FPGA-based High-speed Reservoir Computing for MIMO Channel Prediction

Chunxiao Lin, Ummay Sumaya Khan, Lingjia Liu, *IEEE Fellow*, Yang Yi, *Senior Member, IEEE*
Department of Electrical and Computer Engineering, Virginia Tech, VA, USA
Email: {chunxiaol, ummaysumaya, ljliu, yangyi8}@vt.edu

***Abstract*—Real-time data processing has become a cornerstone of next-generation wireless communication systems because emerging 5G/6G applications demand high data rates, low latency, and reliable service. In such environments, multiple-input multiple-output (MIMO) channels vary rapidly, so predicting channel state information ahead of time is essential for maintaining link quality. To address this challenge, we propose a real-time channel prediction framework based on the Echo State Network (ESN). The ESN is a reservoir computing model with a fixed recurrent layer that offers low training overhead and computational complexity, making it well-suited for capturing temporal correlations in channel coefficients. Leveraging this advantage, our design delivers sub-frame-level prediction accuracy for MIMO channels across diverse mobility scenarios. We further realize the ESN predictor on a Field-Programmable Gate Array (FPGA) with an optimized reservoir neuron architecture. The FPGA implementation exhibits high throughput and low resource utilization, yielding low bit-error rates and efficient hardware usage. These results demonstrate that an ESN-based MIMO channel predictor not only surpasses conventional autoregressive methods in accuracy but also achieves real-time performance on resource-constrained hardware, offering a promising solution for future 5G/6G communication systems.**

***Index Terms*—Reservoir computing, Echo state network, Channel prediction, Machine learning, FPGA acceleration**

I. Introduction

WITH The rapid advancement of wireless communication technologies, particularly in the context of 5G and beyond-5G networks, the requirements for wireless data processing tasks continue to grow significantly. Channel prediction, a critical task essential for maintaining reliable communication links by forecasting channel state information (CSI) ahead of time, has become increasingly challenging in dynamic wireless environments. Conventional approaches, typically based on autoregressive or statistical models, struggle to deliver the necessary prediction accuracy and speed when channel states rapidly fluctuate, thus hindering their effectiveness in modern wireless systems.

In high mobility scenarios, such as vehicle-to-everything communications, high-speed trains, and nonterrestrial networks, channel conditions change dramatically within extremely short intervals, which greatly increases the complexity of channel prediction. Under these conditions, machine learning (ML)-based methods have increasingly become the preferred alternative due to their ability to capture non-linear and highly dynamic channel characteristics. However, traditional ML models, such as recurrent neural networks (RNNs), typically require extensive training data, high computational resources, and substantial training times, all of which are constrained in rapidly changing, real-time wireless environments. In this landscape, Echo State Networks (ESNs), a type of neuroscience-inspired neural networks, emerge as a promising solution, due to their minimal training requirements and ability to perform complex time-serial tasks with limited computational resources.

ESNs are a specialized form of RNNs with a simplified framework. Unlike traditional RNNs, ESNs utilize a fixed, randomly initialized internal layer known as the reservoir, which makes ESNs well-suited for time-series tasks. The distinctive characteristic of ESNs is their ability to learn quickly, as only the output weights are trained, leading to faster convergence and lower computational requirements. This efficiency positions ESNs as an optimal solution for low-power and time-intensive tasks. Especially in Next-G wireless systems, extreme channel dynamics introduce significant uncertainty in ML model generalization, along with the challenge of limited training data. ESN-based methods are therefore regarded as a top choice for online real-time learning in wireless tasks due to their lightweight training requirements [1]. Moreover, the theoretical foundation of ESNs is also explored, revealing an easier generalization than vanilla RNNs [2], [3].

To further meet the low-power and cost-efficiency requirements of Edge AI devices, researchers are exploring various ESN implementations, which have been proposed in the literature using approaches such as optical implementations [4], memristor-based platforms [5], and classical digital hardware [6]–[8]. Among these options, Field-Programmable Gate Arrays (FPGAs) stand out as a promising platform for designers, enabling them to leverage the parallelism of ESNs and benefit from quick development cycles and prototyping. Compared to other conventional hardware platforms such as GPUs and ASICs, FPGA-based implementations offer a good balance between performance and flexibility, making them an attractive choice for the acceleration of ESN models.

