Comparison of most used activation functions in deep neural networks and their circuit realizations in analog and digital neural networks

ELEC-L352001: Postgraduate Course in Electronic Circuit Design

Andrei Spelman

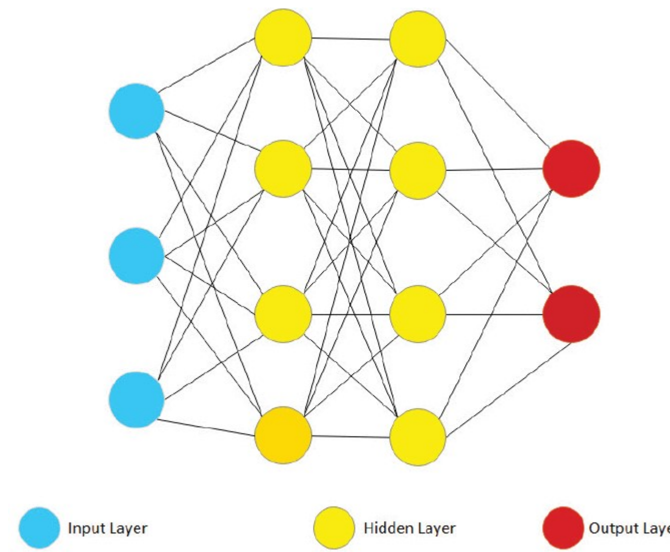
18.5.2022

Aalto-yliopisto
Aalto-universitetet
Aalto University

- **Introduction**
- **Theory**
  - Sigmoid function
  - Hyperbolic tangent
  - Rectified linear units
- **Comparison**
- **Hardware**
- **Assignment**

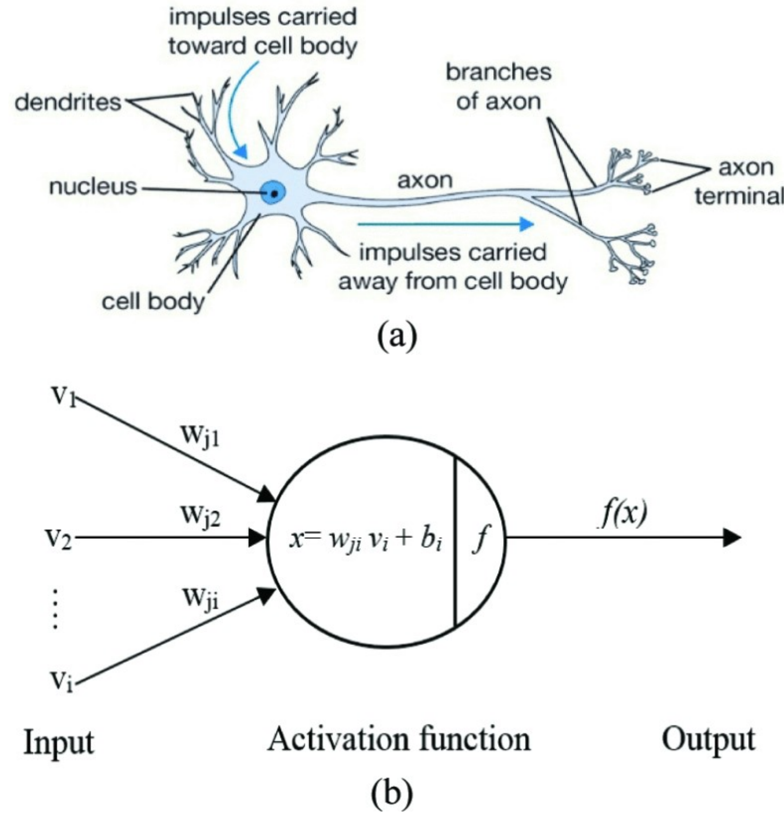
Introduction

- Main functional elements of neural networks are neurons
- Used for solving complex problems such as pattern classification, clustering, prediction, control and function approximation
- Deep networks have layers between input and output
- More calculations, more layer, more complex



Neurons

- Inputs are weighted
- Adder sums input data together
- Activation function decides if neuron activated or not
- Activation functions can be linear or non-linear
- Non-linear are used for more complex calculations



Most used activation functions

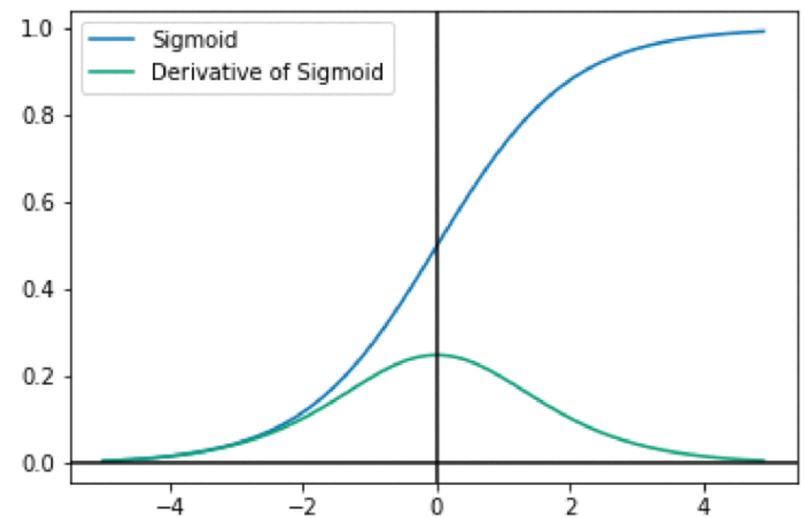
- Sigmoid function
- Hyperbolic tangent
- Rectified linear units

Theory

Sigmoid function

$$f(x) = \frac{1}{1 + e^{-x}}$$

- Saturated
- Centered at 0.5
- Gradient vanished especially in deep networks
- Soft saturation results in the difficulties of training a deep neural network
- Not used in deep networks
- Hard to optimize



Improvements

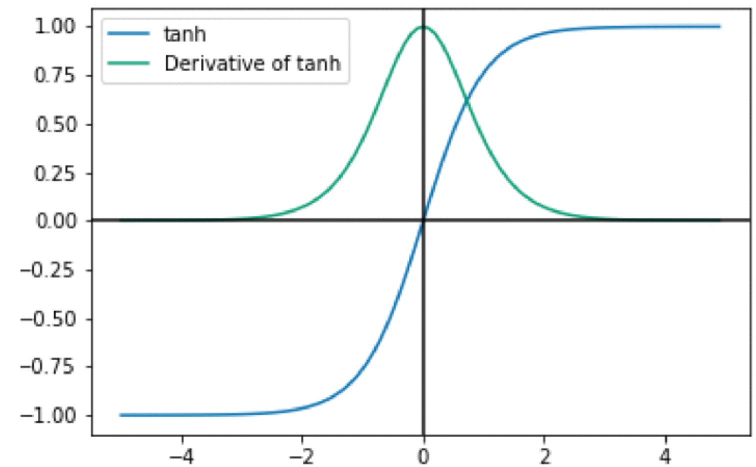
- Orthogonal weight initialization can increase performance of sigmoid network
- Pre-training
- Adding noise to activation function
- Hyperbolic tangent

