# An On-chip learning Neuromorphic Accelerator for Wireless Edge AI application


## Muhammad Farhan Azmine


- <u>Committee Members</u>
- Dr. Yang (Cindy) Yi
- Dr. Dong Sam Ha
- Dr. Creed F. Jones
- Dr. Xiaoting Jia
- Dr. Jeffrey Walling

# Outline

- Introduction
- Challenges
- State of the Art
- Research improvement (Concept)
- Detailed Hardware Implementation (Novelty)
- Performance Analysis
- Summary
- Future Work
- Acknowledgement
- Reference

# About me

- **Educational Background**
  - **BSc in Electrical and Electronic Engineering in Bangladesh University of Engineering & Technology (BUET)**
  - **Direct PhD (Fourth Semester) in Virginia Tech CPE**
- **Research Interest**
  - **Hardware Accelerator for AI**
  - **Spiking Neural Network Accelerator on FPGA**
- **Publications**
  - Lin, Chunxiao, Muhammad Farhan Azmine, and Yang Yi. "Accelerating Next-G Wireless Communications with FPGA-Based AI Accelerators." *2023 IEEE/ACM International Conference on Computer Aided Design (ICCAD)*. IEEE, 2023.
  - Lin, Chunxiao, Muhammad Farhan Azmine, Yibin Liang, and Yang Yi. "Leveraging neuro-inspired AI accelerator for high-speed computing in 6G networks." *Frontiers in Computational Neuroscience* 18 (2024): 1345644.

# Introduction

## Importance of Edge Computing based AI hardware

- **Exponential growth** of IoT (Internet of Things) applications
- Extensive **workload on Data centers**
- Incurs **high latency**
- **High network bandwidth** usage
- Industry and research effort to push AI computing to network edge

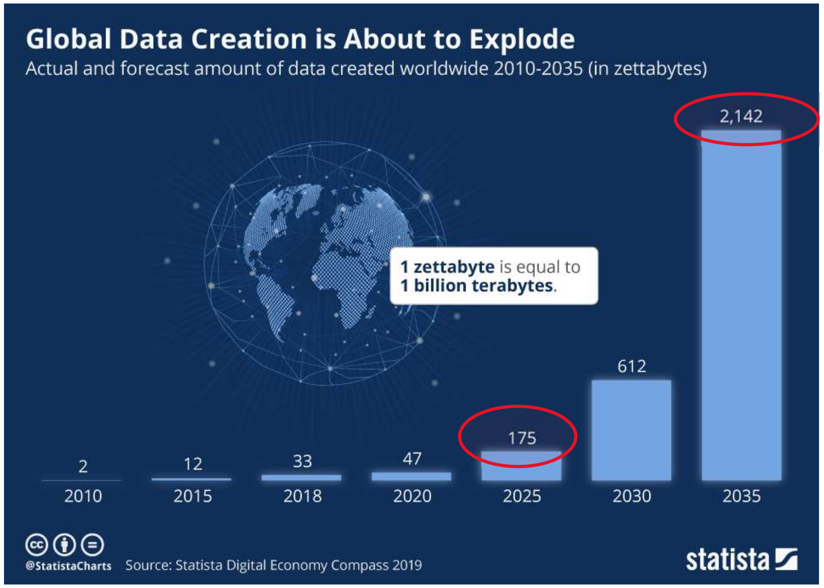



Figure: Global estimation data generation [1]

https://www.statista.com/chart/17727/global-data-creation-forecasts/

# What is Edge AI and Why ?

- Combination of **computation on end edge** of network and **AI applications**
- **Traditional AI applications** are cloud driven
- Worsens **latency** in network connection
- Increased **communication** cost
- **Privacy** concerns

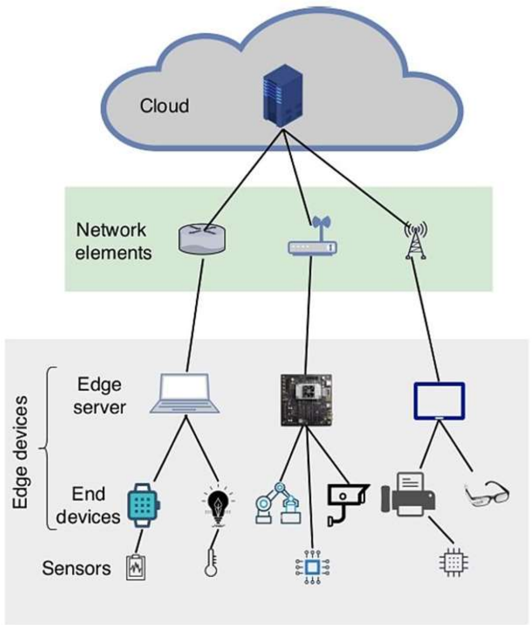



Figure: Network hierarchy for edge AI applications [2]

https://viso.ai/edge-ai/edge-ai-applications-and-trends/

# Edge cloud and Edge AI [2]

- Computations being performed as close to data sources as possible
- Edge computing can **decentralize** the cloud to edge nodes
- Data can be sent to edge nodes to be processed for machine learning
- Creating **edge cloud**

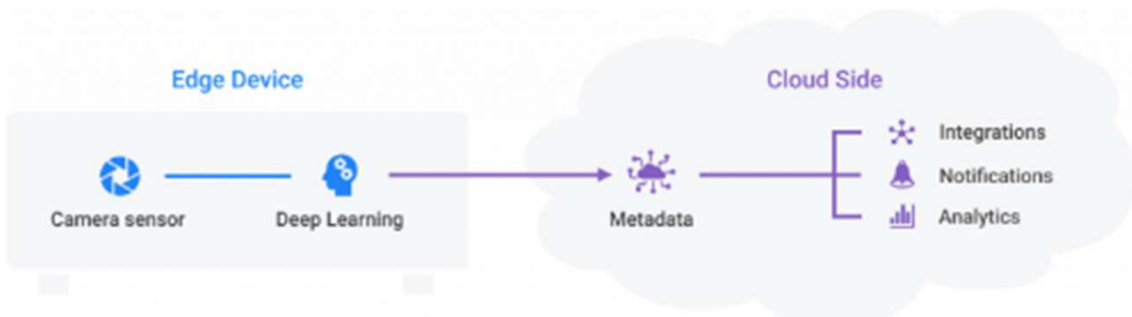


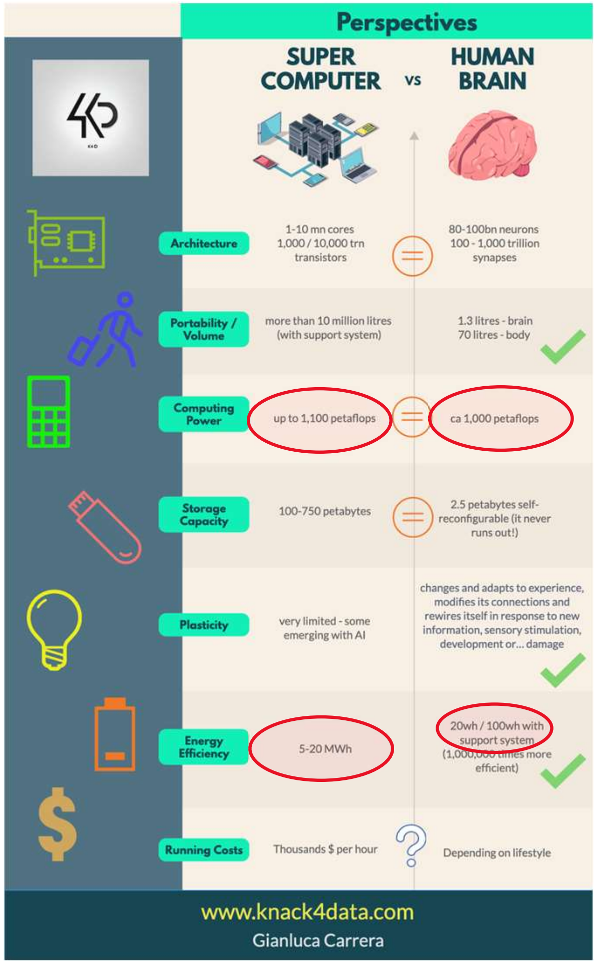

Figure: Data is analyzed on-device, and the processed insights of multiple edge devices are gathered in the cloud [2]

https://viso.ai/edge-ai/edge-ai-applications-and-trends/ [2]

