Chemical Reaction Network Implementation of Logic Gates and Neural Networks Using a Molecular Exchange Mechanism

Muhtasim Ishmum Khan [*1] **Moshiur Rahman** [*1] **Md. Shahriar Karim** [1]

Abstract

The emergence of synthetic biological circuits opens up a wide range of possibilities for maneuvering and controlling cellular behavior for many applications, including therapeutics in humans. These applications require remarkable reproducibility, and the logic circuit must demonstrate resilience against various perturbations. In this paper, we design the fundamental logic gates using a biomolecular perceptron based on the Molecular Exchange Mechanism (MEM) as its core module and extend the design to produce a winner-take-all (WTA) network implementation. The efficacy of the designs is evaluated using synthetic Boolean data and pattern classification tasks, such as the MNIST handwritten digits dataset and the CalTech-101 Silhouettes dataset. Interestingly, the designs demonstrate improved reproducibility in comparison to other design choices, suggesting the MEM-perceptron as a potential choice for biomolecular logic implementation and pattern recognition in a molecular environment.

1. Introduction

Living cells are computational devices that sense, process, and transmit information at different stages of tissue, organ, and species development (Bray, 1995; Green et al., 2017). For example, cells from the embryonic phase to the later stages follow a standard template for performing their stipulated tasks, such as growth, cell differentiation, and apoptosis. In many systems, cells decode the information received using a threshold-based mechanism to turn on/off downstream gene expression (Ashe & Briscoe, 2006). Synthetic logic circuits have been of immense interest for controlling cellular behavior in therapeutic applications, requiring precision and reliability in its performance. So, reprogramming cellular behavior through a synthetic circuits must demonstrate remarkable reproducibility and robustness against intrinsic and extrinsic fluctuations that a circuit experiences (Mukherji & Van Oudenaarden, 2009; Thattai & Van Oudenaarden, 2001).

Activation and inactivation in threshold-based cell functioning mimic the Boolean representation in computing devices. This suggests a potential role of biocomputers in computation and manipulation at the cellular level. The biocomputers consist of biomolecular logic gates that can link different regulatory factors and program cellular behaviors to harness therapeutic advantages (Ma et al., 2022). Regardless of whether the cell-free or within-cell computation is performed (Miyamoto et al., 2013), the synthesizing of biomolecular logic circuits adopts a nucleic acid-based or a protein-based computation following enzymatic interactions (Unger & Moult, 2006). In these designs, input concentrations to any logic gates, if perturbed, would result in an erroneous logic output. Although input and output in genetic logic circuits are primarily molecular concentrations, input fluctuation can switch threshold-based decoding between 1 and 0 and alter output, triggering an erroneous consequence in decision-making. Precisely, the inputs in these programmable gene circuits may vary because of perturbations in pH, extracellular factors, temperature, production rate of signaling molecules (Miyamoto et al., 2013), etc. These affect the reproducibility and reliability of the outputs, making the programmable circuits vulnerable.

Moreover, irreproducibility of discerning outputs due to their variability over multiple runs under identical training and resource specifications (D'Amour et al., 2022; Summers & Dinneen, 2021), has always been an issue in neural networks. Partial solutions (Anil et al., 2018; Shamir & Coviello, 2020) to reproducibility have been attempted, but the cost, maintenance, and complexity incurred make them untenable (Shamir et al., 2020). As studied, the role of the standard Rectified Linear Unit (ReLU) is instrumental in the success of the prevalence of neural networks, but also has the downside of contributing to the problem of

[*]Equal contribution [1]Department of Electrical and Computer Engineering, North South University, Dhaka, Bangladesh. Correspondence to: Md. Shahriar Karim <shahriar.karim@northsouth.edu>.

Proceedings of the Workshop on Generative AI for Biology at the 42nd International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025.

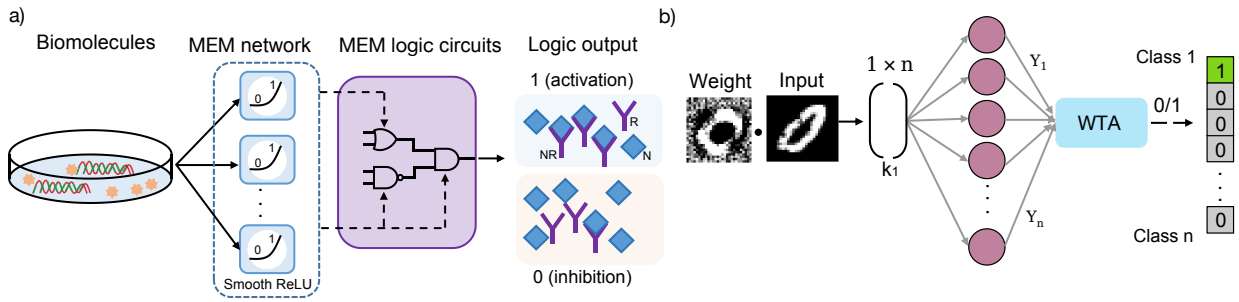

Figure 1. a) The MEM is used to design logic circuits that produce binary outputs based on the concentration of the decision-making species NR. By tuning the formation and decay rates of NR, these circuits can implement threshold-based decision rules, effectively encoding logic operations such as AND, OR, and XOR. When the concentration of NR exceeds a predefined threshold, the output is interpreted as "1"; otherwise, it is interpreted as "0". b) Image inputs and their corresponding weights are fed into the generalized Winner-Take-All (WTA) framework used to train the MEM-based Biomolecular Neural Network (BNN). The dot product of the input and weight vectors, which give the weighted sum of inputs, is used as the formation rate of the decision-making species, denoted as k_1. This rate, a parameter of MEM, governs the production of Y_i, which represents both the concentration of the decision-making species NR_i and the output of node i. The outputs from all nodes are then passed through a WTA mechanism, which selects the node with the highest output as the "winner." In this example, the digit "0" corresponds to the first class in the MNIST dataset, and a correct prediction results in activation of the first output node.