Hyperbolic tangent

$$tanh(x) = \frac{sinh(x)}{cosh(x)} = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

$$tanh(x) = 2sigmoid(2x) - 1$$

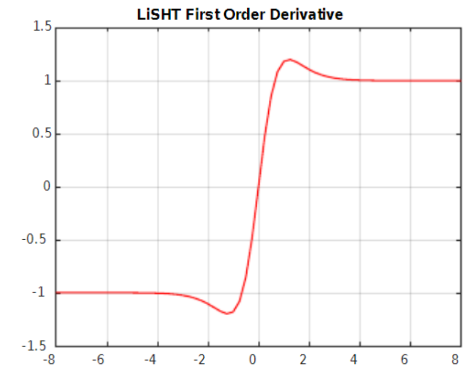
- Saturated
- Centered at zero
- Derivative is steeper
- Faster than sigmoid
- Lower error
- Still vanisher in deep networks

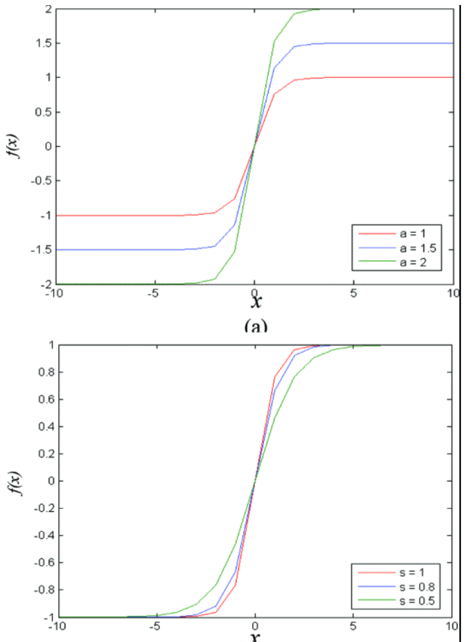


Improvements

- Linear scaling to tackle its gradient diminishing problems
- Linearly Scaled Hyperbolic Tangent (LiSHT)
- Adaptive hyperbolic tangent

$$f_8(x) = a\frac{\exp(2sx) - 1}{\exp(2sx) + 1}$$





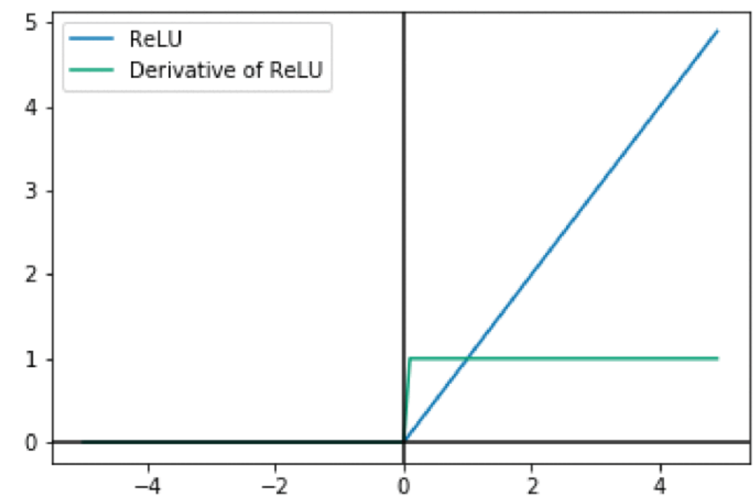
(a)
(b)

Aalto-yliopisto
Aalto-universitetet
Aalto University

Rectified linear units

- ReLU
- Most popular
- Simple and fast in training
- Not saturated
- Linear for positive values
- Zero for negative values
- "Dead neuron" or "dying ReLU"
- Bias shift

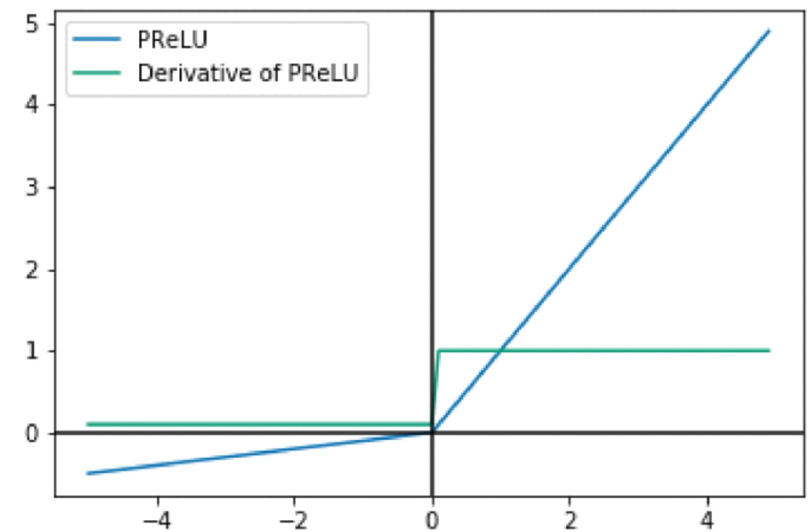
$$\mathrm{ReLU}(x) = \max(0,\, x)$$



Leaky ReLU

$$\mathrm{PReLU}(x) = \max(0,\ x) + \alpha * \min(0,\ x)$$

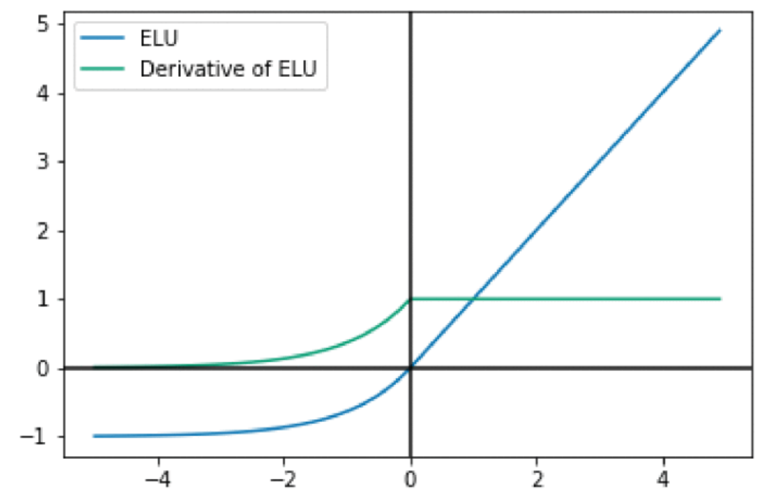
- **Leaky ReLU, LReLU**
- **Parametric leaky ReLU, PReLU, if α is learnable**
- **Attempt to fix "dying ReLU"**
- **Possible to perform back propagation**



