The following publication P. Sun, J. Wu, M. Zhang, P. Devos and D. Botteldooren, "Delayed Memory Unit: Modeling Temporal Dependency Through Delay Gate," in IEEE Transactions on Neural Networks and Learning Systems, vol. 36, no. 6, pp. 10808-10818, June 2025 is available at https://doi.org/10.1109/TNNLS.2024.3490833.

Delayed Memory Unit: Modelling Temporal Dependency Through Delay Gate

Pengfei Sun, Jibin Wu*, Malu Zhang, Paul Devos, and Dick Botteldooren

***Abstract*—Recurrent Neural Networks (RNNs) are widely recognized for their proficiency in modeling temporal dependencies, making them highly prevalent in sequential data processing applications. Nevertheless, vanilla RNNs are confronted with the well-known issue of gradient vanishing and exploding, posing a significant challenge for learning and establishing long-range dependencies. Additionally, gated RNNs tend to be over-parameterized, resulting in poor computational efficiency and network generalization. To address these challenges, this paper proposes a novel Delayed Memory Unit (DMU). The DMU incorporates a delay line structure along with delay gates into vanilla RNN, thereby enhancing temporal interaction and facilitating temporal credit assignment. Specifically, the DMU is designed to directly distribute the input information to the optimal time instant in the future, rather than aggregating and redistributing it over time through intricate network dynamics. Our proposed DMU demonstrates superior temporal modeling capabilities across a broad range of sequential modeling tasks, utilizing considerably fewer parameters than other state-of-the-art gated RNN models in applications such as speech recognition, radar gesture recognition, ECG waveform segmentation, and permuted sequential image classification.**

***Index Terms*—recurrent neural network, delay gate, delay line, speech recognition, time series analysis**

I. INTRODUCTION

RECURRENT neural networks (RNNs) have demonstrated remarkable performance in processing sequential data. Notable applications include speech recognition [16], [37], gesture recognition [29], [52], and time series analysis [22]. However, despite being equipped with advanced optimization algorithms, vanilla RNN remains susceptible to vanishing and exploding gradient problems [2]. These issues hinder their ability to learn and establish long-range temporal dependencies. Specifically, when gradients vanish during backpropagation, small changes to the weights have a minimal effect on distant future states. On the other hand, when gradients explode, gradient-based optimization algorithms encounter challenges in smoothly navigating the loss surface [31].

To address these issues, numerous novel neural architectures and training methods have emerged in recent decades. Notably, seminal neural architectures like the long short-term memory (LSTM) [20], have introduced memory cells along with various gating mechanisms to facilitate the retention of historical information over extended periods. These gating mechanisms control information updates to the memory cells, thereby preventing rapid gradient vanishing or exploding over time. However, these gated RNNs often suffer from over-parameterization. For instance, in some speech processing tasks, it has been observed that both the update and reset gates exhibit similar behaviors, resulting in a large number of redundant parameters [34]. Moreover, within the LSTM framework, the presence of three gates alongside memory cells leads to increased model evaluation time [14]. These challenges pose significant obstacles in terms of computational efficiency and may potentially give rise to overfitting issues. Consequently, ongoing research endeavors have been dedicated to developing alternative architectural solutions. One prominent example is the Gated Recurrent Unit (GRU), which reduces the number of gating units in LSTM to two [5]. However, it is crucial to acknowledge that despite these advancements, some degree of redundancy in parameterization still exists within these gating mechanisms [33].

The Transformer, a model introduced by Vaswani et al. [43], has firmly established itself as a powerful solution for sequence modeling. Its success can be largely attributed to the incorporation of the self-attention mechanism, which excels in establishing temporal dependencies and facilitating temporal parallelization. While Transformers have demonstrated impressive performance across a range of sequential tasks, recent studies have highlighted potential challenges they may face when applied to smaller datasets or deployed on low-power devices [9], [13], [51]. Unlike Transformers, RNNs offer the advantage of parameter sharing, enabling them to flexibly handle sequences of varying lengths and facilitating efficient deployment in practical scenarios.

Recently, there has been a growing interest in incorporating neuronal delays into neural networks to enhance their temporal modeling capabilities. Studies on biologically plausible spiking neural networks (SNNs) suggest that the integration of delay line structures, represented by learnable synaptic or axonal delay variables, can significantly improve the temporal modeling capability of SNNs [17], [21], [38]–[42], [50]. Moreover, research demonstrates that employing delay lines with a fixed number of delays proves effective for tasks such as sound source

This work was supported in part by the Research Foundation - Flanders under grant number G0A0220N and the Flemish Government under the "Onderzoeksprogramma Artificiele Intelligentie (AI) Vlaanderen", National Natural Science Foundation of China (Grant No. 62306259 and 62106038), Research Grants Council of the Hong Kong SAR (Grant No. C5052-23G and PolyU25216423), and the Sichuan Science and Technology Program under Grant 2023YFG0259.

Pengfei Sun, Paul Devos, and Dick Botteldooren are with the Department of Information Technology, WAVES Research Group, Ghent University, Technologiepark Zwijnaarde 126, 9052 Ghent, Belgium (e-mail: pengfei.sun@ugent.be; p.devos@ugent.be; dick.botteldooren@ugent.be).

Jibin Wu is with the Department of Data Science and Artificial Intelligence and the Department of Computing, The Hong Kong Polytechnic University, Hong Kong SAR. (*Corresponding Author: jibin.wu@polyu.edu.hk)

Malu Zhang is with the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 610054, China (e-mail: maluzhang@uestc.edu.cn)

localization [3], [23], [30], [44]. In these models, neurons equipped with different delay parameters are utilized for coincidence detection, allowing for the detection of interaural time differences (ITD) between spatially distributed microphone pairs. While these delay-line models excel in feature extraction, their potential impact on the network dynamics of RNNs remains unclear.

In this paper, we address this research question by exploring the interplay between the delay line and RNN, which leads to the development of a novel RNN model, referred to as the Delayed Memory Unit (DMU). The DMU can effectively facilitate temporal information interaction and alleviate the challenge of learning long-range temporal dependencies within conventional RNNs. Different from other gated RNNs that employ parameterized memory units to store historical information, DMU utilizes delay lines to directly propagate information into future time steps. This not only reduces the model parameters but also facilitates temporal information processing. Notably, converting a vanilla RNN to the proposed DMU is straightforward, requiring only the addition of delay gates. We conducted extensive experiments on a range of benchmark tasks, including audio processing, radar gesture recognition, waveform segmentation, and sequence classification. Notably, DMU surpassed other state-of-the-art (SOTA) gated RNNs, achieving superior performance with significantly reduced parameters. We also performed ablation studies to understand the impact of various hyperparameters associated with the delay gate. To summarise, our contributions are threefold:

- We proposed a novel RNN model for sequential modeling, dubbed DMU. This model integrates a delay line with a vanilla RNN to enhance temporal modeling capabilities. Theoretical analysis confirms the effectiveness of DMU in tackling the long-range temporal credit assignment problem.
- We have introduced two methodologies to reduce the computational cost of the DMU: integrating dilated delay and implementing thresholding schemes. These methodologies promise to enhance computational efficiency while preserving the commendable performance of DMU.
- We demonstrate the superior temporal modeling performance of DMU across a range of benchmark tasks, consistently outperforming state-of-the-art RNN models while utilizing significantly fewer parameters.

The rest of this paper is organized as follows. In Section II, we first introduce the formulation of the proposed DMU, followed by an in-depth analysis of its effectiveness in facilitating temporal credit assignment. Furthermore, we conduct an analysis on the model complexity and provide two methods to enhance the computational efficiency of DMU. In Section III, we evaluate the proposed DMU across a diverse range of temporal processing tasks. Furthermore, in Section IV, we conduct a comprehensive study on the effectiveness of the proposed DMU model as well as its associated hyperparameters. Finally, we conclude the paper in Section V.

II. Method

A. Delayed Memory Unit

Given an input $x_t \in \mathbb{R}^M$ and the hidden state $h_{t-1} \in \mathbb{R}^N$ at time step $t-1$, the hidden state at time step t can be updated as:

$$\tilde{h}_t = \sigma_g(W_h x_t + U_h h_{t-1} + b_h), \tag{1}$$

where σ_g denotes a nonlinear activation function. The matrices $W_h \in \mathbb{R}^{N\times M}$ and $U_h \in \mathbb{R}^{N\times N}$ represent the feedforward and recurrent weights, respectively, while $b_h \in \mathbb{R}^N$ is the bias term associated with the hidden neurons.

