Adaptive Synapse Generation in Cortical Learning Algorithm with Dynamic Range

Takeru Aoki
Department of Information and Computer Technology
Faculty of Engineering, Tokyo University of Science
Tokyo, Japan
aoki@rs.tus.ac.jp

Tomoaki Tatsukawa
Department of Information and Computer Technology
Faculty of Engineering, Tokyo University of Science
Tokyo, Japan
tatsukawa@rs.tus.ac.jp

***Abstract*—The Cortical Learning Algorithm (CLA) is a brain-inspired framework modeled after the human neocortex. By representing and predicting time-series data using sparse distributed representations, CLA achieves lower computational costs and superior adaptability to shifting trends compared to conventional artificial neural networks, making it well-suited for online time-series forecasting. However, CLA-DR (Dynamic Range), which eliminates predefined input range constraints by dynamically adding synapses, often suffers from decreased prediction accuracy due to underutilization of the allocated predictor. To address this issue, this study proposes an adaptive synapse generation method for the CLA-DR based on bias in the input data distribution. Experimental results using both synthetic and real-world datasets demonstrate that the proposed method achieves superior prediction accuracy compared to existing CLA models and a related LSTM approach by utilizing the predictor more effectively.**

***Index Terms*—Cortical Learning Algorithm, time-series forecasting, continuous learning**

I. Introduction

Time-series forecasting is a fundamental technology for decision-making and anomaly detection. In this field, continuous learning is crucial because the volume of time-series data grows over time and the diverse non-stationarity inherent in such data is difficult to eliminate. One framework for continuous learning and time-series forecasting is Hierarchical Temporal Memory (HTM) [1], [2], which is modeled after the functions of the human neocortex. The Cortical Learning Algorithm (CLA) [3]–[5] is one of the algorithms based on HTM and is an effective method for continuous learning. CLA converts the temporal context of inputs into sparse distributed representations, enabling it to construct a predictor that is resilient to shifting data trends. Furthermore, CLA achieves online learning by continuously updating the predictor as it receives time-series data. While Long Short-Term Memory (LSTM) [6], a type of recurrent neural network, is a widely used method for time-series forecasting, it is primarily suited for offline learning. Consequently, CLA has been reported to outperform LSTM in online forecasting tasks such as taxi demand and electricity consumption [7], [8].

The authors previously proposed CLA with Dynamic Range (CLA-DR) [9], which eliminates the predefined input range constraints required in CLA. However, this method fails to account for biases in the input data distribution. Consequently, parts of the predictor remain underutilized when the input data exhibit such bias. In this study, we propose a method for adaptive synapse generation based on input bias to address this limitation and improve prediction accuracy through more effective utilization of the predictor. To verify the effectiveness of the proposed method, we compare its prediction accuracy with conventional CLA and LSTM across multiple synthetic and real-world time-series datasets.

This work was supported by JSPS KAKENHI Grant Number 25K21303.

II. Cortical Learning Algorithm (CLA)

This section reviews the conventional CLA [3]–[5] as an existing method, and CLA with Dynamic Range (CLA-DR) [9] as a previously proposed method.

A. Conventional CLA

The structure of the CLA predictor is shown in Fig. 1. The predictor consists of n_c columns, and each column contains n_r cells. Column synapses connect input bits to columns, while cell synapses connect each cell to cells in other columns. The input value $X(t)$ is represented as a bit string of length n, denoted as $\boldsymbol{x}(t) = (x_0, x_1, \ldots, x_{n-1}) \in \{0,1\}^n$. The value of each bit x_i at time t is determined by the following equation:

$$x_i = \begin{cases} 1, & \text{if } h(X(t)) < i \le h(X(t)) + \delta \\ 0, & \text{otherwise} \end{cases} \quad (i = 0, 1, \ldots, n-1) \tag{1}$$

