IEEE Xplore® Digital Library

2020 IEEE International Symposium on Circuits and Systems
Virtual, October 10-21, 2020

Low-Cost Reservoir Computing Using Cellular Automata and Random Forests

Ángel López García-Arias*,

Jaehoon Yu †, Masanori Hashimoto*

* Osaka University, † Tokyo Institute of Technology

2020 IEEE International Symposium on Circuits and Systems
Virtual, October 10-21, 2020

大阪大学
OSAKA UNIVERSITY

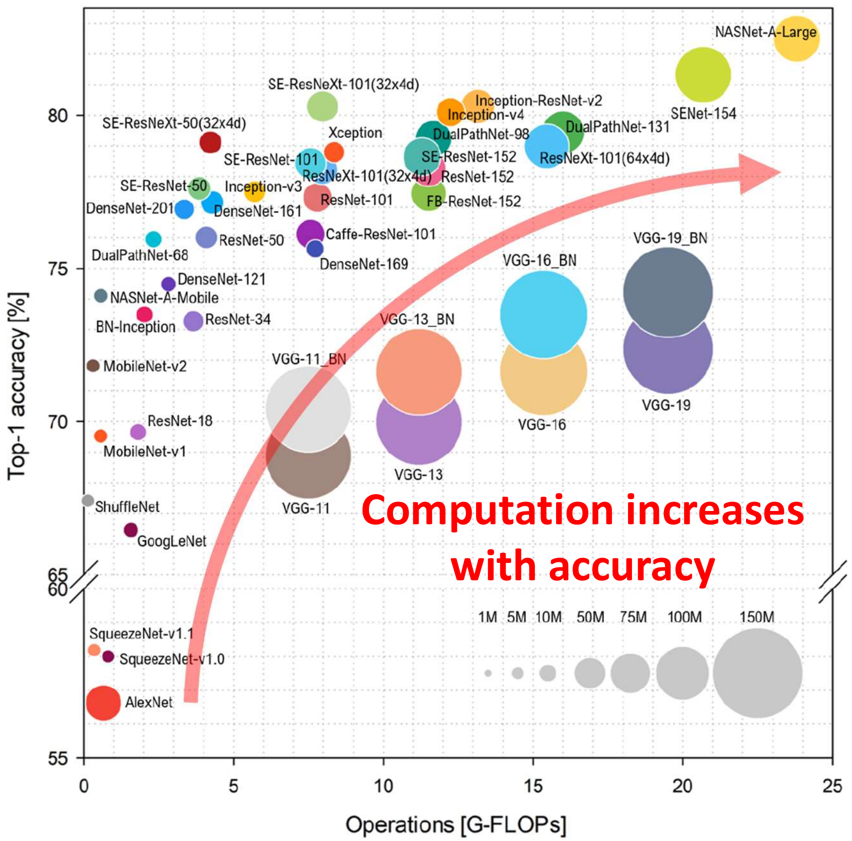
東京工業大学
Tokyo Institute of Technology

Outline

- Background
  - **Reservoir Computing**
  - **Cellular Automata**
  - **ReCA Image Classifier**
- Proposed Model Optimizations
  - **Random Forest Classifier**
- Architecture for Feature Generation
- Conclusion