The key feature of our proposed DMU is the incorporation of the delay gate d_t, which serves the purpose of dynamically controlling the propagation of the hidden state $\tilde{h}_t$ to future time steps. This gating mechanism is mathematically expressed as follows:

$$d_t = \sigma_h(W_d x_t + U_d h^d_{t-1} + b_d), \tag{2}$$

where $W_d \in \mathbb{R}^{n\times M}$ and $U_d \in \mathbb{R}^{n\times n}$ are the parameters governing the delay gate d_t. Here, n represents the number of delays within the delay line. The value of n can be adjusted according to the specific characteristic of the task at hand, and setting $n = 0$ will transform the DMU to the conventional RNN. To ensure that the output of the delay gates falls within a proper range, the softmax activation function σ_h is applied. The hidden state for the delay gate, denoted as h^d_{t-1}, is updated in a similar manner to the hidden state $\tilde{h}_t$. This state serves two purposes. Firstly, it helps to alleviate the workload on $\tilde{h}_t$, allowing it to primarily store the information relevant to the task. Secondly, it introduces momentum and prevents abrupt changes in the delay gate outputs, thus enhancing noise robustness.

As shown in Fig. 1, the neuron at time t receives information not only from the previous hidden state h_{t-1} but also from the delayed information controlled by the corresponding delay gates. This can be represented by:

$$h_t = \tilde{h}_t + \sum_{i=t-n\tau}^{t-\tau} d_i^{(t-i)/\tau} * \tilde{h}_i, \tag{3}$$

where $d_i^{(t-i)/\tau}$ is the delay gate output at the delay line index $(t-i)/\tau$ for time step i. The τ is the delay line dilation/skipping factor that represents the temporal resolution of the delay line.

For efficient implementation of the proposed delayed gate, we employ a sliding window memory $m_t \in \mathbb{R}^{N\times n}$ to store the delayed information. This memory is updated at each time step as:

$$m_t = \tilde{h}_t \otimes d_t + m_{t-1}\ll 1, \tag{4}$$

where $\otimes$ represents the Kronecker product, and $\ll 1$ denotes the operation of shifting the contents within the sliding window one step forward in time. Following this implementation, Eq. 3 can be updated according to:

$$h_t = \tilde{h}_t + m^1_t, \tag{5}$$

where m^1_t represents the information that is to be incorporated into the current time step t. Specifically, m^1_t equals the sum of the delayed information, which is described by the second term of Eq. 3. This formulation allows the current network

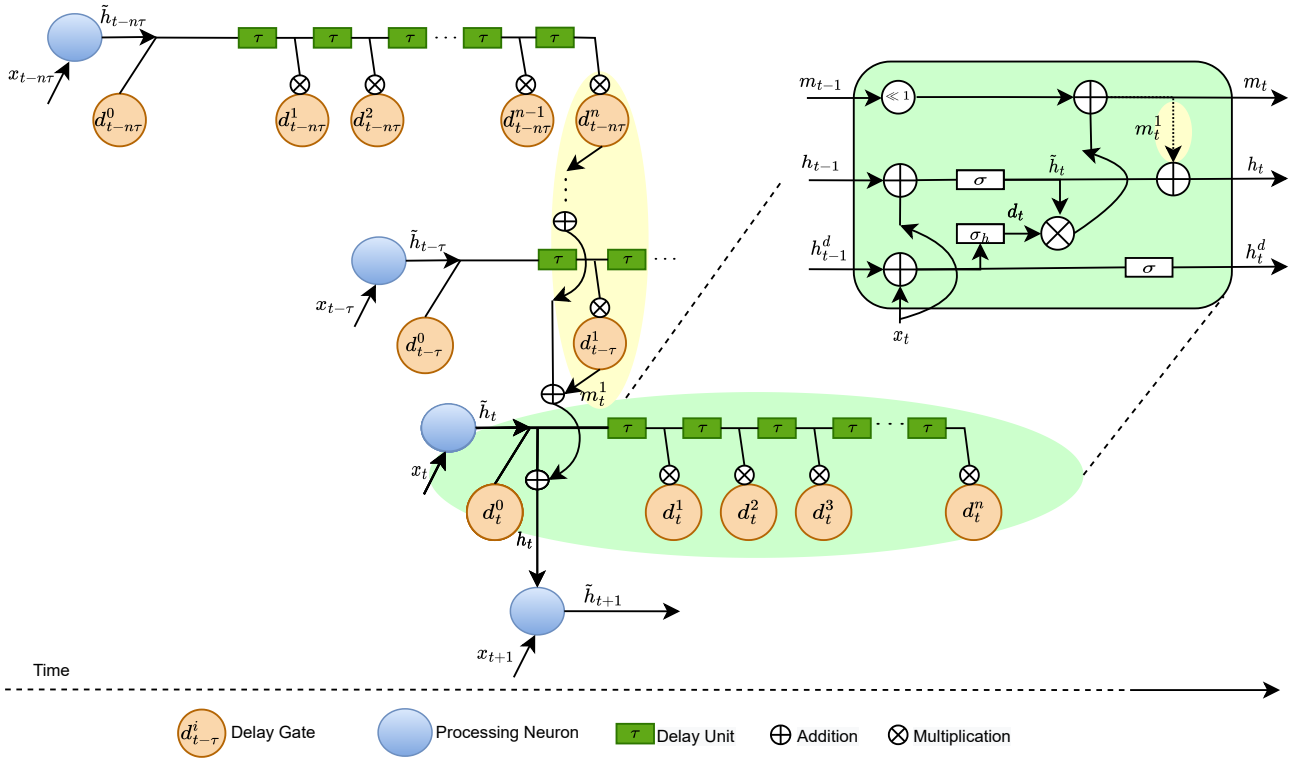

Fig. 1. Illustration of the proposed DMU. The hidden state passes through a delay line, along which the delay unit continuously applies a fixed delay of τ to the signal. At each point along the line, the corresponding delay cell gates the information from the state. The light blue area on the right side illustrates the time-unrolled DMU internal operation. This area corresponds to another light blue section at the bottom of the illustration. Similarly, the segment highlighted in light yellow within the figure represents the first state of the sliding memory window, m_t^1.

state to be effectively influenced by the previous states ranging from time step $t - n\tau$ to $t - \tau$. Unlike the vanilla RNN and other gated variants, the DMU enables a more direct influence of historical network states on future states. This feature significantly facilitates temporal credit assignment, as elaborated in the subsequent section.

It is worth noting that the delay gate mechanism proposed in this work exhibits similarities to the skip connections originally introduced for image recognition tasks [19]. In essence, both designs address the vanishing gradient problem by establishing shortcut connections that aid in the propagation of information. However, it is important to note that while skip connections in image recognition primarily focus on information propagation across the spatial dimension, the delay gate mechanism is specifically designed to address the temporal dimension.