# Introduction

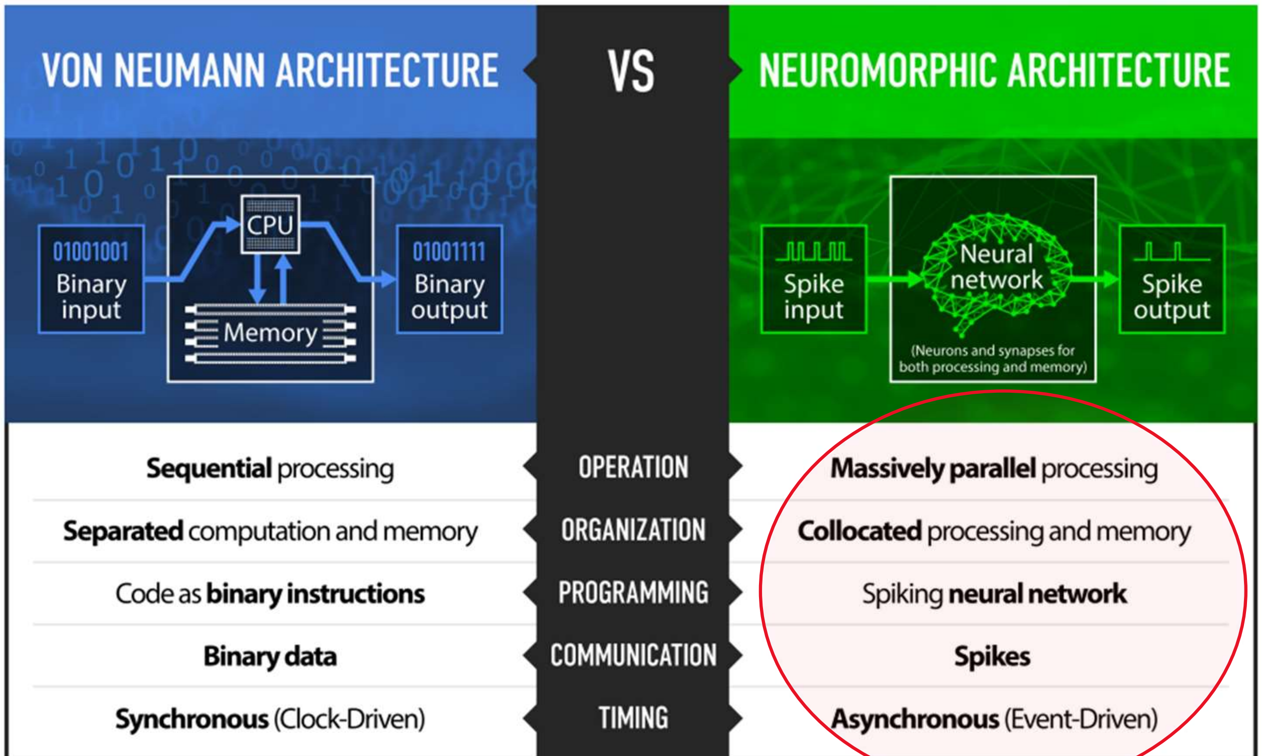
## Spike based-Computing for Edge AI

- Building up dedicated energy efficient accelerator for AI training is crucial
- Human brain can perform **1000 petaflops** of instructions against supercomputers which perform **1100 petaflop** instructions !!
- Human brain can perform using only **20 Wh/100 Wh** power against supercomputers that uses 5-20 MWh power




https://www.knack4data.com/celebrating-the-human-brain-brain-versus-machine-a92ecf8e3b92 [3]

# Introduction



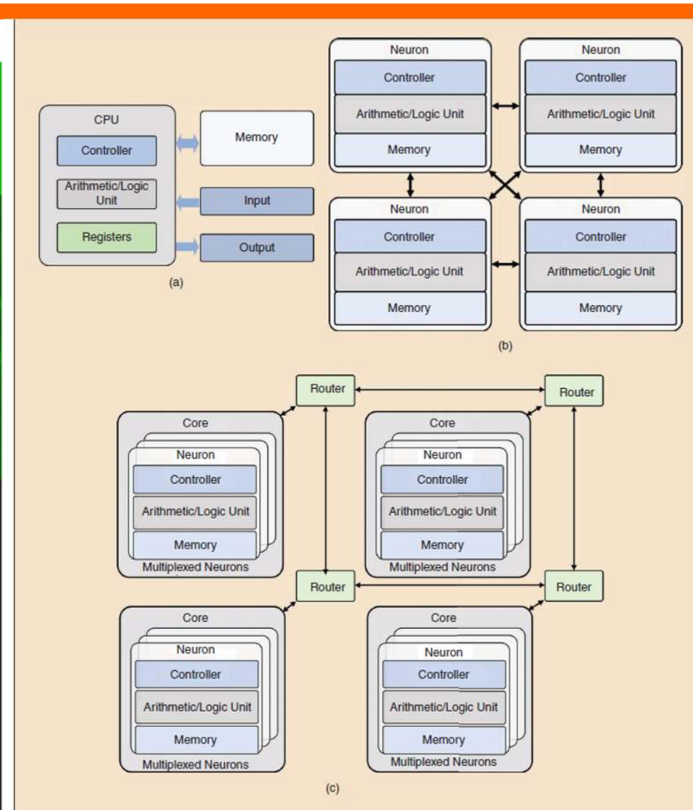




**Figure 9.** (a) Von Neumann Architecture (b) Ideal Neuromorphic Architecture (c) Practical Neuromorphic Core.

Schuman, Catherine D., Shruti R. Kulkarni, Maryam Parsa, J. Parker Mitchell, and Bill Kay. "Opportunities for neuromorphic computing algorithms and applications." *Nature Computational Science* 2, no. 1 (2022): 10-19. [4]

Shrestha, Amar, et al. "A survey on neuromorphic computing: Models and hardware." *IEEE Circuits and Systems Magazine* 22.2 (2022): 6-35. [5]

# Spike Based On Chip Training

- Leveraging **on-chip training** on hardware accelerator can benefit **Edge AI**
- So far, the ANN based accelerators use **off-line software training** and inference on hardware
- Spiking networks use **local learning** vs propagation-based learning of ANN

# Introduction

## Why LSM ?

- **Simpler Network (Less parameters)**
- **Sparse Spike based computation**
- **Local learning (No back propagation needed)**
- **Avoidance of Overfitting**
- **Hardware friendly design**

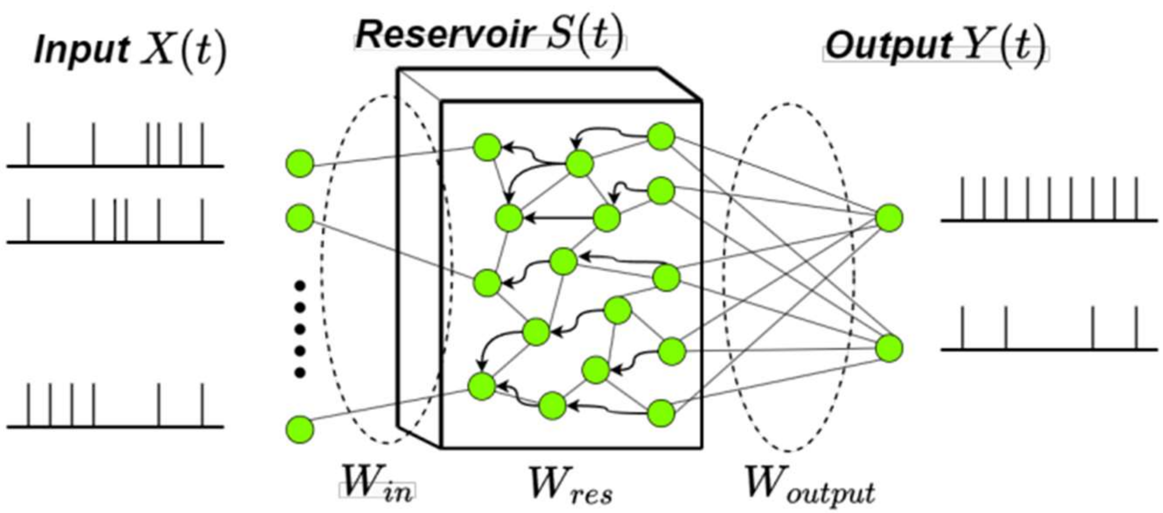



Figure : Liquid State Machine Network

- $s[t] = (1-c)s[t-1] + c \cdot f\_activation(W_{in}^{T}x[t] + W_{res}^{T}s[t-1]);$
- $y[t] = (W\_readout)^{T} * (s[t])$

# Introduction

## Challenges ?

- **Accuracy : Poorer performance because of low complex learning**

- **Solution: Find out combination of unsupervised local learning in reservoir and supervised learning in readout layer**