In response to these challenges, this article presents a high-performance FPGA-based ESN specifically developed for real-time MIMO channel prediction tasks. The key contributions of this work are summarized as follows:

1) We propose a low-latency, high-reliability ESN-based method tailored for MIMO channel prediction, demonstrating superior prediction accuracy compared to the conventional autoregressive (AR)-based approach.
2) We implement and validate the proposed ESN architecture on FPGA hardware, optimizing the design to achieve high throughput and compute density suitable for real-time applications.
3) We demonstrate considerable improvements in digital signal processing (DSP) slice utilization efficiency and system throughput, highlighting the practical advantages

of our FPGA-based ESN design in handling high-dimensional, real-time MIMO channel data.

II. Preliminaries

A. Echo State Networks

A typical ESN structure consists of three layers: the input layer with N_i input units, the reservoir layer with N_r units, and the output layer with N_o output units. At each discrete time step t, the input, reservoir, and output units are represented by $\mathbf{x}(t) = (x_1(t), \ldots, x_{N_i}(t))$, $\mathbf{r}(t) = (r_1(t), \ldots, r_{N_r}(t))$, and $\mathbf{y}(t) = (y_1(t), \ldots, y_{N_o}(t))$, respectively.

The architecture of the ESN is defined by three key weight matrices: $\mathbf{W}_{in}$, an $N_r \times N_i$ matrix that maps the inputs to the internal units; $\mathbf{W}_x$, an $N_r \times N_r$ matrix that defines the recurrent connections within the reservoir; and $\mathbf{W}_{out}$, an $N_o \times N_r$ matrix that maps the reservoir states to the output units. The dynamics of the internal reservoir states at the next time step $t+1$ are governed by the following equation:

$$\mathbf{r}(t+1) = f\left(\mathbf{W}_{in}\mathbf{x}(t+1) + \mathbf{W}_x\mathbf{r}(t)\right) \tag{1}$$

where f is the non-linear activation function applied to the reservoir. The output is subsequently computed through a linear readout mechanism:

$$\mathbf{y}(t+1) = \mathbf{W}_{out}\left[\mathbf{r}(t+1); \mathbf{x}(t+1)\right] \tag{2}$$

Online training Algorithms such as Recursive Least Square (RLS) are used to update $\mathbf{W}_{out}$ incrementally, enabling the ESN to dynamically adapt to changes in the input data.

B. State-of-the-art

In adaptive MIMO systems operating in real time, the transmitter's ability to access precise CSI is vital for maintaining performance. In time-division duplexing (TDD) systems, channel reciprocity allows the transmitter to infer CSI directly from uplink measurements, thereby avoiding explicit feedback and associated delays. Nevertheless, in scenarios involving rapid user mobility, processing latency can cause the inferred CSI to deviate from the true channel conditions. In frequency-division duplexing (FDD) systems, the receiver estimates the CSI and relays it back to the transmitter. Here, the feedback loop inevitably introduces a delay, which means the CSI may no longer reflect the actual channel when it is applied—a phenomenon known as channel aging.

Historically, high-mobility conditions have often been addressed with autoregressive (AR) or other parametric modeling approaches, valued for their relatively low computational overhead compared to machine learning (ML) methods. More recently, ML-based predictors have demonstrated superior accuracy over traditional AR and parametric algorithms. For example, [9] utilized ARMA models combined with Kalman filtering for MIMO channel forecasting, while [10] applied recurrent neural networks (RNNs) to the same problem. In [11], a specialized neural network was designed for CSI extrapolation, enabling prediction of future channel states. Furthermore, [12] integrated convolutional neural networks (CNNs) with RNN and AR techniques for enhanced MIMO channel prediction.