irreproducibility (Shamir & Coviello, 2020). A biomolecular neural network (BNN) with a smooth-ReLU activation function has been shown to perform better than the ReLU activation function (Anderson et al., 2021), making the smooth-ReLU activation function a choice that optimizes the reproducibility-accuracy trade-off. Considering issues that affect the reliability of these logic circuits, we use a biomolecular perceptron model, namely the MEM-perceptron (Rahman et al., 2024), designed from a molecular exchange mechanism capable of reducing noise (Karim et al., 2012; Singh, 2011) to implement all fundamental logic gates with a few sample logic circuits presented here. We also implement a winner-take-all (WTA) type network (Cherry & Qian, 2018) and the efficacy study demonstrates that the WTA is capable of recognizing image classes in different datasets. As has been studied by Rahman et al. (2024), the MEM-perceptron achieves smooth-ReLU behavior using a sequence of biomolecular interactions that allow the exchange of a siganlling particle through the formation of an intermediate complex. The designed biomolecular logic circuits and networks implement the smooth-ReLU activation function through the MEM-perceptron and demonstrate comparatively better reproducibility over other competing biomolecular perceptron models across several image classification datasets.

2. Methods and Results

2.1. Molecular Exchange Mechanism

The molecular exchange mechanism (MEM) is inspired by interactions that occur in various biophysical systems such as the synaptic cleft of the brain and early embryonic development (Papouin et al., 2012; Karim et al., 2012). There are three species that work in MEM: N (signaling agent, which could be a neurotransmitter), R (receptor) and E (exchanger molecule). The complex NR is the decision-making species. The complex NE, formed through the interaction of the signaling agent N and exchange molecule E, is used to impose negative feedback on the formation rate of NR. In the MEM dynamics, both NR and NE subsequently form the intermediate complex NRE, through which exchange of R and E occurs. Together, these interactions produce a smooth ReLU-like behavior of the species NR in steady state, similar to the activation function described in Anderson et al. (2021).

2.1.1. MEM CHEMICAL REACTION NETWORK

$$\text{r1: } N + R \underset{k_{\text{NR}}^{\text{off}}}{\overset{\hat{k}_{\text{NR}}^{\text{on}}}{\rightleftharpoons}} NR, \quad \text{r2: } N + E \underset{k_{\text{NE}}^{\text{off}}}{\overset{k_{\text{NE}}^{\text{on}}}{\rightleftharpoons}} NE$$

$$\text{r3: } NR + E \underset{k_{\text{NRE}}^{\text{off}}}{\overset{k_{\text{NRE}}^{\text{on}}}{\rightleftharpoons}} NRE, \quad \text{r4: } NE + R \underset{k_{\text{NER}}^{\text{off}}}{\overset{k_{\text{NER}}^{\text{on}}}{\rightleftharpoons}} NRE$$

$$\text{r5: } NR, NE, NRE \xrightarrow{\delta} \emptyset$$

$$[R_{\text{TOT}}] = [R] + [NR] + [NRE] \tag{1}$$

Upon formation of NE, the species negatively regulate the NR formation rate constant $k_{\text{NR}}^{\text{on}}$ following a Hill equation:

$$\hat{k}_{\text{NR}}^{\text{on}} = k_{\text{NR}}^{\text{on}} \left(\frac{K^{\text{n}}}{K^{\text{n}} + [NE]^{\text{n}}} \right) \tag{2}$$

where $k_{\text{NR}}^{\text{on}}$ is the basal NR production rate.

