Memory-Augmented Autoencoder with Reservoir Computing for Edge-Based Anomaly Detection in Autonomous Systems

Fabiha Nowshin, *Virginia Tech, Blacksburg, VA, 24060, USA*
Zheng Dong, *Wayne State University, Detroit, MI, 48201, USA*
Yang Yi, *Virginia Tech, Blacksburg, VA, 24060, USA*

Abstract—Real-time anomaly detection and trajectory prediction in autonomous vehicles demand models both computationally efficient and accurate. Traditional deep learning approaches often suffer from high latency, energy inefficiency, and complex training, limiting their suitability for edge-based inference. We propose a hardware-optimized architecture combining an autoencoder (AE) and reservoir computing (RC) implemented on a 22nm CMOS ASIC. The AE extracts spatial features from multi-sensor data, while the RC captures temporal patterns for robust anomaly detection and trajectory forecasting. With ultra-low power consumption and high-speed processing, our system is well-suited for resource-constrained IoT environments like autonomous vehicles. Evaluated on the KITTI dataset, the AE-RC model achieves 96.8% accuracy, with 5.6mW power usage and 5μs inference latency. This work advances energy-efficient, high-performance AI hardware for safety-critical automotive applications and enables scalable edge intelligence in next-generation IoT systems.

Keywords: *Autonomous Driving, ASIC design, edge computing*

The rapid advancement of autonomous vehicles (AVs) has driven the need for efficient, real-time anomaly detection and trajectory prediction to ensure safe navigation in complex environments. AVs rely on high-dimensional data from sensors such as LiDAR, radar, cameras, GPS, and IMUs,[1] generating over 2 GB/s of data that needs to be processed with ultra-low latency for timely decisions.[2] Cloud-based solutions struggle with latency, bandwidth, and security limitations,[3] prompting a shift toward edge-based AI for on-device inference.[4] While deep learning techniques are widely used,[5] conventional models like Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) face trade-offs. CNNs excel at spatial feature extraction but lack temporal context,[6] whereas LSTM[7] and GRU[8] models capture temporal dynamics but are computationally intensive, limiting their suitability for low-power, real-time edge deployment.

To overcome such limitations, we propose a novel Autoencoder-Reservoir Computing (AE-RC) model implemented on a 22nm Application-Specific Integrated Circuit (ASIC) designed for low-power, high-speed AI processing at the edge. Our approach leverages the energy efficiency of ASIC hardware and the computational simplicity of reservoir computing, a class of recurrent neural networks that eliminates the need for costly backpropagation through time (BPTT) training.[9] In reservoir computing (RC), a high-dimensional dynamical reservoir, constructed with random, static connections, embeds temporal features with a simple linear read-out, greatly reducing training complexity. While RC variants such as Echo State Networks and Liquid State Machines rely on densely connected node arrays, our Delayed Dynamical Feedback Reservoir (DDFR) replaces hundreds of physical neurons with a single nonlinear element plus a delay line, reproducing rich temporal dynamics with far less circuitry and power. The autoencoder (AE) module extracts low-dimensional latent representations of vehicle sensor

XXXX-XXX © IEEE
Digital Object Identifier 10.1109/XXX.0000.0000000

data using Multiply and Accumulate (MAC) operations, significantly reducing computational overhead. The DDFR in our system enhances stability and adaptability, mitigating issues of gradient vanishing and computational bottlenecks commonly encountered in deep recurrent architectures. We use a Support Vector Machine (SVM) readout layer to classify anomalies. Since the RC fixes its internal weights, only the SVM needs training, avoiding costly end-to-end gradient updates while preserving accuracy.

While reservoir computing efficiently captures temporal dependencies, its deployment in AVs requires hardware capable of real-time, low-power inference. GPUs[10] and FPGAs[11] offer AI acceleration but suffer from high power usage and latency. In contrast, ASICs provide tailored hardware optimized for operations like matrix multiplications and activation functions,[12] enabling energy-efficient, real-time sensor processing. Integrating our AE-RC model into an ASIC framework allows on-device anomaly detection and trajectory prediction with minimal latency, removing dependence on cloud-based inference.

The contributions of this work are summarized as follows. We present the first AE–RC ASIC in 22 nm CMOS for real-time AV anomaly and obstacle detection. The autoencoder uses MAC crossbars to condense high-dimensional sensor data into compact, noise-reduced latent space, cutting computational cost. A delayed-feedback reservoir then captures temporal dependencies without backpropagation, enabling low-latency, minimal-training anomaly and trajectory inference. We evaluate the architecture using the large-scale KITTI dataset,[13] which includes diverse driving scenarios like pedestrians and oncoming vehicles, treated as anomalies. Our ASIC-based AE-RC design demonstrates high accuracy, superior energy efficiency, and significantly reduced inference latency, outperforming state-of-the-art models in real-world edge AI applications for autonomous driving.