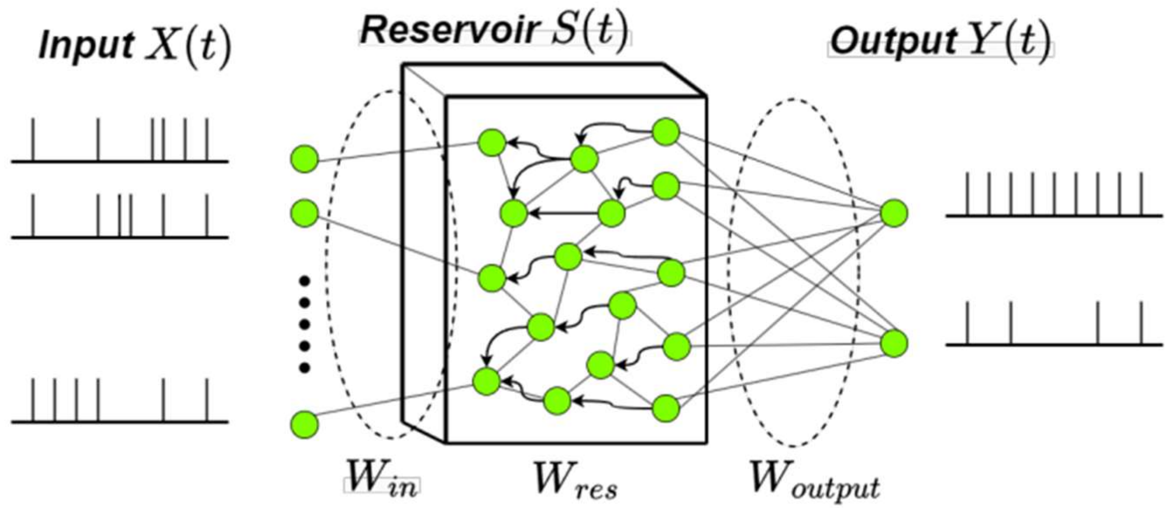



Figure : Liquid State Machine Network

- $s[\mathbf{t}] = (\mathbf{1}-c)s[\mathbf{t}-\mathbf{1}] + c \cdot f\text{\_activation}(W_{\mathbf{in}}{}^{\mathsf{T}} x[\mathbf{t}] + W_{\mathbf{res}}{}^{\mathsf{T}} s[\mathbf{t}-\mathbf{1}])$;
- $y[t] = (W\_readout)^{T} * (s[t])$

# State of the Art

- **Peng Li et ai. [6] Developed LSM in FPGA for speech signal recognition**
- **Developed two types of separate neuron named as LE (Learning Element) and OE (Output Element)**

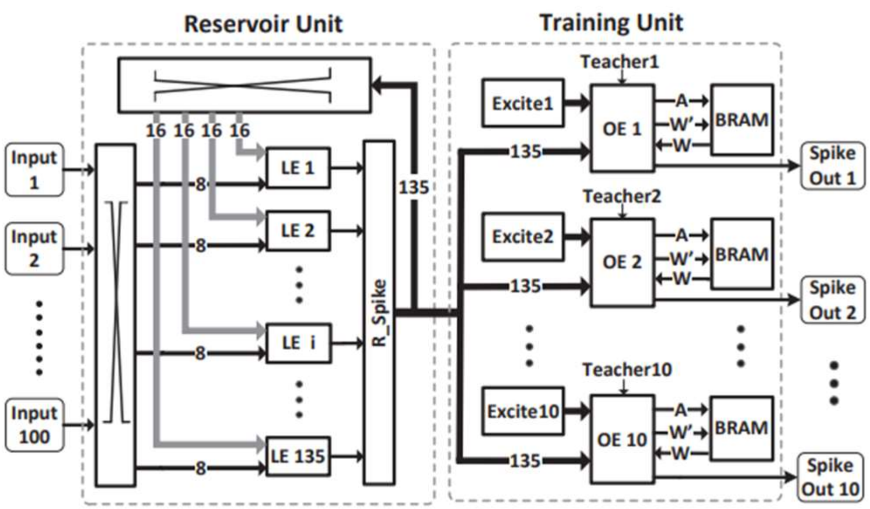



Figure : LSM network by Peng Li et ai [6]

# State of the Art

## Neuron Structure (LE and OE)

- **LIF spiking equation :**

$$V_{mem}(t) = V_{mem}(t-1) - \frac{V_{mem}(t-1)}{\tau} + R_{+} - R_{-}$$

$$R_{+} = \frac{ES_{+} - ES_{-}}{\tau_{ES_{+}} - \tau_{ES_{-}}}, \quad R_{-} = \frac{IS_{+} - IS_{-}}{\tau_{IS_{+}} - \tau_{IS_{-}}}$$

- **SRU (Synaptic Response Unit) calculation :**

$$\begin{cases} ES_{+}(t) = ES_{+}(t-1)(1 - 1/\tau_{ES_{+}}) + \sum w_i \cdot E_{+}(i) \\ ES_{-}(t) = ES_{-}(t-1)(1 - 1/\tau_{ES_{-}}) + \sum w_i \cdot E_{+}(i) \\ IS_{+}(t) = IS_{+}(t-1)(1 - 1/\tau_{IS_{+}}) + \sum w_i \cdot E_{-}(i) \\ IS_{-}(t) = IS_{-}(t-1)(1 - 1/\tau_{IS_{-}}) + \sum w_i \cdot E_{-}(i) \end{cases}$$

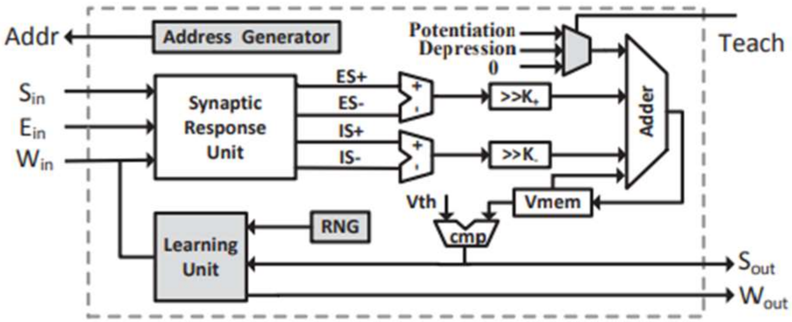



Figure : Digital Neuron by Peng Li et ai [7]

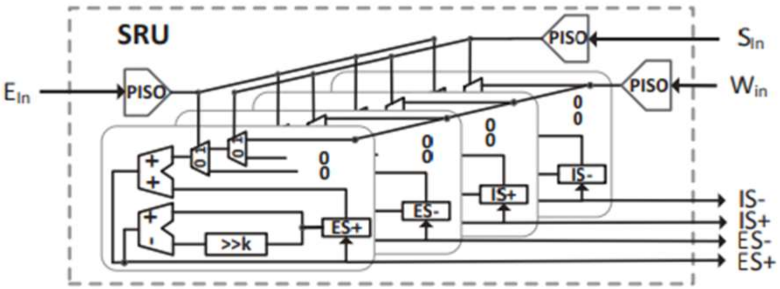



Figure : SRU unit by Peng Li et ai [7]

# State of the Art

## LSM Learning (Spike-timing-dependent-plasticity)

- **STDP Equation**

Reward-based STDP equation:

$$\text{Weight}_{ij} = \text{Weight}_{ij} + \eta \cdot \Delta w_{ij}^{\text{potentiation/depression}}$$

Where:

$$\Delta w_{ij}^{potentiation} = A_{\text{pos}} \cdot \exp(-\frac{\Delta t_{potentiation}}{\tau_{\text{pos}}})$$

$$\Delta w_{ij}^{depression} = -A_{\text{neg}} \cdot \exp(-\frac{\Delta t_{depression}}{\tau_{\text{neg}}})$$

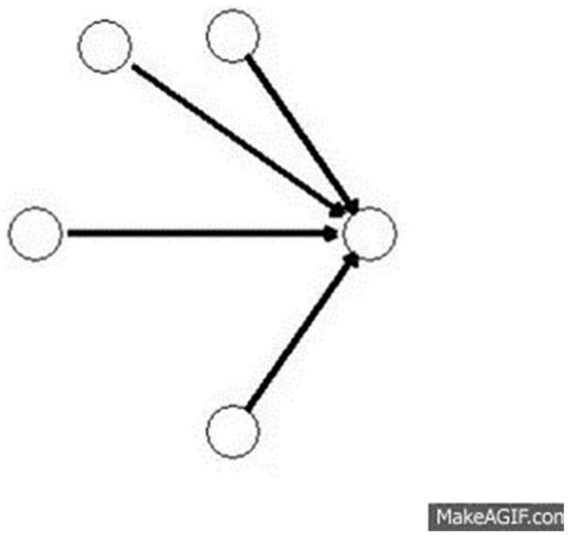



Figure : STDP GIF

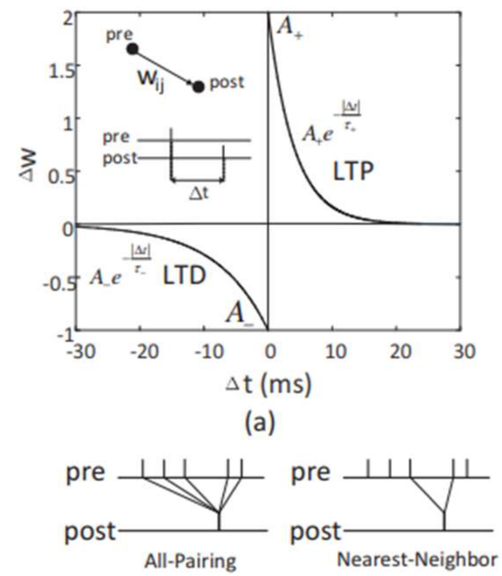



Figure : STDP nearest neighbor by Y. Jin et ai [8]

