the future frames concatenated along the temporal dimension, and augments the channel dimension with a binary mask channel (1 for past frames and 0 for future frames) to distinguish between conditioning and prediction targets.

The diffusion timestep t is embedded using sinusoidal position encodings followed by a two-layer MLP that projects the embedding to a 256-dimensional vector. This embedding is injected into every residual block to enable adaptive conditioning on the current stage of the diffusion process.

The encoder begins with an initial 3D convolutional block that projects the (2+1) input channels (real, imaginary, and mask) into 32 feature channels. It then applies three downsampling stages with output widths of 64, 128, and 256, respectively. Each 3D convolutional block contains two 3×3×3 convolutions, each followed by Group Normalization and SiLU activations, with the timestep embedding added between the two convolutions. After each stage, the spatial resolution is reduced by half via maxpooling, while preserving the temporal dimension. Skip connections are extracted from the outputs of each 3D convolutional block before pooling for use in the decoder.

At the bottleneck, the network applies another 3D convolutional block with 256 channels, followed by a self-attention layer that performs multi-head self-attention only along the temporal axis while treating each spatial location independently. This allows the model to explicitly capture temporal dependencies between CSI frames while keeping the computational cost independent of the spatial size.

The decoder mirrors the encoder and consists of three upsampling stages. Each block begins with a 3D transposed convolutional block layer that upsamples only the spatial dimensions, concatenates the result with the corresponding skip connection, and processes the combined features to fuse spatial and temporal information. The channel dimensions are reduced progressively from 256 → 128 → 64 → 32, while the temporal resolution remains constant.

Finally, a 1×1×1 convolution maps the 32 feature channels to the two output channels representing the real and imaginary parts of the predicted future CSI. Since the input volume includes both past and future frames, only the last frames of the output are retained to produce predictions.

Within every 3D convolutional block, the operations are applied in the following order: Convolution $\rightarrow$ Group Normalization $\rightarrow$ time-embedding addition $\rightarrow$ SiLU $\rightarrow$ Convolution $\rightarrow$ Group Normalization $\rightarrow$ SiLU. Thus, the SiLU activation is consistently used in all residual blocks across the encoder, bottleneck, and decoder. The multi-head self-attention layer does not use any activation internally and relies on the surrounding residual blocks for nonlinearity. No activation is applied after the final convolution to allow the network to produce unconstrained outputs.

References

[1] E. Biglieri, R. Calderbank, A. Constantinides, A. Goldsmith, A. Paulraj, and H. V. Poor, *MIMO Wireless Communications*. Cambridge, U.K.: Cambridge Univ. Press, 2007.

[2] B. Hassibi and B. M. Hochwald, "How much training is needed in multiple-antenna wireless links?" *IEEE Trans. Inf. Theory*, vol. 49, no. 4, pp. 951–963, 2003.

[3] F. Boccardi, R. W. Heath, A. Lozano, T. L. Marzetta, and P. Popovski, "Five disruptive technology directions for 5G," *IEEE Commun. Mag.*, vol. 52, no. 2, pp. 74–80, 2014.

[4] A. Alkhateeb, O. El Ayach, G. Leus, and R. W. Heath, "Channel estimation and hybrid precoding for millimeter wave cellular systems," *IEEE J. Sel. Topics Signal Process.*, vol. 8, no. 5, pp. 831–846, 2014.

[5] M. Sternad, T. Svensson, T. Ottosson, A. Ahlén, A. Svensson, and A. Brunstrom, "Towards systems beyond 3G based on adaptive OFDMA transmission," *Proc. IEEE*, vol. 95, no. 12, pp. 2432–2455, 2007.

[6] 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.

[7] 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.

[8] I. C. Wong and B. L. Evans, "Joint channel estimation and prediction for OFDM systems," in *Proc. IEEE Global Telecommun. Conf. (GLOBECOM)*, 2005, pp. 5 pp.–2259.

[9] L. S. Muppirisetty, T. Svensson, and H. Wymeersch, "Spatial wireless channel prediction under location uncertainty," *IEEE Trans. Wireless Commun.*, vol. 15, no. 2, pp. 1031–1044, 2016.

[10] H. Guo, B. Makki, D.-T. Phan-Huy, E. Dahlman, M.-S. Alouini, and T. Svensson, "Predictor antenna: A technique to boost the performance of moving relays," *IEEE Commun. Mag.*, vol. 59, no. 7, pp. 80–86, 2021.

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

[12] T. Fischer and C. Krauss, "Deep learning with long short-term memory networks for financial market predictions," *Eur. J. Oper. Res.*, vol. 270, no. 2, pp. 654–669, 2018.

[13] W. Kong, Z. Y. Dong, Y. Jia, D. J. Hill, Y. Xu, and Y. Zhang, "Short-term residential load forecasting based on LSTM recurrent neural network," *IEEE Trans. Smart Grid*, vol. 10, no. 1, pp. 841–851, 2019.