RELATED WORKS

Common AI accelerators in AVs include Graphics Processing Units (GPUs), Field Programmable Gate Arrays (FPGAs), and Application-Specific Integrated Circuits (ASICs). GPUs, such as in NVIDIA's Drive PX2 platform, are widely used for parallel tasks like object detection and sensor fusion,[10] but their high power consumption and general-purpose design limit suitability for energy-constrained, latency-critical applications. FPGAs, such as Xilinx's Zynq UltraScale MPSoC, offer reconfigurability and improved efficiency for real-time workloads,[11] but are hindered by complex programming, higher cost, and notable power demands, reducing their scalability in AV systems.

ASIC-based computing offers an optimal solution for AI acceleration in AVs by providing hardware tailored to specific autonomous driving workloads. These custom chips maximize efficiency, minimize energy use, and enable low-latency, real-time inference.[12] Companies like Mobileye, NVIDIA, and Qualcomm have developed ASICs with dedicated accelerators for tasks such as computer vision, sensor fusion, and trajectory prediction.[14] By leveraging ASICs, AVs benefit from deterministic inference times, essential for safety and decision-making. This work builds on these advantages by implementing a custom 22nm AI accelerator to execute an AE-RC model for real-time edge-based anomaly detection.

THE AE-RC ARCHITECTURE

The AE-RC architecture, shown in Figure 1, processes sensor data from autonomous vehicles using an autoencoder to extract spatial features through dimensionality reduction. These features are then passed to the DDFR for temporal feature extraction. The combined output is classified by an SVM to detect anomalies and trajectory deviations during obstacle encounters.[15]

The Autoencoder Layer

The autoencoder layer compresses multi-modal sensor inputs from LiDAR or cameras, into a compact latent space, denoising and distilling essential spatial features such as speed and object positions.[16] By mapping high-dimensional data into a lower-dimensional representation, it reduces computational load while preserving critical information for the DDFR downstream. The encoding phase process is mathematically represented as:

$$Z = \mathrm{ReLU}(W_e X + b_e), \tag{1}$$

where X represents the input feature vector, W_e and b_e are the learnable weight matrix and bias terms of the encoder, while ReLU is the activation function for nonlinearity and robustness.

The decoder reconstructs the input from the latent space in compressed form, reducing dimensionality and noise, as described by (2):

$$X_{rec} = \mathrm{ReLU}(W_d Z + b_d) \tag{2}$$

where W_d and b_d are the decoder weights and bias, and Z is the latent vector fed into decoder. We use

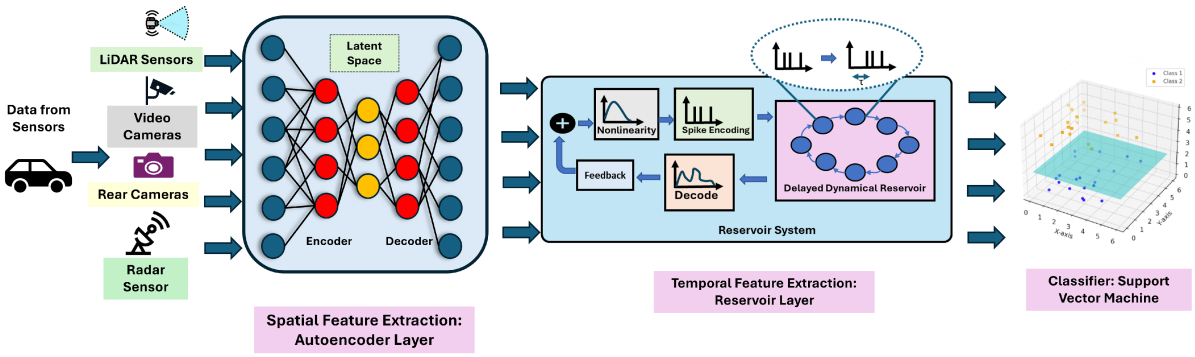

FIGURE 1. Architecture of the autoencoder-based reservoir computing model

ReLU for its simplicity, sparsity, and hardware-friendly analog implementation due to its piecewise linear nature. It also helps avoid vanishing gradients during training. The autoencoder minimizes the reconstruction error from the original input data, expressed using (3),

$$L = \|X - X_{rec}\|^2. \tag{3}$$

During inference, input data is passed through the autoencoder to produce a reconstructed output. Anomalies, unseen or irregular patterns, result in higher reconstruction errors. Comparing this error to a threshold allows detection, with outliers flagged when the error exceeds the limit. This threshold is empirically determined based on training data to balance detection sensitivity and minimize false positives.

The core mathematical operation of the autoencoder from (1) and (2) involves MAC operations, which can be realized using a crossbar architecture. To implement the autoencoder in ASIC, the weight matrices, W_e and W_d, can be mapped onto these crossbar architectures, into the conductance values of each cell, as demonstrated in Figure 2. We first discuss the design of our individual cell properties and their weight storage phenomenon to demonstrate how the autoencoder is implemented via our crossbar.