Trends in Computer Vision

- Convolutional Neural Networks (CNNs) lead the field



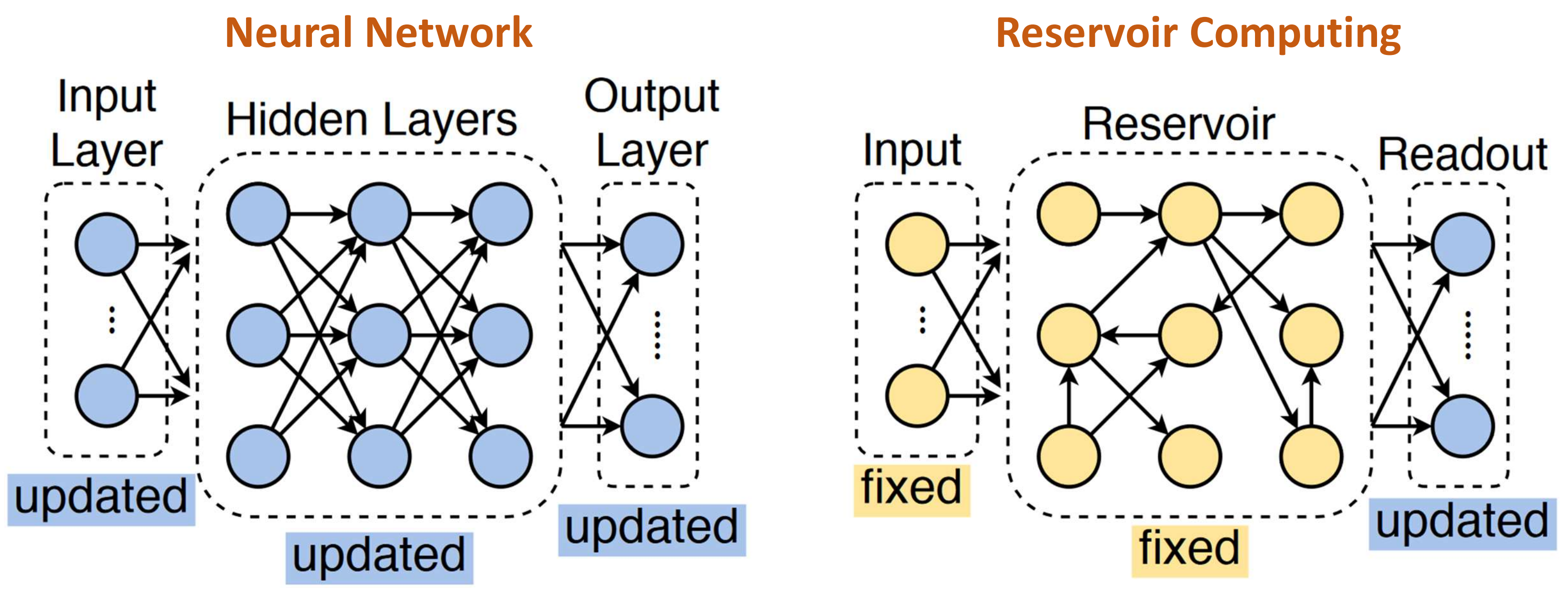
Trend: improve accuracy by growing size

↓

Too expensive for mobile systems

S. Bianco, R. Cadene, L. Celona and P. Napoletano, "Benchmark Analysis of Representative Deep Neural Network Architectures," in IEEE Access, vol. 6, pp. 64270-64277, 2018.

Reservoir Computing (RC)

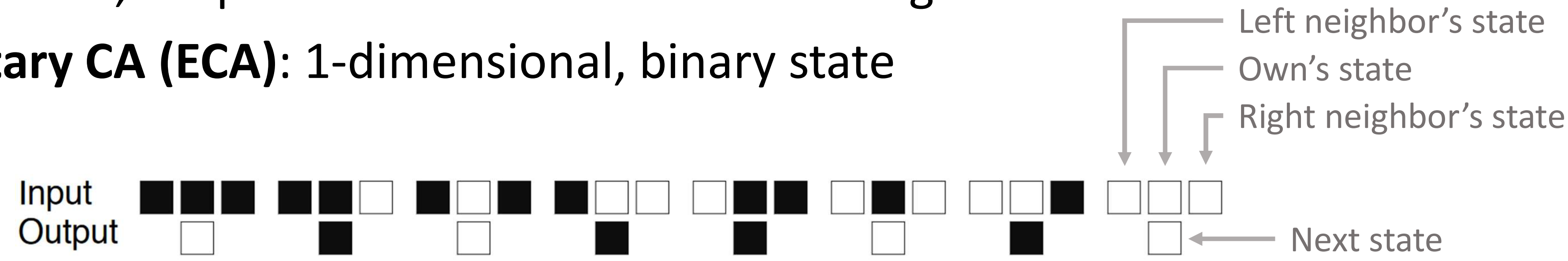


- **Reservoir (does not learn)**: augments features non-linearly
- **Readout (learns): Only updates output layer weights**
- Possibility of physical reservoirs (non-digital) [1]