B. Temporal Credit Assignment within DMU

Consider h_t and h_T as the hidden unit vectors at time steps t and T, respectively, where $t \ll T$. Let L be the loss function that we aim to minimize at time T. The error gradient backpropagated from T to t can be expressed as:

$$\begin{aligned}\frac{\partial L}{\partial h_t} &= \frac{\partial L}{\partial h_T}\frac{\partial h_T}{\partial h_t} \\ &= \frac{\partial L}{\partial h_T}\frac{\partial(\tilde{h}_T + \sum_{i=T-n\tau}^{T-\tau} d_i^{(T-i)/\tau}\tilde{h}_i)}{\partial h_t} \\ &= \frac{\partial L}{\partial h_T}\left(\frac{\partial \tilde{h}_T}{\partial h_t} + \frac{\partial \sum_{i=T-n\tau}^{T-\tau} d_i^{(T-i)/\tau}\tilde{h}_i}{\partial h_t}\right).\end{aligned} \tag{6}$$

To simplify our analysis, let's disregard the term $\frac{\partial L}{\partial h_T}$, as it is not recursively affected by time. The second term within the bracket can be expressed as follows:

$$\begin{aligned}\alpha_T =& \lambda_T\alpha_{T-1} + \sum_{i=T-n\tau}^{T-\tau} d_i^{(T-i)/\tau}\lambda_i \\ =& \prod_{k=t+1}^{T}\lambda_k + \sum_{i=t+2}^{T}\left((\prod_{j=i}^{T}\lambda_j)\sum_{l=j-n\tau-1}^{j-\tau-1} d_l^{(T-l)/\tau}\lambda_l\right) \\ &+ \sum_{i=T-n\tau}^{T-\tau} d_i^{(T-i)/\tau}\lambda_i,\end{aligned} \tag{7}$$

where $\alpha_k = \frac{\partial h_k}{\partial h_t}$ and $\lambda_k = \frac{\partial \tilde{h}_k}{\partial h_{k-1}}$. $\prod_{k=t+1}^{T}\lambda_k = \prod_{k=t+1}^{T}\mathrm{diag}(\sigma'(h_k))U_h^T$ represents the gradient of the vanilla RNN, and $\mathrm{diag}(\sigma'(h_{k+1}))$ is the Jacobian matrix of the pointwise activation function. From this equation, it's evident

TABLE I
THE NETWORK UPDATE FORMULATION AND PARAMETER COUNT FOR VANILLA RNNS, DMU, AND LSTM MODELS.

Formulation	Parameters
RNN	
$h_t = \sigma_g(W_h x_t + U_h h_{t-1} + b_h)$	$N^2 + MN + N$
DMU	
$\tilde{h}_t = \sigma_g(W_h x_t + U_h h_{t-1} + b_h)$ $d_t = \sigma_h(W_d x_t + U_d h^d_{t-1} + b_d)$ $h^d_t = \sigma_g(W_d x_t + U_d h^d_{t-1} + b_d)$ $m_t = d_t \otimes \tilde{h}_t + m_{t-1} \ll 1$ $h_t = \tilde{h}_t + m^1_t$	$N^2 + MN + N + Mn$ $+n^2 + n$
LSTM	
$i_t = \sigma_g(W_i x_t + U_i h_{t-1} + b_i)$ $f_t = \sigma_g(W_f x_t + U_f h_{t-1} + b_f)$ $o_t = \sigma_g(W_o x_t + U_o h_{t-1} + b_o)$ $z_t = \tanh(W_c x_t + U_c h_{t-1} + b_c)$ $c_t = f_t \circ c_{t-1} + i_t \circ z_t$ $h_t = o_t \circ \tanh(c_t)$	$4 * (N^2 + MN + N)$

Notations	Descriptions
$x_t \in \mathbb{R}^M$	Input
$\tilde{h}_t$, $h_t \in \mathbb{R}^N$	The hidden state
$h^d_t \in \mathbb{R}^n$	The delayed hidden state
$m_t \in \mathbb{R}^{N \times n}$	The sliding memory window
$d_t \in \mathbb{R}^n$	The delay gate
n	The delay line length
τ	The dilation factor
$W \in \mathbb{R}^{N \times M}$	Learnable feedforward weight
$U \in \mathbb{R}^{N \times N}$	Learnable recurrent weight
$b \in \mathbb{R}^N$	Learnable bias
$\otimes$	The kronecker product
$\circ$	The hadamard product
$\sigma_g(\cdot)$	The tanh activation function
$\sigma_h(\cdot)$	The softmax activation function
$\ll 1$	The sliding operation

that the delay gate introduces an addition term that can facilitate the smoother propagation of error gradients across longer time spans. This mechanism effectively addresses the issue of vanishing gradients, which can hinder the training process of RNNs.

C. Model Complexity Analysis

Here, we compare the model complexity of the proposed DMU against vanilla RNN and LSTM. As shown in Table I, the parameter count for the vanilla RNN is $N^2 + MN + N$, while an LSTM unit has $4 * (N^2 + MN + N)$ parameters. On the other hand, the DMU introduces an additional $Mn + n^2 + n$ parameters compared to the vanilla RNN. However, given that $n \ll M \ll N$, the DMU only adds a slightly higher number of parameters compared to the RNN unit, while still maintaining approximately $4\times$ fewer parameters than the LSTM unit. Although our approach requires additional memory to retain information in the sliding window memory, we provide two solutions that can effectively address this challenge in the following section.

D. Thresholding DMU and Dilated Delay

As given earlier, the memory usage for the sliding window memory m_t is $N * n$, and it can be efficiently managed by reducing the length of the delay line n. To this end, we propose two strategies. Firstly, we introduce a thresholding mechanism to skip less significant delay gates, which can be described as follows:

$$d_t = \begin{cases} d_t & \text{if } d_t \geq \theta \\ 0 & \text{if } d_t < \theta \end{cases}, \tag{8}$$

where the threshold θ is utilized to determine whether the output of the delay gate should be skipped. Specifically, any delay gate outputs falling below the value of θ will be considered insignificant, which can be safely ignored without affecting the task performance too much.

Additionally, as depicted by the second term of Eq. 3, dilated delay can be utilized to skip elements on the delay line at an interval of τ. This method retains the same temporal duration covered by the delay line, while reducing the total memory consumption by a factor of τ. The impact of these two strategies on memory consumption and task performance are analyzed in details in Sections IV-B and IV-C.

III. EXPERIMENTAL RESULTS

In this section, we evaluate the performance of the proposed DMU on real-world temporal classification tasks. We demonstrate the superiority of the proposed DMU by comparing its efficacy against other SOTA baseline models.

A. Experimental Setups and Training Configuration

For our model evaluations, we selected benchmark tasks encompassing an inherent temporal dimension. These tasks span areas such as speech processing, gesture recognition, waveform classification, and more. We employed the Pytorch library for implementing all models, except for the TIMIT phoneme recognition task where the PyTorch-Kaldi library [32] was used. Across all tasks, we set the nonlinear activation function σ_g to be $tanh(\cdot)$. However, for the TIMIT phoneme recognition task, we followed the default Pytorch-Kaldi library configuration, which employs $relu(\cdot)$. We selected the $softmax(\cdot)$ function for σ_h to ensure a proper probability distribution. For all experiments, the delay line resolution, denoted by τ, was set to 1 time step.

All models were trained using the Adam optimizer [24], with each training batch consisting of 128 data samples. We used a constant learning rate of 0.001 across 120 training epochs. We adopted the Kaiming weight initialization approach [18] for both network weights and bias. The sliding window memory was initialized with zeros. Notably, benefiting from the enhanced temporal credit assignment facilitated by the delay gate, DMU sidesteps the necessity for other advanced optimization methods, such as weight normalization, gradient clipping, and weight regularization, etc. All model training was conducted on Nvidia Geforce GTX 1080Ti GPUs, each equipped with 12 GB of memory.

TABLE II
COMPARISON OF MODEL PERFORMANCE ACROSS TIMIT, SHD, HEY SNIPS, SOLI, QTDB, AND PSMNIST DATASETS. THE TERMS #UNITS/#PARAMS DENOTE THE NUMBER OF HIDDEN NEURONS IN THE RECURRENT LAYER AND THE TOTAL PARAMETERS OF THE NETWORK, RESPECTIVELY. PER STANDS FOR PHONE ERROR RATE. FOR THE PSMNIST TASK, WE REPLICATED THE RESULTS OF LIPSCHITZ RNN, CORNN, AND τ-GRU USING PUBLICLY AVAILABLE SOURCE CODES.