In the context of FPGA-accelerated Echo State Networks (ESNs), most recent research has emphasized reducing hardware cost and improving throughput in the reservoir computation stage. Alomar et al. [13] presented a compact ESN hardware implementation that minimizes area by lowering weight precision and substituting multipliers with shift-and-add units. Honda and Tamukoh [14] achieved further efficiency gains by adopting ternary weight encoding (0, ±1) along with fixed-point arithmetic, simplifying the hardware and lowering computational demand. Kleyko et al. [15] proposed an alternative design by merging hyperdimensional computing (HDC) principles with integer-based reservoirs, yielding an energy-efficient ESN architecture. In our own earlier work, we explored DSP48E1 slice utilization [16] and bit-serial reservoir architectures [17] independently, achieving both low-power and resource-efficient ESN implementations.

III. Methodology

A. Realistic channel generation

For the channel prediction task, channel data have been generated with Network Simulator-3 (NS-3), which utilizes a 3GPP Spatial channel model (SCM) to simulate vehicle-to-vehicle (V2V) channel conditions [18]. The settings involve a MIMO system with one transmitter and one receiver, each equipped with two antennas.

The channel generation configuration is detailed in the following table I:

TABLE I
Channel Generation Configuration

Parameter	***Value***
Channel scenario	V2V-Urban
Transmit Antenna	2
Receive Antenna	2
Speed	5/10/20 km/h
Distance between BS and UE	500 m
Number of subframes	50
Number of subcarriers	8
Number of symbols per subframe	14
Number of pilot symbols per subframe	4

B. Problem formulation

In a MIMO system, the received signals can be modeled as the following:

$$\mathbf{r(t)} = \mathbf{H(t)x(t)} + \mathbf{n(t)} \tag{3}$$

Where, $\mathbf{r(t)} = [r_1(t), \ldots, r_{Nr}(t)]$ denotes the receive signal vector of size $N_r \times 1$ at time t, $\mathbf{x(t)} = [x_1(t), \ldots, x_{Nt}(t)]$ denotes transmitted signal vector of size N_tx1, and n(t) represents additive gaussian noise vector. Channel matrix $\mathbf{H(t)} = [h_{n_r,n_t}(t)]_{N_r \times N_t}$ represents the continuous-time channel impulse responses.

The channel information used at the transmitter at time $t+\tau$ due to feedback and processing delay τ, becomes outdated, which means $H(t) \neq H(t+\tau)$. The objective of channel prediction is to estimate a future channel state $H(t+\tau)$ at time t, such that the predicted channel $\hat{H}(t+\tau)$ closely matches the actual future channel state, meaning $\hat{H}(t+\tau) \approx H(t+\tau)$.

Assume that the transmitter has access to historical channel information from the previous n time slots, represented as

$\{H_{t_1}, H_{t_2}, \ldots, H_{t_n}\}$. The goal is to predict the CSI for the upcoming time slot, and the predicted channel can be represented as $\hat{H_{t_{n+1}}}$. The key objective of channel prediction is minimizing the difference between the actual future channel and predicted future channel. This leads to the following minimization problem:

$$\text{minimize} \quad \left\| H_{n+1} - \hat{H}_{n+1} \right\|^2$$

NMSE is also used to measure the difference between the predicted and actual channels as the key performance metric, which can be defined as NMSE$(H, \hat{H})$. Here, $\hat{H}$ is the predicted value, and H is the true value.

C. Conventional method for channel prediction

The impulse response of a time-varying channel can be an autoregressive (AR) process, which can capture the relationship between current and previous channel states using a linear combination. In [19], Kalman filter (KF) is utilized to estimate the AR coefficients, which predict the future channel by the linear combination of current weighted channel data with a sequence of past channel data observations. The researchers in [20] demonstrated the performance of the AR-based channel predictor for narrowband single input single output (SISO) channels. For MIMO systems, the channels are approximated as a set of parallel SISO channels, which discards the spatial relation of the antenna arrays. A general AR-based channel predictor is as follows:

$$\hat{H}[t] = \sum_{k=1}^{p} a_k \hat{H}[t-k] \tag{4}$$

where $\hat{H}[t]$ represents the predicted value at time t, and a_k are the AR coefficients determined from previous channel observations. The parameter p is the order of the process, meaning the number of previous time steps used for the prediction.