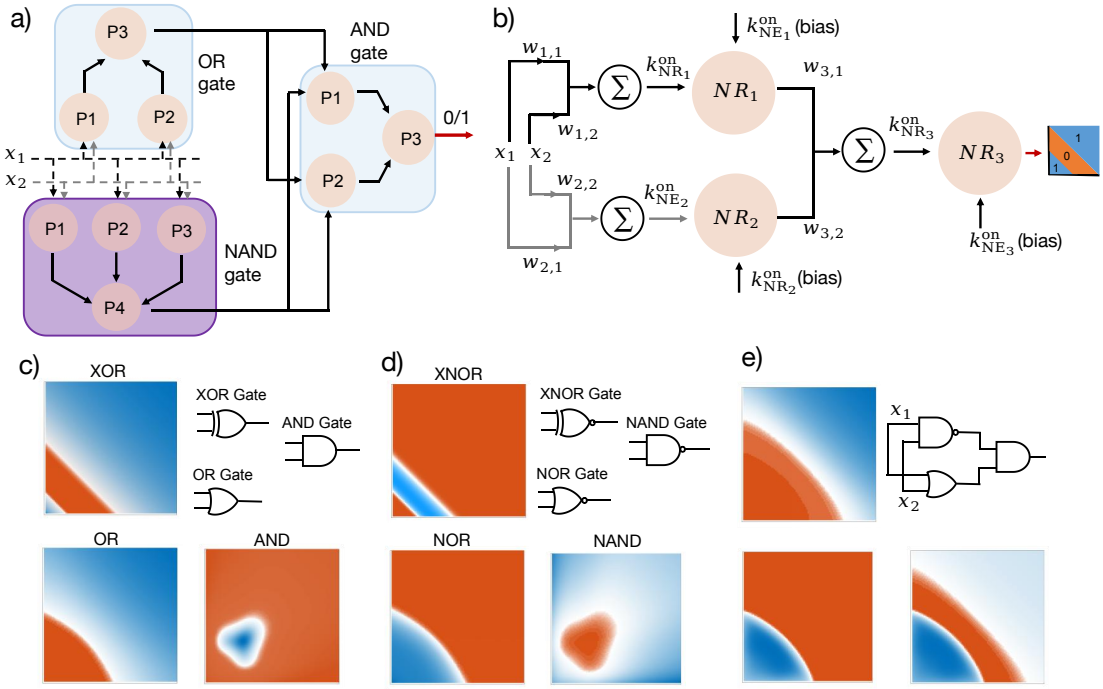

Figure 2. a) The schematic representation of the XOR logic circuit using the AND, NAND and OR logic circuits constructed using MEM peceptrons. b) The internal network of the MEM-XOR using MEM perceptrons. Node 1 (NR_1) is an activating perceptron while node 2 (NR_2) is an inhibiting perceptron, where the former node receives the weighted sum of inputs as k^{on}_{NR} and a constant bias while the latter node receives the weighted sum as k^{on}_{NE} and the activation rate takes the constant value of the bias. c,d) Different logic outputs obtained by changing the weights of the perceptrons in the respective logic gate circuits. e) The logic circuit in (a) being realized by using the gates in (d) and changing the AND gate from a 3-boundary to a 2-boundary AND gate to obtain a similar XOR output as (c).

2.2. Logic Gate Implementation

The MEM has demonstrated the ability to make both linear and non-linear classification through the implementation of different logic gate circuits. Initially, we design a 3-node network (2 hidden layer nodes and 1 output layer node) and a 4 node network (3 hidden layer nodes and 1 output layer node) to produce a logical OR and AND output respectively (see Appendix B for tuned weights). Following the validation of linear classification through these logic gates, we design a 3-node network (same design as OR gate) to demonstrate XOR logic output, realizing non-linear classification. In all cases, the system receives two inputs and the hidden layer produces different decision planes which are combined by tuning the weights of the output node to produce the desired logic circuit behavior. Furthermore, changing the design of the output layer node from an activating to an inhibiting one and subsequently tuning the weights allows each logic gate to behave as its negated versions, that is, NAND, NOR and XNOR.

To manifest the inverted logic gate implementation, the formation rate of the decision-making species NR is kept constant at the value of the bias to the respective node and the weighted sum is assigned to the species NE (Moorman et al., 2019), which is responsible for the negative feedback in the system (Eq. 4. In the MEM ODE, k^{on}_{NR}, which is the activation rate of NR, takes the weighted sum of the inputs and k^{on}_{NE}, which is the activation rate of NE, is the bias to the system for an activating node. Therefore, in order to implement the inhibiting logic and following this modification, the perceptron condition, as derived by Rahman et al. (2024), is also inverted (Eq. 3 and Eq. 4) (see Fig. 2b).

$$k^{on}_{NE} = W_1X_1 + W_2X_2 + ...W_nX_n \tag{3}$$

$$k^{on}_{NR} = W_0 \text{ (bias)}; \quad k^{on}_{NR} \geq \Omega k^{on}_{NE}; \quad k^{on}_{NR} < \Omega k^{on}_{NE} \tag{4}$$

Interestingly, the regions corresponding to the implemented logical operations exhibit soft boundaries between the 0 and 1 classes, unlike the sharp divisions expected from ideal binary logical outputs. We hypothesize that this behavior arises from the smooth nature of the ReLU activation function, which causes the network to produce gradual transitions rather than strict binary separations. While this characteristic introduces a degree of noise, it may also confer benefits (Bishop, 1995) and this remains part of our ongoing exploration.

2.3. Training Biomolecular Network

2.3.1. Network design and datasets

The activation function of the MEM-perceptron resembles a smooth ReLU similar to the work done by Anderson et al. (2021), which we hypothesize to be a contributing factor to its ability to absorb noise, providing more stability in decision-making. Therefore, for noisy inputs (such as gene expression data), the control over noise is likely to allow consistency while predicting molecular patterns. To verify this hypothesis, we design a single layer network with the MEM-perceptron as the building block. We train the single layer network following the winner-take-all (WTA) principle (Maass, 2000), where the output with the maximum value "wins", and remains active, while all other outputs "lose" and are turned off. We choose two datasets for our experiments: The MNIST dataset (LeCun et al., 1998), which consists of 10 classes of 28x28 grayscale images of handwritten digits (0-9). For benchmarking, we select molecular sequestration (Moorman et al., 2019) which has been shown to behave as a BNN, and repeat all experiments as done with MEM.

2.3.2. WTA implementation

The winner-take-all (WTA) circuit, as implemented in this study, is a single layer neural network with the number of nodes representing the number of classes in the dataset. In this WTA implementation, we demonstrate the detection of 10 number classes of the MNIST dataset with each pixel of the image sample being fed into the nodes as the weighted sum to each node. The learning principle allows only one output node to remain active (high: 1) and all other nodes remain inactive (low: 0) in response to an input. The active node is the denoted as the "winner" and indicates the predicted class of the provided input. Previous work demonstrates that training a chemical ODE-based WTA network by comparing the weights with the inputs instead of the predicted outputs eliminates the variability-dependence of dynamic range of output concentration common in such networks (Arcadia et al., 2021). Interestingly, accommodating this concentration independence also reshapes the weights towards resembling the input following the sequential updating after each iteration. To harness these properties, we develop and train a WTA network using MEM perceptrons, which maintain noise under tight control in a context-specific manner (Rahman et al., 2024; Khan et al., 2024). In the proposed WTA, the weighted sum of inputs produces the vector containing the formation rate ($k^{\text{on}}_{\text{NR}}$) of the decision-making species NR in a biomolecular interaction ($N + R \rightleftharpoons NR$) of each MEM-perceptron node. We optimize the weights using stochastic gradient descent (Ruder, 2016) to minimize the error between this NR formation rate vector (which is the input to the MEM system) and the ground truth, given by the mean squared error(MSE) function in Eq. 6. The formation rate of the species NE, $k^{\text{on}}_{\text{NE}}$ is the MEM perceptron parameter that corresponds to the bias (W_0, Fig. 2b) in the perceptron. We set this bias to a constant 0.5 because the negative feedback on NR is dependent on the concentration of NE and varying the bias does not adapt the decision plane as per the perceptron condition (Khan et al., 2024). In order to account for the constant bias, we normalize the weighted sum for each iteration. This training strategy pushes the formation rate of the node corresponding to the correct class toward 1 and the formation rates of all other nodes toward 0, so that the output of the node corresponding to the correct class becomes maximum and it becomes the "winner".