A MOSFET operating in the deep triode region behaves as a voltage-controlled resistor and is utilized to store autoencoder weight values in our CMOS ASIC implementation. As shown in Figure 2, each weight cell is programmable using our custom cell design. When the enable signal V_e activates the write transistors, a voltage V_w charges the capacitor, setting a specific drain-to-source voltage, V_{ds}. This voltage determines the effective channel resistance of the MOSFET, enabling analog weight storage. In this regime, the MOSFET's current–voltage relationship is defined by:

$$I_d = \mu C_{ox} \left(\frac{W}{L}\right) (V_{gs} - V_{th}) V_{ds}, \tag{4}$$

where μ is the carrier mobility, C_{ox} is the gate oxide capacitance, W/L is the width-to-length ratio of the transistor, and V_{th} is the threshold voltage. In our architecture, the input signal V_x is applied as the gate-source voltage V_{gs}, modulating the output current I_d based on the stored weight V_{ds}. This current represents the product of input and weight from (1) and (2), which is collected at the column line of the crossbar, enabling analog parallel vector-matrix multiplication. The precision of this operation is affected by device resistances and channel length modulation, which are minimized through optimized transistor sizing and layout techniques.

To model activation and bias in hardware, we design the circuit in Figure 2. Outputs from each cell are summed and passed through a voltage-to-current converter that includes ReLU activation. The resulting current is combined with a bias and sent to the output module. Resistor R_1 converts current to voltage, which drives MOSFET N_3 in a transconductance amplifier setup, effectively modeling the ReLU function such that:

$$V_{out} = R_2 \mu C_{ox} \left(\frac{W}{L}\right) (V_{in} - V_{th}), V_{in} > V_{th}. \tag{5}$$

The Delayed Dynamical Feedback Reservoir

The autoencoder output is passed through a masking module to enhance virtual node diversity before entering the Delayed Dynamical Feedback Reservoir (DDFR) in 22nm CMOS to extract temporal features. We adopt binary masking for its low hardware cost and effectiveness in spreading input signals across reservoir states.[17] Masking ensures each node re-

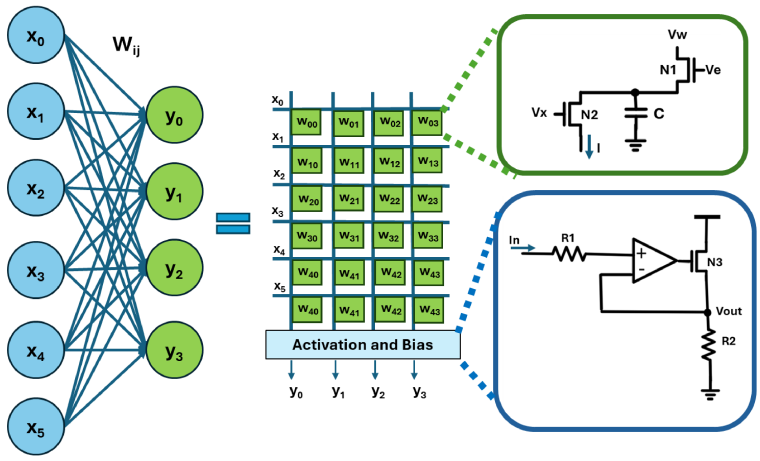

FIGURE 2. Modeling the autoencoder using cell and crossbar design

sponds uniquely, maximizing state space utilization. This operation is expressed in (6),

$$V_{\text{out}}(t) = \sum_{m=1}^{N} V_{\text{in}}^{m}(t) \cdot V_{\text{msk}}^{m}(t) \tag{6}$$

where $V_{\text{in}}^{m}(t)$ represents the input for the m^{th} virtual node, $V_{\text{msk}}^{m}(t)$ is the masking sequence, and $V_{\text{out}}(t)$ is the modulated signal.

To achieve the masking functionality we use our design in Figure 3, where the input signals are sampled at a rate of CLK_m and the differential pairs are applied to the gates of transistors M_2 and M_9. The differential mask signal V_{msk+} and V_{msk-}, which carry the variations, are applied to the transistors M_6, M_{13}, and M_7, M_{12}. The mask signal pairs control the amount of current flowing through the output nodes of the circuit, modulating the drain current through M_3, M_7, M_8, and M_{12}. We formulate the relationship between the input and output signals such that,

$$V_{out} = \sqrt{K\left(\frac{W}{L}\right)_{3,8}}(V_{msk} - V_{tp})2V_{in}R_2 \parallel r_{o3} \parallel r_{o7} \tag{7}$$

where R_2 corresponds to the load resistances, and r_{o3} and r_{o7} are the output resistances of the transistors in the differential pair, V_{tp} and V_{msk} is the threshold of the transistor and mask signal respectively. Using this technique, (7) can be implemented by our masking design from Figure 3, where the input signal is amplified and imprinted with the mask.