Methods	**#Params**	**PER(%)**
RNN*	3.23M	18.8%
GRU*	7.51M	18.7%
LSTM*	9.66M	17.6%
Bidirectional-RNN*	5.20M	17.7%
Bidirectional-GRU*	14.03M	16.4%
Bidirectional-LSTM*	14.36M	15.9%
DMU	4.57M	17.5%
Bidirectional-DMU	**6.59M**	**15.6%**

∗ Reproduce through Pytorch-Kaldi [32]

(a) TIMIT

Methods	**#Params**	**Acc.**
FSNN [7]*	0.09M	48.10%
RSNN [7]*	1.79M	83.20%
Adaption RSNN [48]*	0.14M	84.40%
Attention RSNN [46]*	0.19M	90.02%
Bi-LSTM [49]	1.10M	87.20%
LSTM	0.56M	79.90%
DMU	0.16M	90.81%
DMU	0.24M	**91.48%**

∗ Spiking Neural Network

(b) SHD

Methods	**#Units/#Params**	**Acc.**
Spiking CNN [47]	-/583k	95.06%
Vanilla RNN	64/88k	95.59%
LSTM	64/347k	95.64%
GRU	64/261k	95.64%
DMU	64/113k	**95.98%**
DMU	185/346k	**96.29%**

(c) Hey Snips

Methods	**#Params**	**Acc.**
CNN Deep [45]	17.04M	48.18%
CNN-LSTM [45]	2.52M	87.17%
SRNN [48]	1.05M	91.90%
Vanilla RNN	1.05M	90.38%
LSTM	3.42M	92.62%
DMU	1.10M	**94.78%**

(d) SoLi

Methods	**#Unit/#Params**	**Acc.**
ECGNet [1]	-/8.64k	81.09%
Vanilla RNN	32/3.43k	90.56%
GRU	32/9.96k	93.73%
LSTM	32/13.22k	94.18%
DMU	32/5.82k	94.48%
DMU	52/12.33k	**94.97%**

(e) QTDB

Methods	**#Unit/#Params**	**Acc.**
GRU [4]	-/165k	92.39%
LSTM [4]	200/165k	89.86%
Lipschitz RNN [10]	200/82k	96.40%
coRNN [35]	200/82k	96.06%
τ-GRU [11]	200/164k	**96.97%**
DMU	200/49k	96.39%

(f) PSMNIST

B. Speech Processing

Phoneme Recognition: The TIMIT dataset [15] is a classical benchmark used for phoneme recognition tasks. In this dataset, each utterance contains 16-bit speech waveforms sampled at 16 kHz, along with time-aligned orthographic, phonetic, and word transcriptions. We extract 39-dim Mel-frequency cepstral coefficients (MFCCs) [8] from the raw audio waveforms and use them as input to our neural network-based classifier. To facilitate comparison with other baseline models, we follow the network and training configurations provided by the Pytorch-Kaldi library[1]. For the DMU model, we set the number of delays to $n = 45$.

As shown in Table II**(a)**, the proposed DMU model achieves a superior PER compared to other strong baseline models. Specifically, with only an additional 1.39 M parameters introduced by the delay gate, the PER is reduced by 2.1% compared to the vanilla bidirectional-RNN model. Furthermore, our DMU model surpasses LSTM and GRU models by 0.3% and 0.8%, respectively. These results underscore the effectiveness of the proposed delay gate in tackling the challenging speech processing task.

Wake-word Detection: The Hey Snips dataset [6] has been widely used for the wake-word detection task, which contains utterances collected from nearly 1,800 speakers. Following the experimental settings of Yılmaz et al. [47], we extract 40-dim MFCC features from raw audio waveforms, with a frame length of 30 ms and a frame shift of 10 ms, resulting in a total of 98 frames for each 1-second input. For this task, we adopt a convolutional recurrent neural network (CRNN) architecture, where convolutional layers are introduced at the front-end to extract local spectrotemporal features and recurrent layers are further used to handle the global temporal transitions as described in [47]. Our CRNN consists of one convolutional layer with 32 filters that have a kernel size of 5 and 10 for the frequency and time dimensions, respectively, followed by two recurrent layers with 64 neurons each. For the delay gate, the total number of delays n is set to 20 due to the short duration of each sample. The other training configurations are consistent

[1]Pytorch-Kaldi is publicly available at: https://github.com/mravanelli/pytorch-kaldi

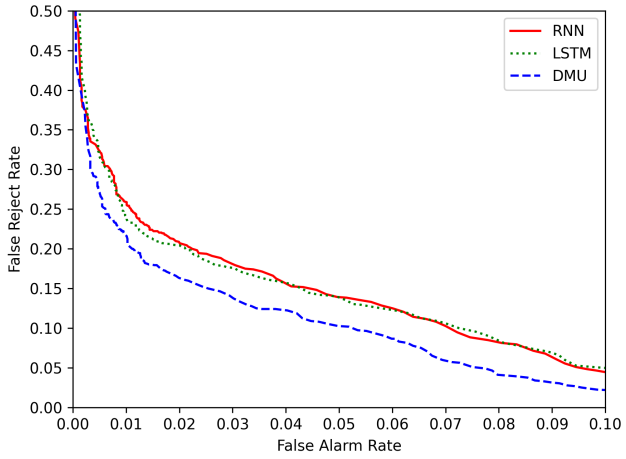

Fig. 2. Comparison of the false reject rate and false alarm rate between the proposed DMU model, RNN, and LSTM models on the wake-word detection task.

with those of Yılmaz et al. [47].

As presented in Table II**(c)**, our results demonstrate that the proposed DMU consistently outperforms other baseline models when using the same number of neurons. Moreover, as shown in Fig. 2, we notice that DMU exhibits higher robustness compared to RNN and LSTM models in terms of the False Rejection Rate (FRR) and False Alarm Rate (FAR).

Event-based Spoken Word Recognition: The Spiking Heidelberg Dataset (SHD) [7] was introduced to study event-based speech processing. It is constructed by converting raw audio waveforms into an event-based representation using a biologically plausible artificial cochlear model. In this study, we utilize the SHD dataset to evaluate the applicability and generalization of the DMU to other data modalities. Each sample in the SHD dataset comprises 700 channels, each corresponding to distinct frequency-sensitive neurons in the peripheral auditory system. To ensure sufficient temporal resolution while minimizing post-processing workload, we aggregate the spikes within each 10 ms time bin, leading to a total of 100 time steps per sample. For this task, we employ a two-layer recurrent neural network and utilize the readout integrator [49] for decoding, which has been shown to outperform the last time step decoding method [7]. The total number of delays n is set to 30 to encourage the DMU to learn short- and mid-range temporal dependencies between different phonetic units.

The results of the DMU, along with other SOTA recurrent spiking neural networks (RSNNs) and LSTM models, are provided in Table II**(b)**. Notably, our proposed DMU model has achieved the SOTA test results on this dataset, surpassing LSTM and Bi-LSTM models by 11.58% and 4.28%, respectively, while using fewer than half and a quarter of their respective network parameters. This result also competes favorably with the SOTA RSNN model that employs temporal attention and advanced data augmentation techniques [46]. It's worth noting that when networks are trained with the last time step loss, all SNNs fail due to the requirement of long-range temporal credit assignment. Similarly, the LSTM can only achieve a test accuracy of 56.63%, while our DMU attains 78.71%.

C. Radar Gesture Recognition

Here, we perform a radio-frequency-based gesture recognition task using the SoLi dataset [45]. In this dataset, a millimeter-wave radar captures reflected energy, mapping it to pixel intensity. We design the task to process radar samples sequentially, frame by frame, making a decision after the last frame has been read. Following the recommendation by Yin et al. [49], we employ a single channel instead of four, as it has been demonstrated to contain ample discriminative information [49]. Our network architecture consists of a recurrent layer with 512 neurons, followed by a dense layer with 512 neurons, and a final classification layer with 11 neurons. To capture the desired temporal dependencies of this dataset effectively, we set the total number of delays to 40, ensuring an adequate range of temporal information integration.

As the test results reported in Table II**(d)**, CNN models are inferior to both vanilla and gated RNN models in this context. This aligns with the insight that the radar sensor mainly captures hand pose changes, which convey more information compared to absolute positions [28]. Same as the previous tasks, the DMU outperforms the vanilla RNN by 4.4%, with only a slight increase in the total parameter count. Additionally, it surpasses the performance of the LSTM model by 2.16%, utilizing less than a third of the LSTM model's parameters.

D. ECG Waveform Segmentation

The electrocardiogram (ECG) is a vital tool used by clinicians to evaluate a patient's cardiovascular system. For this task, we employ the QTDB dataset [25] to assess the predictive capability of our proposed DMU model. The objective is to segment the ECG signal into distinct characteristic waveforms (Normal, P, QR, RS, and T). Clinicians can then diagnose based on the shape and duration of these waveforms. The QTDB dataset comprises 105 ECG records, each with a duration of 15 minutes. With a sampling frequency of 250 Hz, we pad shorter signals with zeros to achieve a consistent length of 300 data points. Each recording in the QTDB dataset features two input channels. Inspired by the design of ECGNet [1], our network incorporates two recurrent layers, each containing 32 neurons. Considering the short- to mid-range temporal dependencies required in this task, we set the DMU delay to 30. As evidenced in Table II**(e)**, the DMU consistently outperforms other competitive baseline models, with similar or fewer parameters, showcasing its superior temporal modeling capability.