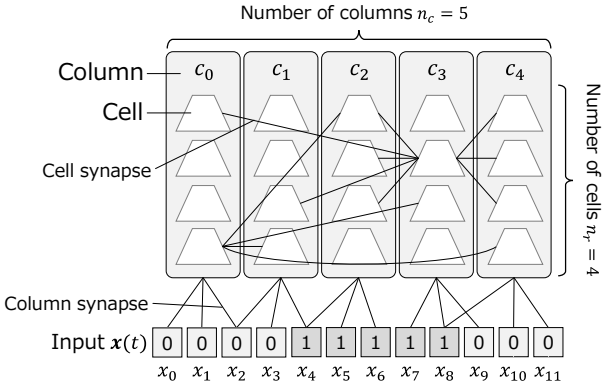


Fig. 1. Structure of CLA predictor

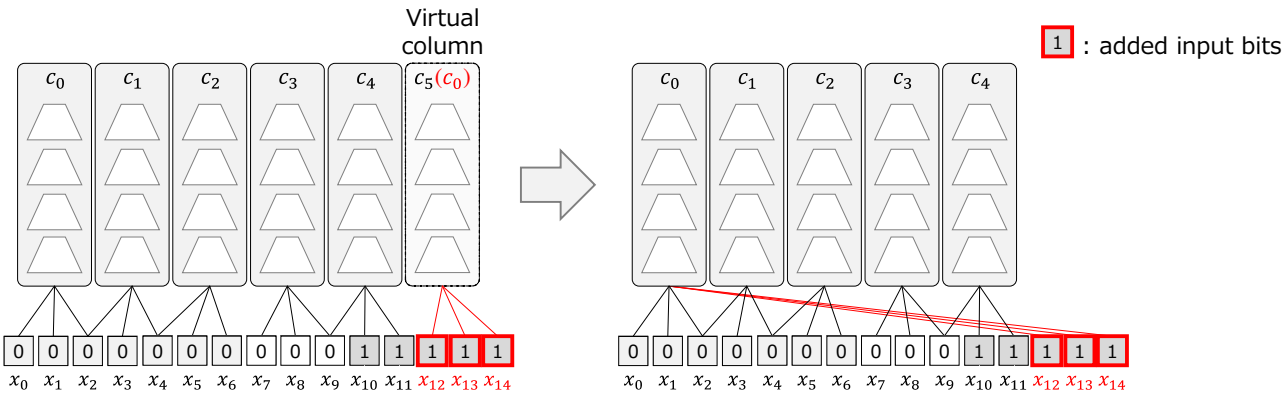


Fig. 2. CLA-DR: add column synapse

$$h(X(t)) = \frac{(X(t) - X^{\min}) \cdot (n - \delta)}{X^{\max} - X^{\min}} - 0.5 \qquad (2)$$

where δ denotes the length of a chunk of 1s in the bit string. $X^{\max}$ and $X^{\min}$ represent the maximum and minimum values of the input, respectively.

At each time step t, the input data are represented by a set of active columns. This is achieved by activating n_{ac} columns that have the largest number of column synapses connected to input bits with a value of 1. Furthermore, the column synapses of the active columns are updated so that the same set of columns is activated in response to the same input value.

Next, the input data context is represented by a set of active cells. Specifically, if an active column contains predictive cells, those cells become active; otherwise, all cells in the column are activated. Furthermore, the cell synapses of the active cells are updated so that the same set of cells is activated in response to the same input context. Then, cells with n_p synapses connected to active cells become predictive, representing the input data for the next time step $t + 1$.

Finally, the set of predictive cells is converted into a predicted value at time $t + 1$. In this study, we employ CLA-CD (Column-based Decoder) [10], which computes predicted values using the column synapses constructed in the above procedure. CLA repeats this procedure at each time step to achieve online learning.

B. CLA-DR

The conventional CLA requires pre-setting the maximum value $X^{\max}$ and minimum value $X^{\min}$ to convert input values into bit strings. This requirement poses difficulties for online learning and leads to degraded prediction accuracy. To address these issues, we have previously proposed CLA-DR, which eliminates the need for upper and lower bounds on input values.

First, input bits are added when an input value exceeds $X^{\max}$ or falls below $X^{\min}$. Specifically, the range of i in Eq. (1) is expanded from $[0, n-1]$ to $i \in \mathbb{Z}$. By changing only the range of the input conversion, values outside the original bounds are converted into bit strings while preserving representations within the original range.

Next, segments, which are bundles of column synapses, are added to correspond to the added input bits. Specifically, as shown in Fig. 2, virtual columns are added according to the added bits, and segments are generated to connect these virtual columns to those bits. Then, assuming that the existing set of columns has a periodic or spiral structure, the generated segments are mapped onto the existing columns corresponding to the virtual columns. This allows the added input bits to be integrated into the internal representation of the predictor without adding new columns. Furthermore, the assumption of a periodic structure maintains the continuity of the input values represented by each column.