Exponential Linear Unit

$$\mathrm{ELU}(x) = \max(0,\ x) + \min(0,\ \alpha(e^x - 1))$$

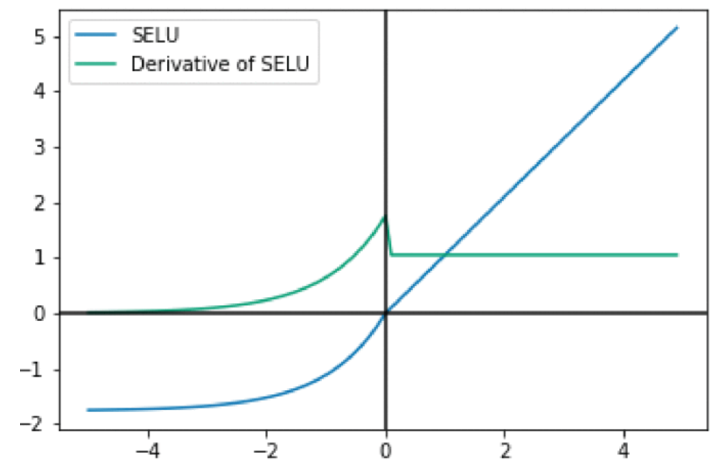
- ELU
- For negative values increases exponentially
- Same benefits as from Leaky ReLU
- Reduces bias shift problem, which is defined as the change of a neuron's mean value due to weights update



Scaled Exponential Linear Unit

$$\mathrm{SELU}(x) = \gamma * (\max(0, x) + \min(0, \alpha(e^x - 1)))$$

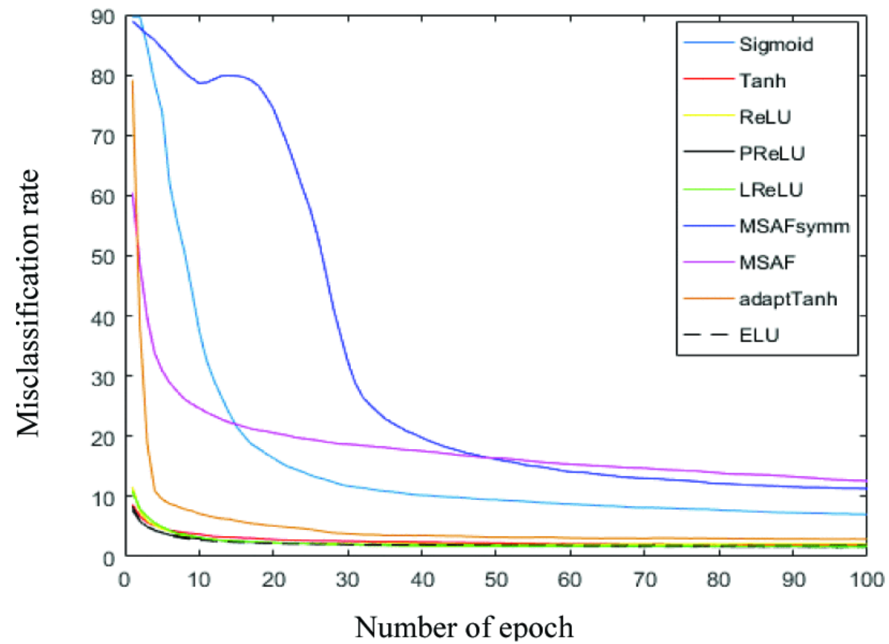
- **SELU**
- **Self-normalizing**
- **Converges towards zero mean and unit variance even under the presence of noise**



Comparison 1

- M. M. Lau and K. Hann Lim
- Four layers, feedforward
- Initialization for saturated activation functions with small random Gaussian weight initialization
- Unsaturated activation function, the weight initialization were using Xavier weight initialization
- Training: 60000 images
  Testing: 10000 images

Activation Functions	Misclassification rate	Pre-train
Sigmoid	7.01	Yes
Hyperbolic Tangent	1.86	Yes
MSAF	12.59	Yes
MSAF_symmetrical	11.28	Yes
ReLU	2.08	No
LReLU	1.68	No
PReLU	1.6	No
ELU	1.88	No
Adaptive tanh	2.93	No



Comparison 2

- B. Ding, H. Qian and J. Zhou
- Deep convolutional neural network
- Training: 60000 samples
  Testing: 10000 samples

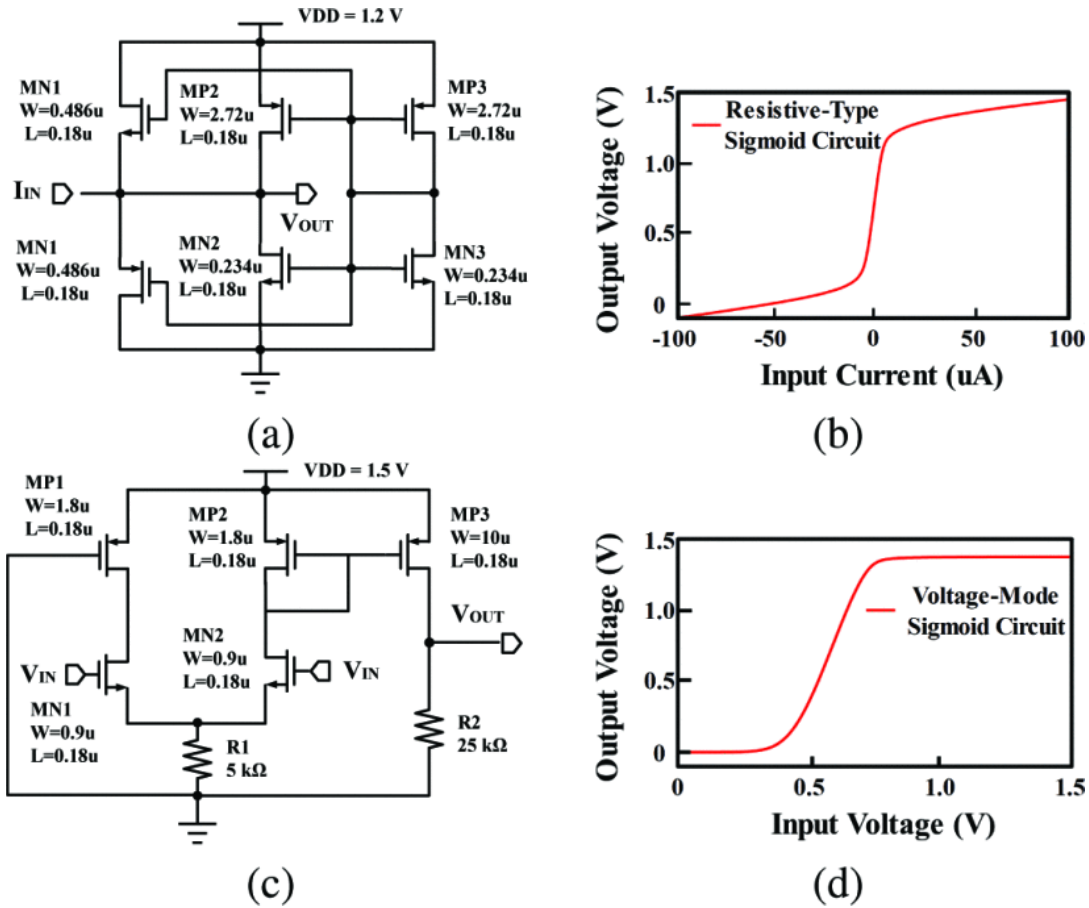
Activation function	Parameter	Error (%)
Sigmoid	-	1.15
Tanh	-	1.12
ReLU	-	0.8
RReLU	$a = 0.5$	0.99
ELU	$\alpha = 1$	1.1