E. Permuted Sequential Image Classification

The permuted sequential MNIST (PS-MNIST) dataset [26] has been designed to evaluate the capability of RNN models in modeling long-range temporal dependencies. This dataset is derived from the MNIST dataset by first flattening each image into a 784-dim vector, then shuffling its spatial information

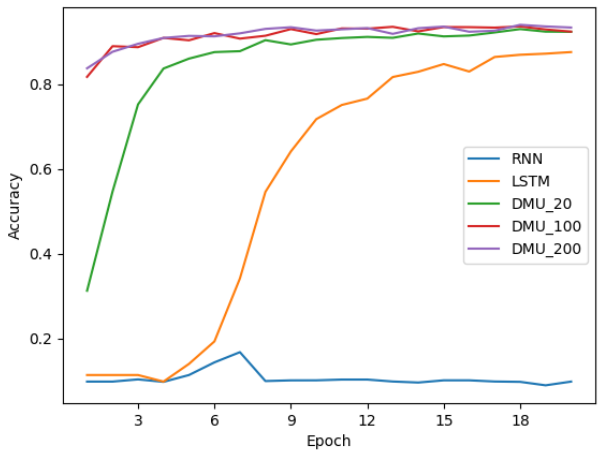

Fig. 3. Comparison of the learning curves of different models on the PS-MNIST dataset with a temporal duration of 784 time steps. This dataset has been specifically designed to test the model's ability to retain long-term memory between pixels that may be widely separated. All the models use 200 hidden neurons, and DMU_n denotes the DMU model with the number of delays of n.

with a consistent permutation vector. In our experiments, the elements of this vector are sequentially presented to the network, with decisions made after all pixels have been processed. This task demands the model to decipher the original temporal order and capture dependencies between different pixels. Following the experimental setup of Chandar et al. [4], our DMU is designed with a single recurrent layer with the number of delays n set to 80.

Results in Table II**(f)** highlight that our proposed DMU can achieve a test accuracy of 96.39% using only $49K$ parameters. This represents a substantial accuracy improvement of 6.53% over LSTM, despite employing only 30% of its parameters. Furthermore, our DMU is competitive with other recently introduced RNN models engineered for learning long-term dependencies, such as Lipschitz RNN, coRNN, τ-GRU [10], [11], [35], [36]. Although the τ-GRU demonstrates remarkable performance in this challenging task, it requires specialized training techniques to achieve the peak performance. We also provide the learning curves in Fig. 3 to compare the learning dynamics of our DMU with other baseline models. Notably, under identical training conditions and neuron count, the DMU converges rapidly within ten epochs, significantly surpassing the LSTM model.

IV. Discussions

In this section, we begin by analyzing the relationship between the DMU and recurrent memory. Following this, we conduct an ablation study to investigate the impact of various hyperparameters employed in the DMU. Additionally, we introduce a simple yet effective thresholding scheme that can significantly reduce the computational cost of the proposed DMU. Furthermore, we extend the application of the delay line structure to popular LSTM and GRU models, showcasing its generalizability. Finally, we provide evidence that the DMU enhances the predictive capability of the model.

A. Understand The Interplay Between Recurrent Memory and Delay Line

To delve deeper into the relationship between the memory established by recurrent connections and the delayed lines, we incorporated the DMU into a one-layer Independently Recurrent Neural Network (Indrnn) comprising 512 neurons [27]. The Indrnn architecture is defined as $h_t = \sigma(W_h x_t + U_h \odot h_{t-1})$, where U_h represents the self-recurrent weight vector that has the same dimensionality as the hidden state h_{t-1}, and $\odot$ denotes the Hadamard product. Notably, in an IndRNN layer, each neuron operates independently and maintains its own memory state, thereby facilitating independent analysis. In this analysis, we use the SHD dataset and the loss function is derived from the output of the last time step. Thus, the network is trained to retain long-term memory.

Fig. 4 illustrates the histograms of the learned recurrent weights U_h. While the conventional IndRNN layer achieves a test accuracy of 53%, integrating IndRNN with DMU leads to a significant performance enhancement, with accuracy reaching 78%. This notable improvement underscores the efficacy of the proposed DMU in enhancing temporal modeling. As shown in Fig. 4(a), a substantial portion of the weights in the vanilla Indrnn are close to 1, indicating these neurons are trained to retain long-term memory. In contrast, upon integration with DMU, the distribution of recurrent weights becomes more dispersive, suggesting a shift of the burden of modeling long-term dependencies to the delay line. Furthermore, the mean value of weights decreases with the increase in delay line length, attributed to the delay line's facilitation of more direct temporal credit assignment. Consequently, the model is encouraged to rely more on the delay line for establishing temporal dependencies.

B. Effect of the Number of Delays and Delay Dilation Factors

To investigate the impact of the number of delays n and the dilation factor τ on temporal classification performance, we conducted an ablation study on the SHD dataset. The network architecture chosen for this study can be represented as 700-128-128-20, wherein the numbers indicate the number of neurons in each respective layer. Additionally, we compared two decoding methods: last-time step (Last) and readout integrator (All) [49]. As depicted in Fig. 5**(a)**, without utilizing the delay line ($n = 0$), the classification accuracy remains at 5%. This observation suggests that incorporating the delay line significantly enhances the capability of the vanilla RNN in sequential modeling. Furthermore, the 'Last' decoding approach demonstrates improved performance with longer delay lines, indicating the importance of long-range temporal dependency for this task, as longer delay lines facilitate the establishment of such dependency. In contrast, the 'All' method utilizes both the integrator and the delay lines to establish temporal dependencies. Our experimental results suggest that incorporating longer delays generally improves performance, though it also leads to higher memory consumption. Once the