To train the MEM-based WTA network, first, we compute the dot product of the weights and the inputs, which we then provide as the formation rate of the decision-making species

Algorithm 1 WTA Network Training Algorithm

Input: Training dataset $\mathcal{D} = \{(X_i, Y_i)\}_{i=1}^{N}$
Initialize:
- CRN parameters (reaction rates, constants)
- NN hyperparameters: learning rate η, batch size B, number of epochs E
- Number of input/output nodes
- Weights $\mathbf{W} \sim \mathcal{U}(0, 1)$, Bias $B = 0.5$

for epoch $= 1$ **to** E **do**
  Shuffle $\mathcal{D}$
  Split $\mathcal{D}$ into mini-batches of size B
  for each mini-batch $(X^{(b)}, Y^{(b)})$ **do**
    for each input $x^{(i)}$, label $y^{(i)}$ in mini-batch **do**
      for each output node j **do**
        $k^{\text{on}}_{\text{NR}} \leftarrow W_j \cdot x^{(i)}$
        Normalize $k^{\text{on}}_{\text{NR}}$
        $k^{\text{on}}_{\text{NE}} \leftarrow B$
        if $k^{\text{on}}_{\text{NR}} < 0$ **then**
          $k^{\text{on}}_{\text{NR}} \leftarrow 10^{-5}$
        end if
        Solve the system of ODEs
        $o_j \leftarrow$ Final concentration of NR
      end for
      $\hat{y}^{(i)} \leftarrow \arg\max_j(o_j)$ ▷ WTA decision
    end for
    Compute loss L ▷ Eq. 6
    Compute gradient $\frac{\partial L}{\partial \mathbf{W}}$ ▷ Eq. 7
    Update weights $\mathbf{W}$ ▷ Eq. 8
  end for
end for
Output: Trained weight matrix $\mathbf{W}_{\text{final}}$

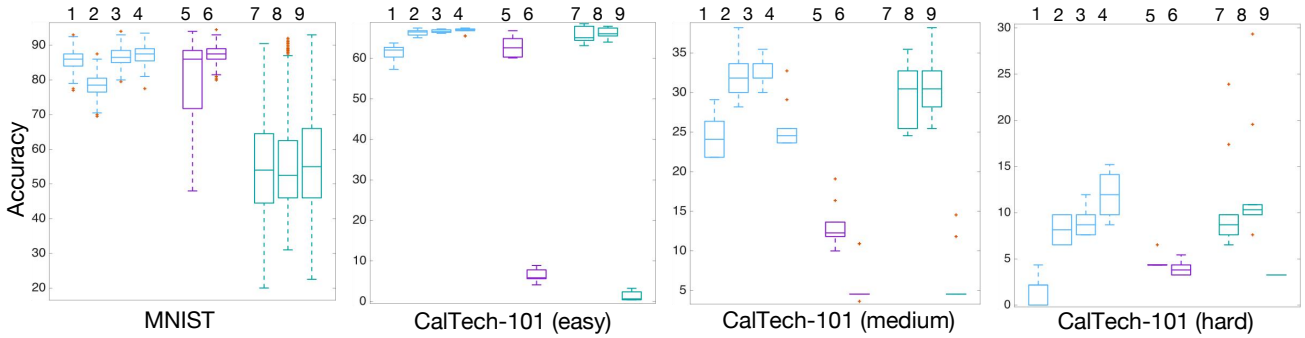

Figure 3. The box plots show the deviation in test accuracies of the MEM-WTA network, the molecular sequestration-based WTA network, and the WTA network based on the CRN in Anderson et al. (2021). In order to demonstrate the degree of variation of the accuracies of each model, we used the MNIST and CalTech101 dataset, with the latter dataset being split into three subclasses (easy, medium, hard) based on the classification results of a generic multi-layer perceptron (details given in appendix C). Each model is run 10 times with different learning rates (detailed values given in Table 1). The box plots labeled from 1 to 4 represent the MEM-WTA, 5 and 6 correspond to the molecular sequestration-based WTA network and 7 to 9 show the WTA network of the CRN in Anderson et al. (2021).

NR of MEM (Eq. 5).

$$k_{\text{NR}}^{\text{on}} = \mathbf{W}^T\mathbf{X} = W_1X_1 + \ldots + W_nX_n \tag{5}$$

Then, we calculate the error between the formation rate and the ground truth (Eq. 6).

$$L = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{2}||k_{\text{NR}_i}^{\text{on}} - \hat{y}_i||^2 \tag{6}$$

After that, we calculate the gradient of the loss with respect to the weights (Eq. 7).

$$\frac{\partial L}{\partial \mathbf{W}} = \frac{\partial L}{\partial k_{\text{NR}}^{\text{on}}}\frac{\partial k_{\text{NR}}^{\text{on}}}{\partial \mathbf{W}} = (k_{\text{NR}}^{\text{on}} - \hat{y})X \tag{7}$$

We update the weights following Eq. 8.

$$\mathbf{W} = \mathbf{W} - \eta\frac{\partial L}{\partial \mathbf{W}} \tag{8}$$

We perform these steps in batches, as we employ stochastic gradient descent to train our WTA network. By following this training strategy, we try to maximize the formation rate of the decision-making species of the node corresponding to the correct class, and make it the "winner". In the above equations, $k_{\text{NR}}^{\text{on}}$ is the formation rate of the decision-making species NR, $\mathbf{W}$ is the weight matrix, $\mathbf{X}$ is the input data vector, $\hat{y}$ is the ground truth, and η is the learning rate.