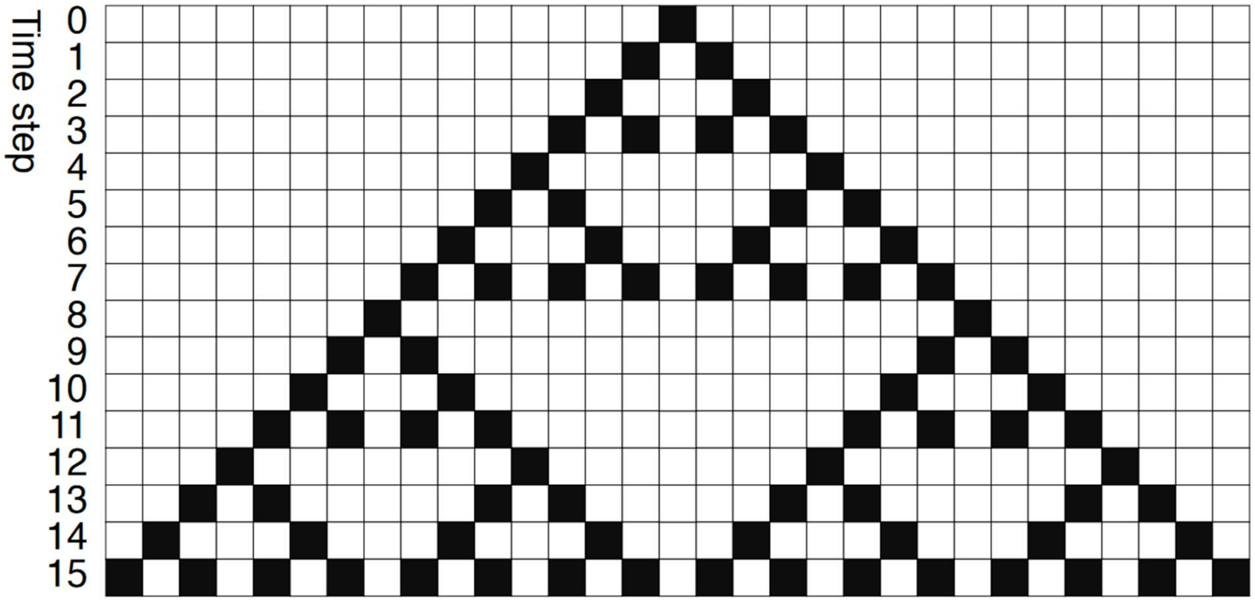
[1] G. Tanaka, T. Yamane, et al., “Recent advances in physical reservoir computing: A review,” Neural Networks, Jul 2019.

Cellular Automata (CA)

- Discrete state, simple evolution rules based on neighbors
- **Elementary CA (ECA)**: 1-dimensional, binary state

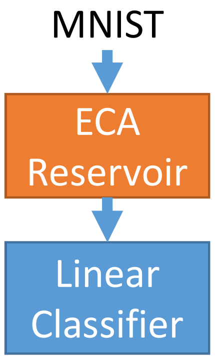


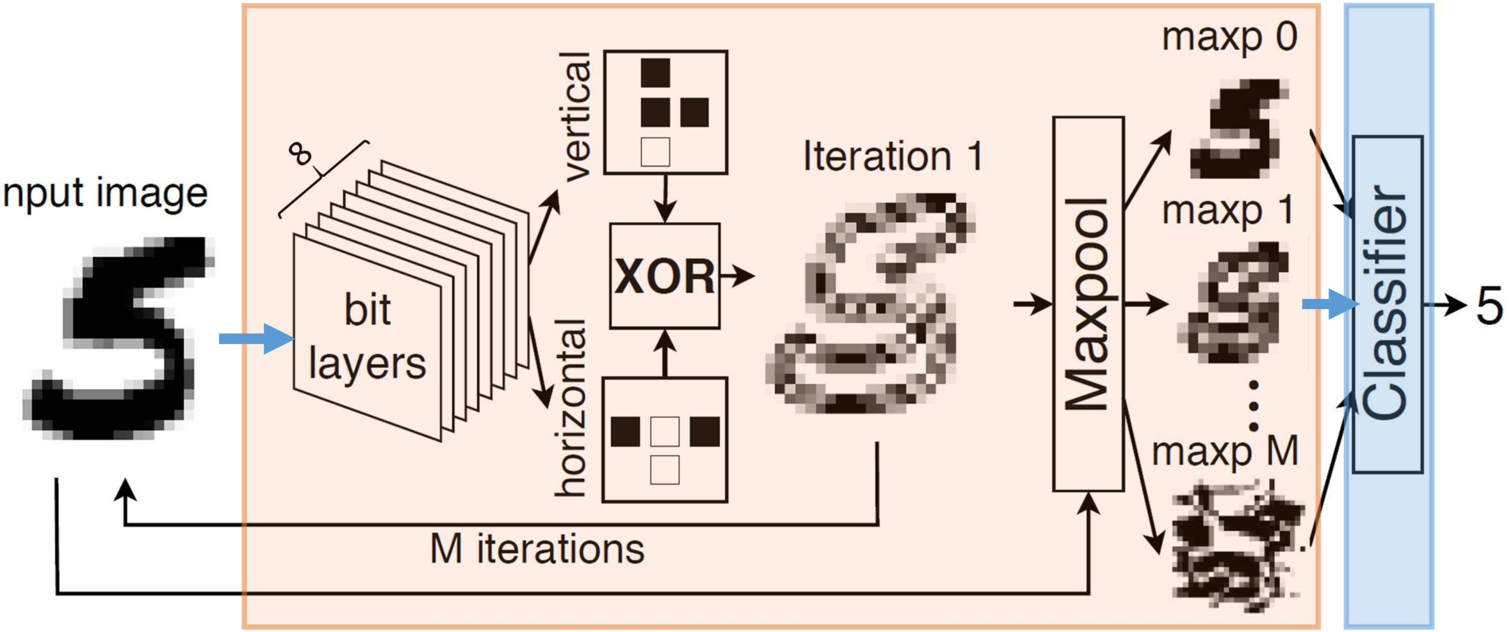
Simple Computation
↓
Complex Behaviour



Rule 90
↓
Fractal
(Sierpiński Triangle)