Hardware

Hardware implementations

- **May be categorized into three approaches:**
    - Approximation
        - Taylor
        - Piecewise linear
        - Approximation of first derivative
    - Lookup Table (LUT) based
    - Hybrid Approaches

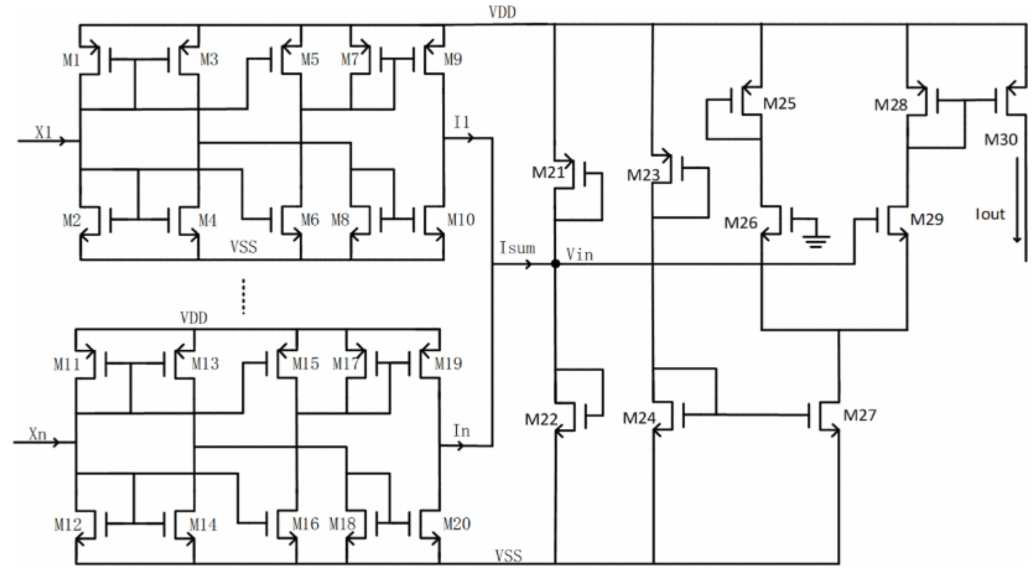
Sigmoid

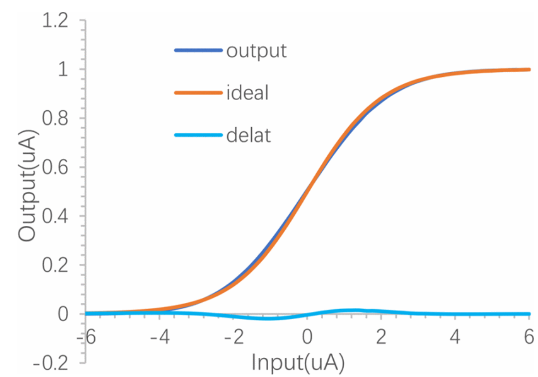


VDD = 1.2 V
MN1
W=0.486u
L=0.18u
MP2
W=2.72u
L=0.18u
MP3
W=2.72u
L=0.18u
IIN
VOUT
MN1
W=0.486u
L=0.18u
MN2
W=0.234u
L=0.18u
MN3
W=0.234u
L=0.18u
(a)
Resistive-Type
Sigmoid Circuit
Output Voltage (V)
1.5
1.0
0.5
0
-100
-50
0
50
100
Input Current (uA)
(b)
VDD = 1.5 V
MP1
W=1.8u
L=0.18u
MP2
W=1.8u
L=0.18u
MP3
W=10u
L=0.18u
VOUT
MN2
W=0.9u
L=0.18u
VIN
VIN
MN1
W=0.9u
L=0.18u
R1
5 kΩ
R2
25 kΩ
(c)
Voltage-Mode
Sigmoid Circuit
Output Voltage (V)
1.5
1.0
0.5
0
0
0.5
1.0
1.5
Input Voltage (V)
(d)

Sigmoid, high-precision

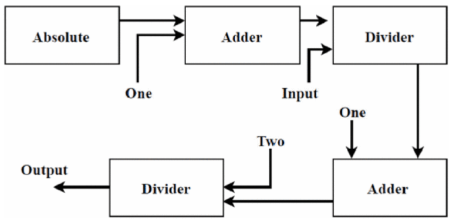
- M1-M10 for linear weight
- M21 and M22 I-V circuit
- M23, M24 and M27 current bias
- Rest is differential pairs