# State of the Art

## LSM Learning (Spike-timing-dependent-plasticity)

- **STDP Equation**

Reward-based STDP equation:

$$\text{Weight}_{ij} = \text{Weight}_{ij} + \eta \cdot \Delta w_{ij}^{\text{potentiation/depression}}$$

Where:

$$\Delta w_{ij}^{potentiation} = A_{\text{pos}} \cdot \exp(-\frac{\Delta t_{potentiation}}{\tau_{\text{pos}}})$$

$$\Delta w_{ij}^{depression} = -A_{\text{neg}} \cdot \exp(-\frac{\Delta t_{depression}}{\tau_{\text{neg}}})$$

- **Supervised STDP Learning**
  - **Uses a teacher signal (CT)**
  - **Follows one hot encoding**

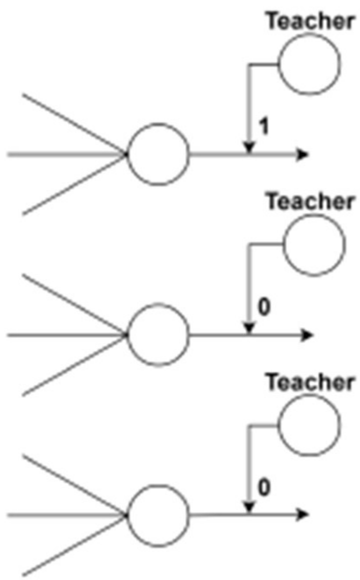



Figure [10] : Supervised STDP (Classification Teacher)

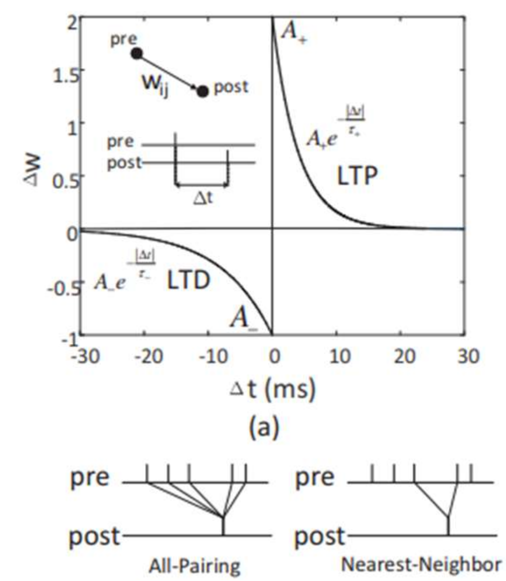



Figure : STDP nearest neighbor by Y. Jin et ai [8]

# State of the Art

## Learning Engine (Unsupervised LE)

- **STDP Equation curve:**

$$\Delta w^{+} = A_{+}(w) \cdot e^{-\frac{|\Delta t|}{\tau_{+}}} \, if \, \Delta t > 0$$

$$\Delta w^{-} = A_{-}(w) \cdot e^{-\frac{|\Delta t|}{\tau_{-}}} \, if \, \Delta t < 0,$$

- **Hardware STDP LUT:**

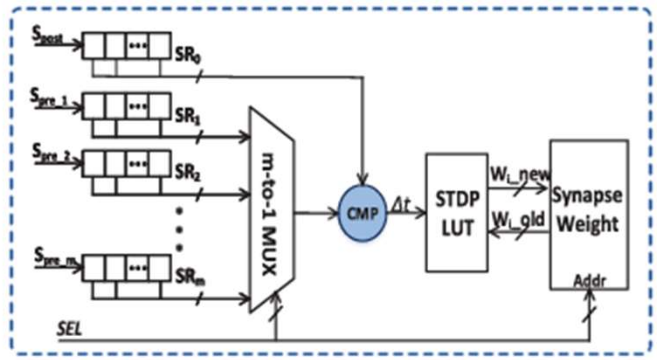
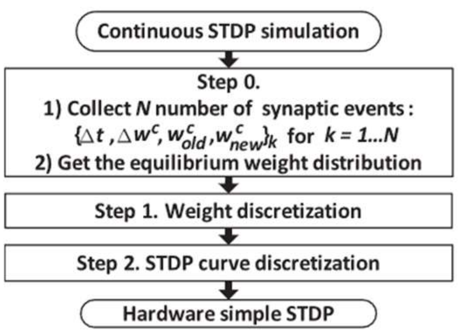
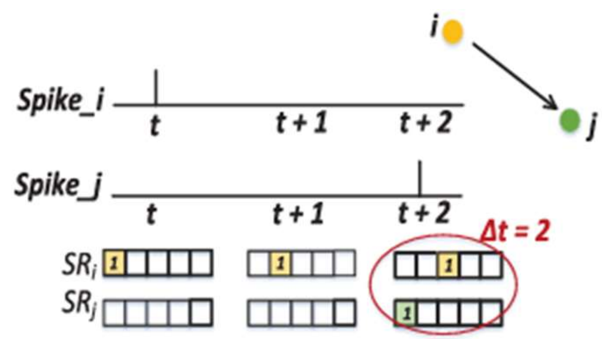



Figure : Unsupervised Learning unit by Peng Li et ai [6]



# State of the Art

## Learning Engine (Supervised OE)

- **STDP Equation :**

$$\Delta w^{+} = A_{+}(w) \cdot e^{-\frac{|\Delta t|}{\tau_{+}}} \; if \Delta t > 0$$

$$\Delta w^{-} = A_{-}(w) \cdot e^{-\frac{|\Delta t|}{\tau_{-}}} \; if \Delta t < 0,$$

- **Activity based Probabilistic-STDP :**

$$w \leftarrow w + \Delta W with \; p \propto |\Delta w^{+}|$$

$$w \leftarrow w - \Delta W with \; p \propto |\Delta w^{-}|$$

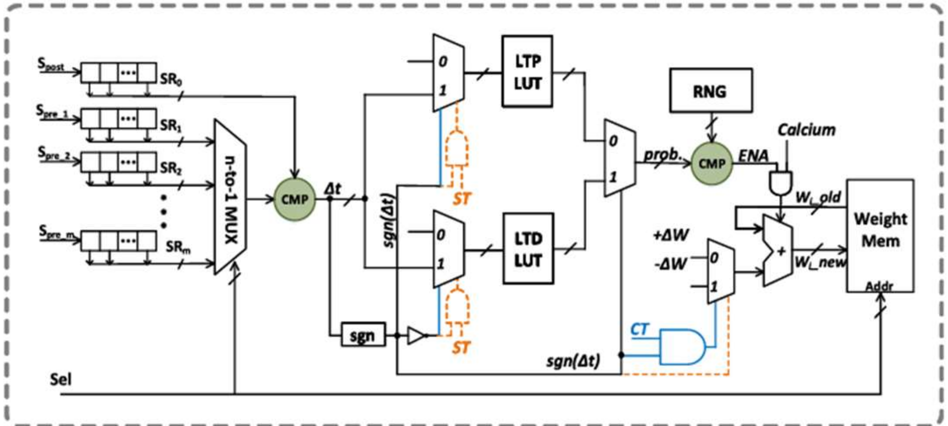



Figure : Supervised Learning unit by Peng Li et ai [6]

- **Two stages** of training :
  - ➢ **Sparsification** training period
  - ➢ **Classification** training period
- Uses two **look up tables** for AP-STDP

# Research improvement (Concept)

## Simplified LIF neuron vs SRU

- **Membrane Potential Equation:**

$$V_{mem}(t) = V_{mem}(t-1) - \frac{V_{mem}(t-1)}{\tau_{mem}} + \sum W_i * S_i$$

- **Advantages :**
  - **Vmem is directly calculated from weight and input spikes**
  - **Simplified digital neuron structure**
  - **Does not exacerbate performance**
  - **No need of Synaptic Response Unit (SRU)**