D. Echo state network for channel prediction

ESN-based channel prediction is a learning-driven technique that does not require an explicit model to describe the relationship between past and future channel states. This model can learn a direct mapping from historical channel data to future channel data, effectively addressing the challenges posed by conventional methods.

In this work, the size of input and output features of the model are determined from the product of the number of transmit antennas, receive antennas, and sub-carriers. The subcarriers are provided as features to capture the dynamics of the frequency domain channel data. For a real-valued ESN model, the real and imaginary components of the complex-valued channel data are separated, stacked, and then provided to the model as input features. As a result, the input and output feature sizes are $T_x\_ant \times R_x\_ant \times N_{sc} \times 2$, where T_x_ant is the transmit antenna, R_x_ant is the receive antenna, and N_{sc} is the number of subcarriers.

The proposed ESN-based channel predictor is implemented in a symbol-by-symbol prediction manner. The channel data for each symbol is used to predict the channel data after 20 symbols. The input and the label for the training dataset, where the input data, D corresponds to the channel data from the current subframe, and the label I, is the channel data from the future subframe, can be shown as follows:

$\Phi\{I, D\} = \{[H_{t20}, H_{t21}, H_{t22}, \ldots], [H_{t1}, H_{t2}, H_{t3}, \ldots]\}$

where $H_{t1}, H_{t2}, H_{t3}, \ldots$ represents the current channel data for symbols in the subframe, and $H_{t20}, H_{t21}, H_{t22}, \ldots$ are the channel data after 20 symbols.

For 2×2 MIMO system, the ESN parameters are provided in the table II.

TABLE II
ESN MODEL PARAMETERS

Parameter	***Value***
Input neuron	64
Output neuron	64
Reservoir neuron	8
Spectral radius	0.7
Sparsity	0.8
Input scaling	10
Learning rate	0.001

IV. Proposed ESN Architecture

In this section, we present a detailed description of our proposed FPGA-based ESN architecture, which integrates an optimized reservoir neuron design for efficient real-time computation. First, we introduce a DSP-based multiply-accumulate (MAC) block, named **MAC_DA**, which serves as the fundamental computational element for reservoir neuron operations, ensuring efficient arithmetic processing on FPGA hardware. Subsequently, we describe a low-power nonlinear function approximator implemented using a range-addressable look-up table (RALUT), optimized for high-accuracy nonlinear activations within the reservoir. Finally, we summarize the resource utilization and throughput performance of the implemented FPGA design, demonstrating its effectiveness and suitability for real-time MIMO channel prediction tasks.

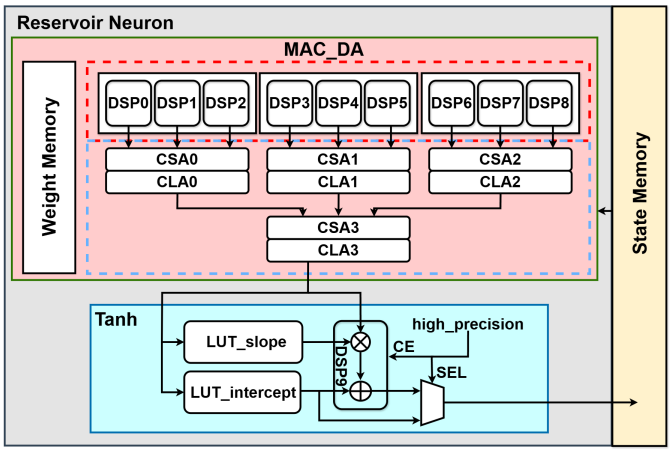

Fig. 1. Proposed reservoir neuron architecture

A. MAC_DA Module Design

Our software simulations showed that effective performance in the channel prediction task can be achieved with a small

reservoir size. Consequently, we implement a MAC_DA unit for each reservoir neuron, consisting of a 9-DSP array and a high-speed adder tree to perform a two-stage MAC operation. The architecture is designed to optimize the utilization of the DSP_macro IP while minimizing the demand on Configurable Logic Blocks (CLBs), thereby achieving high throughput with relatively low CLB usage.