ReCA Image Classifier [2]





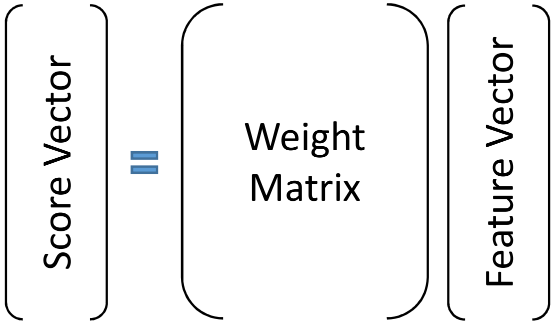
- Simple reservoir + simple classifier
- **Rule 90** is the most efficient reservoir
- Very low energy, **97’3% acc. on MNIST**

[2] A. Morán, C. F. Frasser, and J. L. Rosselló, “Reservoir computing hardware with cellular automata,” 2018.

ReCA: Classifier's Cost

- [2] uses **linear combination:**

$$y = \text{argmax}(\text{W}\bar{\text{x}})$$



- **Classifier spoils reservoir**'s low computation:

$$\#\ \text{multiplications} = \text{f} \cdot \text{c},$$

$$\#\ \text{additions} = (\text{f} - 1) \cdot \text{c},$$

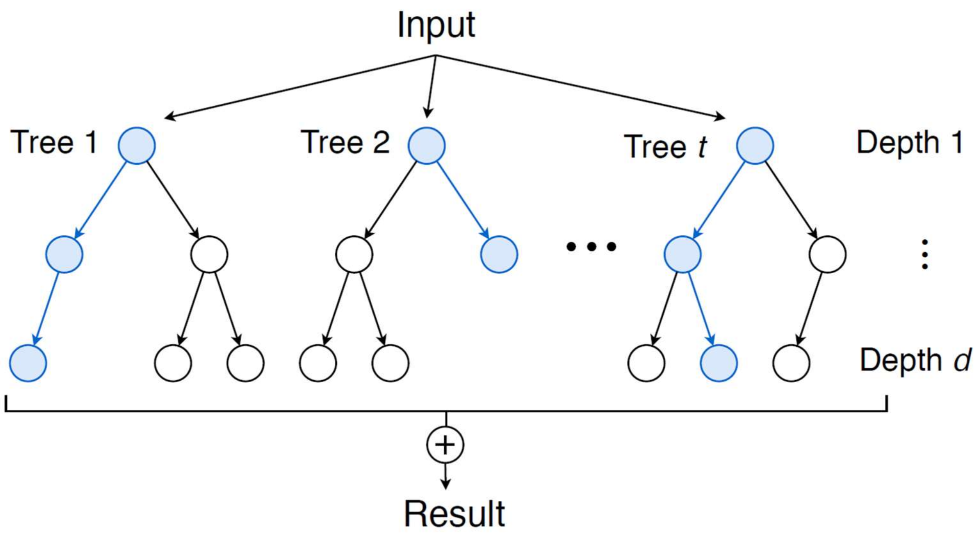
f: # features, c:# classes.

→ MNIST case: $\#\ \text{mult.} = 33{,}320,\ \ \#\ \text{add.} = 33{,}310$

- # Operations **scales with # features**

[2] A. Morán, C. F. Frasser, and J. L. Rosselló, "Reservoir computing hardware with cellular automata," 2018.

Random Forest Classifier (RF)



- # Operations (worst case):

$$\textbf{\# multiplications} = \mathbf{0},$$

$$\text{\# additions (\# comparisons)} = \mathrm{t} \cdot \mathrm{d},$$

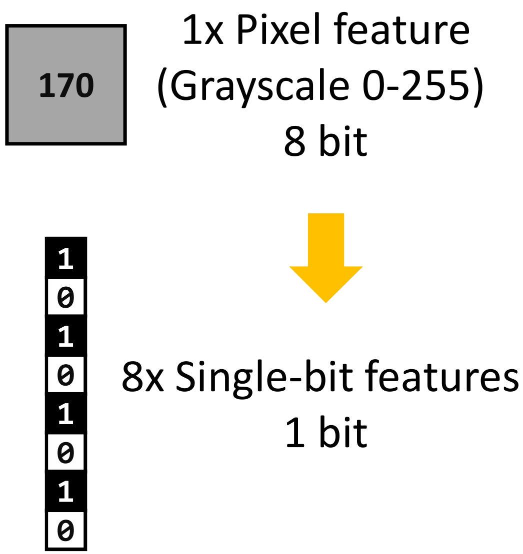
t: # trees, d: max. depth.