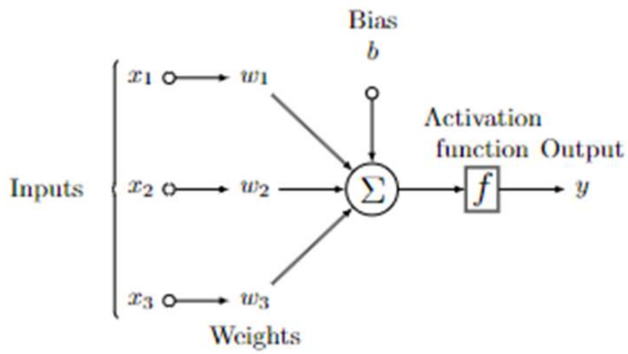



Figure [10]: Spiking Neuron Model

# Research improvement (Concept)

## Triplet STDP vs Duplet STDP in Unsupervised LE

- **STDP Equation curve:**

$$\Delta w = \begin{cases} \Delta w^+ = e^{-\Delta t_1/\tau^+}(A_2^+ + A_3^+ e^{-\Delta t_2/\tau^y}) \\ \Delta w^- = -e^{\Delta t_1/\tau^-}(A_2^- + A_3^- e^{-\Delta t_3/\tau^x}) \end{cases}$$

- **Advantages :**
  - **Considers 3 spike event instead of two spike events**
  - **Real-time exponential approximation**
  - **Encoder based asynchronous architecture**
  - **More accurate weight update**

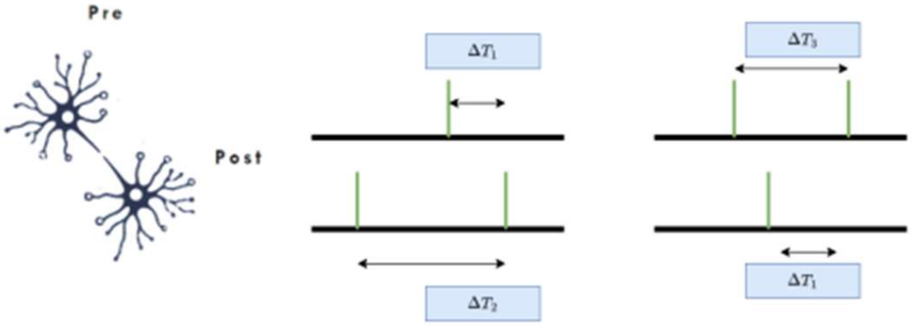



Figure : TSTDP Timing Differences for pre-post-pre spiking and post-pre-post spiking respectively

# Research improvement (Concept)

## Adaptive threshold vs Sparsification in Supervised OE:

- **Adaptive Threshold Equation:**

$$V_{th}(t) = V_{th}(t-1) - \frac{V_{th}(t-1)}{\tau_{th}} + C_{th}$$

- **Advantages :**
  - **Cth input is directly the spike event input**
  - **Helps to avoid weight saturation and thus overfitting**
  - **Reduces training time period as no sparsification mode needed**
  - **Used in typical software SNN models**

# Research improvement (Concept)

## Hardware Friendly Loss function

Reward-based STDP equation:

$$\text{Weight}_{ij} = \text{Weight}_{ij} + \eta \cdot \Delta w_{ij}^{\text{potentiation/depression}}$$

$$\Delta w_{ij}^{potentiation} = A_{\text{pos}} \cdot \exp(-\frac{\Delta t_{potentiation}}{\tau_{\text{pos}}})$$

$$\Delta w_{ij}^{depression} = -A_{\text{neg}} \cdot \exp(-\frac{\Delta t_{depression}}{\tau_{\text{neg}}})$$

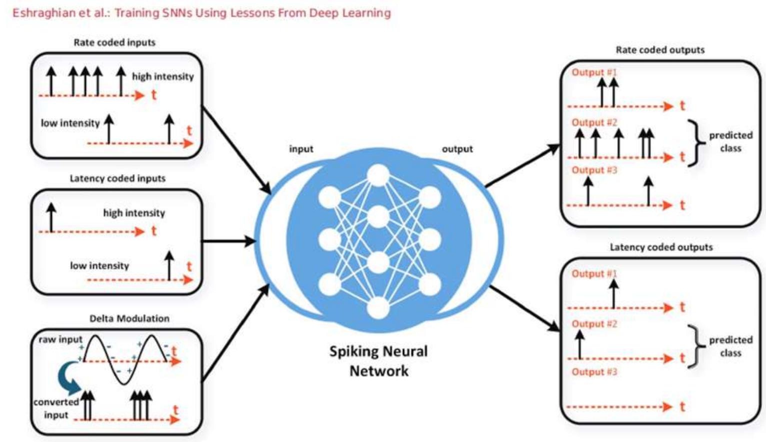



Figure [11] : Spike Output Encoding Scheme

Predicted Output $$\hat{y} = \sum_{t=0}^{T} S[t].$$

# Research improvement (Concept)

## Hardware friendly reward prediction error (RPE)

**Algorithm 1** Adaptive Learning with Weight Reversion

Initialize weights Wmatrix
Initialize other parameters and hyper-parameters
loss constant $\boldsymbol{C}$
**for** each iteration $i \rightarrow$ maximum number of iteration **do**
  Weight$_{temp} \leftarrow$ Wmatrix$_i$
  loss$_i \leftarrow 0$
  **for** each sample $j \rightarrow$ all the samples **do**
    $Wmatrix_i \leftarrow Wmatrix_i +$ reward
    $loss_i \leftarrow loss_i + (y_j - \hat{y}_j)$
  **end for**
  **if** loss$_i < loss_{i-1} - \boldsymbol{C}$ **then**  Wmatrix$_i \leftarrow Weight_{temp}$
**end if**
**end for**

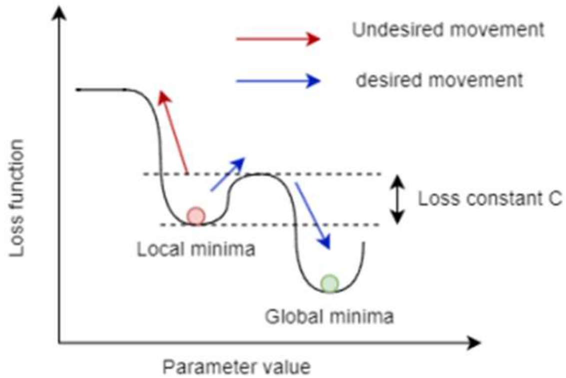



Figure : Loss function landscape

Predicted Output $$\hat{y} = \sum_{t=0}^{T} S[t].$$

Loss function $$L_{\text{loss}} = \sum_{i=1}^{\text{I}} (y_i - \hat{y}_i)$$

- Advantages
  - Guides to reach Global Minima
  - Helps reduce training period
  - Helps retaining optimized weights in hardware

# Detailed Hardware Implementation (Proposed)

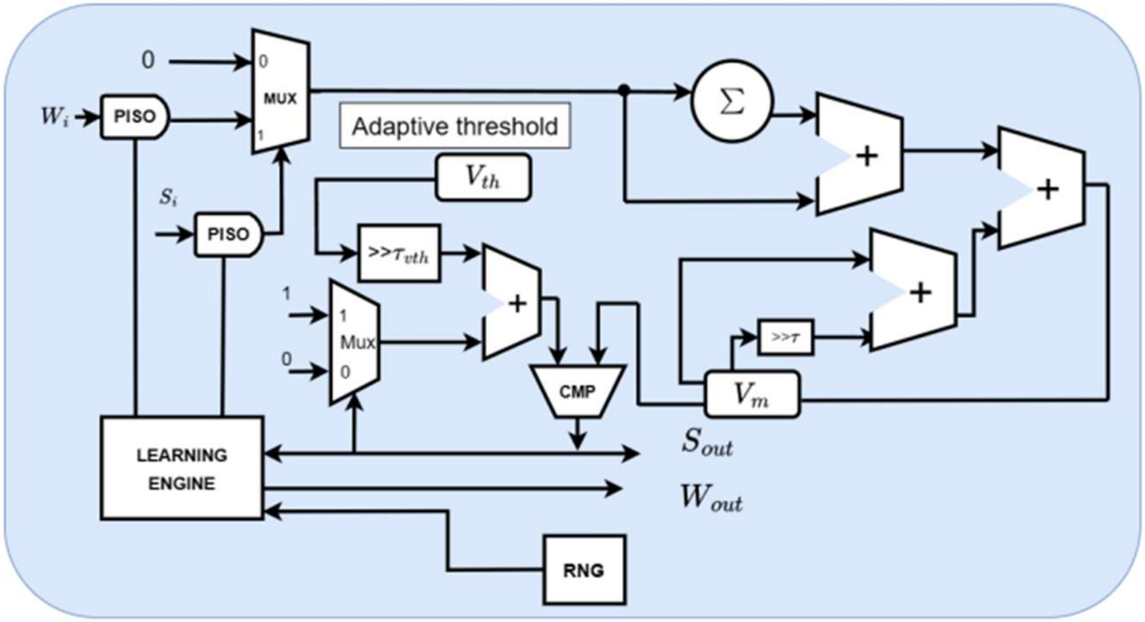
## Digital LIF neuron

- **Membrane Potential Calculation**

$$V_{mem}(t) = V_{mem}(t-1) - \frac{V_{mem}(t-1)}{\tau_{mem}} + \sum W_i * S_i$$

- **Adaptive threshold Calculation :**

$$V_{th}(t) = V_{th}(t-1) - \frac{V_{th}(t-1)}{\tau_{th}} + C_{th}$$




- No SRU needed
- Learning engine differs for LE and OE
- No sparsification stage needed