In each reservoir neuron, the state updating process is performed as outlined in Equation 1, which consists of the sum of two vector multiplications followed by a nonlinear function applied to the resulting scalar. Through mathematical reformulation, the sum of vector multiplications, $\mathbf{w}_{\text{in,i}} \cdot \mathbf{x}(t+1) + \mathbf{w}_{\text{x,i}} \cdot \mathbf{r}(t)$, can be implemented as a single MAC operation, represented as $\{\mathbf{w}_{\text{in,i}}; \mathbf{w}_{\text{x,i}}\} \cdot \{\mathbf{x}(t+1); \mathbf{r}(t)\}$, with concatenation of the weights and vectors accordingly. Here, $\mathbf{w}_{\text{in},i}$ and $\mathbf{w}_{\text{x},i}$ denote the i-th rows of the corresponding weight matrices $\mathbf{W}_{\text{in}}$ and $\mathbf{W}_{\text{x}}$, respectively.

As shown in Figure 1, a MAC_DA unit utilizes nine DSPs to perform nine independent MAC operations in parallel, repeating this process during the first stage until all vector inputs are processed. The resulting nine partial sums of products (SoPs) are subsequently compressed twice through a fast adder tree to generate the final SoP. To efficiently handle the number of partial results, 3-input carry-save adders (CSAs) are used in conjunction with a final carry-lookahead adder (CLA) layer. Specifically, four CSA-CLA adder blocks are employed to complete the compression stage.

As a result, the proposed MAC_DA unit takes $\frac{\lceil N_i+N_r \rceil}{9}+2$ cycles to finish the entire MAC operation and then forward the SoP to the nonlinear activation function.

B. RALUT-based Hyperbolic Tangent design

For the nonlinear activation function, a LUT-based hyperbolic tangent (tanh) function is implemented. The tanh function is preferred in ESNs because of its natural bounds of -1 to 1 and its rich but controlled nonlinearity, which help maintain the echo state property and enable stable and expressive reservoir dynamics. In our design, the sum-of-products (SoP) result s from the MAC_DA unit is a 52-bit fixed-point value, where $s[51]$ is the sign bit, $s[50:34]$ represents the integer part, and $s[33:0]$ corresponds to the fractional part. Exploiting the centrosymmetric property of the tanh function, we compute the output only for the absolute value of s and apply a sign inversion if $s[51]=1$. Moreover, since $\tanh(s) \approx 1$ for $s > 8$ (with an error less than 10^{-6}), we truncate the LUT input range to $[0, 8]$.

A LUT called LUT_intercept with a depth of 256 is used to store precomputed tanh values corresponding to 256 evenly distributed samples over this interval, where $s_c = |s|[36:29]$ are used for indexing. For input values within the range $[0, 8]$ that do not correspond exactly to the sampled points stored in the LUT, linear interpolation is employed to approximate the true tanh values. To enable this, an additional LUT—referred to as LUT_slope—is used to store the slope of the tanh function at each sampled input point. Both the slope and intercept are indexed using the same sample code s_c. The final tanh value for an arbitrary input x is then computed using the linear approximation:

$$\tanh(x) \approx \text{slope}_s \times \Delta_s + \text{intercept}_s \tag{5}$$

where $\Delta_s = |s|[28:21]$ is the offset between the actual input s and the nearest sampled point. This linear approximation is performed with another DSP.

Moreover, we further optimize the tanh function using a **range-addressable LUT** mechanism. This approach is based on the observation that the rate of change of $\tanh(x)$ varies across the interval $[0, 8]$; specifically, in regions where Δ_s is close to zero, the change in $\tanh(x)$ is minimal, making linear interpolation unnecessary. To exploit this, we divide the input range into a high-precision region $[0, t)$ and a low-precision region $[t, 8]$, where t is a predefined threshold. For inputs $s < t$, the DSP is enabled to perform linear interpolation, and the DSP output is selected as $\tanh(s)$. For inputs $s \geq t$, the DSP is bypassed, and the value from the LUT_intercept is directly used as the output. This selective activation significantly reduces the memory usage and dynamic power consumption in the tanh block while incurring only a negligible loss in accuracy. In our case, by adopting RALUT in the tanh function, the required BRAM can be reduced from two 256-entry BRAMs to three 128-entry BRAMs, achieving a 25% reduction in BRAM usage.