- **Does not scale with # features**

Single-Bit Features

- RF: no arithmetic operations with features
- RF: # operations independent of # features

→ **treat each 8-bit pixel as eight 1-bit features**



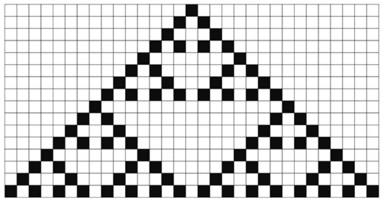
Advantages in specialized hardware:

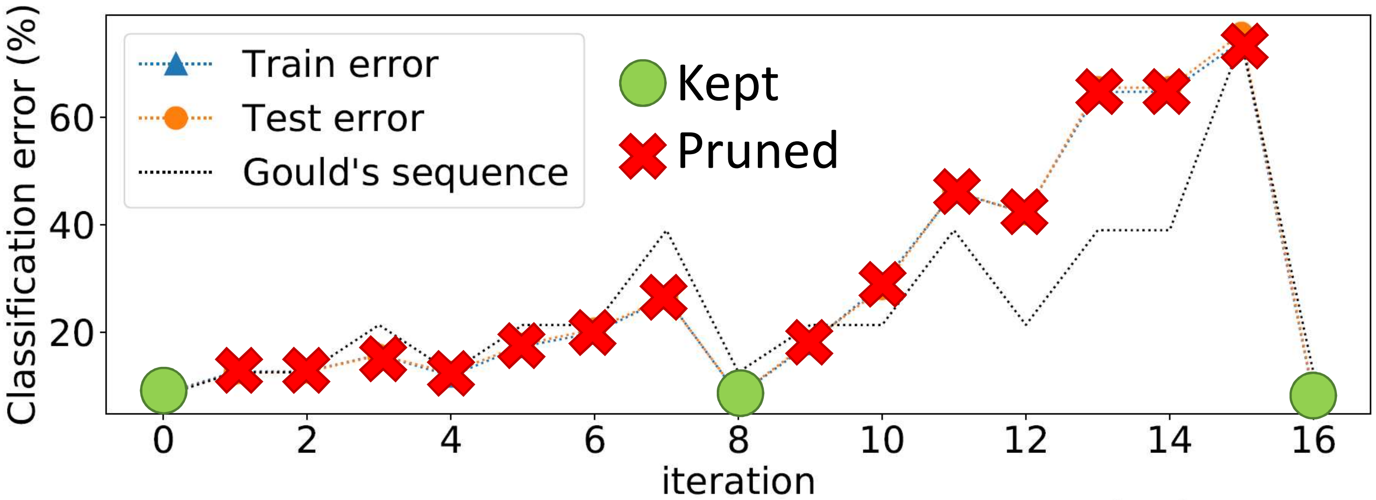
- Single-bit add. → **8x less computation per add.**

  Since 8-bit adder is made of 8x 1-bit adder
- 1-bit RF weights
- Better pruning granularity