Figure: Spiking Neuron Model

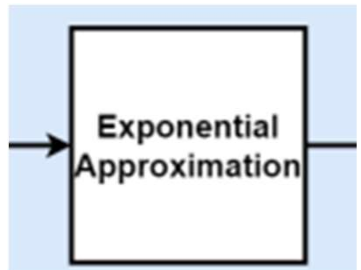
# Detailed Hardware Implementation (Proposed)

## Triplet STDP in Unsupervised LE

- **Finalized Triplet Equation**

$$\Delta w^{+} = A_2^{+} 2^{-1.4375\frac{\Delta t_1}{\tau^{+}}} + A_3^{+} 2^{-1.4375\left(\frac{\Delta t_2}{\tau_y} + \frac{\Delta t_1}{\tau_+}\right)}$$

$$\Delta w^{-} = -A_2^{-} 2^{-1.4375\frac{\Delta t_1}{\tau^{-}}} + A_3^{-} 2^{-1.4375\left(\frac{\Delta t_3}{\tau_x} - \frac{\Delta t_1}{\tau_-}\right)}$$



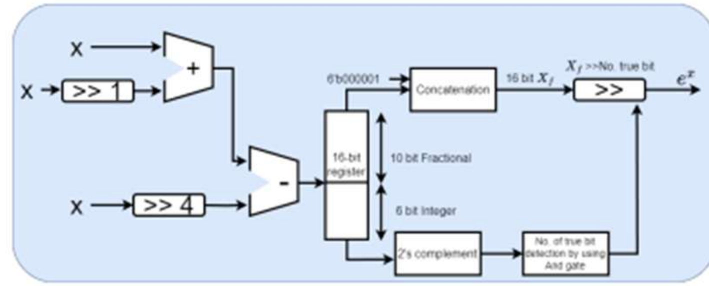




Figure: Implementation of exponential approximation in Triplet-STDP

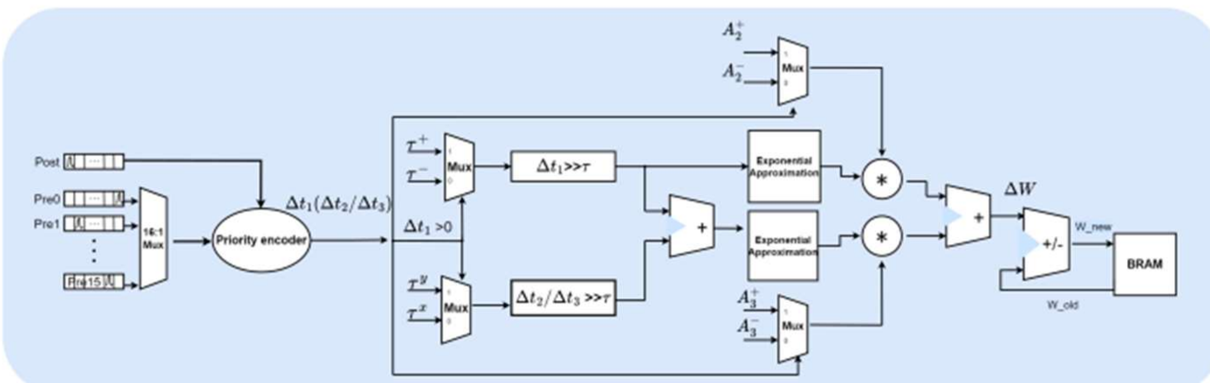



Figure: Hardware implementation of Triplet-based STDP

# Detailed Hardware Implementation (Proposed)

## Supervised Learning Engine (SLE)

- **Simplified hardware design**
  - **Less look up tables**
  - **No sparsification gates or modes**
- **Implements the loss function weight adaptation**

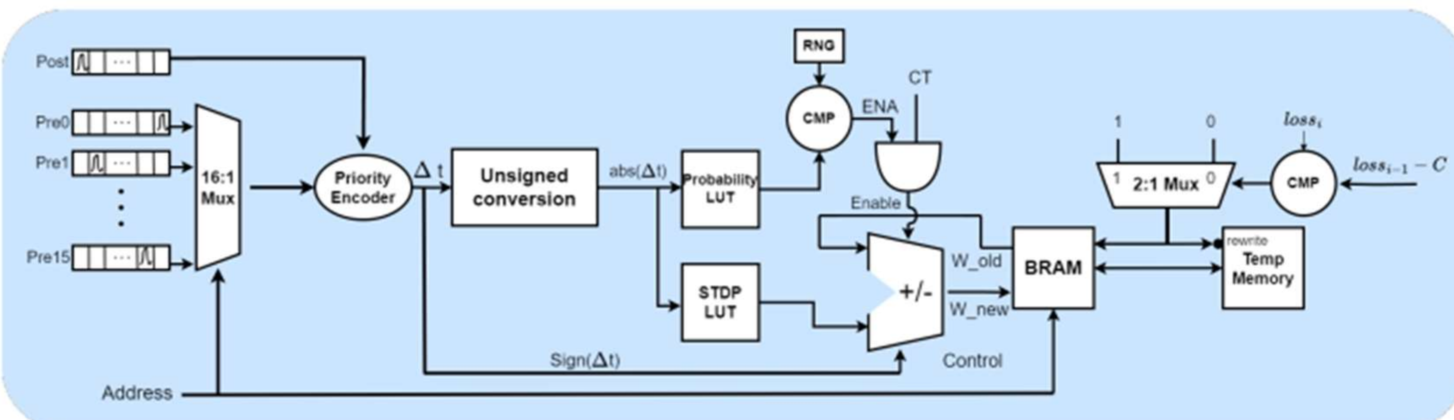
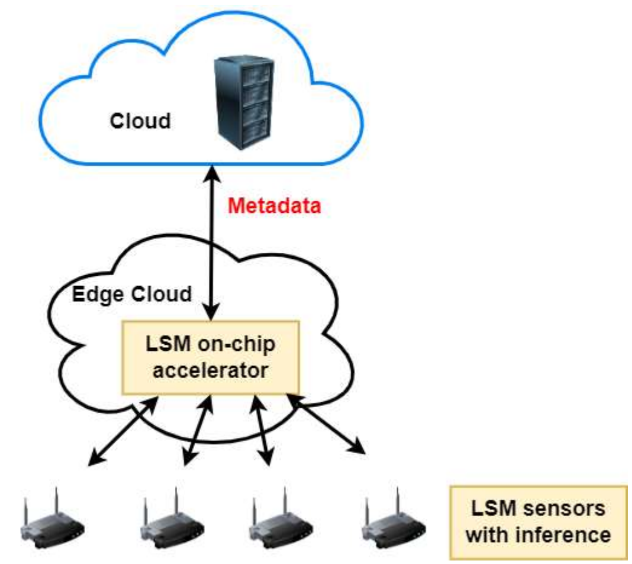



Figure: Implementation of exponential approximation in Triplet-STDP

# Performance Analysis

## Spectrum Sensing dataset from RWTH Achen University

- Input contains one feature data **of received energy signal**
- Output contains **binary targe**t label (1/0)
- Chosen for **testing performance against other accelerators**
- Central Frequency of channel (**fc=3750 MHz**)
- Bandwidth (**Bw=1500 MHz**)
- Frequency resolution (**fr=200 KHz**)
- Total Samples (7500)
- Train data (6000) (80%)
- Test data (1500) (20%)




Wang, L., Hu, J., Jiang, R., & Chen, Z. (2024). A Deep Long-Term Joint Temporal–Spectral Network for Spectrum Prediction. *Sensors*, *24*(5), 1498 [12].