Finally, the range of the predicted value is restricted to the vicinity of the current input value. This is because a single column corresponds to multiple values due to the periodic structure assumption, necessitating the determination of an appropriate predicted value. Specifically, the maximum distance from the center bit of the chunk of 1s in the input bit string $\boldsymbol{x}(t)$ is set as a parameter $range$, and CLA-CD is applied within this prediction range.

III. Limitation of CLA-DR

In this study, we focus on the issue of inefficient column utilization in CLA-DR due to biases in the distribution of input values. As mentioned above, CLA-DR adds segments based on the assumption of a periodic structure and reassigns the added input bits. At this stage, the frequency distribution of each input value range is unknown. If this frequency is biased, it leads to imbalances in the frequency with which columns become active. CLA is designed under the assumption that all columns are utilized uniformly. Consequently, over-utilized columns fail to represent inputs with sufficient precision, while under-utilized columns fail to contribute effectively to the internal representation. This imbalance leads to degraded forecasting accuracy. Although this issue also occurs in the conventional CLA, it becomes more pronounced in CLA-DR because the total number of added input bits is also unknown.

On the other hand, CLA-AC (CLA with Adaptive Columns) [11] is an existing method that addresses the issues mentioned above. However, while CLA-AC assumes that each column contains only a single segment, CLA-DR allows each column

TABLE I
COMPARISON OF THE PROPOSED AND EXISTING METHODS

		Handling of bias in distribution of input range	
		Without	With
Multiple segments on each column	With	Conventional CLA	CLA-AC
	Without	CLA-DR	**Proposed Method**

to contain multiple segments. Therefore, directly integrating CLA-AC with CLA-DR is challenging. Table I illustrates the relationship between the existing methods and the proposed method.

IV. Proposed method : adaptive synapse generation in CLA-DR

To address this issue, we propose a method to adaptively generate column synapses according to the input distribution.

Fig. 3 shows an overview of the proposed method. First, the source and destination columns are determined based on the activation frequency a_i of each column over the recent t_w time steps. Specifically, columns where $a_i \leq \Theta_l$ are selected as source columns $\boldsymbol{C}^s$, while those where $a_i > \Theta_h$ are selected as destination columns $\boldsymbol{C}^d$. Here, Θ_l and Θ_h are hyperparameters. In Fig. 3, the blue columns c_0, c_1, and c_3 represent the source columns, while c_2 and c_4 represent the destination columns. Next, the number of generated segments g_i for each destination column is calculated using the following equation:

$$g_i = |\boldsymbol{C}^s| \frac{a_i - \Theta_h}{\Sigma_{c_j \in \boldsymbol{C}^d} a_j - \Theta_h} \ (c_i \in \boldsymbol{C}^d) \tag{3}$$

where $|\boldsymbol{C}^s|$ denotes the total number of source columns.

Then, segments are replicated from the destination columns to the source columns. For each destination column d, the segment $s_{d,j}$ that contributes most to its activation is selected for addition. In Fig. 3, segment $s_{2,0}$ from column c_2 and $s_{4,0}$ from column c_4 are selected and added to the source columns according to g_i. By selecting added segments in this manner, we aim to suppress the increase in the total number of segments and limit the corresponding input range for each column while effectively utilizing all columns.

Furthermore, the proposed method is applied at time steps $t = \{t \mid t \bmod t_w = 0,\ t \neq 0\}$, using the periodicity parameter t_w. This aims to stabilize the input distribution and ensure sufficient time for learning through synapse updates after generation.

One difference from the existing method, CLA-AC, is the handling of original segments in the source columns. In CLA-AC, since each column has only one segment, the original segment is deleted and replaced by a new one. In contrast, the proposed method retains the original segments and adds new ones, based on the assumption that columns may contain multiple segments. In addition, the proposed method includes a procedure to select which segments to replicate from the destination columns when multiple segments exist. These differences enable the generation of segments tailored to the underlying CLA structure.