The output from the masking stage is passed through a nonlinearity module implementing the Mackey-Glass function. As shown in Figure 3, the analog circuit exhibits a piecewise response, at low input voltages, transistor M_1 operates in subthreshold, allowing the output to follow the input with a positive gradient. Once the threshold is exceeded, M_1 turns on fully, and M_2 discharges the output, creating a negative gradient. This threshold corresponds to the transistor's inherent V_{th}, defined by the 22nm CMOS process. This compact design replicates the non-monotonic behavior of the Mackey-Glass function.

The outputs are then fed into the Analog-to-Spike Encoder circuit, which begins with a voltage-to-current converter that activates neurons $N1$ and $N2$, as shown in Figure 4. The high input impedance of the transimpedance amplifier directs the signal to transistor $M2$, where matched PMOS transistors M_4 and M_7 and diode-connected transistors (M_5/M_8/M_{10}) ensure linear and stable current scaling, mitigating channel length modulation. The resulting mirrored currents, I_{out1} and I_{out2}, drive two leaky integrate-and-fire (LIF) neurons, introducing spatiotemporal encoding. These neurons integrate current over time on a membrane capacitor, with V_c evolving as:

$$\frac{dV}{dt} = \frac{1}{C}\left(\frac{(W/L)}{(W/L)}I_1 - I_{\text{leak}}\right). \tag{8}$$

When V_c crosses a threshold set by V_k, inverters generate spikes, regulated by an external clock V_{clk}. A positive feedback loop triggers transistor M_1 to discharge the capacitor and reset the neuron. The spike outputs from $N1$ and $N2$ are combined into a unified spike train that encodes the extracted temporal features, such as motion dynamics and temporal correlations, into a spatiotemporal format for neuromorphic processing.

This spike train is then injected into the reservoir core, where delayed, recurrent spiking activity simulates temporal feedback and memory. Each neuron

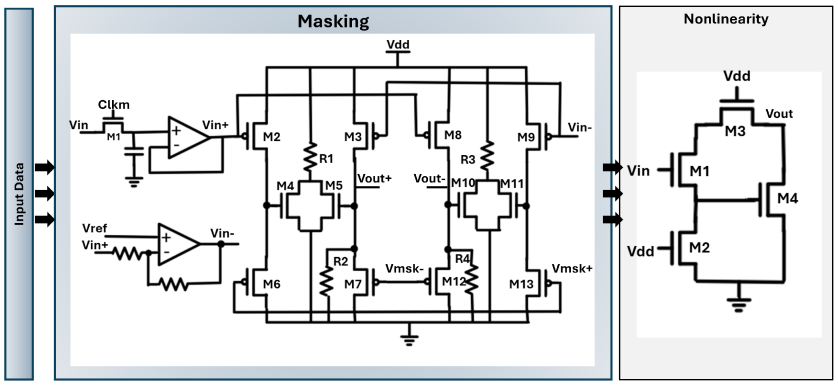

FIGURE 3. The input module of the DDFR

output acts as a clock for the next, introducing delay τ and generating output spikes offset by θ. The spike train is finally decoded into an analog signal using a charge pump, where each spike *Spk* triggers the transistor to turn on and incrementally raises the output voltage across the capacitor. A buffer amplifier stabilizes the output, and a feedback loop via a summing amplifier combines the current input with a scaled version of the prior output, enhancing the network's ability to learn time-varying patterns.

Support Vector Machine Classifier

The final layer utilizes an SVM classifier with a Radial Basis Function (RBF) kernel to identify anomalies in AV environments. The SVM operates by finding an optimal decision boundary that separates normal and anomalous driving behaviors, which can be mathematically formulated as an optimization problem,

$$\min_{\mathbf{w},b} \quad \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{n}\xi_i \tag{9}$$

where $\mathbf{w}$ and b define the decision hyperplane, C is a regularization parameter balancing classification accuracy and margin maximization, and ξ_i are slack variables allowing flexibility in handling misclassified samples. The input feature vectors $\mathbf{x}_i$ represent processed driving patterns, while y_i denotes the respective class labels, distinguishing normal vehicle behavior from hazardous situations.

To enhance the classification capability, the RBF kernel function is employed to map the input data into a higher-dimensional space, allowing the SVM to separate complex patterns effectively. The kernel function is given by (10),

$$K(x_i, x) = \exp\left(-\gamma||x_i - x||^2\right) \tag{10}$$

where γ is a hyperparameter controlling the influence of training samples on the decision boundary, and $||x_i - x||^2$ represents the squared Euclidean distance between two feature vectors.