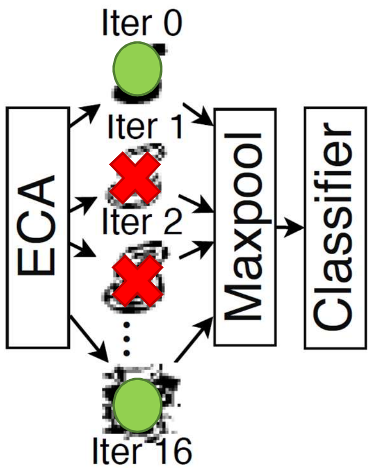
Iteration Pruning

- Some ECA iterations contribute little



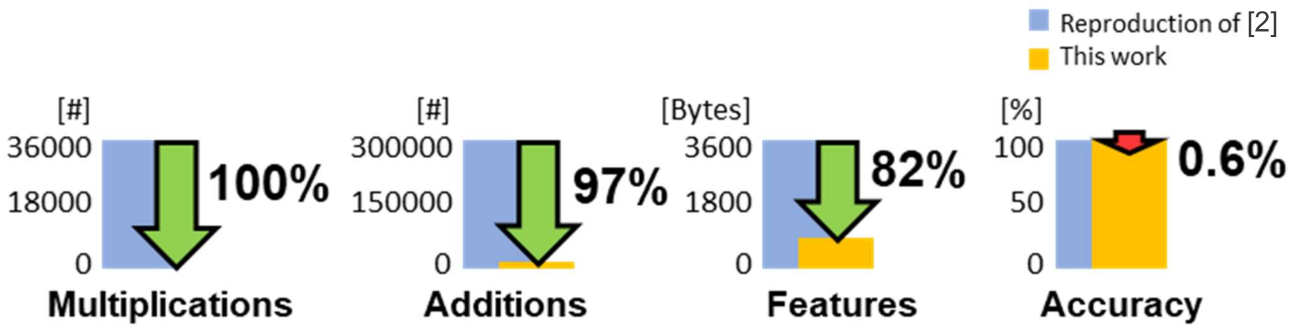


- Only keep iterations with lower error
- **82% of features are pruned**
- Error/iter. follows a **predictable trend**



Results

Classifier	Accuracy	# Mult.	# 1b Add.	Features (Bytes)
Linear Combination [2]	**97.3%**	**33,320**	**266,480**	**3,332**
Random Forest	97.0%	0	64,000	3,332
RF, binary features	96.0%	0	8,000	3,332
RF, pruned bin. features	**96.7%**	**0**	**8,000**	**588**



[2] A. Morán, C. F. Frasser, and J. L. Rosselló, "Reservoir computing hardware with cellular automata," 2018.

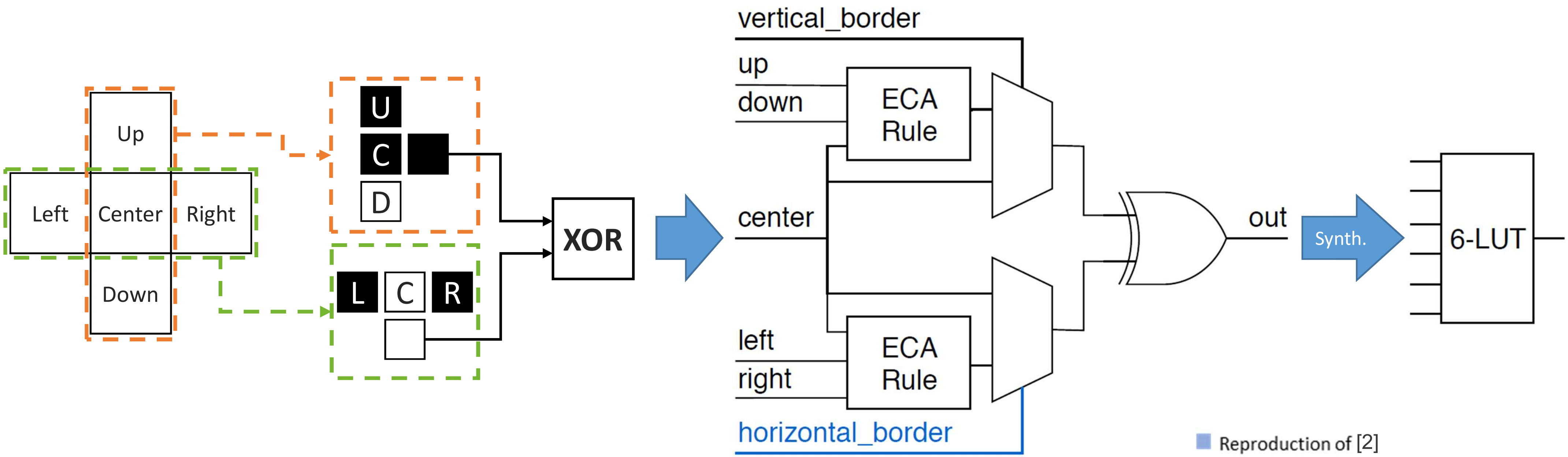
Architecture for Feature Generation

ECA is well-suited for hardware optimization

- **Simple** computation → efficient use of **resources**
- **Local** computation→ minimal **data transfers**
- Intrinsically **parallel** → high **throughput**

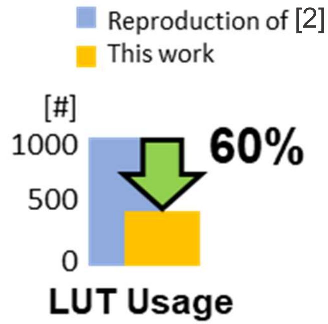
Simple Computation → Efficient use of Resources