V. Experimental settings

A. Datasets

As synthetic time-series datasets, we use three types of sine waves to verify the effectiveness of the proposed method under different trends in input value ranges: a sine wave with a constant range $X_{\text{sin}}(t)$, a sine wave with a gradually shifting trend $X_{\text{trend}}(t)$, and a sine wave with a sudden range change $X_{\text{change}}(t)$. The value of each input at time t is calculated using the following equations:

$$X_{sin}(t) = \frac{1}{2} \cdot \sin\left(\frac{(t-1)\cdot\pi}{50}\right) + \frac{1}{2} \tag{4}$$

$$X_{trend}(t) = \frac{t}{20000} + X_{sin}(t) \tag{5}$$

$$X_{change}(t) = \begin{cases} X_{sin}(t), & \text{if } 0 \leq t \leq 20000, \\ 1 + X_{sin}(t), & \text{if } 20001 \leq t \leq 40000 \end{cases} \tag{6}$$

The number of time steps is set to $t \in [1, 20, 000]$ for $X_{\text{sin}}(t)$ and $X_{\text{trend}}(t)$, and $t \in [1, 40, 000]$ for $X_{\text{change}}(t)$.

As a real-world dataset, we use electricity load data from the New York Independent System Operator (NYISO) [12]. Following a previous study [8], we applied preprocessing steps, including outlier removal, missing value imputation, and a moving average filtering [13]. The dataset was collected in New York City, collected at 15-minute intervals from May 1, 2007, to February 28, 2019, totaling $414,914$ time steps.

B. Algorithms

To verify the effectiveness of the proposed method, we compare four types of CLAs on the three synthetic datasets: conventional CLA [3]–[5], CLA-AC [11], CLA-DR [9], and CLA-DR-ASG (Adaptive Synapse Generation), which integrates the proposed method into CLA-DR. For the electricity load data, we further compare these CLAs with six LSTM models [6]. These LSTM models incorporate combinations of three optimization algorithms and two input window sizes.

C. Parameters and Evaluation Metric

Regarding the parameters, the input range $\{X^{\max}, X^{\min}\}$ for the conventional CLA and CLA-AC is set to the maximum and minimum values of each input. In contrast, CLA-DR and CLA-DR-ASG use $\{X^{\max}, X^{\min}\} = \{0, 1\}$ for the three synthetic datasets based on the base sine wave, and $\{X^{\max}, X^{\min}\} = \{4000, 8000\}$ for the electricity load data, reflecting the range in which most data points are concentrated. Common CLA parameters are set as follows: number of columns $n_c = 2,048$, number of cells per column $n_r = 32$, input bit length $n = 421$, chunk length $\delta = 21$, number of active columns $n_{ac} = 40$, and the threshold for predictive cells $n_p = 15$. For CLA-DR, the decoder prediction range is set to $range = 110$. These values are consistent with the default parameters of the conventional CLA [5] and those used in previous studies [8], [9], [11]. For the proposed method, the activation frequency thresholds for the source and destination columns are set to $\Theta_l = 0.01$ and $\Theta_h = 0.02$, respectively. These values are determined based on the average column