The SVM makes its final classification decision based on the function:

$$f(x) = \sum_{i=1}^{N}\alpha_i y_i K(x_i, x) + b \tag{11}$$

where α_i are Lagrange multipliers learned during training. The model classifies new driving behavior as normal or anomalous according to (12),

$$\hat{y} = \begin{cases} +1, & f(x) > 0 \quad \text{(Normal Driving Condition)} \\ -1, & f(x) \leq 0 \quad \text{(Obstacle).} \end{cases} \tag{12}$$

Performance Analysis of the Model

Measurement Results from the ASIC Implementation

We validate our design by applying sample inputs to the 32×32 crossbar in Figure 2. Different write voltages at V_w terminal program the weights, and the resulting column output currents serve as inputs to the ReLU activation module. Input voltages range from $400mV$ to $800mV$. Figures 5a and 5b show the input voltages and corresponding output currents across various weight values, confirming that the outputs depend on both the stored V_w weights and the input signals, effectively realizing matrix multiplication.

To test the modules of our DDFR on 22nm technology with a supply voltage of $800mV$, we apply a $1MHz$ sinusoidal wave with $100mV$ amplitude as input, emulating an analog signal. A binary mask waveform is applied to the input terminals of the masking circuit

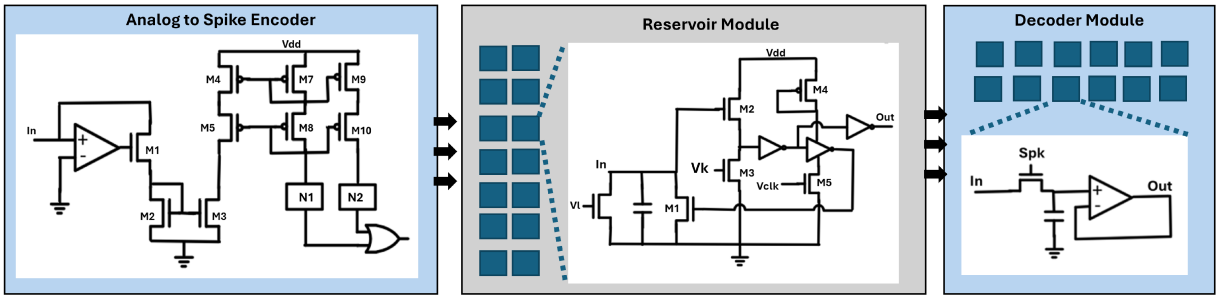

FIGURE 4. The output modules of the DDFR

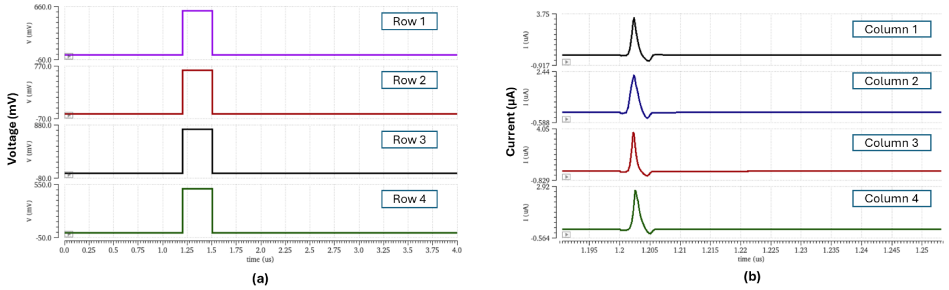

FIGURE 5. a. Input voltages at the rows of the crossbar. b. Output currents from the columns of the crossbar.

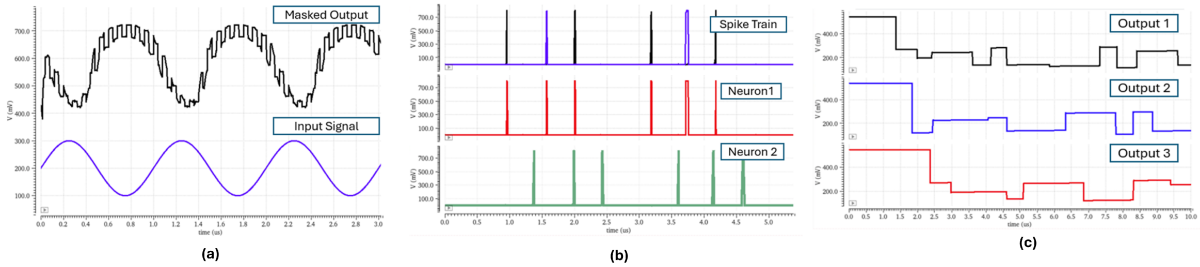

FIGURE 6. Outputs from the DDFR. a. Masking output. b. Analog to spike train and the corresponding outputs from neurons. c. Final outputs from decoder.

which then modulates the amplitude of the input signal adding variability and increasing the dimensions in the state space. The corresponding masked output is demonstrated in Figure 6a. After the mackey-glass nonlinearity is applied to the output, the signal goes through the analog to spike converter module, generating the spike train from Figure 6b. This spike train is injected onto the neuron core forming the reservoir with a 500ns delay between each neuron. The outputs from the decoder module is shown in Figure 6c from 3 blocks inside the core, demonstrating the reservoir's capability of combining the current and past outputs. At a supply voltage of 800ṁV, the mask consumes the most power of 1.2mW. The neurons and decoder modules which form the bulk of the reservoir only consume 3.3μW and 48μW respectively. The layout of the DDFR in 22nm technology is demonstrated in Figure 7a. Area analysis in Figure 7b shows the mask and encoder modules dominate overall area usage, with the neuron and decoder modules occupying minimal area.