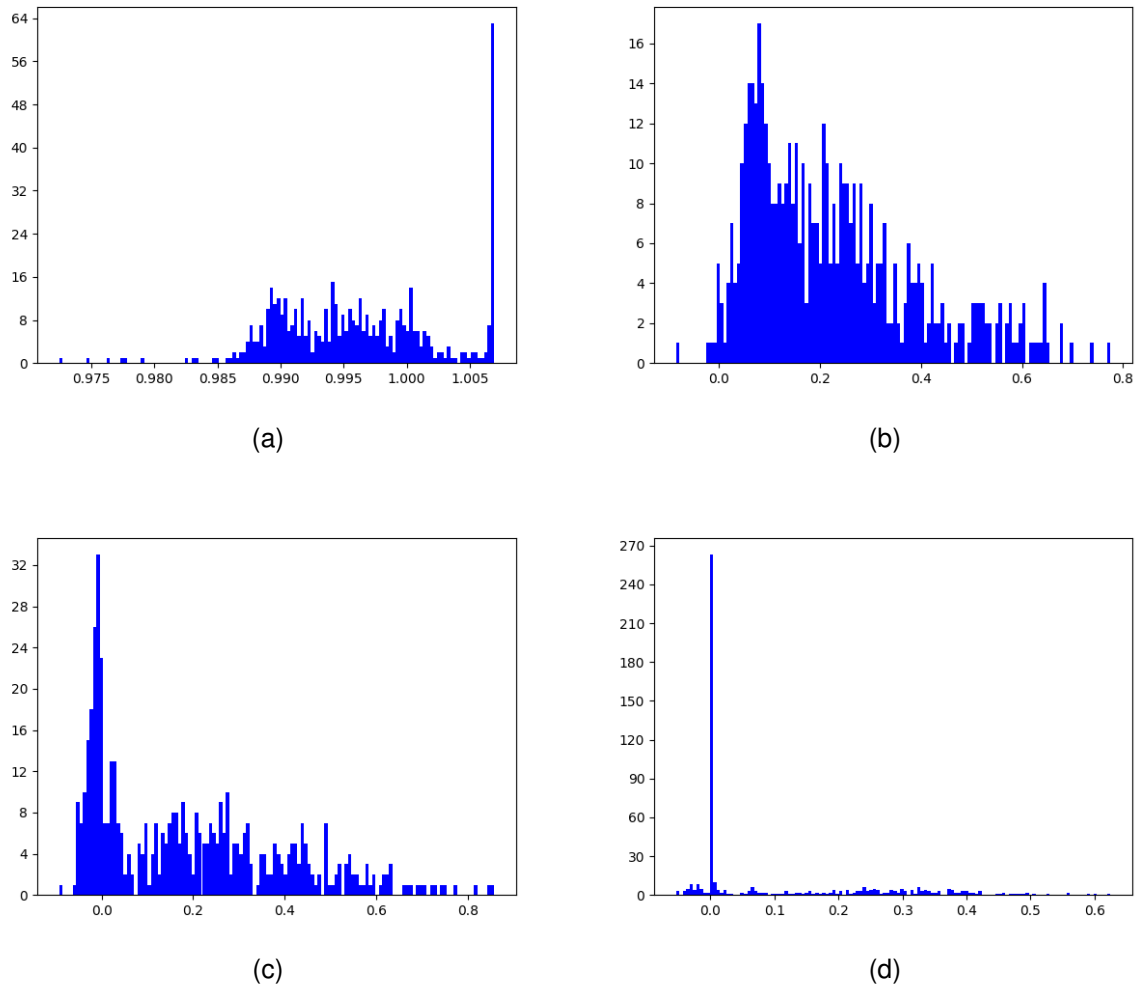

Fig. 4. Histograms of the learned recurrent weights on the SHD dataset. (a) Indrnn. (b) Indrnn+DMU (20 delays). (c) Indrnn+DMU (60 delays). (d) Indrnn+DMU (100 delays). The x-axis represents the value of the recurrent weights and the y-axis represents the frequency.

delays surpass a certain threshold, the model's performance begins to saturate since the delay line has already fulfilled the necessary range of temporal dependencies as shown in both Fig. 5**(a)** and Fig. 3. In practice, we have discovered that setting delay lengths between 30 and 80 time steps proves effective in achieving competitive results compared to gated RNNs across various tasks and datasets.

Figure 5**(b)** presents the results of the 'Last' decoding method using different configurations of τ. Our experimental results demonstrate that performance can be maintained at a promising level when τ is limited to a value of five or smaller. However, it is important to note that excessively high values beyond this threshold, which correspond to prolonged skip intervals, can result in the loss of fine-grained temporal information. This loss, in turn, has a detrimental effect on the model's performance.

C. Delay Gate Thresholding Scheme to Reduce Memory Usage

Here, we conduct a detailed analysis on how the proposed thresholding scheme affects both model performance and memory usage. In Fig. 6, we present the results of applying various hard thresholds θ directly during inference for the SHD dataset. In the inset, it can be observed that the accuracy shows a slight improvement as the threshold value increases. This enhancement can be attributed to certain delay gates that unintentionally allow noise to pass through. By shutting these gates, the network effectively mitigates such noise interference, leading to improved performance. Interestingly, there is an initial significant reduction in computational overhead as the threshold increases. It should be noted that even when only 4 delay gates remain active, the proposed DMU achieves an accuracy surpassing 87%, which represents SOTA performance. This highlights the effectiveness of the strategy, providing similar effect to the introduction of delay gate dilation. However, the proposed thresholding scheme represents a more dynamic and adaptable approach to achieving these improvements.

D. Empowering Gated RNNs with DMU's Delay Line Mechanism

To further demonstrate the effectiveness of the DMU's delay line mechanism, we integrate it into well-established

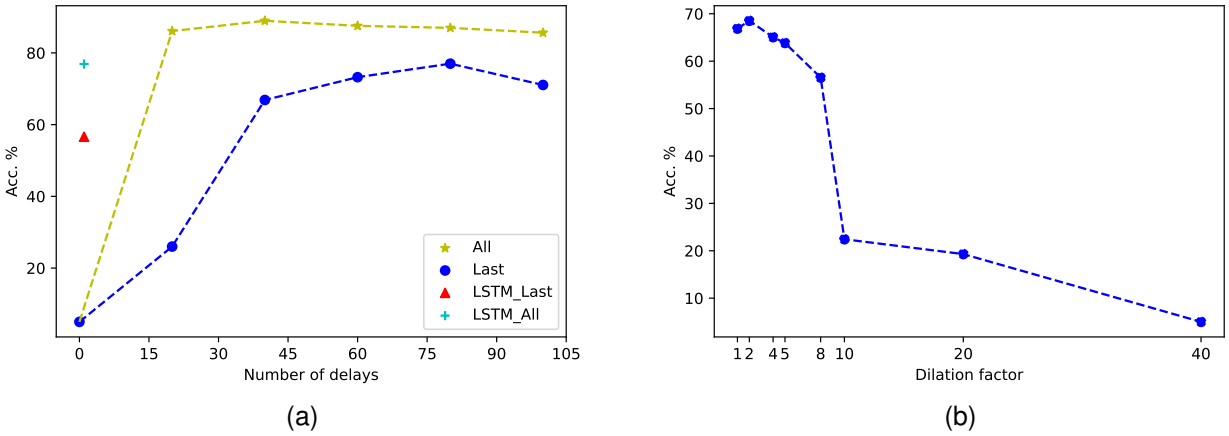

Fig. 5. **(a)** Comparison of performance for two decoding methods in DMU: Last time step loss (Last) and readout integrator (All). The X-axis represents the number of delays n in a delay line and the Y-axis indicates classification accuracy. "LSTM_Last" represents the LSTM coupled with the "Last" decoding method, while "LSTM_All" stands for LSTM coupled with the "All" decoding method. **(b)** The effect of the dilation factor τ on the classification accuracy. A fixed total delay of 40 has been used.

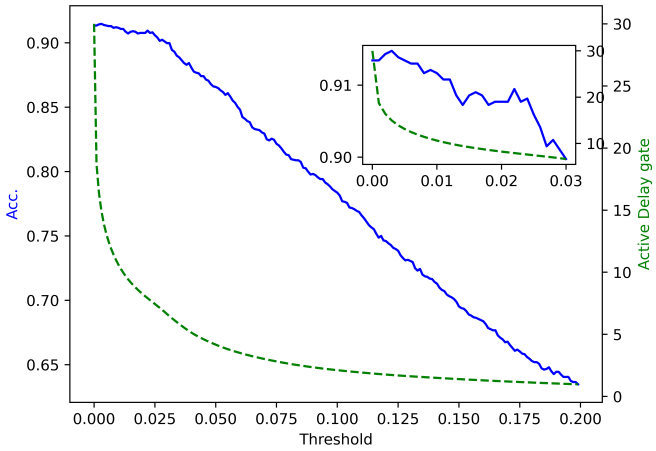

Fig. 6. Illustration of the impact of delay gate threshold on classification accuracy and the number of active delay gates. The total delay line length $n = 30$. The left Y-axis corresponds to accuracy, while the right Y-axis represents the number of active delay gate.

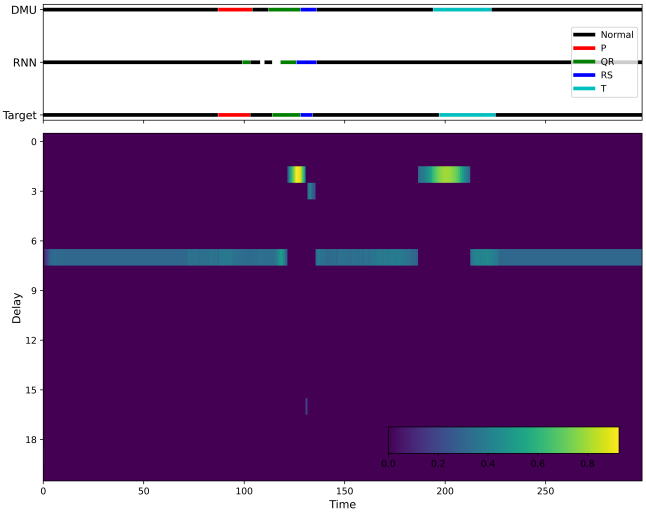

Fig. 7. Illustration of the delay gate values over time. A delay line with 21 delays has been used in this study. The top figure shows the output, presented from top to bottom: the target, the predicted label from the RNN model, and the predicted label from the DMU.

gated RNN architectures, namely GRU and LSTM. In this study, both models are configured to have a single recurrent layer consisting of 512 units. The results obtained on the SHD dataset, as presented in Table III, clearly illustrate the advantages of incorporating the delay lines. Specifically, with a modest increase of only 0.88% in parameters, the LSTM model achieves a remarkable improvement of 15.99% in accuracy on the SHD dataset. Similarly, the GRU model, when combined with the delay line mechanism, also exhibits substantial performance enhancements of nearly 10%.