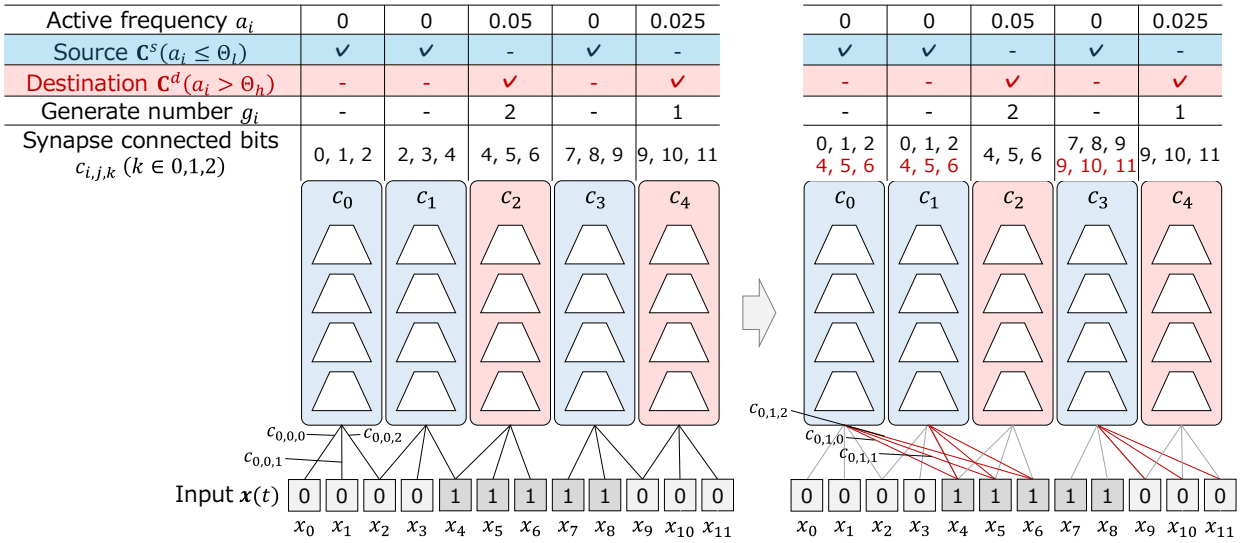


Fig. 3. Proposed method : adaptive synapse generation in CLA-DR

activation frequency $\frac{n_{ac}}{n_c} \approx 0.02$. Furthermore, the periodic parameter is set to $t_w = 1,000$ for the synthetic datasets and $t_w = 35,040$ for the electricity load data. These values are chosen considering the 100-step cycle of the sine waves and the one-year cycle of the electricity load data (4 steps/hour $\times$ 24 hours $\times$ 365 days $= 35,040$ steps). For LSTM, two window sizes $w = \{1, 100\}$ are applied to the input layer, and min-max normalization is applied to all inputs. The architecture consists of two hidden layers: a fully connected layer with 100 units using a linear activation function, and an LSTM layer with 100 units using a tanh activation function. The output layer uses a fully connected layer with a single unit and no activation function. To account for the influence of online learning, three optimization algorithms are applied: Adam [14], RMSprop [15], and SGD. The loss function uses the mean squared error.

For the evaluation metric, the prediction error at each time step is measured using Mean Absolute Error (MAE), where a lower MAE indicates higher prediction accuracy. Each method is evaluated over 30 independent trials for all inputs, and results are compared using the average MAE. Furthermore, we analyze the activation frequency of each column. The closer this frequency is to the average value ($\frac{n_{ac}}{n_c} \approx 0.02$), the more effectively the predictor is considered to be utilized.

VI. Experimental Results and Discussion

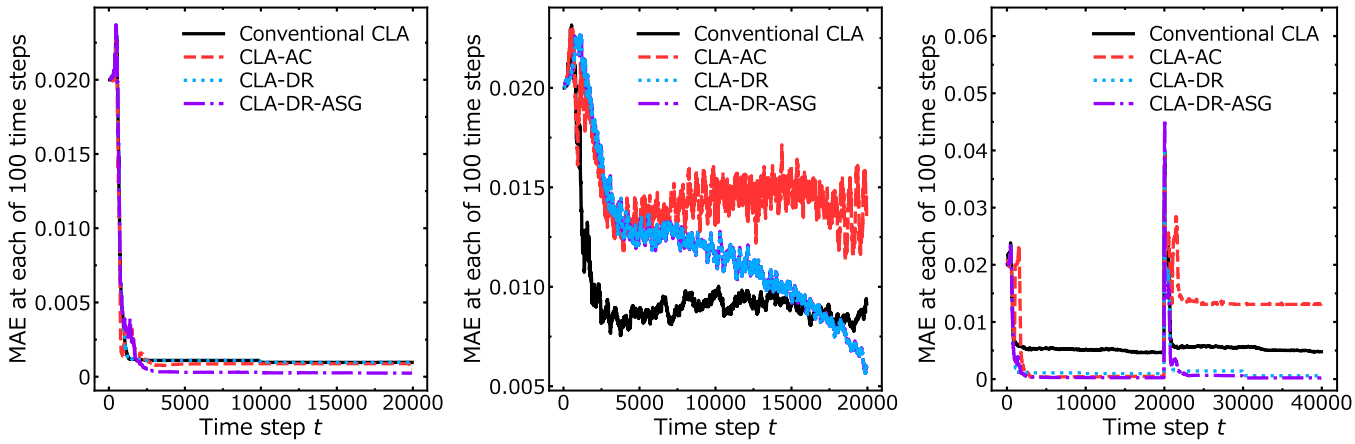
A. Results on synthetic Time-series Datasets

Fig. 4 shows the MAE for each CLA on the synthetic time-series datasets, and Fig. 5 shows the activation frequency of the columns. The MAE is calculated as an average over every 100 time steps, corresponding to one cycle of the data.