Evaluation on the KITTI Dataset for Anomaly Detection

We evaluate our anomaly detection framework using the KITTI dataset on a 12GB NVIDIA Tesla K80 GPU

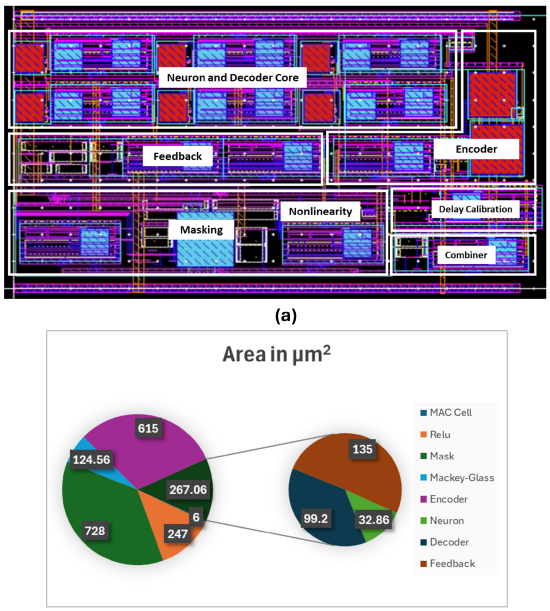

FIGURE 7. a. Layout of our DDFR design. b. Area consumption of each block.

with 13GB RAM. The dataset includes 7,480 training and 7,517 testing RGB images labeled for object detection. All images are resized to 128×128 (16,384) and normalized to [0,1]. Object categories are parsed from the label files, with outliers flagged as anomalies. The dataset is randomly split 80/20 for training and validation.

The autoencoder compresses the 16,384-dimensional input into a 128-dimensional latent space, with a compression ratio of 128:1, preserving key structural features like edges and contours, while filtering noise. The encoder has fully connected layers sized 16,384×4096, 4096×1024, 1024x512, and 512×128, where these matrix sizes translate to the number of crossbar cells in hardware. This AE is trained independently, using mean squared error (MSE) loss over 100 epochs, achieving a final reconstruction error of 0.0303.

The decoder is symmetric to the encoder, reconstructing the images from the latent space, to be fed into the DDFR which introduces memory via delayed feedback loops. The DDFR has a controlled feedback loop with a delay parameter, allowing the network to retain past states while processing new inputs. The reservoir state evolves according to:

$$r(t) = f(W_{\text{res}} \cdot r(t-1) + W_{\text{in}} \cdot x(t) + W_{\text{fb}} \cdot r(t-d) + b) \quad (13)$$

TABLE 1. Comparison of state-of-the-art models.

Network	Data	Accuracy	Loss	Task
GRU[8]	Image	–	0.03	Trajectory Prediction
LSTM + GRU[7]	Radar Signal	96%	–	Lane Change
RCNN[18]	Image	96%	0.06	Signal Recognition
GSAN+ RNN[19]	Image+ LiDAR	92.7%	0.43	Lane Change
Variance AE[20]	LiDAR	69.2%	0.45	Trajectory Prediction
This Work	Image	96.8%	0.03	Anomaly Detection

where W_{res} is the recurrent weight matrix, W_{in} is the input weight matrix, W_{fb} represents the delayed feedback connections, d is the feedback delay, and $f(\cdot)$ is the nonlinear activation function. The DDFR consists of 100 neurons, where the critical parameters are the spectral radius and leak rate, tuned to control the reservoir's temporal memory depth and stability. This structure enables the reservoir to encode spatiotemporal dependencies in the dataset, improving anomaly detection performance.

The final anomaly classification stage is performed using an SVM with the RBF kernel. The reservoir-transformed features are used as input to the classifier, enabling non-linear decision boundaries for anomaly detection. The model achieves an accuracy of 96.8% on the training set and 94.2% on the validation set. Table 1 includes a comparison of our model with diverse set of approaches applied to the different tasks, highlighting the breadth of AV inference challenges. Our model outperforms other models, achieving 96.8% accuracy and an MAE loss of 0.03.