[14] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting," in *Proc. Int. Joint Conf. Artif. Intell. (IJCAI)*, 2018, pp. 3634–3640.

[15] C. Meijer and L. Y. Chen, "The rise of diffusion models in time-series forecasting," 2024, arXiv:2401.03006.

[16] K. Rasul, C. Seward, I. Schuster, and R. Vollgraf, "Autoregressive denoising diffusion models for multivariate probabilistic time series forecasting," 2021, arXiv:2101.12072.

[19] G. Liu, Z. Hu, L. Wang, J. Xue, H. Yin, and D. Gesbert, "Spatio-temporal neural network for channel prediction in massive MIMO-OFDM systems," *IEEE Trans. Commun.*, vol. 70, no. 12, pp. 8003–8016, 2022.

[17] 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.

[18] O. Stenhammar, G. Fodor, and C. Fischione, "A comparison of neural networks for wireless channel prediction," *IEEE Wireless Commun.*, vol. 31, no. 3, pp. 235–241, 2024.

[20] W. Jiang and H. D. Schotten, "Recurrent neural network-based frequency-domain channel prediction for wideband communications," in *Proc. IEEE Veh. Technol. Conf. (VTC Spring)*, 2019, pp. 1–6.

[21] ——, "Deep learning for fading channel prediction," *IEEE Open J. Commun. Soc.*, vol. 1, pp. 320–332, 2020.

[22] S. Fan, Z. Liu, X. Gu, and H. Li, "CSI-LLM: A novel downlink channel prediction method aligned with LLM pre-training," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, 2025, pp. 1–6.

[23] H. Kim, J. Choi, and D. J. Love, "Machine learning for future wireless communications: Channel prediction perspectives," 2025, arXiv:2502.18196.

[24] W. Xu, J. An, Y. Xu, C. Huang, L. Gan, and C. Yuen, "Time-varying channel prediction for RIS-assisted MU-MISO networks via deep learning," *IEEE Trans. Cogn. Commun. Netw.*, vol. 8, no. 4, pp. 1802–1815, 2022.

[25] 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.

[26] Y. Jin, Y. Wu, Y. Gao, S. Zhang, S. Xu, and C.-X. Wang, "Linformer: A linear-based lightweight transformer architecture for time-aware MIMO channel prediction," *IEEE Trans. Wireless Commun.*, pp. 1–1, 2025.

[27] C. Jiang, J. Guo, C.-K. Wen, and S. Jin, "Enhancing reliability in AI-based CSI prediction: A proxy-based performance monitoring approach," *IEEE Trans. Commun.*, vol. 73, no. 4, pp. 2602–2615, 2025.

[28] M. Akrout, F. Bellili, A. Mezghani, and R. W. Heath, "Next-slot OFDM-CSI prediction: Multi-head self-attention or state space model?" 2024, arXiv:2405.11072.

[29] V. Rizzello, B. Böck, M. Joham, and W. Utschick, "Reverse ordering techniques for attention-based channel prediction," *IEEE Open J. Signal Process.*, vol. 5, pp. 248–256, 2024.

[30] Z. Zhao, F. Meng, Z. Lyu, H. Li, X. Li, and G. Zhu, "CSI-BERT2: A BERT-inspired framework for efficient CSI prediction and classification in wireless communication and sensing," 2025, arXiv:2412.06861.

[31] S. Mourya, P. Reddy, S. Amuru, and K. K. Kuchi, "Spectral temporal graph neural network for massive MIMO CSI prediction," *IEEE Wireless Commun. Lett.*, vol. 13, no. 5, pp. 1399–1403, 2024.

[32] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 33, 2020, pp. 6840–6851.

[33] J. Song, C. Meng, and S. Ermon, "Denoising diffusion implicit models," 2022, arXiv:2010.02502.

[34] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," 2015, arXiv:1505.04597.

[35] Ö. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger, "3D U-Net: Learning dense volumetric segmentation from sparse annotation," in *Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Interv. (MICCAI)*, 2016, pp. 424–432.

[36] W. Peebles and S. Xie, "Scalable diffusion models with transformers," 2023, arXiv:2212.09748.

[37] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W. Wong, and W. Woo, "Convolutional LSTM network: A machine learning approach for precipitation nowcasting," 2015, arXiv:1506.04214.

[38] "Study on channel model for frequencies from 0.5 to 100 GHz," 3GPP, Tech. Rep. TR 38.901, Mar. 2022, available: https://www.3gpp.org/ftp/Specs/archive/38_series/38.901/.

[39] B. Fesl, B. Böck, F. Strasser, M. Baur, M. Joham, and W. Utschick, "On the asymptotic mean square error optimality of diffusion models," 2025, arXiv:2403.02957.

[40] P. J. Huber, "Robust estimation of a location parameter," *Ann. Math. Statist.*, vol. 35, no. 1, pp. 73–101, 1964.

[41] A. Nichol and P. Dhariwal, "Improved denoising diffusion probabilistic models," 2021, arXiv:2102.09672.