First, Fig. 4(a) shows the MAE for the sine wave with a constant range $X_{\sin}(t)$. It indicates that the prediction errors of the conventional CLA and CLA-DR are comparable. This is because the input lacks range shifts, and removing upper and lower bound constraints has no effect. In contrast, CLA-AC achieves a smaller prediction error after convergence than these two methods. Furthermore, the proposed CLA-DR-ASG achieves the best prediction accuracy among the compared methods. Regarding the activation frequency of columns in Fig. 5(a), the conventional CLA and CLA-DR have several columns whose activation frequencies significantly exceed the mean value of 0.02. On the other hand, such columns disappear in CLA-AC, and the number of unused columns is also reduced. This suggests that the predictor is utilized more effectively by arranging synapses to match the bias in the sine wave's range. Moreover, the proposed CLA-DR-ASG eliminates the unused columns that remain in CLA-AC, demonstrating that all columns are fully utilized. This indicates that allowing multiple segments per column leads to more efficient utilization of the available columns.

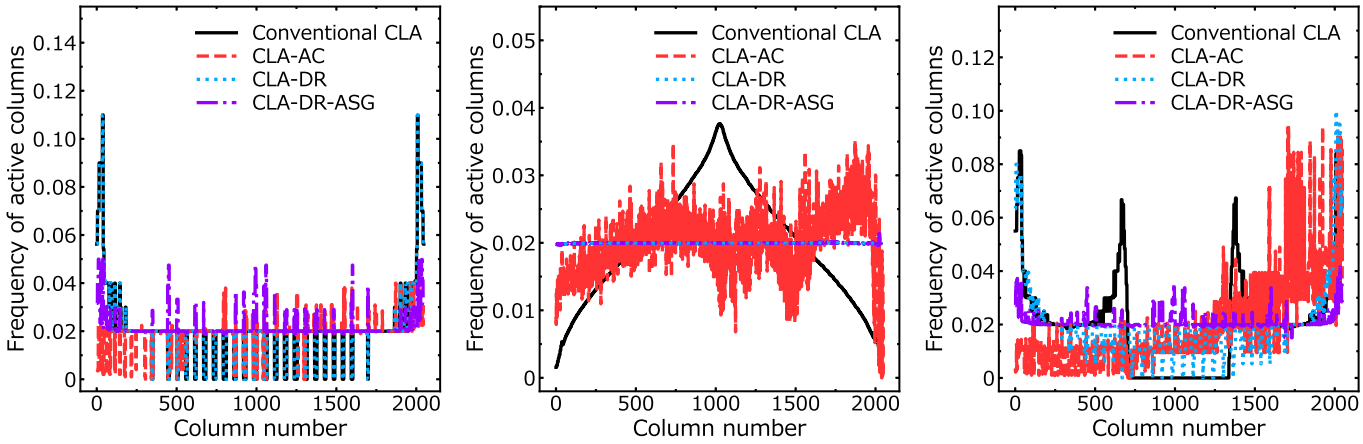
Next, Fig. 4(b) shows the MAE for the sine wave with a gradually shifting trend $X_{\text{trend}}(t)$. It can be observed that the prediction error of CLA-AC is larger than that of the conventional CLA. This is because CLA-AC does not adapt to shifts in the input range, leading to frequent and destabilizing changes in synapse placement. On the other hand, the prediction errors of CLA-DR and CLA-DR-ASG are comparable, and both achieve better accuracy than the conventional CLA after $t = 18,000$. As shown in Fig. 5(b), both CLA-DR and CLA-DR-ASG utilize all columns uniformly and effectively. This implies that the proposed method, which adds segments based on the input distribution, has limited effect due to the constantly changing input range. However, it also shows that the original segment groups remain unchanged, indicating that the proposed method does not adversely affect CLA-DR.

Finally, Fig. 4(c) shows the MAE for the sine wave with sudden range changes $X_{\text{change}}(t)$. This indicates that the prediction accuracy of the conventional CLA over the entire period and that of CLA-AC after the change are poor. This is attributed to their inability to adapt to shifts in the input range. In contrast, CLA-AC and CLA-DR-ASG achieve the best prediction accuracy before the change, while CLA-DR-



((a)) Sine wave $X_{sin}(t)$ ((b)) Sine wave with a trend $X_{trend}(t)$ ((c)) Sine wave with a change point $X_{change}(t)$

Fig. 4. MAE on synthetic time-series datasets



((a)) Sine wave $X_{sin}(t)$ ((b)) Sine wave with a trend $X_{trend}(t)$ ((c)) Sine wave with a change point $X_{change}(t)$

Fig. 5. The frequency of active columns on synthetic time-series datasets

ASG remains the most accurate after the change. As shown in Fig. 5(c), CLA-DR-ASG achieves the most uniform activation across all columns. This suggests that the balanced utilization of the available columns directly contributes to the improvement in prediction accuracy.