# Performance Analysis

## Comparative Analysis (Accuracy)

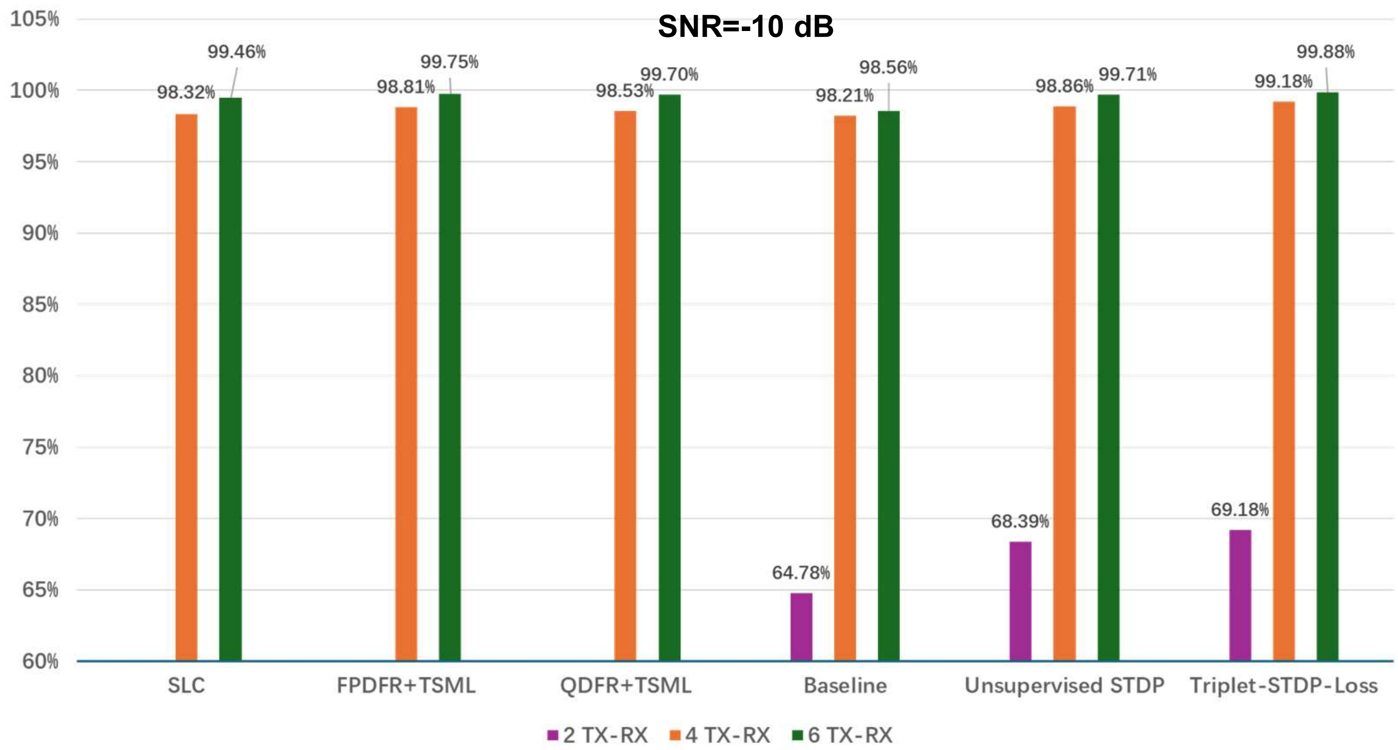



Figure: Accuracy comparison of different models

# Performance Analysis

## Comparative Analysis (Accuracy)

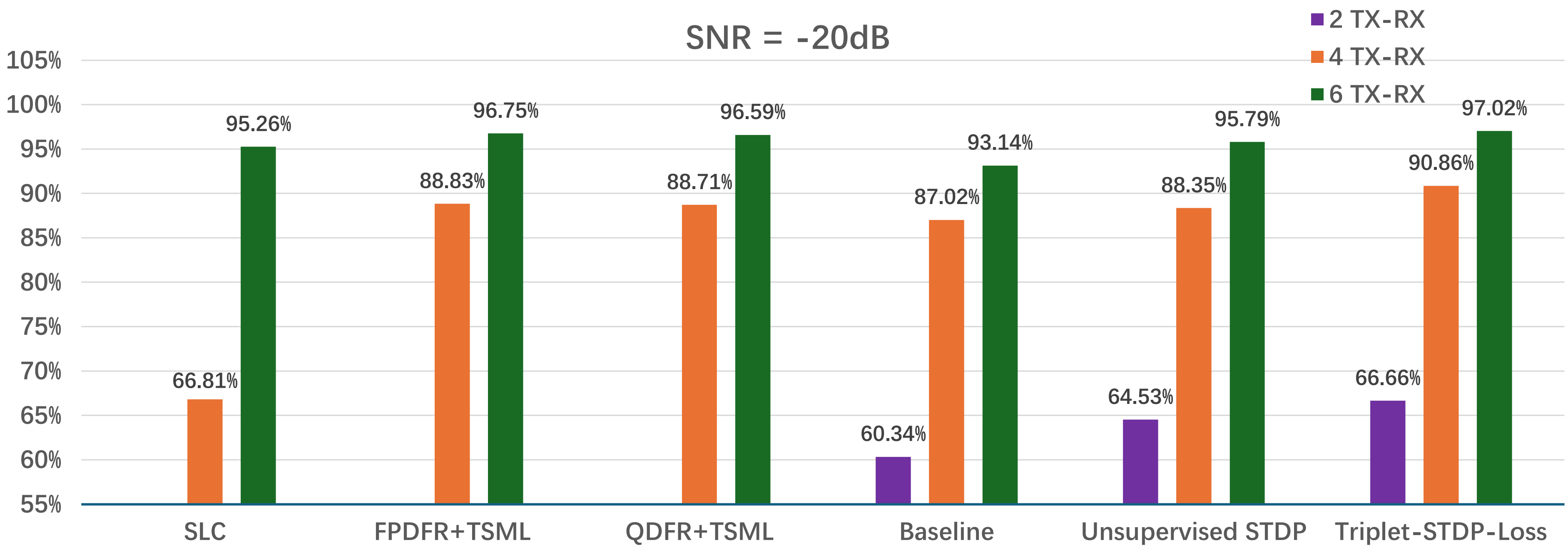



Figure: Accuracy comparison of different models

# Performance Analysis

## Loss function Training period improvement (%)

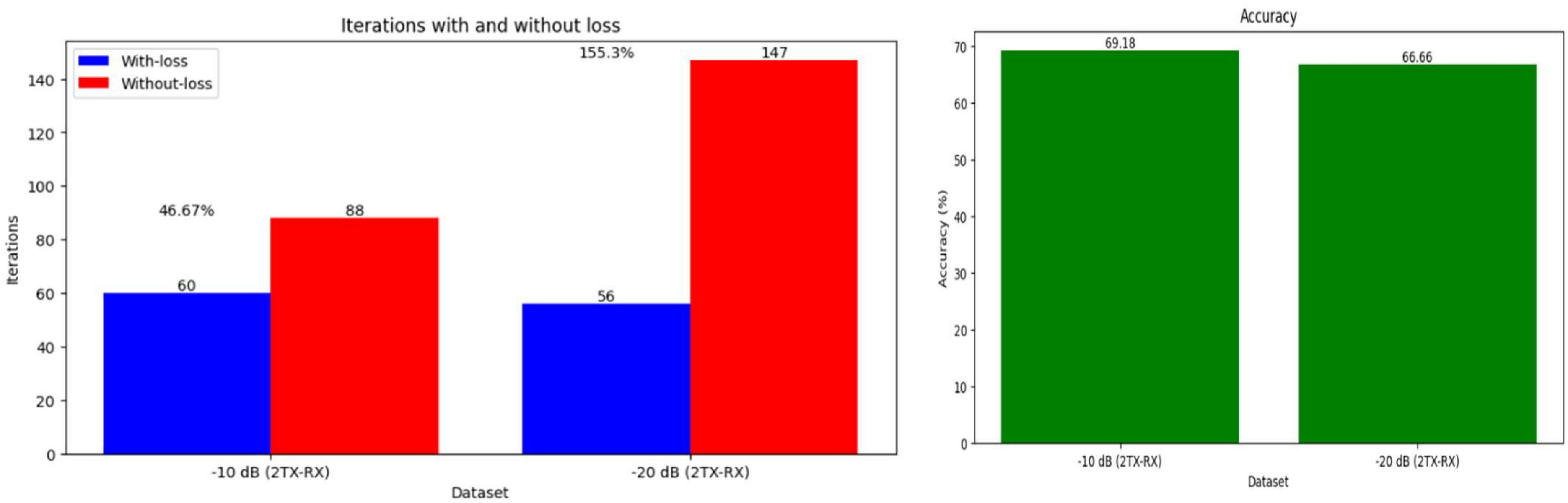
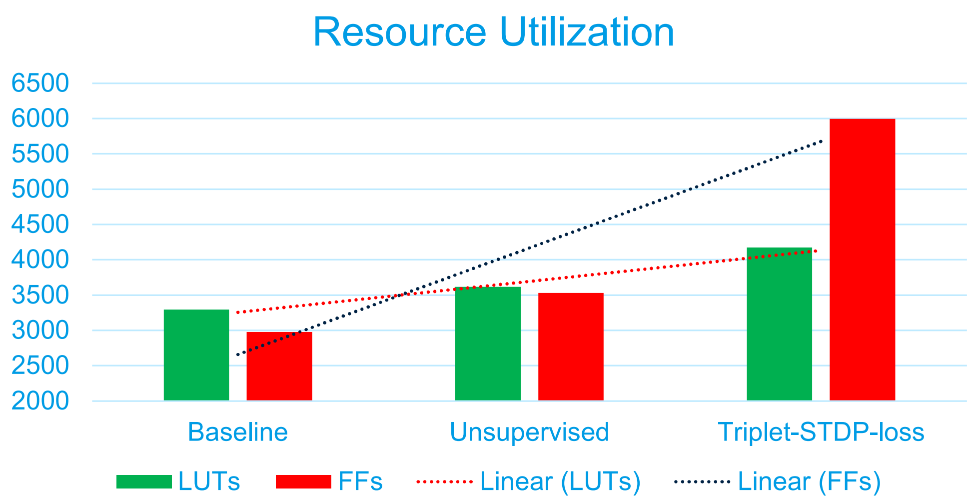