ECA PU maps optimally to FPGA 6-LUT



FPGA RESOURCE UTILIZATION COMPARISON

	# CLB LUTs	# CLB Registers	# BRAM
This work	392	822	8
Reproduction of [2]	971	822	8

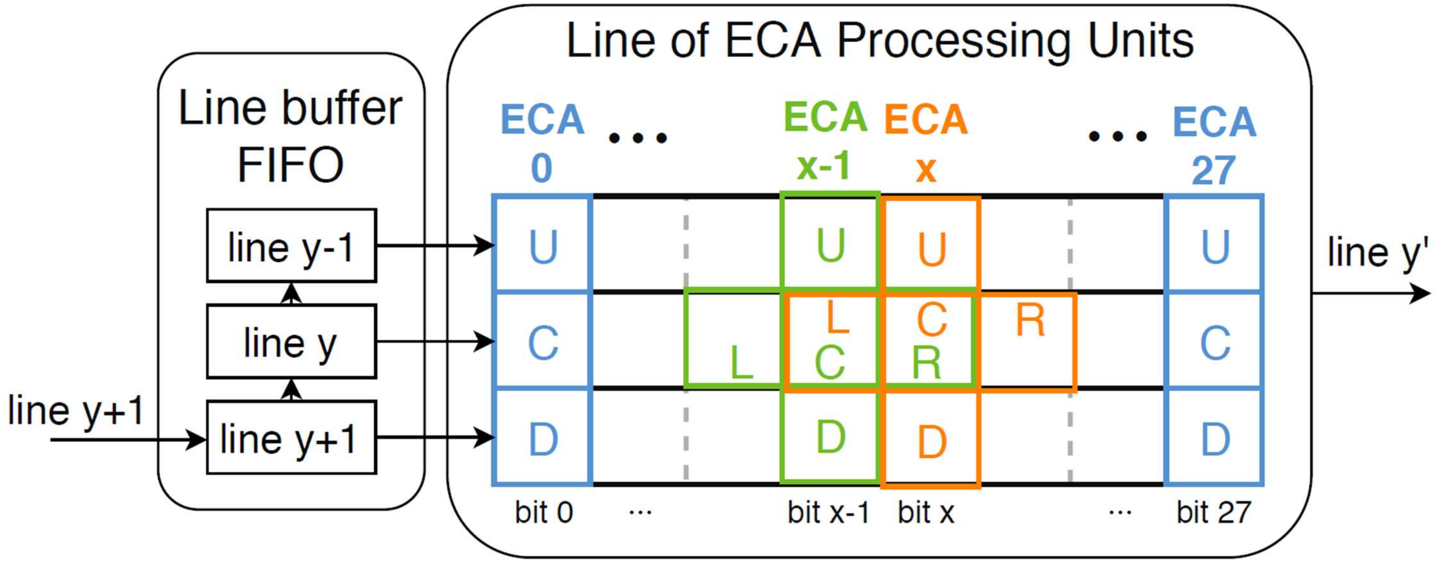


PU: Processing Unit, 6-LUT: 6-input Look Up Table; U: up; C: center; D: down; L: left; R: right

Local Computation → Minimal Data Transfers

Each pixel is only read once

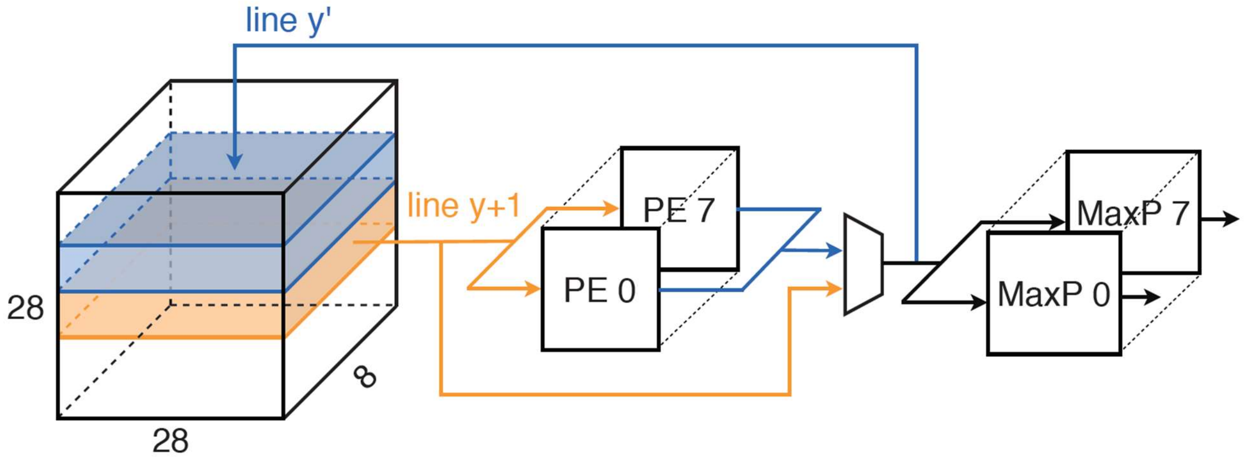
- PE: 3-line buffer FIFO + Line of 28 ECA PUs
- Minimizes data transfers (most expensive)