In the WTA training setup, we initialize the weights randomly following a normal distribution. The input data values are non-negative as they represent image pixel intensity. We normalize the inputs to stay in the range [0 1]. As training progresses, the distribution of the weights changes to fit the data, and some of the weight values become negative. This might seem contradictory as the system we are using to implement the WTA network is a model of a chemical system, and in a chemical system, there can be no negative values (Samaniego et al., 2021). Previous works have taken measures to keep weight values to a fixed range for in vitro implementation of a chemical neural network, such as clipping negative values to 0 and normalizing values so that weights remain in the range [0 1] (Arcadia et al., 2021). In the MEM-based WTA network, the weights and the input data do not directly map to any component of the chemical system. However, the dot product of the weight matrix and input data vector maps to the formation rates of the decision-making species NR, and we ensure that all values of the formation rate vector are always non-negative in our training algorithm. In fact, negative weights have been shown to be useful for learning more generalizations (Wang et al., 2023). Whether negative weights have any useful contribution in the learning process of the WTA network constructed using MEM is subject to further investigation.

2.3.3. Reproducibility of Performance

Reproducibility is a well-known challenge in training neural networks, largely due to the inherent randomness in processes like weight initialization and mini-batch selection (D'Amour et al., 2022; Summers & Dinneen, 2021). The MEM demonstrates robustness against noise and is thus hypothesized to produce more reproducible results compared to similar biophysically inspired models. To explore this, we build a Winner-Take-All (WTA) network using MEM-based perceptrons and train it on both the MNIST dataset and three subsets (easy, medium, and hard) of the CalTech-101 Silhouettes dataset (Marlin et al., 2010). The categorization of these subsets is explained in Appendix C, Fig. 5. We run

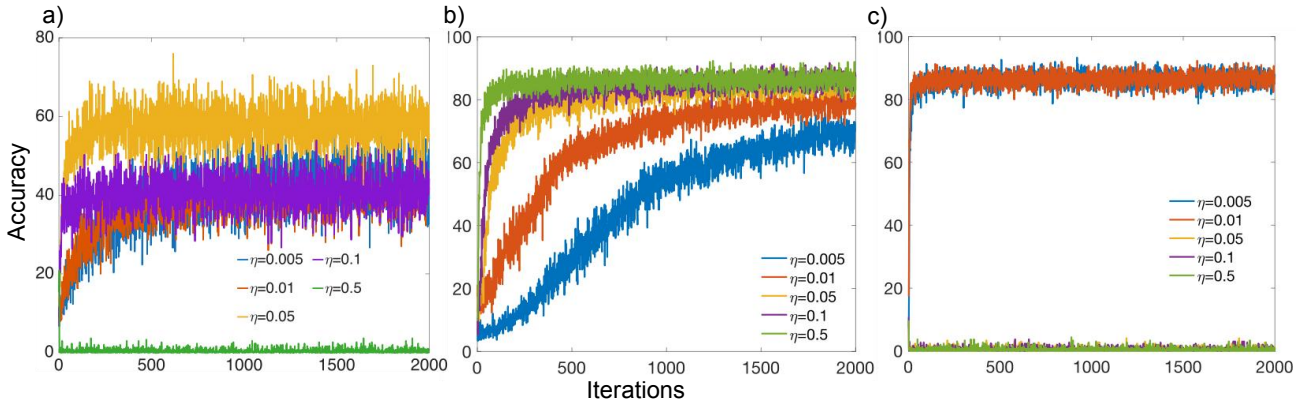

Figure 4. Training accuracy comparison between WTA networks constructed using a) CRN model as shown in Anderson et al. (2021), b) MEM, c) Molecular Sequestration (Moorman et al., 2019). All models are trained over 2000 iterations using 5 different learning rate values. MEM demonstrates learning for all values of learning rate that was tried. However, the other two models do not learn for higher values of learning rate. This shows that MEM is more robust to learning rate perturbations, and is more consistent compared to the other models.

Model	Learning rate	Mean test accuracy			
		MNIST	Easy	Medium	Hard
MEM	0.01	0.784 ± 0.030	0.616 ± 0.020	0.246 ± 0.029	0.019 ± 0.016
	0.05	0.857 ± 0.026	0.664 ± 0.008	0.323 ± 0.032	0.082 ± 0.014
	0.1	0.866 ± 0.025	0.667 ± 0.004	0.324 ± 0.016	0.090 ± 0.014
	0.5	0.873 ± 0.024	0.668 ± 0.007	0.256 ± 0.030	0.119 ± 0.023
(Moorman et al., 2019)	0.005	0.875 ± 0.024	0.628 ± 0.027	0.132 ± 0.027	0.046 ± 0.007
	0.01	0.807 ± 0.111	0.064 ± 0.015	0.057 ± 0.028	0.039 ± 0.008
(Anderson et al., 2021)	0.005	-	0.656 ± 0.020	0.298 ± 0.040	0.109 ± 0.055
	0.01	0.549 ± 0.157	0.663 ± 0.012	0.307 ± 0.038	0.128 ± 0.066
	0.05	0.556 ± 0.139	0.014 ± 0.011	0.063 ± 0.037	0.033 ± 0.000
	0.1	0.562 ± 0.160	-	-	-

Table 1. Performance and reproducibility comparison of WTA networks constructed using MEM, Molecular Sequestration, and the CRN as shown in Anderson et al. (2021). MEM exhibits learning in both low and high values of the learning rate. The other two models do not learn for high values of learning rate, e.g., 0.5, and training accuracy stays around 0-1% for the MNIST dataset. Upon using the three subclasses of the CalTech-101 Silhouettes dataset, both the molecular sequestration-based WTA and the CRN-based WTA of Anderson et al. (2021), demonstrate learning for lower learning rates, while falling to significantly low accuracy levels when the learning rate was increased beyond 0.01.