TABLE III
COMPARISON OF LSTM AND GRU MODEL PERFORMANCE ON THE SHD DATASET WITH AND WITHOUT INCORPORATING DELAY LINES.

Method	Accuracy	Increased Parameters
GRU	78.87%	-
DMU-GRU (n = 30)	**88.43%**	1.17%
LSTM	73.28%	-
DMU-LSTM (n = 30)	**89.27%**	0.88%

E. DMU Enhance Model's Predictive Capability

In contrast to conventional RNNs that summarize and store historical information in hidden states, the proposed DMU possesses the unique capability of directly projecting historical information into the future. This enables direct interaction and integration of temporally separated information, which greatly facilitates temporal pattern recognition. To demonstrate this capability, we conducted an experiment on the ECG dataset

using a 1-layer RNN combined with a delay line of $n = 21$. Accurately pinpointing the starting and ending points of the 'RS' and 'T' waves in this task is challenging due to their potentially biphasic nature [12]. As illustrated in Fig. 7, it is evident that the vanilla RNN struggles in accurately identifying the 'RS' and 'T' waves. In contrast, when equipped with the DMU, the network reliably projects information 7 time steps ahead, aligning with the temporal gap between the subsequent labeling points, which is approximately 7. Furthermore, the DMU dynamically toggles to a delay of 2 as the label is going to switch to 'RS' and 'T'. These sequential projections of information allow for the aggregation of relevant information, enabling the accurate classification of the waveform.

V. Conclusion

In this study, we introduce DMU, a novel RNN architecture tailored for sequential modeling tasks. This model facilitates the direct establishment of temporal dependencies through its unique delay gate mechanism. Analysis of the distribution of learned memory and gate values suggests that the DMU enables the network to account for past events compared to conventional recurrent integration methods. Our experiments, spanning across audio processing (including classification and keyword detection), radar gesture recognition, ECG streaming classification, and Permuted sequential MNIST, have confirmed the effectiveness of the proposed methods. Notably, the DMU outperforms SOTA gated-RNN models in these tasks with significantly reduced model parameters. Furthermore, we propose two effective strategies to reduce the memory cost of DMU, including dilated delay line and delay gate thresholding schemes. In the present study, the delay line operates at a uniform temporal scale governed by the dilation factor of the delay gate, which strikes a balance between computational efficiency and temporal modeling performance. However, real-world temporal signals exhibit intricate multiscale temporal dynamics. For instance, speech signals encompass various levels of structure, including phonemes, syllables, and words. Effectively and efficiently capturing such multiscale temporal patterns using a delay line remains an open question that necessitates further exploration in subsequent research efforts.

References

[1] Hedayat Abrishami, Matthew Campbell, Chia Han, Richard Czosek, and Xuefu Zhou. P-qrs-t localization in ecg using deep learning. In *2018 IEEE EMBS International Conference on Biomedical & Health Informatics (BHI)*, pages 210–213. IEEE, 2018.

[2] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.

[3] Catherine E Carr and Masakazu Konishi. Axonal delay lines for time measurement in the owl's brainstem. *Proceedings of the National Academy of Sciences*, 85(21):8311–8315, 1988.

[4] Sarath Chandar, Chinnadhurai Sankar, Eugene Vorontsov, Samira Ebrahimi Kahou, and Yoshua Bengio. Towards non-saturating recurrent units for modelling long-term dependencies. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3280–3287, 2019.

[5] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[6] Alice Coucke, Mohammed Chlieh, Thibault Gisselbrecht, David Leroy, Mathieu Poumeyrol, and Thibaut Lavril. Efficient keyword spotting using dilated convolutions and gating. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6351–6355. IEEE, 2019.

[7] Benjamin Cramer, Yannik Stradmann, Johannes Schemmel, and Friedemann Zenke. The heidelberg spiking data sets for the systematic evaluation of spiking neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

[8] Steven Davis and Paul Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE transactions on acoustics, speech, and signal processing*, 28(4):357–366, 1980.

[9] Gregoire Deletang, Anian Ruoss, Jordi Grau-Moya, Tim Genewein, Li Kevin Wenliang, Elliot Catt, Chris Cundy, Marcus Hutter, Shane Legg, Joel Veness, et al. Neural networks and the chomsky hierarchy. In *The Eleventh International Conference on Learning Representations*, 2022.

[10] N Benjamin Erichson, Omri Azencot, Alejandro Queiruga, Liam Hodgkinson, and Michael W Mahoney. Lipschitz recurrent neural networks. In *International Conference on Learning Representations*, 2020.

[11] N Benjamin Erichson, Soon Hoe Lim, and Michael W Mahoney. Gated recurrent neural networks with weighted time-delay feedback. *arXiv preprint arXiv:2212.00228*, 2022.

[12] Derek V Exner. Noninvasive risk stratification after myocardial infarction: rationale, current evidence and the need for definitive trials. *Canadian Journal of Cardiology*, 25:21A–27A, 2009.

[13] Aysu Ezen-Can. A comparison of lstm and bert for small corpus. *arXiv preprint arXiv:2009.05451*, 2020.

[14] Amin Faraji, Sayed Alireza Sadrossadat, Weicong Na, Feng Feng, and Qi-Jun Zhang. A new macromodeling method based on deep gated recurrent unit regularized with gaussian dropout for nonlinear circuits. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 2023.

[15] John S Garofolo. Timit acoustic phonetic continuous speech corpus. *Linguistic Data Consortium, 1993*, 1993.

[16] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6645–6649. Ieee, 2013.

[17] Ilyass Hammouamri, Ismail Khalfaoui-Hassani, and Timothée Masquelier. Learning delays in spiking neural networks using dilated convolutions with learnable spacings. *arXiv preprint arXiv:2306.17670*, 2023.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[20] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[21] Shaista Hussain, Arindam Basu, Runchun Mark Wang, and Tara Julia Hamilton. Delay learning architectures for memory and classification. *Neurocomputing*, 138:14–26, 2014.

[22] Fatih Ilhan, Oguzhan Karaahmetoglu, Ismail Balaban, and Suleyman Serdar Kozat. Markovian rnn: An adaptive time series prediction network with hmm-based switching for nonstationary environments. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–14, 2021.

[23] Lloyd A Jeffress. A place theory of sound localization. *Journal of comparative and physiological psychology*, 41(1):35, 1948.

[24] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[25] Pablo Laguna, Roger G Mark, A Goldberg, and George B Moody. A database for evaluation of algorithms for measurement of qt and other waveform intervals in the ecg. In *Computers in cardiology 1997*, pages 673–676. IEEE, 1997.

[26] Quoc V Le, Navdeep Jaitly, and Geoffrey E Hinton. A simple way to initialize recurrent networks of rectified linear units. *arXiv preprint arXiv:1504.00941*, 2015.

[27] Shuai Li, Wanqing Li, Chris Cook, Ce Zhu, and Yanbo Gao. Independently recurrent neural network (indrnn): Building a longer and deeper rnn. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5457–5466, 2018.