Figure: Training period improvement

# Performance Analysis

## FPGA resource utilization



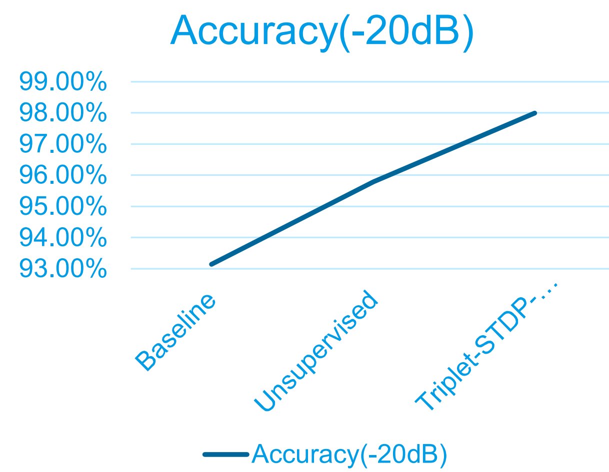




Table : Resource Comparisons of different architecture and accuracy comparison for -20 db dataset

# Performance Analysis

## Power Report

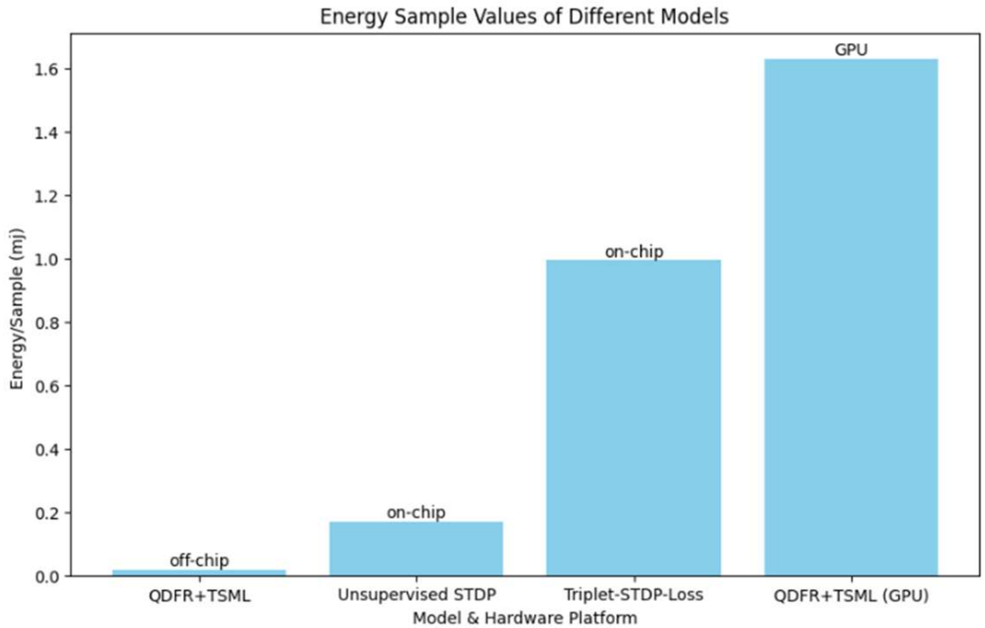



Table : Power consumption of different accelerators for spectrum sensing

# Performance Analysis

## Inference Time

TABLE VII: Inference Speed Comparison Table

| Model | Inference time |
|---|---|
| Unsupervised STDP | 1 |
| Triplet-STDP-without loss | 1.2x |
| Triplet-STDP-with loss | 1.2084x |

Table : Latency time during inference mode

# Summary

- **Improved accuracy in real-time on-chip training**

- **Reduced training period for worst case scenarios**

- **No sparsification training period needed**

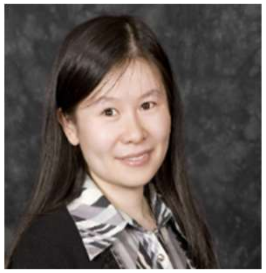
- **Achieved satisfactory power  and design optimization for on-chip training hardware**

# Future Work

- **Introduce local supervised learning with better and hardware friendly loss function to increase accuracy**
- **Build an LSM with Time-to-first-spike (TTFS) based R-STDP**
- **Simplify the reservoir structure with smarter unsupervised learning method and architectural optimization**
- **Increase power efficiency of the design using power clock gating**

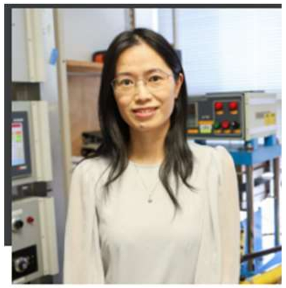
# Acknowledgments

## Advisory Committee


Dr. Yang (Cindy) Yi


Dr. Creed F. Jones


Dr. Dong Ha


Dr. Xiaoting Jia


Dr. Jeffrey Walling

# Reference


1. https://www.statista.com/chart/17727/global-data-creation-forecasts/
2. https://viso.ai/edge-ai/edge-ai-applications-and-trends/
3. https://www.knack4data.com/celebrating-the-human-brain-brain-versus-machine-a92ecf8e3b92
4. Schuman, Catherine D., Shruti R. Kulkarni, Maryam Parsa, J. Parker Mitchell, and Bill Kay. "Opportunities for neuromorphic computing algorithms and applications." Nature Computational Science 2, no. 1 (2022): 10-19.
5. Shrestha, Amar, et al. "A survey on neuromorphic computing: Models and hardware." IEEE Circuits and Systems Magazine 22.2 (2022): 6-35.
6. Y. Liu, S. S. Yenamachintala, and P. Li, “Energy-efficient fpga spiking neural accelerators with supervised and unsupervised spike-timingdependent-plasticity,” ACM Journal on Emerging Technologies in Computing Systems (JETC), vol. 15, no. 3, pp. 1–19, 2019.
7. Wang, Qian, Yingyezhe Jin, and Peng Li. "General-purpose LSM learning processor architecture and theoretically guided design space exploration." In 2015 IEEE Biomedical Circuits and Systems Conference (BioCAS), pp. 1-4. IEEE, 2015.
8. Jin, Y., & Li, P. (2016, July). AP-STDP: A novel self-organizing mechanism for efficient reservoir computing. In *2016 International Joint Conference on Neural Networks (IJCNN)* (pp. 1158-1165). IEEE.
9. Lagani, Gabriele, et al. "Spiking Neural Networks and Bio-Inspired Supervised Deep Learning: A Survey." *arXiv preprint arXiv:2307.16235* (2023).
10. Shrestha, Amar, et al. "Approximating back-propagation for a biologically plausible local learning rule in spiking neural networks." *Proceedings of the International Conference on Neuromorphic Systems*. 2019.
11. Eshraghian, Jason K., et al. "Training spiking neural networks using lessons from deep learning." *Proceedings of the IEEE* (2023).
12. Wang, L., Hu, J., Jiang, R., & Chen, Z. (2024). A Deep Long-Term Joint Temporal–Spectral Network for Spectrum Prediction. Sensors, 24(5), 1498.

# Contribution

## Gauri Sharma

- Software design of LSM for Algorithm Verification
- Triplet STDP RTL design
- Integration of Triplet STDP in LSM reservoir RTL
- Encoder RTL designs

## Muhammad Farhan Azmine

- Fixed point software design of Baseline LSM reservoir for hardware verification
- RTL design of LIF neuron, SRU, Learning engines of both Unsupervised and Supervised algorithm and verification
- Integration of LSM reservoir RTL and verification of baseline
- Integration of Triplet STDP in LSM reservoir RTL