each experiment 10 times using a range of learning rates (0.01, 0.05, 0.1, 0.5) and track the variation of the test accuracy across runs. For comparison, we repeat the same setup with two alternative WTA networks: one using a molecular sequestration-based perceptron with a ReLU activation function (Moorman et al., 2019), and another using a chemical reaction network (CRN) model that implements a smooth-ReLU function (Anderson et al., 2021). Our findings show that the MEM-based network trains reliably across all tested learning rates. In contrast, the other two models struggle to converge consistently under the same conditions. Moreover, we analyze the deviation in test accuracy of the three WTA networks trained with different learning rates, and we find that the MEM-based network performs more consistently with having similar accuracy results as that of the generic multilayer perceptron used to create the subclass of the CalTech-101 Silhouettes dataset.

3. Discussion

The envisaged applications of synthetic biological circuits in regulating and controlling cellular behavior leave no room for faltering under perturbations. So, it is imperative to design a synthetic circuit that harnesses and mitigates biological noise to maintain the decision concentration within a certain tolerance. In designing biological circuits, we

apply molecular exchange mechanisms and negative feedback, exhibiting kinetic-dependent noise mitigation abilities. We devised the underlying perceptron circuits and the relevant weight parameters for most logic gates. We extended further to realize synthetic logic circuits using molecular exchange-based perceptron that resonates to a smoothed ReLU as the fundamental building block. An artificially smoothed ReLU exploits the reproducibility-accuracy tradeoff and ameliorates the irreproducibility problem of the accuracy of neural network models. This work also provides a WTA-type network consisting of a MEM-perceptron. We applied the WTA for pattern recognition of image classes in three image datasets, and the WTA of the MEM-perceptron demonstrated better reproducibility. The proposed work provides an avenue for exploration that ensures robust and reproducible biomolecular logic and circuit design requirements. Though all the implementations here rely on weight search, weight tuning, and updates through image training, but learning through backpropagation remains unexplored, training a full-fledged multi-layer feedforward network capable of learning through backpropagation is a part of our ongoing work.

References

Anderson, D. F., Joshi, B., and Deshpande, A. On reaction network implementations of neural networks. *Journal of the Royal Society Interface*, 18(177):20210031, 2021.

Anil, R., Pereyra, G., Passos, A., Ormandi, R., Dahl, G. E., and Hinton, G. E. Large scale distributed neural network training through online distillation. *arXiv preprint arXiv:1804.03235*, 2018.

Arcadia, C. E., Dombroski, A., Oakley, K., Chen, S. L., Tann, H., Rose, C., Kim, E., Reda, S., Rubenstein, B. M., and Rosenstein, J. K. Leveraging autocatalytic reactions for chemical domain image classification. *Chemical Science*, 12(15):5464–5472, 2021.

Ashe, H. L. and Briscoe, J. The interpretation of morphogen gradients. 2006.

Bishop, C. M. Training with noise is equivalent to tikhonov regularization. *Neural computation*, 7(1):108–116, 1995.

Bray, D. Protein molecules as computational elements in living cells. *Nature*, 376(6538):307–312, 1995.

Cherry, K. M. and Qian, L. Scaling up molecular pattern recognition with dna-based winner-take-all neural networks. *Nature*, 559(7714):370–376, 2018.

D'Amour, A., Heller, K., Moldovan, D., Adlam, B., Alipanahi, B., Beutel, A., Chen, C., Deaton, J., Eisenstein, J., Hoffman, M. D., et al. Underspecification presents challenges for credibility in modern machine learning. *Journal of Machine Learning Research*, 23(226):1–61, 2022.

Green, A. A., Kim, J., Ma, D., Silver, P. A., Collins, J. J., and Yin, P. Complex cellular logic computation using ribocomputing devices. *Nature*, 548(7665):117–121, 2017.

Karim, M. S., Buzzard, G. T., and Umulis, D. M. Secreted, receptor-associated bone morphogenetic protein regulators reduce stochastic noise intrinsic to many extracellular morphogen distributions. *Journal of The Royal Society Interface*, 9(70):1073–1083, 2012.

Khan, M. I., Rahman, M., and Karim, M. S. Development of a molecular exchange mechanism-based biomolecular neural network. In *ICLR 2024 Workshop on Generative and Experimental Perspectives for Biomolecular Design*, 2024. URL https://openreview.net/forum?id=dm5E5eKCDk.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Ma, Q., Zhang, M., Zhang, C., Teng, X., Yang, L., Tian, Y., Wang, J., Han, D., and Tan, W. An automated dna computing platform for rapid etiological diagnostics. *Science Advances*, 8(47):eade0453, 2022.

Maass, W. On the computational power of winner-take-all. *Neural computation*, 12(11):2519–2535, 2000.

Marlin, B., Swersky, K., Chen, B., and Freitas, N. Inductive principles for restricted boltzmann machine learning. In Teh, Y. W. and Titterington, M. (eds.), *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pp. 509–516, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR. URL https://proceedings.mlr.press/v9/marlin10a.html.

Miyamoto, T., Razavi, S., DeRose, R., and Inoue, T. Synthesizing biomolecule-based boolean logic gates. *ACS synthetic biology*, 2(2):72–82, 2013.

Moorman, A., Samaniego, C. C., Maley, C., and Weiss, R. A dynamical biomolecular neural network. In *2019 IEEE 58th conference on decision and control (CDC)*, pp. 1797–1802. IEEE, 2019.

Mukherji, S. and Van Oudenaarden, A. Synthetic biology: understanding biological design from synthetic circuits. *Nature Reviews Genetics*, 10(12):859–871, 2009.