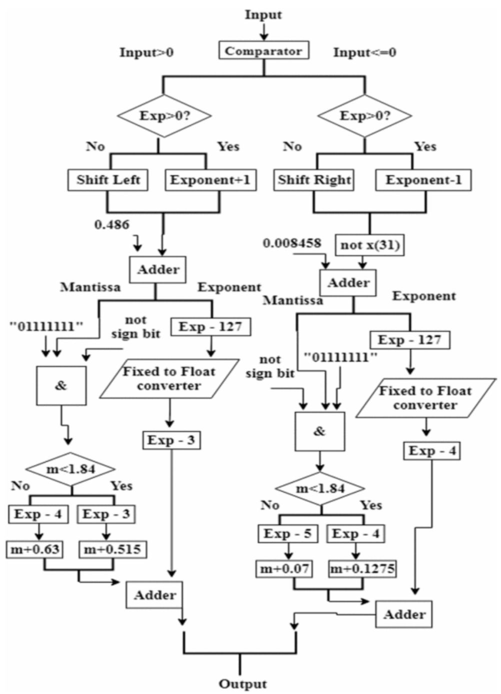


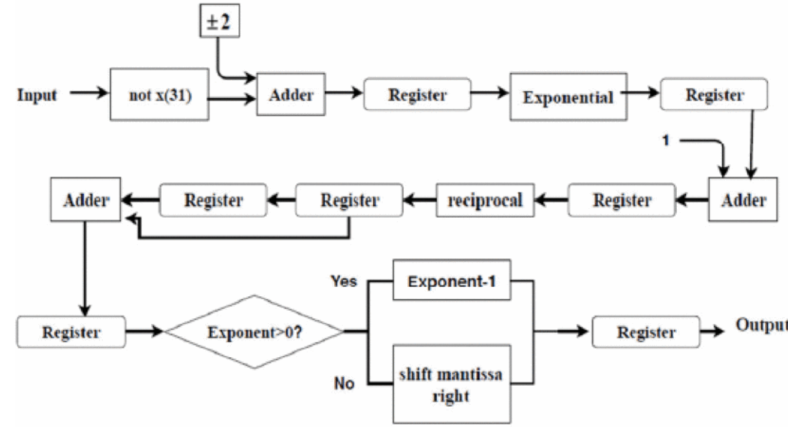


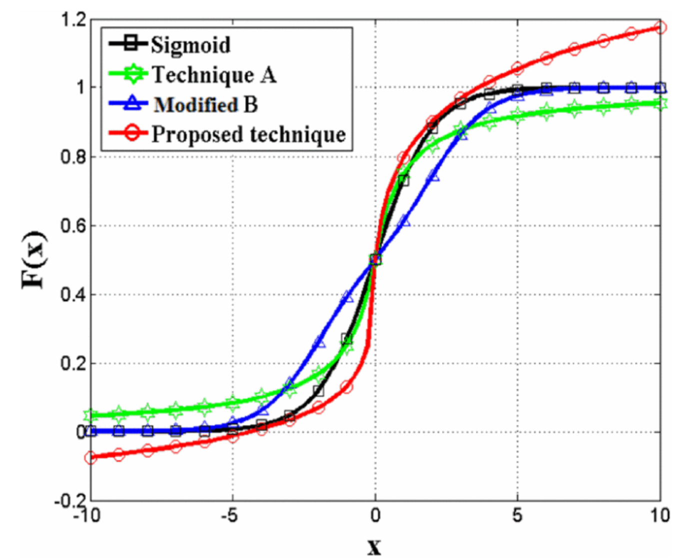
Aalto-yliopisto
Aalto-universitetet
Aalto University

Sigmoid, digital





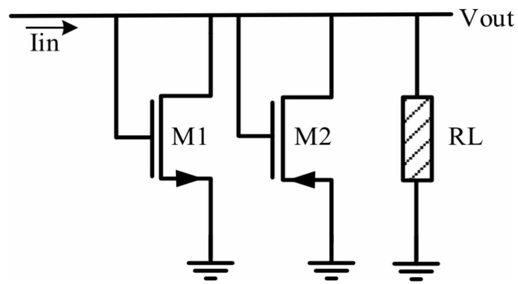




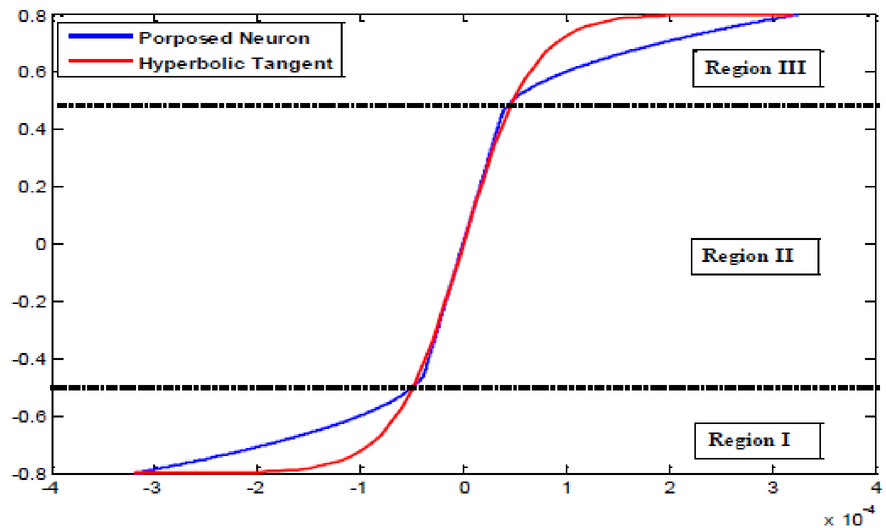
Aalto-yliopisto
Aalto-universitetet
Aalto University

Hyperbolic tangent

- **Passive resistive**
- **Max error 19.7%**
- **Average error 6.88%**

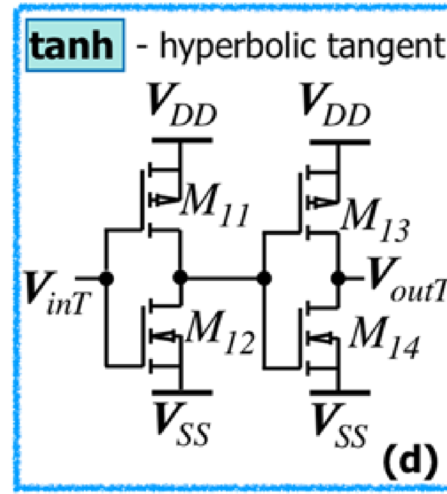


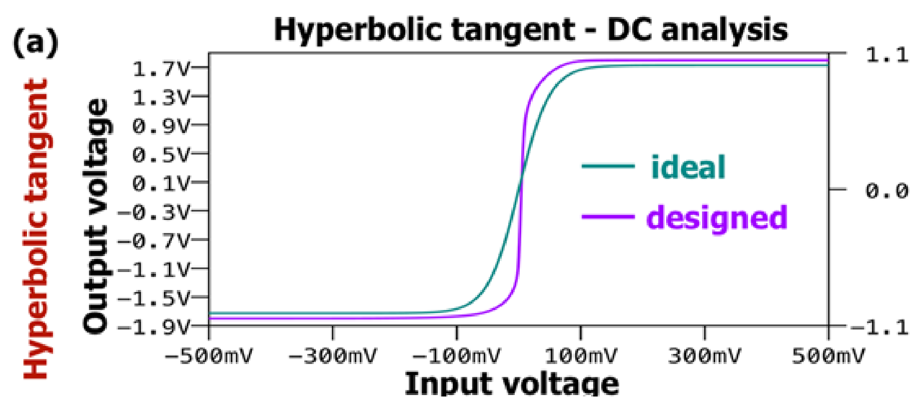
Region	V_{out}	M_1	M_2
I	$V_{out}<-V_{tp}$	OFF	Sat
II	$-V_{tp}<V_{out}<V_{tn}$	OFF	OFF
III	$V_{out}>V_{tn}$	Sat	OFF