This work is the first to validate an AE–RC anomaly-detection model in 22nm CMOS ASIC hardware. To meet physical and energy constraints, we simplify the input by extracting grayscale maps from RGB images in the KITTI dataset and resizing them to 5×5 pixels. This reduces input dimensionality to 25, enabling efficient mapping onto the crossbar. The encoder compresses this into a 4-dimensional latent space using 25×8 and 8×4 cells, achieving a 6.25:1 compression ratio. This simplified grayscale, low-resolution design enables efficient ASIC deploy-

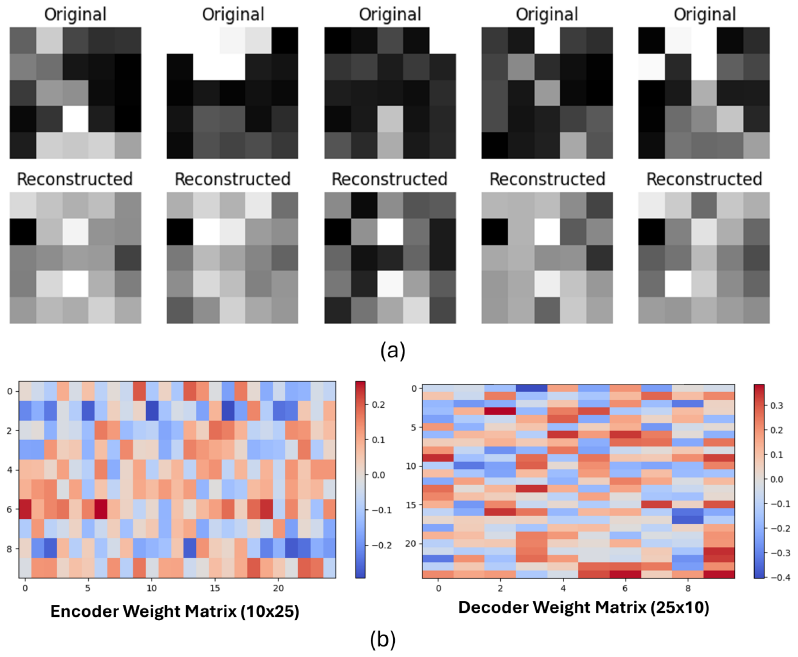

FIGURE 8. a. Original and reconstructed images from the hardware AE-RC model b. Extracted weight matrix from the crossbar.

ment, with some trade-off in accuracy. Temporal feedback in the DDFR helps preserve anomaly detection performance, validating the AE-RC pipeline as a lightweight, scalable solution. The extracted spatial features and required crossbar cells are adaptable based on deployment needs. The model is first trained offline, after which the learned weight values are mapped onto the crossbar hardware.

Figure 8a presents the resized input images alongside their reconstructed counterparts and 8b shows the crossbar mapping for our ASIC implementation. The weight values are linearly mapped onto the crossbar, and the processed input images are applied accordingly. After the accumulated currents are computed, as demonstrated in Figure 5, the signals pass through the reservoir computing layer. The outputs from the reservoir are extracted and subsequently classified using an SVM classifier following our previous setup. When tested with 50 images, the ASIC system successfully identified 30 normal images and correctly detected 20 anomalous images, with a power consumption of $5.6mW$ at a latency of $5\mu s$.

Our model functions as a low-latency, power-efficient auxiliary perception layer that enhances the AV's core perception and control stack. Its modular design allows it to run in parallel across different sensor inputs. This enables strategies like combining results over time or from multiple sensors, which helps reduce false negatives and makes the system more reliable. Furthermore, the use of an SVM with an RBF kernel enables tunable confidence thresholds, allowing the model to be conservatively biased in favor of minimizing unsafe outcomes.

Conclusion

This work introduces a hardware-efficient anomaly detection framework combining an autoencoder with a delayed feedback reservoir and SVM classifier. Implemented in 22nm technology, the design enables ultra-low-power, real-time inference for autonomous and IoT-enabled systems, achieving 96.8% accuracy on the KITTI dataset. ASIC validation confirms its feasibility for edge deployment in reliable applications.

ACKNOWLEDGMENTS

This work was supported in part by the U.S. National Science Foundation (NSF) under Grant CCF-1750450, Grant ECCS-1731928, Grant ECCS-2128594, Grant ECCS-2314813, Grant CCF-1937487, Grant CNS-2103604 and Grant CNS-2231523.