Papouin, T., Ladépêche, L., Ruel, J., Sacchi, S., Labasque, M., Hanini, M., Groc, L., Pollegioni, L., Mothet, J.-P., and Oliet, S. H. Synaptic and extrasynaptic nmda receptors

are gated by different endogenous coagonists. *Cell*, 150 (3):633–646, 2012.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Rahman, M., Khan, M. I., and Karim, M. S. Design of a molecular exchange-based robust perceptron for biomolecular neural network. In *The Second Tiny Papers Track at ICLR 2024*, 2024.

Ruder, S. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

Samaniego, C. C., Moorman, A., Giordano, G., and Franco, E. Signaling-based neural networks for cellular computation. In *2021 American Control Conference (ACC)*, pp. 1883–1890. IEEE, 2021.

Shamir, G. I. and Coviello, L. Anti-distillation: Improving reproducibility of deep networks. *arXiv preprint arXiv:2010.09923*, 2020.

Shamir, G. I., Lin, D., and Coviello, L. Smooth activations and reproducibility in deep networks. *arXiv preprint arXiv:2010.09931*, 2020.

Singh, A. Negative feedback through mrna provides the best control of gene-expression noise. *IEEE transactions on nanobioscience*, 10(3):194–200, 2011.

Summers, C. and Dinneen, M. J. Nondeterminism and instability in neural network optimization. In *International Conference on Machine Learning*, pp. 9913–9922. PMLR, 2021.

Thattai, M. and Van Oudenaarden, A. Intrinsic noise in gene regulatory networks. *Proceedings of the National Academy of Sciences*, 98(15):8614–8619, 2001.

Unger, R. and Moult, J. Towards computing with proteins. *Proteins: Structure, Function, and Bioinformatics*, 63(1): 53–64, 2006. doi: 10.1002/prot.20725.

Wang, Q., Powell, M. A., Geisa, A., Bridgeford, E., Priebe, C. E., and Vogelstein, J. T. Why do networks have inhibitory/negative connections? In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 22551–22559, 2023.

A. ODE model for the MEM logic gates

The logic circuits are constructed using the following set of ODEs which are interconnected to develop the circuit formation.

$$\begin{aligned}
\dot{[NR]}_i &= \hat{k}^{\text{on}}_{\text{NR}_i}[N][R]_i + k^{\text{off}}_{\text{NRE}_i}[NRE]_i - k^{\text{off}}_{\text{NR}_i}[NR]_i - k^{\text{on}}_{\text{NRE}_i}[NR]_i[E]_i - \delta[NR]_i \\
\dot{[NE]}_i &= k^{\text{on}}_{\text{NE}_i}[N][E]_i + k^{\text{off}}_{\text{NER}_i}[NRE]_i - k^{\text{off}}_{\text{NE}_i}[NE]_i - k^{\text{on}}_{\text{NER}_i}[NE]_i[R]_i - \delta[NE]_i \\
\dot{[NRE]}_i &= k^{\text{on}}_{\text{NRE}_i}[NR]_i[E]_i + k^{\text{on}}_{\text{NER}_i}[NE]_i[R]_i - (k^{\text{off}}_{\text{NRE}_i} + k^{\text{off}}_{\text{NER}_i} + \delta)[NRE]_i
\end{aligned} \tag{9}$$

where i denotes the node number and the total quantity of the nodes to implement a specific logic circuit is dictated by the number of decision boundaries that are required to show the classification output of the logic gate.

A.1. ODEs of the MEM-XOR

Considering the 2-input, 3-node XOR logic implementation, nodes 1 and 2 receive the system input and generate two decision boundaries separately. The concentration output from each of these nodes is then used to calculate the weighted sum that is given as input to the node 3 in the network as the formation rate of NR. Finally, node 3 produces the final output that corresponds to the XOR logic. Therefore, the connectivity between nodes within any logic circuit is established through the formation rate of the decision-making species (NR). The ODEs for each node and the generalized basal formation rate equation is given as

Node 1

$$\begin{aligned}
\dot{[NR]}_1 &= \hat{k}^{\text{on}}_{\text{NR}_1}[N][R]_1 + k^{\text{off}}_{\text{NRE}_1}[NRE]_1 - k^{\text{off}}_{\text{NR}_1}[NR]_1 - k^{\text{on}}_{\text{NRE}_1}[NR]_1[E]_1 - \delta[NR]_1 \\
\dot{[NE]}_1 &= k^{\text{on}}_{\text{NE}_1}[N][E]_1 + k^{\text{off}}_{\text{NER}_1}[NRE]_1 - k^{\text{off}}_{\text{NE}_1}[NE]_1 - k^{\text{on}}_{\text{NER}_1}[NE]_1[R]_1 - \delta[NE]_1 \\
\dot{[NRE]}_1 &= k^{\text{on}}_{\text{NRE}_1}[NR]_1[E]_1 + k^{\text{on}}_{\text{NER}_1}[NE]_1[R]_1 - (k^{\text{off}}_{\text{NRE}_1} + k^{\text{off}}_{\text{NER}_1} + \delta)[NRE]_1
\end{aligned} \tag{10}$$

Node 2

$$\begin{aligned}
\dot{[NR]}_2 &= \hat{k}^{\text{on}}_{\text{NR}_2}[N][R]_2 + k^{\text{off}}_{\text{NRE}_2}[NRE]_2 - k^{\text{off}}_{\text{NR}_2}[NR]_2 - k^{\text{on}}_{\text{NRE}_2}[NR]_2[E]_2 - \delta[NR]_2 \\
\dot{[NE]}_2 &= k^{\text{on}}_{\text{NE}_2}[N][E]_2 + k^{\text{off}}_{\text{NER}_2}[NRE]_2 - k^{\text{off}}_{\text{NE}_2}[NE]_2 - k^{\text{on}}_{\text{NER}_2}[NE]_2[R]_2 - \delta[NE]_2 \\
\dot{[NRE]}_2 &= k^{\text{on}}_{\text{NRE}_2}[NR]_2[E]_2 + k^{\text{on}}_{\text{NER}_2}[NE]_2[R]_2 - (k^{\text{off}}_{\text{NRE}_2} + k^{\text{off}}_{\text{NER}_2} + \delta)[NRE]_2
\end{aligned} \tag{11}$$