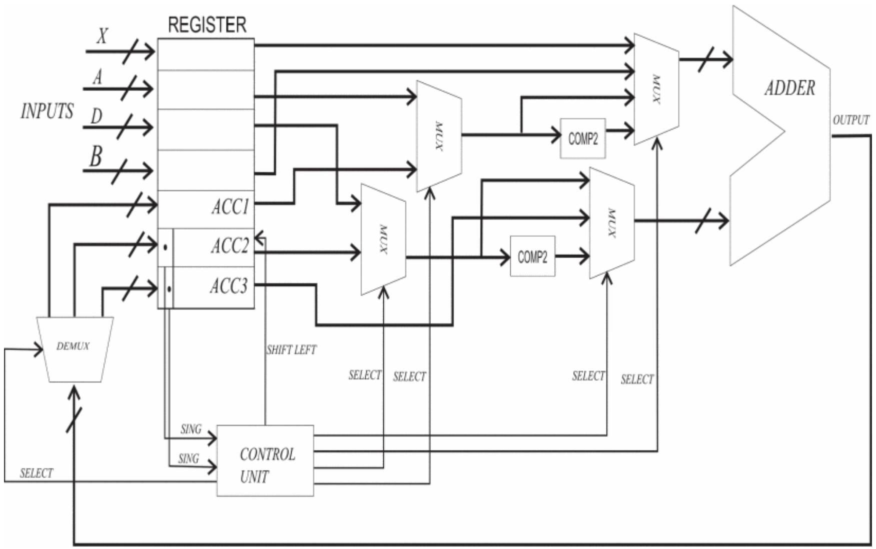
Hyperbolic tangent

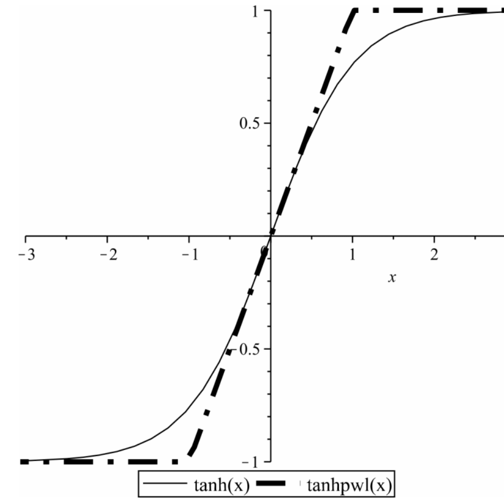
- "Hard" tanh
- Two adjusted inverters
- Small on-chip area and power consumption compared to other traditional tanh





Hyperbolic tangent, PWL

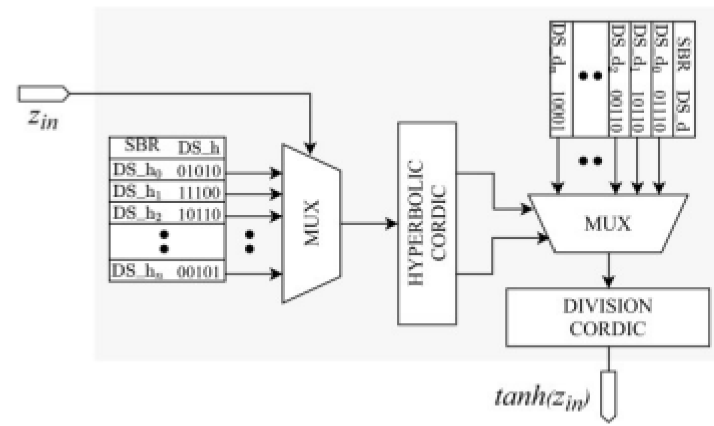




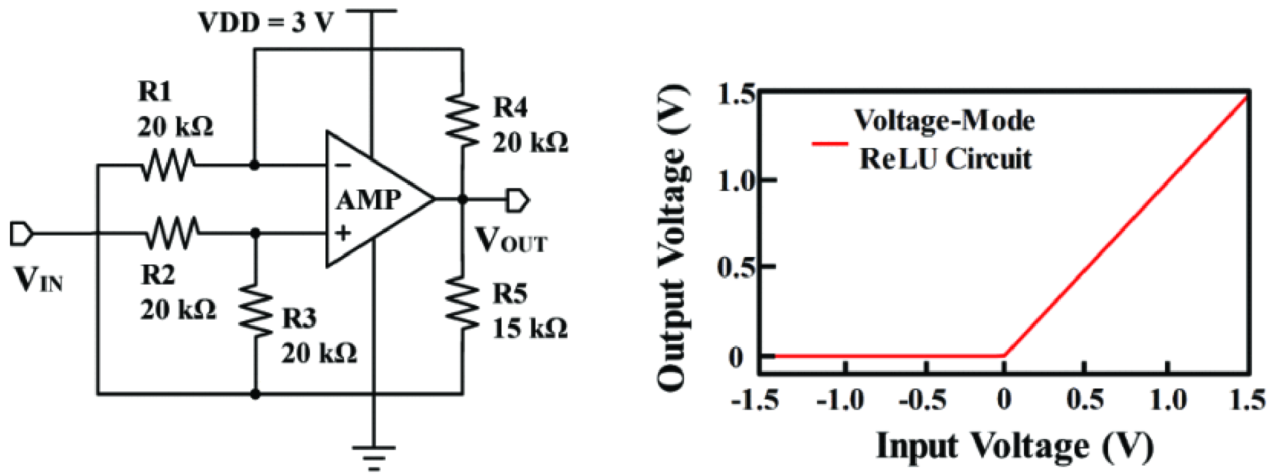
Format	I-bits	F-bits	Range	Max.error	Av.error
(2,6)	2	6	0,1.89750	0.53000	0.31623097
(3,5)	3	5	0,3.98675	0.238405844	0.08753644
(4,4)	4	4	0,7.93750	0.238405844	0.08649234

Hyperbolic tangent, CORDIC

- Coordinate Rotation DIgital Computer (CORDIC)
- CORDICs used for example in a transmitters



ReLU

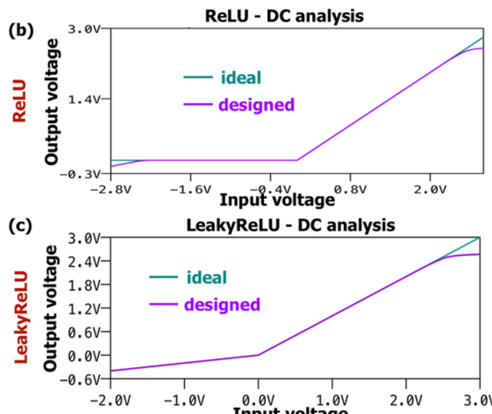


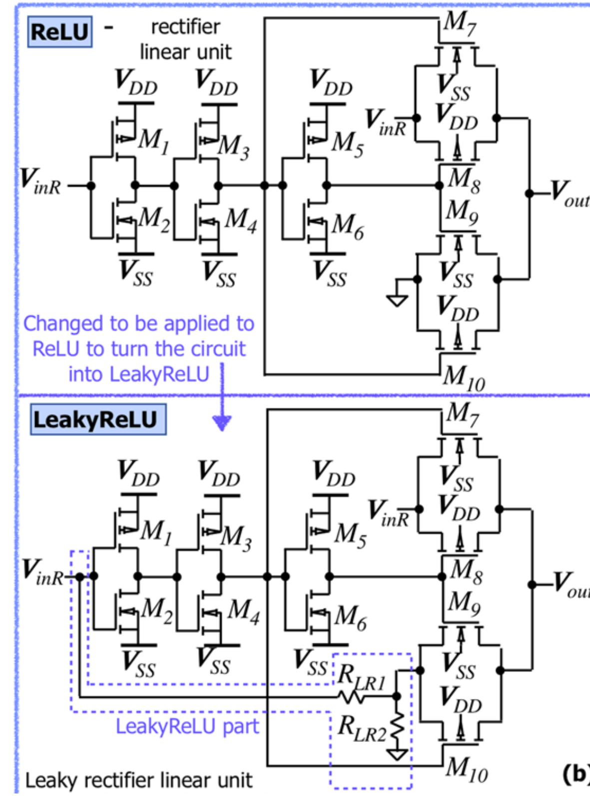
(a) (b)