C. FPGA Synthesis Results

Our proposed ESN implementation is evaluated on the Xilinx Virtex-7 VC707 development platform, which provides 2,800 DSP slices and 485K logic cells, making it particularly well suited for high-throughput real-time applications.

1) Matrix-vector Multiplication: As one of the key modules, we compare the synthesis results and performance of the proposed MAC_DA unit with the baseline MAC architecture in [16] for the same MVM operation. For the multiplication between a 72×72 matrix and a 72×1 vector, we use 9 DSP_SA blocks, 8 MAC-DSP blocks, and 8 baseline MAC units, respectively, ensuring that the total number of DSPs remains comparable across all designs. The comparison results are summarized in Table III.

TABLE III
COMPARISON OF THE MAC BLOCK DESIGNS FOR MATRIX-VECTOR MULTIPLICATION

Metric	Baseline [16]	MAC_DA
LUT	9176 (3.02%)	25016 (8.24%)
FF	328 (0.05%)	13688 (2.24%)
DSP	72 (2.57%)	80 (2.86%)
BRAM	36 (3.49%)	36 (3.49%)
DSP Efficiency (MOPS/DSP)	65.0	81.1
Power	0.183W	0.253W
T_process	1040 ns	900 ns

It can be observed that the MAC_DA-based design incurs an increase in LUT and FF utilization, along with a 38% rise in static power compared to the baseline design, which employs a fully DSP-based MAC architecture. The baseline leverages DSP48 slices on the VC707 to minimize CLB usage, but its pipeline does not achieve high DSP utilization efficiency. In contrast, the proposed architecture balances CLB and DSP usage via CLB-based fast adders, improving DSP efficiency

from 65.0 to 81.1 MOPS/DSP and reducing processing time by 13.5%—an important gain for the high-speed application. Although LUT and FF usage increase, the overhead represents only a small fraction of the device's available resources, which is acceptable for this task and allows further scaling.

2) MIMO Channel Prediction: In the FPGA on-board verification, we performed real-time operations at a global clock of 100 MHz. The delay of optimized ESN inference is 97 clocks, achieving a processing speed of 65.98M input samples/s, which is 527% faster than the speed of 10.53 input samples/s in the baseline [16]. As shown in Table IV, the resource utilization remains within the limits of the Virtex-7 FPGA board and is scalable for datasets of even higher dimensions.

TABLE IV
AREA AND TIMING CHARACTERISTICS FOR FPGA-BASED ESN SYSTEM IN MIMO CHANNEL PREDICTION

	LUT	FF	BRAM	DSP	Process Speed
Metric	20127 (6.63%)	2927 (0.48%)	40 (3.88%)	152 (5.43%)	65.98M samples/s

V. EXPERIMENTAL EVALUATION

To compare the performance of our on-chip ESN implemented for channel prediction tasks, an AR-based predictor has been implemented as the baseline method for comparison. The order of the AR-predictor has been set to 20, meaning it utilizes data from the previous 20 past time steps to predict the future channel states.

For the experiment, the channel data was generated for three different mobility conditions:

1) Low-mobility scenario: Both the transmitter and receiver have a mobility of 5 km/h.
2) Medium-mobility scenario: Both the transmitter and receiver have a mobility of 10 km/h.
3) High-mobility scenario: Both the transmitter and receiver have a mobility of 20 km/h.

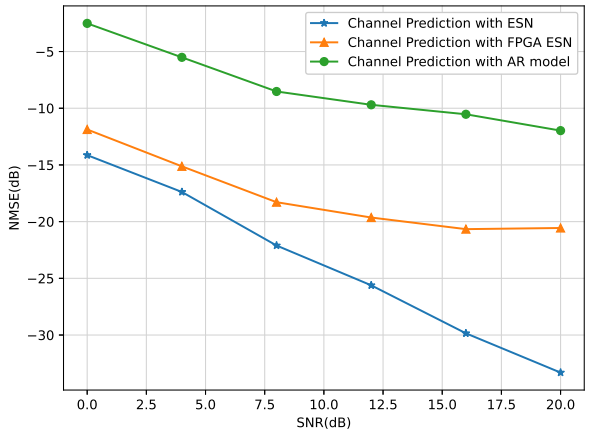

Fig. 2. NMSE Vs SNR performance of AR predictor, SW ESN and FPGA ESN for medium mobility scenario.