Node 3

$$\begin{aligned}
\dot{[NR]}_3 &= \hat{k}^{\text{on}}_{\text{NR}_3}[N][R]_3 + k^{\text{off}}_{\text{NRE}_3}[NRE]_3 - k^{\text{off}}_{\text{NR}_3}[NR]_3 - k^{\text{on}}_{\text{NRE}_3}[NR]_3[E]_3 - \delta[NR]_3 \\
\dot{[NE]}_3 &= k^{\text{on}}_{\text{NE}_3}[N][E]_3 + k^{\text{off}}_{\text{NER}_3}[NRE]_3 - k^{\text{off}}_{\text{NE}_3}[NE]_3 - k^{\text{on}}_{\text{NER}_3}[NE]_3[R]_3 - \delta[NE]_3 \\
\dot{[NRE]}_3 &= k^{\text{on}}_{\text{NRE}_3}[NR]_3[E]_3 + k^{\text{on}}_{\text{NER}_3}[NE]_3[R]_3 - (k^{\text{off}}_{\text{NRE}_3} + k^{\text{off}}_{\text{NER}_3} + \delta)[NRE]_3
\end{aligned} \tag{12}$$

$$k^{\text{on}}_{\text{NR}_i} = W_1X_1 + W_2X_2 + ...W_nX_n \tag{13}$$

where, n represents the number of inputs to the particular node being considered. With respect to the XOR logic implementation, nodes 1 and 2 receive two inputs X_1 and X_2 and the output node, receives $X_1 = NR_1$ and $X_2 = NR_2$.

B. Weights for the MEM logic gates

Gate	Node 1	Node 2	Node 3	Node 4
NAND	0.4, 1, 1	0.05, 0.08, 0.2	0.05, 2, 0.05	0.75, 10, 3, 3
AND	0.4, 1, 1	0.05, 0.08, 0.2	0.05, 2, 0.05	0.9, 10, 3, 3
OR	0.5, 0.5, 1	0.5, 1, 0.5	0.05, 10, 10	0,0,0,0
NOR	0.5, 0.5, 1	0.5, 1, 0.5	0.05, 10, 10	0,0,0,0
XOR	0.5, 1, 1	0.2, 1, 1	0.02, 10, 1	0,0,0,0
XNOR	0.5, 1, 1	0.2, 1, 1	0.05, 10, 1	0,0,0,0

Table 2. Tuned weights for biomolecular logic gate implementation using MEM.

C. CalTech-101 Dataset Categorization

Easy Classes		**Medium Classes**		**Hard Classes**	
ID	**Name**	**ID**	**Name**	**ID**	**Name**
20	ceiling fan	97	wild cat	90	sunflower
16	butterfly	18	cannon	68	okapi
53	joshua tree	88	stop sign	60	lotus
66	nautilus	7	ant	91	tick
5	accordion	65	minaret	11	binocular
22	chair	100	yin yang	25	cougar face
33	dolphin	64	metronome	19	car side
30	cup	80	scissors	63	menorah
62	mayfly	32	dollar bill	29	crocodile head
49	hedgehog	83	snoopy	8	barrel

Table 3. Caltech-101 Silhouettes Dataset: Easy, Medium, and Hard Classes.

We evaluate MEM and other BNN models using the MNIST handwritten digits dataset and three customized subsets of the Caltech-101 Silhouettes dataset. To construct these subsets, we first train a multilayer perceptron (MLP) implemented via the scikit-learn library (Pedregosa et al., 2011) on the entire Caltech-101 Silhouettes dataset. Using the trained MLP, we compute the recall for each class and rank the classes accordingly. The 10 classes with the highest recall values are designated as the ”easy” subset, while the 10 classes with the lowest recall values form the ”hard” subset. The ”medium” subset consists of 10 classes selected from the middle of the ranked list (positions 46 to 55). This categorization is illustrated in Fig. 5, with the specific classes listed in Table 3.

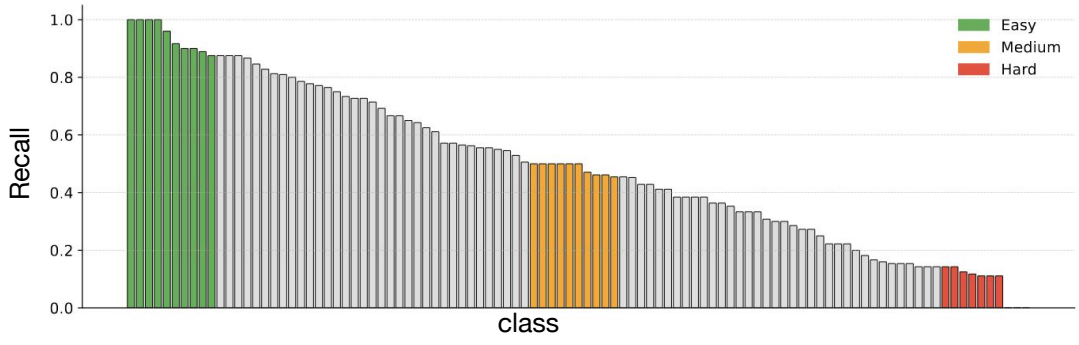

Figure 5. Categorization of the CalTech-101 Silhouettes dataset into easy, medium, and hard subsets based on the per-class recall value achieved by training an MLP on the dataset.