B. Results on Electricity Load Dataset

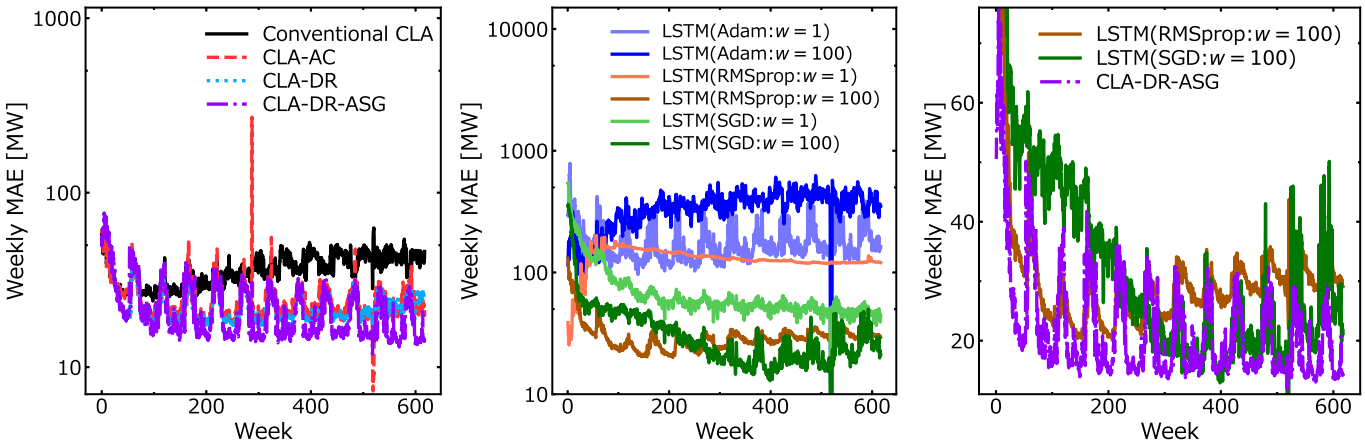
Fig. 6 shows the MAE for the electricity load dataset, and Fig. 7 shows the activation frequency of the columns for each CLA. To account for the periodicity of the data, the MAE is calculated as an average over $4 \times 24 \times 7 = 672$ time steps, corresponding to one week.

First, a comparison of the MAE across CLAs in Fig. 6(a) shows that the proposed CLA-DR-ASG achieves the lowest prediction error. Furthermore, Fig. 7 reveals that CLA-DR-ASG activates all columns most uniformly, indicating effective utilization of the predictor. Next, a comparison of the MAE across LSTM models in Fig. 6(b) shows that prediction accuracy varies significantly depending on the optimization algorithm and the window size w. Specifically, the LSTM using RMSprop with $w = 100$ converges the fastest, whereas the LSTM using SGD with $w = 100$ achieves the lowest prediction error after convergence. Finally, a closer comparison of the MAE for the best-performing CLA and LSTM models in Fig. 6(c) demonstrates that the proposed CLA-DR-ASG achieves the highest prediction accuracy at most time steps. It can also be observed that while the LSTM temporarily achieves the best accuracy around week 400, its performance subsequently deteriorates. This suggests that LSTM models struggle to retain diverse time-series patterns during online learning, whereas CLA is better suited for this task.

Based on the results above, the proposed method is shown to improve the prediction accuracy of CLA for time-series data with periodic structures and biased input distributions by ensuring effective column utilization.

VII. Conclusion

For CLA-DR, which eliminates the upper and lower bound constraints of the input, we propose a method to additionally arrange synapses based on the activation frequency of columns. This approach aims to achieve synapse placement that adapts to biases in the input range and to improve



((a)) Comparison in CLA ((b)) Comparison in LSTM ((c)) Excerpts from Selected Methods

Fig. 6. MAE on real-world electrivity load prediction

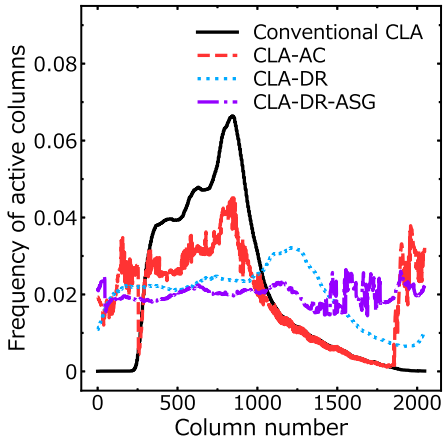


Fig. 7. The frequency of active columns on real-world electrivity load prediction

prediction accuracy through more effective utilization of the predictor.