The FPGA-based ESN for channel prediction is compared with both Python-based ESN and baseline AR-based predictors for medium mobility conditions and demonstrated in figure 2. The signal-to-noise ratio (SDR) in the dB scale is varied from 0 to 20 dB to show the NMSE in the dB scale for the predicted channel and true channel.

From figure 2, it can be shown that both ESN-based methods always outperform the conventional AR predictor. The Python-based ESN performs better than the FPGA-based ESN channel predictor, with slight differences in low SNR scenarios and increased differences in higher SNR scenarios, which is expected. These results demonstrate the effectiveness of the ESN model in channel prediction and confirm the reliable implementation of ESN on an FPGA.

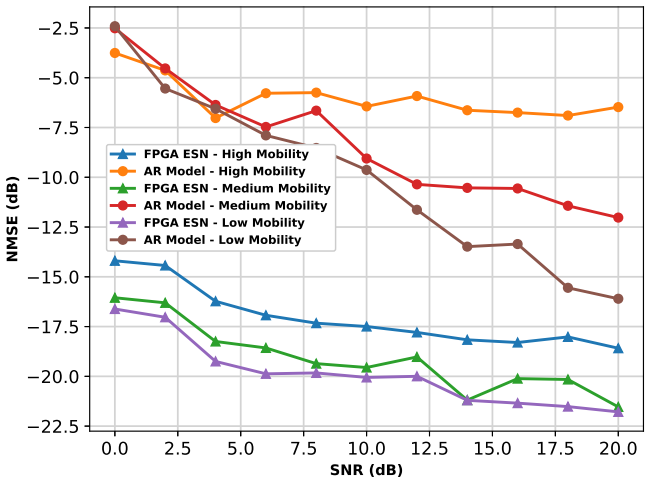

Fig. 3. NMSE Vs SNR performance of FPGA-based ESN and AR predictor for different mobility scenarios.

In figure 3, the performance for the FPGA-based ESN predictor and AR-based predictor for three mobility scenarios has been demonstrated. For AR predictor in a high mobility scenario, the performance is lossy in the low SNR region. However, from the result, it is evident that, for all mobility scenarios, the FPGA-based channel predictor is always performing better than the conventional AR channel predictor.

VI. CONCLUSION

This paper presents an FPGA-based ESN system with optimized reservoir neuron architecture, designed for the MIMO channel prediction task. An optimized vector multiplier MAC_DA is proposed to support the high-speed processing of high-dimensional data in ESN inference. The results demonstrate that the proposed design effectively supports the channel prediction task with high-dimensional data (64 inputs/outputs) and achieves a processing speed of 65.98M input samples/s. Furthermore, the proposed ESN-based channel predictor is proven to outperform the conventional AR-based method in NMSE performance.

REFERENCES

[1] J. Xu, S. Jere, Y. Song, Y.-H. Kao, L. Zheng, and L. Liu, "Learning at the speed of wireless: Online real-time learning for ai-enabled mimo in nextg," *IEEE Communications Magazine*, 2024.

[2] S. Jere, H. M. Saad, and L. Liu, "Error bound characterization for reservoir computing-based ofdm symbol detection," in *ICC 2022-IEEE international conference on communications*. IEEE, 2022, pp. 1349–1354.