- Voltage-mode
- Due to op amp, good linearity and operating range

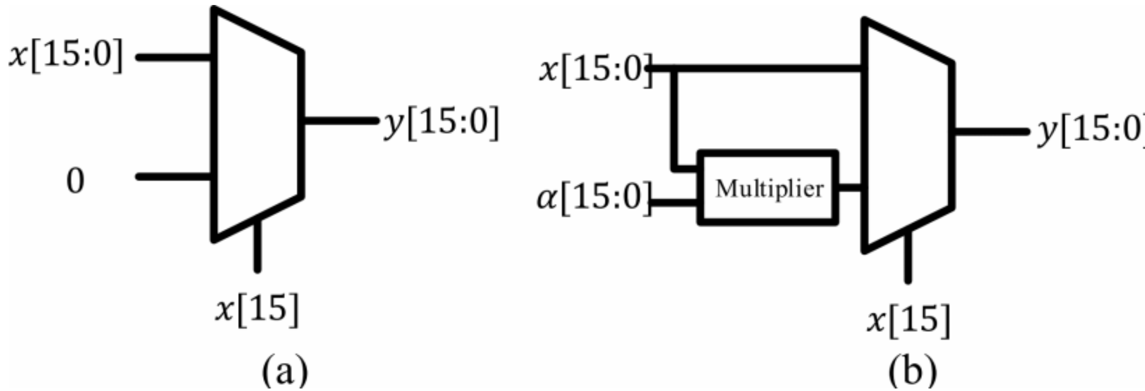
ReLU

- Based on transmission gate (M7-M8 and M9-M10)
- Inverters' threshold voltage is zero
- Adding voltage divider to "negative" transmission gate makes ReLU leaky





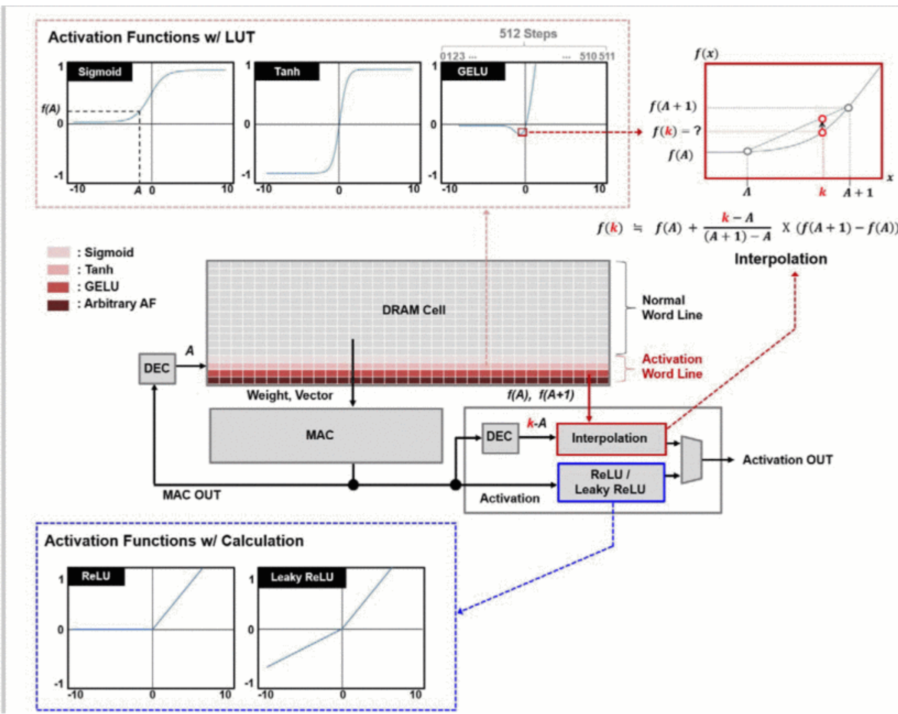
ReLU, digital



LUT

- LUT for every neuron present in the network
- Range addressable LUTs to reduce the LUT size
- A furthermore reduction in LUT is achieved by linearizing the activation function (Hybrid)
- Can have arbitrary activation function
- Simple, faster, and provide reasonable accuracy
- Only involves delay of one-memory access time to output the result, which is less than the usual computation time needed in arithmetic circuits