REFERENCES

1. H. A. Ignatious, M. Khan et al., "An overview of sensors in autonomous vehicles," Procedia Computer Science, vol. 198, pp. 736–741, 2022.
2. S. Liu, L. Liu, J. Tang, B. Yu, Y. Wang, and W. Shi, "Edge computing for autonomous driving: Opportunities and challenges," Proceedings of the IEEE, vol. 107, no. 8, pp. 1697–1716, 2019.
3. F. C. Andriulo, M. Fiore, M. Mongiello, E. Traversa, and V. Zizzo, "Edge computing and cloud computing for internet of things: A review," in Informatics, vol. 11, no. 4. MDPI, 2024, p. 71.
4. C. Sandu and I. Susnea, "Edge computing for autonomous vehicles-a scoping review," in 2021 20th RoEduNet Conference: Networking in Education and Research (RoEduNet). IEEE, 2021, pp. 1–5.
5. L.-H. Wen and K.-H. Jo, "Deep learning-based perception systems for autonomous driving: A comprehensive survey," Neurocomputing, vol. 489, pp. 255–270, 2022.
6. A. Thakur and S. K. Mishra, "An in-depth evaluation of deep learning-enabled adaptive approaches for detecting obstacles using sensor-fused data in autonomous vehicles," Engineering Applications of Artificial Intelligence, vol. 133, p. 108550, 2024.
7. L. Li, W. Zhao, C. Xu, C. Wang, Q. Chen, and S. Dai, "Lane-change intention inference based on rnn for autonomous driving on highways," IEEE Transactions on Vehicular Technology, vol. 70, no. 6, pp. 5499–5510, 2021.
8. P.-Y. Hsu, M.-L. Huang, W.-Y. Wang, and H.-H. Chiang, "Traffic agent trajectory prediction using a time sequence deep learning model with trajectory mapping for autonomous driving," in 2021 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW), 2021, pp. 1– 2.
9. F. Nowshin, Y. Zhang, L. Liu, and Y. Yi, "Recent advances in reservoir computing with a focus on electronic reservoirs," in 2020 11th Inter- national Green and Sustainable Computing Workshops (IGSC). IEEE, 2020, pp. 1–8.
10. D. N.-N. Tran, H.-H. Nguyen, L. H. Pham, and J. W. Jeon, "Object detection with deep learning on drive px2," in 2020 IEEE International Conference on Consumer Electronics-Asia (ICCE-Asia). IEEE, 2020, pp. 1–4.
11. S. M. Wandkar, D. Sanjaya, G. Prajwal, and Basawaraj, "Object tracking system on zynq ultrascale+ mpsoc zcu104," in International Conference on Information and Communication Technology for Intelligent Systems. Springer, 2024, pp. 59–69.
12. B. Liang, "Design of asic accelerators for ai applications," in IET Conference Proceedings CP895, vol. 2024, no. 19. IET, 2024, pp. 147–154.
13. A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in Conference on Computer Vision and Pattern Recognition (CVPR), 2012.
14. J. Ren and D. Xia, "Autonomous driving soc design," in Autonomous driving algorithms and Its IC Design. Springer, 2023, pp. 225–244.
15. F. Nowshin, S. Sethi, Z. Dong, and Y. Yi, "Enhancing driving behavior analysis in autonomous systems: A reservoir computing and temporal-aware machine learning approach," in 2024 IEEE International Conference on Mobility, Operations, Services and Technologies (MOST). IEEE, 2024, pp. 24–33.
16. P. Li, Y. Pei, and J. Li, "A comprehensive survey on design and application of autoencoder in deep learning," Applied Soft Computing, vol. 138, p. 110176, 2023.
17. 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.
18. G.-Z. Tiron and M.-S. Poboroniuc, "Neural network based traffic sign recognition for autonomous driving," in 2019 International Conference on Electromechanical and Energy Systems (SIELMEN), 2019, pp. 1–5.
19. L. Ye, Z. Wang, X. Chen, J. Wang, K. Wu, and K. Lu, "Gsan: Graph self- attention network for learning spatial–temporal interaction representation in autonomous driving," IEEE Internet of Things Journal, vol. 9, no. 12, pp. 9190–9204, 2021.
20. Z. Zhong, Y. Luo, and W. Liang, "Stgm: Vehicle trajectory prediction based on generative model for spatial-temporal features," IEEE Transactions on Intelligent Transportation Systems, vol. 23, no. 10, pp. 18 785–18 793, 2022.

Fabiha Nowshin received the B.S. and M.S degrees in Electrical Engineering from Virginia Tech, Blacksburg, VA, USA. She is currently pursuing her Ph.D. degree at the Bradley Department of Electrical and Computer Engineering at Virginia Tech. Her research interests include autonomous driving, analog-mixed signal circuit design, neuromorphic computing and computing-in-memory architectures. Contact her at fabiha27@vt.edu.

Zheng Dong received the Ph.D. degree from the Department of Computer Science, The University of Texas at Dallas, Richardson, TX, USA, in 2019. He is currently an Assistant Professor with the Department

of Computer Science, Wayne State University, Detroit, MI, USA. His research interests include real-time cyber physical systems and mobile edge computing. Dr. Dong received the Outstanding Paper Award at the 38th IEEE RTSS. Contact him at dong@wayne.edu.

Yang Yi is currently a Professor with the Bradley Department of Electrical Engineering and Computer engineering, Virginia Tech, Alexandria, VA, USA. Her research interests include VLSI circuits and systems, computer-aided design (CAD), neuromorphic architecture for brain-inspired computing systems, and low-power circuits' design with advanced nanotechnologies for high-speed wireless systems. Contact her at yangyi8@vt.edu.