[3] S. Jere, R. Safavinejad, and L. Liu, "Theoretical foundation and design guideline for reservoir computing-based mimo-ofdm symbol detection," *IEEE Transactions on Communications*, vol. 71, no. 9, pp. 5169–5181, 2023.
[4] M. Sorokina, S. Sergeyev, and S. Turitsyn, "Fiber echo state network analogue for high-bandwidth dual-quadrature signal processing," *Optics express*, vol. 27, no. 3, pp. 2387–2395, 2019.
[5] F. Nowshin, Y. Huang, M. R. Sarkar, Q. Xia, and Y. Yi, "Merrc: A memristor-enabled reconfigurable low-power reservoir computing architecture at the edge," *IEEE Transactions on Circuits and Systems I: Regular Papers*, 2023.
[6] K. Bai, Y. Yi, Z. Zhou, S. Jere, and L. Liu, "Moving toward intelligence: Detecting symbols on 5g systems through deep echo state network," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 10, no. 2, pp. 253–263, 2020.
[7] C. Lin, M. F. Azmine, and Y. Yi, "Accelerating next-g wireless communications with fpga-based ai accelerators," in *2023 IEEE/ACM International Conference on Computer Aided Design (ICCAD)*. IEEE, 2023, pp. 1–8.
[8] C. Lin, M. F. Azmine, Y. Liang, and Y. Yi, "Leveraging neuro-inspired ai accelerator for high-speed computing in 6g networks," *Frontiers in Computational Neuroscience*, vol. 18, p. 1345644, 2024.
[9] S. Kashyap, C. Mollén, E. Björnson, and E. G. Larsson, "Performance analysis of (tdd) massive mimo with kalman channel prediction," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 3554–3558.
[10] C. Potter, G. K. Venayagamoorthy, and K. Kosbar, "Rnn based mimo channel prediction," *Signal Processing*, vol. 90, no. 2, pp. 440–450, 2010.
[11] M. Arnold, S. Dörner, S. Cammerer, J. Hoydis, and S. ten Brink, "Towards practical fdd massive mimo: Csi extrapolation driven by deep learning and actual channel measurements," in *2019 53rd Asilomar Conference on Signals, Systems, and Computers*. IEEE, 2019, pp. 1972–1976.
[12] J. Yuan, H. Q. Ngo, and M. Matthaiou, "Machine learning-based channel prediction in massive mimo with channel aging," *IEEE Transactions on Wireless Communications*, vol. 19, no. 5, pp. 2960–2973, 2020.
[13] M. L. Alomar, E. S. Skibinsky-Gitlin, C. F. Frasser, V. Canals, E. Isern, M. Roca, and J. L. Rosselló, "Efficient parallel implementation of reservoir computing systems," *Neural Computing and Applications*, vol. 32, pp. 2299–2313, 2020.
[14] K. Honda and H. Tamukoh, "A hardware-oriented echo state network and its fpga implementation," *Journal of Robotics, Networking and Artificial Life*, vol. 7, no. 1, pp. 58–62, 2020.
[15] D. Kleyko, E. P. Frady, M. Kheffache, and E. Osipov, "Integer echo state networks: Efficient reservoir computing for digital hardware," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 4, pp. 1688–1701, 2020.
[16] V. M. Gan, Y. Liang, L. Li, L. Liu, and Y. Yi, "A cost-efficient digital esn architecture on fpga for ofdm symbol detection," *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, vol. 17, no. 4, pp. 1–15, 2021.
[17] C. Lin, Y. Liang, and Y. Yi, "Fpga-based reservoir computing with optimized reservoir node architecture," in *2022 23rd International Symposium on Quality Electronic Design (ISQED)*. IEEE, 2022, pp. 1–6.
[18] T. Zugno, M. Polese, N. Patriciello, B. Bojović, S. Lagen, and M. Zorzi, "Implementation of a spatial channel model for ns-3," in *Proceedings of the 2020 Workshop on ns-3*, 2020, pp. 49–56.
[19] H. Kim, S. Kim, H. Lee, C. Jang, Y. Choi, and J. Choi, "Massive mimo channel prediction: Kalman filtering vs. machine learning," *IEEE Transactions on Communications*, vol. 69, no. 1, pp. 518–528, 2020.
[20] W. Jiang and H. D. Schotten, "Neural network-based fading channel prediction: A comprehensive overview," *IEEE Access*, vol. 7, pp. 118 112–118 124, 2019.