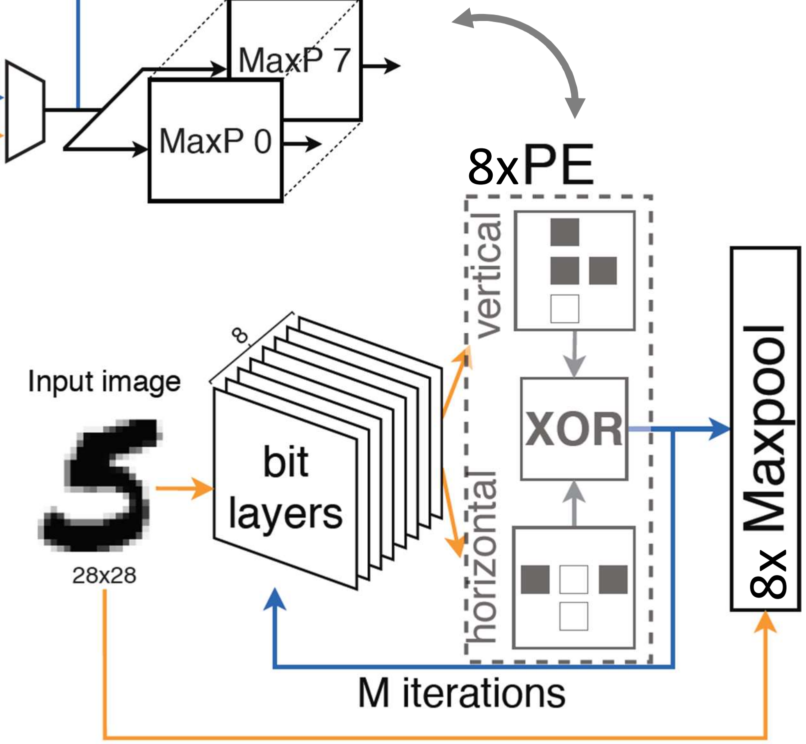


PE: Procesing Element; PU: Processing Unit; U: up; C: center; D: down; L: left; R: right

Intrinsically Parallel → High Throughput

One line is processed every cycle



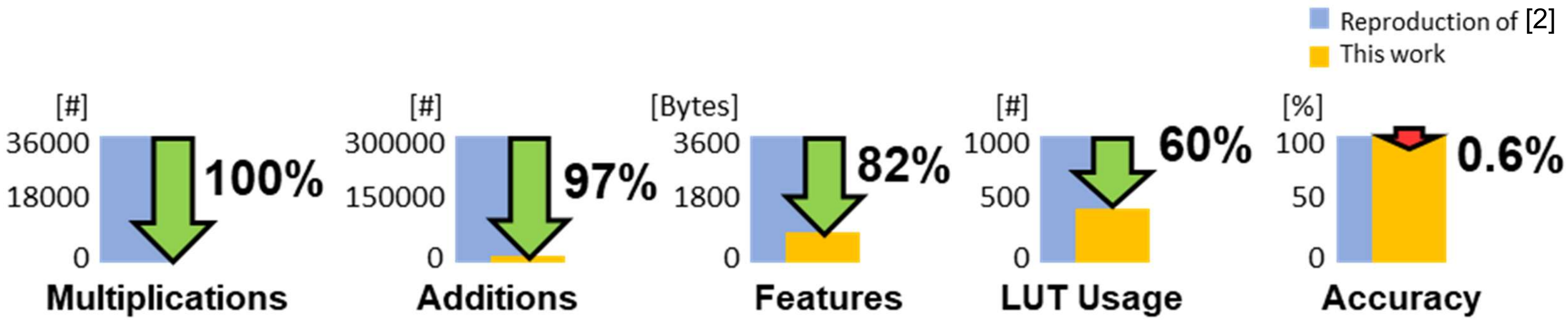


- 8 bit layers processed in parallel
- 1 line every cycle
- 1 iteration every 28 cycles

PE: Processing Element; MaxP: Maxpooling Unit

Conclusion

- **ReCA**: low computational cost image classifier
- **Random Forest** is a better fit for ReCA
- **Architecture design** can improve **efficiency** further
- **Vast computation cuts**, slight accuracy drop
- Further experiments needed with larger and more complex datasets.



[2] A. Morán, C. F. Frasser, and J. L. Rosselló, “Reservoir computing hardware with cellular automata,” 2018.