[28] Jaime Lien, Nicholas Gillian, M Emre Karagozler, Patrick Amihood, Carsten Schwesig, Erik Olson, Hakim Raja, and Ivan Poupyrev. Soli:

Ubiquitous gesture sensing with millimeter wave radar. *ACM Transactions on Graphics (TOG)*, 35(4):1–19, 2016.
[29] Kouichi Murakami and Hitomi Taguchi. Gesture recognition using recurrent neural networks. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 237–242, 1991.
[30] Zihan Pan, Malu Zhang, Jibin Wu, Jiadong Wang, and Haizhou Li. Multi-tone phase coding of interaural time difference for sound source localization with spiking neural networks. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:2656–2670, 2021.
[31] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *International conference on machine learning*, pages 1310–1318. PMLR, 2013.
[32] M. Ravanelli, T. Parcollet, and Y. Bengio. The pytorch-kaldi speech recognition toolkit. In *In Proc. of ICASSP*, 2019.
[33] Mirco Ravanelli, Philemon Brakel, Maurizio Omologo, and Yoshua Bengio. Improving speech recognition by revising gated recurrent units. *arXiv preprint arXiv:1710.00641*, 2017.
[34] Mirco Ravanelli, Philemon Brakel, Maurizio Omologo, and Yoshua Bengio. Light gated recurrent units for speech recognition. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2(2):92–102, 2018.
[35] T Konstantin Rusch and Siddhartha Mishra. Coupled oscillatory recurrent neural network (cornn): An accurate and (gradient) stable architecture for learning long time dependencies. In *International Conference on Learning Representations*, 2020.
[36] T Konstantin Rusch, Siddhartha Mishra, N Benjamin Erichson, and Michael W Mahoney. Long expressive memory for sequence modeling. *arXiv preprint arXiv:2110.04744*, 2021.
[37] Haşim Sak, Andrew Senior, and Françoise Beaufays. Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition. *arXiv preprint arXiv:1402.1128*, 2014.
[38] Sumit B Shrestha and Garrick Orchard. Slayer: Spike layer error reassignment in time. *Advances in neural information processing systems*, 31, 2018.
[39] Pengfei Sun, Yansong Chua, Paul Devos, and Dick Botteldooren. Learnable axonal delay in spiking neural networks improves spoken word recognition. *Frontiers in Neuroscience*, 17:1275944, 2023.
[40] Pengfei Sun, Ehsan Eqlimi, Yansong Chua, Paul Devos, and Dick Botteldooren. Adaptive axonal delays in feedforward spiking neural networks for accurate spoken word recognition. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023.
[41] Pengfei Sun, Longwei Zhu, and Dick Botteldooren. Axonal delay as a short-term memory for feed forward deep spiking neural networks. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8932–8936. IEEE, 2022.
[42] Aboozar Taherkhani, Ammar Belatreche, Yuhua Li, and Liam P Maguire. Dl-resume: A delay learning-based remote supervised method for spiking neurons. *IEEE transactions on neural networks and learning systems*, 26(12):3137–3149, 2015.
[43] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.
[44] Kyriakos Voutsas and JÜrgen Adamy. A biologically inspired spiking neural network for sound source lateralization. *IEEE Transactions on Neural Networks*, 18(6):1785–1799, 2007.
[45] Saiwen Wang, Jie Song, Jaime Lien, Ivan Poupyrev, and Otmar Hilliges. Interacting with soli: Exploring fine-grained dynamic gesture recognition in the radio-frequency spectrum. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, pages 851–860. ACM, 2016.
[46] Man Yao, Huanhuan Gao, Guangshe Zhao, Dingheng Wang, Yihan Lin, Zhaoxu Yang, and Guoqi Li. Temporal-wise attention spiking neural networks for event streams classification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10221–10230, 2021.
[47] Emre Yılmaz, Ozgür Bora Gevrek, Jibin Wu, Yuxiang Chen, Xuanbo Meng, and Haizhou Li. Deep convolutional spiking neural networks for keyword spotting. In *Proceedings of INTERSPEECH*, pages 2557–2561, 2020.
[48] Bojian Yin, Federico Corradi, and Sander M Bohté. Effective and efficient computation with multiple-timescale spiking recurrent neural networks. In *International Conference on Neuromorphic Systems 2020*, pages 1–8, 2020.
[49] Bojian Yin, Federico Corradi, and Sander M Bohté. Accurate and efficient time-domain classification with adaptive spiking recurrent neural networks. *Nature Machine Intelligence*, 3(10):905–913, 2021.
[50] Malu Zhang, Jibin Wu, Ammar Belatreche, Zihan Pan, Xiurui Xie, Yansong Chua, Guoqi Li, Hong Qu, and Haizhou Li. Supervised learning in spiking neural networks with synaptic delay-weight plasticity. *Neurocomputing*, 409:103–118, 2020.
[51] Yibin Zheng, Xinhui Li, Fenglong Xie, and Li Lu. Improving end-to-end speech synthesis with local recurrent neural network enhanced transformer. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6734–6738. IEEE, 2020.
[52] Guangming Zhu, Liang Zhang, Lu Yang, Lin Mei, Syed Afaq Ali Shah, Mohammed Bennamoun, and Peiyi Shen. Redundancy and attention in convolutional lstm for gesture recognition. *IEEE Transactions on Neural Networks and Learning Systems*, 31(4):1323–1335, 2020.

VI. Biography Section

Pengfei Sun is currently a PhD candidate at Ghent University. He was a research engineer at the Agency for Science, Technology and Research (A*STAR), Singapore, and Institute of Neuroinformatics, University of Zurich and ETH Zurich, Switzerland from 2018 to 2021.

His research interests include brain-inspired neural networks, delay learning, neuromorphic applications, EEG decoding, and cognitive modeling.

Jibin Wu received the B.E. and Ph.D degree in Electrical Engineering from National University of Singapore, Singapore in 2016 and 2020, respectively. Dr. Wu is currently an Assistant Professor in the Department of Data Science and Artificial Intelligence and the Department of Computing, The Hong Kong Polytechnic University. His research interests broadly include brain-inspired artificial intelligence, neuromorphic computing, computational audition, speech processing, and machine learning. Dr. Wu has published over 40 papers in prestigious conferences and journals in artificial intelligence and speech processing, including NeurIPS, ICLR, AAAI, TPAMI, TNNLS, TASLP, Neural Networks, and IEEE JSTSP. He is currently serving as the Associate Editors for IEEE Transactions on Neural Networks and Learning Systems and IEEE Transactions on Cognitive and Developmental Systems.

Malu Zhang Malu Zhang (Member, IEEE) received the Ph.D. degree in computer science from the University of Electronic Science and Technology of China, Chengdu, China, in 2019.

From 2019 to 2022, he was a Research Fellow with the HLT Laboratory, Department of Electrical and Computer Engineering, National University of Singapore, Singapore. He is currently a Professor with the Computational Intelligence Laboratory, School of Computer Science and Engineering, University of Electronic Science and Technology of China. His research interests include spiking neural networks, neural spike encoding, and neuromorphic applications. Dr. Zhang is now an Associate Editor of the IEEE Transactions on Emerging Topics in Computational Intelligence.

Devos Paul obtained a MSc degree in Physics (1987) and Physical Engineering (1989), both from Ghent University, and a PhD in Physics (1995) from EPFL (Lausanne, Switzeland). After being active in industry and education, he has become a member of the WAVES research group at Ghent University in 2015, where he is currently appointed as an Associate Professor. His research centers on multidisciplinary studies that place life at the forefront, exploring how acoustics, electronic engineering, and artificial intelligence can make meaningful contributions. He builds on expertise in smart instrumentation, biomedical electronics, signal processing and acoustics, including soundscape studies. He initiated the AcustiCare project, which leverages soundscape interventions to enhance the well-being of individuals with dementia and participates in research projects regarding brain-inspired artificial intelligence. In order to contribute to societal needs his current interest is in smart environmental sensing, including eco- and bioacoustics for both terrestrial and aquatic environments. He is a member of the European Acoustics Association and has shared his research in over 90 journal articles and conference contributions.

Dick Botteldooren obtained a MSc in electronical engineering in 1986 and PhD in engineering in 1990 from Ghent University. After working as a permanent researcher with the Belgian fund for scientific research, he became full professor with Ghent university in 2000.

His research interests include all aspects of environmental sound, perception, impact of sound on people and society, and bio-inspired artificial intelligence. More specifically this has including sound generation (e.g. road traffic), propagation (e.g. sensor networks and modeling), urban sound planning, perception (e.g. soundscape, brain monitoring, and EEG), deviant hearing (e.g. dementia, Parkinson's, hearing loss), and machine listening (e.g. artificial intelligence, human-machine interaction). His research was reported in 218 journal papers and hundreds of conference contributions.

Dick Botteldooren is currently the president of the European Acoustics Association, he was an advisor for national and international health councils and noise policy makers, and he is a member of the EC noise expert group. He is also a fellow of the Acoustical Society of America, a Distinguished Fellow of the International Institute for Acoustics and Vibration, and a member of IEEE.