LUT

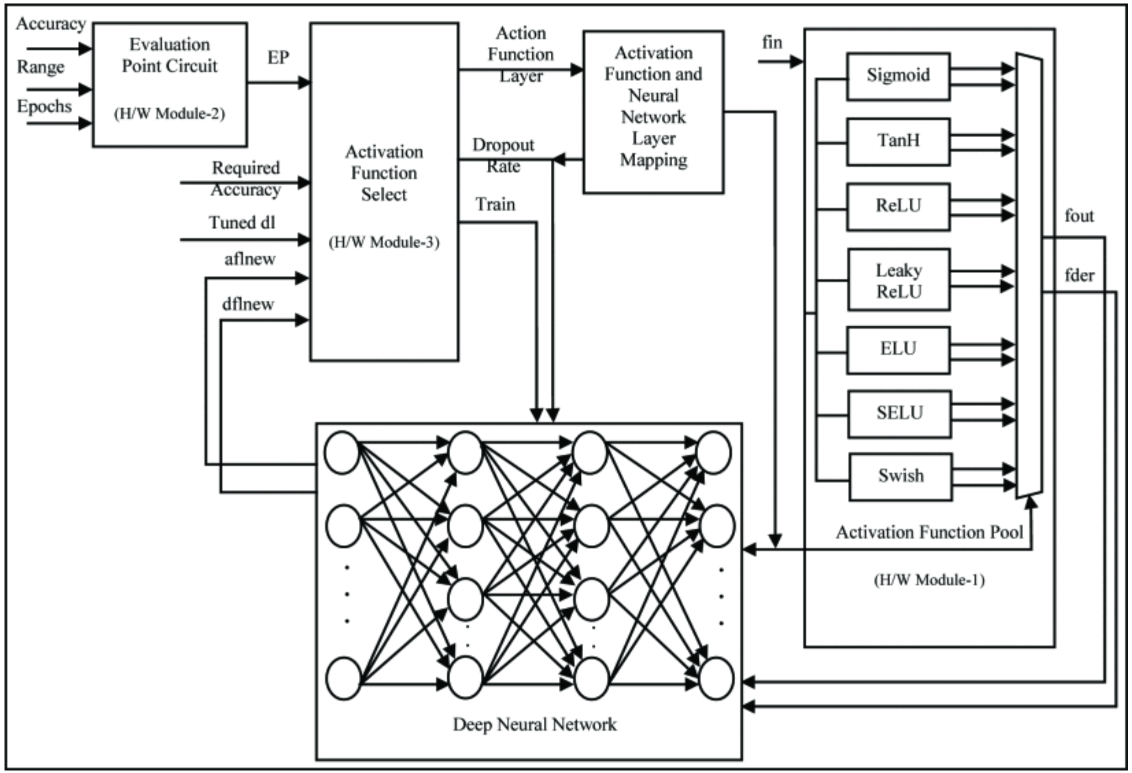


Comparison

	[1]	[2]	[3],[7]	This work
DRAM Type	LPDDR4	DDR4	HBM2	GDDR6
Process	20 nm	2x nm	20 nm	1y nm
Memory Density	8GB/chip (8H 8Gb mono die)	8GB/DIMM	6GB/cube (Buffer die + 4H 4Gb core-die with PCU + 4H 8Gb core-die)	8Gb/chip (4Gb DDP)
Data Rate	3.2Gbps	2.4Gbps	2.4Gbps	16Gbps
Bandwidth	25.6GB/s per chip	19.2GB/s per DIMM	307GB/s per cube	64GB/s per chip
# of Channel	1 per chip	16 per DIMM	8 per cube	2 per chip
# of Processing Unit (PU)	2048 per chip (256 per die)	128 per DIMM (8 per chip)	128 per cube (32 per core-die)	32 per chip (16 per die)
Processing Operation Speed	250MHz	500MHz	300MHz	1GHz
1 PU Throughput	2 GOPS (250MHz x 8byte)	4 GOPS (500MHz x 8byte)	9.6 GFLOPS (300MHz x 32byte)	32 GFLOPS (1GHz x 32byte)
Total Throughput (1 PU Throughput x # of PU)	0.5 TOPS per chip (2 GOPS x 256)	0.5 TOPS per DIMM (4 GOPS x 128)	1.2 TFLOPS per cube (9.6 GFLOPS x 128)	1 TFLOPS per chip (32 GFLOPS x 32)
Operation precision	INT8	INT8	FP16	BF16
Supported Activation Functions	-	-	ReLU	Sigmoid, Tanh, GELU, ReLU, Leaky ReLU, and Arbitrary AF

Aalto-yliopisto
Aalto-universitetet
Aalto University

LUT



Accuracy
Range
Epochs
Evaluation Point Circuit
(H/W Module-2)
EP
Activation Function Select
(H/W Module-3)
Required Accuracy
Tuned dl
aflnew
dflnew
Action Function Layer
Dropout Rate
Train
Activation Function and Neural Network Layer Mapping
fin
Sigmoid
TanH
ReLU
Leaky ReLU
ELU
SELU
Swish
fout
fder
Activation Function Pool
(H/W Module-1)
Deep Neural Network

References

- M. Kaloev and G. Krastev, "Comparative Analysis of Activation Functions Used in the Hidden Layers of Deep Neural Networks," *2021 3rd International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA)*, 2021, pp. 1-5, doi: 10.1109/HORA52670.2021.9461312.
- A. D. Rasamoelina, F. Adjailia and P. Sinčák, "A Review of Activation Function for Artificial Neural Network," 2020 IEEE 18th World Symposium on Applied Machine Intelligence and Informatics (SAMI), 2020, pp. 281-286, doi: 10.1109/SAMI48414.2020.9108717.
- M. M. Lau and K. Hann Lim, "Review of Adaptive Activation Function in Deep Neural Network," 2018 IEEE-EMBS Conference on Biomedical Engineering and Sciences (IECBES), 2018, pp. 686-690, doi: 10.1109/IECBES.2018.8626714.
- B. Ding, H. Qian and J. Zhou, "Activation functions and their characteristics in deep neural networks," 2018 Chinese Control And Decision Conference (CCDC), 2018, pp. 1836-1841, doi: 10.1109/CCDC.2018.8407425.
- R. P. Tripathi, M. Tiwari, A. Dhawan, A. Sharma and S. K. Jha, "A Survey on Efficient Realization of Activation Functions of Artificial Neural Network," 2021 Asian Conference on Innovation in Technology (ASIANCON), 2021, pp. 1-9, doi: 10.1109/ASIANCON51346.2021.9544754.
- S. Lee et al., "A 1ynm 1.25V 8Gb, 16Gb/s/pin GDDR6-based Accelerator-in-Memory supporting 1TFLOPS MAC Operation and Various Activation Functions for Deep-Learning Applications," 2022 IEEE International Solid-State Circuits Conference (ISSCC), 2022, pp. 1-3, doi: 10.1109/ISSCC42614.2022.9731711.
- Krestinskaya, O., Choubey, B. & James, A.P. Memristive GAN in Analog. Sci Rep 10, 5838 (2020). https://doi.org/10.1038/s41598-020-62676-7.
- S. Xing and C. Wu, "Implementation of A Neuron Using Sigmoid Activation Function with CMOS," 2020 IEEE 5th International Conference on Integrated Circuits and Microsystems (ICICM), 2020, pp. 201-204, doi: 10.1109/ICICM50929.2020.9292239.
- J. Shamsi, A. Amirsoleimani, S. Mirzakuchaki, A. Ahmade, S. Alirezaee and M. Ahmadi, "Hyperbolic tangent passive resistive-type neuron," 2015 IEEE International Symposium on Circuits and Systems (ISCAS), 2015, pp. 581-584, doi: 10.1109/ISCAS.2015.7168700.
- T. D. Nguyen, D. H. Kim, J. S. Yang and S. Y. Park, "High-Speed ASIC Implementation of Tanh Activation Function Based on the CORDIC Algorithm," 2021 36th International Technical Conference on Circuits/Systems, Computers and Communications (ITC-CSCC), 2021, pp. 1-3, doi: 10.1109/ITC-CSCC52171.2021.9501440.
- S. M. Waseem, A. Venkata Suraj and S. K. Roy, "Accelerating the Activation Function Selection for Hybrid Deep Neural Networks – FPGA Implementation," 2021 IEEE Region 10 Symposium (TENSYMP), 2021, pp. 1-7, doi: 10.1109/TENSYMP52854.2021.9551000.
- T. K. R. Arvind, M. Brand, C. Heidorn, S. Boppu, F. Hannig and J. Teich, "Hardware Implementation of Hyperbolic Tangent Activation Function for Floating Point Formats," 2020 24th International Symposium on VLSI Design and Test (VDAT), 2020, pp. 1-6, doi: 10.1109/VDAT50263.2020.9190305.
- B. Li, M. Yang and G. Shi, "Design of Analog CMOS-Memristive Neural Network Circuits for Pattern Recognition," 2021 IEEE 14th International Conference on ASIC (ASICON), 2021, pp. 1-4, doi: 10.1109/ASICON52560.2021.9620385.
- P. W. Zaki et al., "A Novel Sigmoid Function Approximation Suitable for Neural Networks on FPGA," 2019 15th International Computer Engineering Conference (ICENCO), 2019, pp. 95-99, doi: 10.1109/ICENCO48310.2019.9027479.
- Y. -H. Wu, W. -H. Lin and S. -H. Huang, "Low-Power Hardware Implementation for Parametric Rectified Linear Unit Function," 2020 IEEE International Conference on Consumer Electronics - Taiwan (ICCE-Taiwan), 2020, pp. 1-2, doi: 10.1109/ICCE-Taiwan49838.2020.9258135.
- R. A. Callejas-Molina, V. M. Jimenez-Fernandez and H. Vazquez-Leal, "Digital architecture to implement a piecewise-linear approximation for the hyperbolic tangent function," 2015 International Conference on Computing Systems and Telematics (ICCSAT), 2015, pp. 1-4, doi: 10.1109/ICCSAT.2015.7362925.

Assignment

For each the most used activation function(Sigmoid, tanh, ReLU), find on the Internet example of the application and why that function is chosen over others.