Experimental results show that on synthetic time-series datasets, CLA-DR-ASG, which integrates the proposed method with CLA-DR, achieves superior prediction accuracy compared to the conventional CLA and existing methods, namely CLA-AC and CLA-DR, regardless of the presence and rate of input distribution shifts. Furthermore, on real-world electricity load data, CLA-DR-ASG demonstrates more stable and higher prediction accuracy than both the other CLA-based methods and the compared LSTM models. Analysis reveals that the proposed method utilizes the predictor more effectively, directly contributing to improved prediction accuracy.

As future work, we plan to explore methods for removing unnecessary synapses. This is because the proposed method continuously adds synapses in response to inputs with shifting ranges, potentially leading to columns possessing an excessive number of synapses. Additionally, we aim to improve practicality by eliminating the need for the parameter $range$, which is currently required in CLA-DR. Furthermore, we plan to further verify the effectiveness of the proposed method using a wider variety of datasets.

REFERENCES

[1] J. Hawkins and S. Blakeslee, *On Intelligence*. New York, NY, USA: Times Books, 2004.

[2] S. Ahmad and J. Hawkins, "Properties of sparse distributed representations and their application to Hierarchical Temporal Memory," Numenta, pp. 1–18, 2015.

[3] J. Hawkins, S. Ahmad, and D. Dubinsky, "Hierarchical temporal memory including HTM cortical learning algorithms," Numenta, Inc., Tech. Rep., 2010.

[4] J. Hawkins and S. Ahmad, "Why neurons have thousands of synapses, a theory of sequence memory in neocortex," *Front. Neural Circuits*, vol. 10, pp. 1–13, 2016.

[5] NuPIC. Accessed: Jan. 31, 2026. [Online]. Available: https://github.com/numenta/nupic

[6] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[7] Y. Cui, S. Ahmad, and J. Hawkins, "Continuous online sequence learning with an unsupervised neural network model," *Neural Comput.*, vol. 28, no. 11, pp. 2474–2504, 2016.

[8] T. Aoki, K. Takadama, and H. Sato, "Adaptive synapse arrangement in Cortical Learning Algorithm," *J. Adv. Comput. Intell. Intell. Inform.*, vol. 25, no. 4, pp. 450–466, 2021.

[9] T. Aoki, Q. Zhang, and T. Tatsukawa, "Overcoming upper and lower bound limitations of input in Cortical Learning Algorithm," *J. Jpn. Soc. Fuzzy Theory Intell. Inform.*, vol. 37, no. 4, pp. 743–750, 2025.

[10] T. Aoki, K. Takadama, and H. Sato, "Column-based decoder of internal prediction representation in Cortical Learning Algorithms," in *Proc. Joint 11th Int. Conf. Soft Comput. Intell. Syst. 21st Int. Symp. Adv. Intell. Syst. (SCIS-ISIS)*, 2020, pp. 1–7.

[11] T. Aoki, K. Takadama, and H. Sato, "Synergistic effect of adaptive synapse arrangement and column-based decoder in Cortical Learning Algorithm," in *Proc. Joint 12th Int. Conf. Soft Comput. Intell. Syst. 22nd Int. Symp. Adv. Intell. Syst. (SCIS-ISIS)*, 2022, pp. 1–6.

[12] New York Independent System Operator. Accessed: Jan. 31, 2026. [Online]. Available: http://mis.nyiso.com/public/

[13] A. Filion. "Data Analytics with MATLAB Webinar Files." MATLAB Central File Exchange. Accessed: Jan. 31, 2026. [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/49063-data-analytics-with-matlab-webinar-files

[14] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Represent. (ICLR)*, 2015, pp. 1–15.

[15] T. Tieleman and G. Hinton, "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude," *COURSERA: Neural Networks Mach. Learn.*, vol. 4, no. 2, pp. 26—31, 2012.