Using Approximate Computing to Improve the Efficiency of LSTM Neural Networks

Seyed Abolfazl Ghasemzadeh

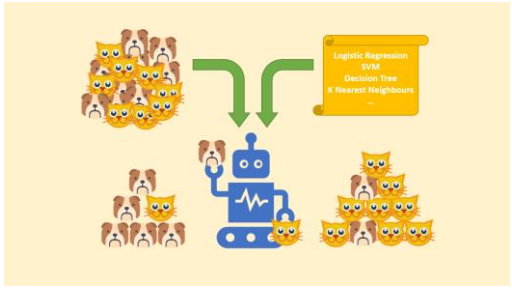
Who Am I?

- M.Sc. in Electrical Engineering – Electronics
- Fields of interest:
  - Approximate Computing in LSTM Neural Networks
  - Machine Learning
  - Internet of Things

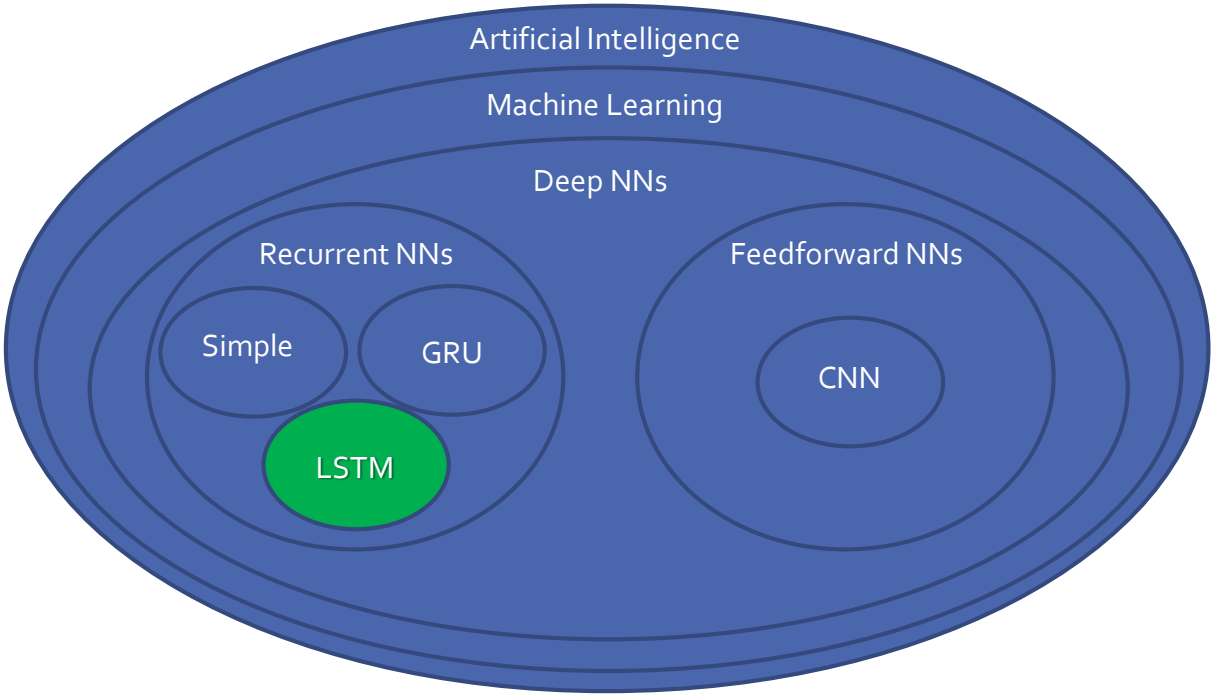
Contents

Neural Networks Applications

Neural Networks

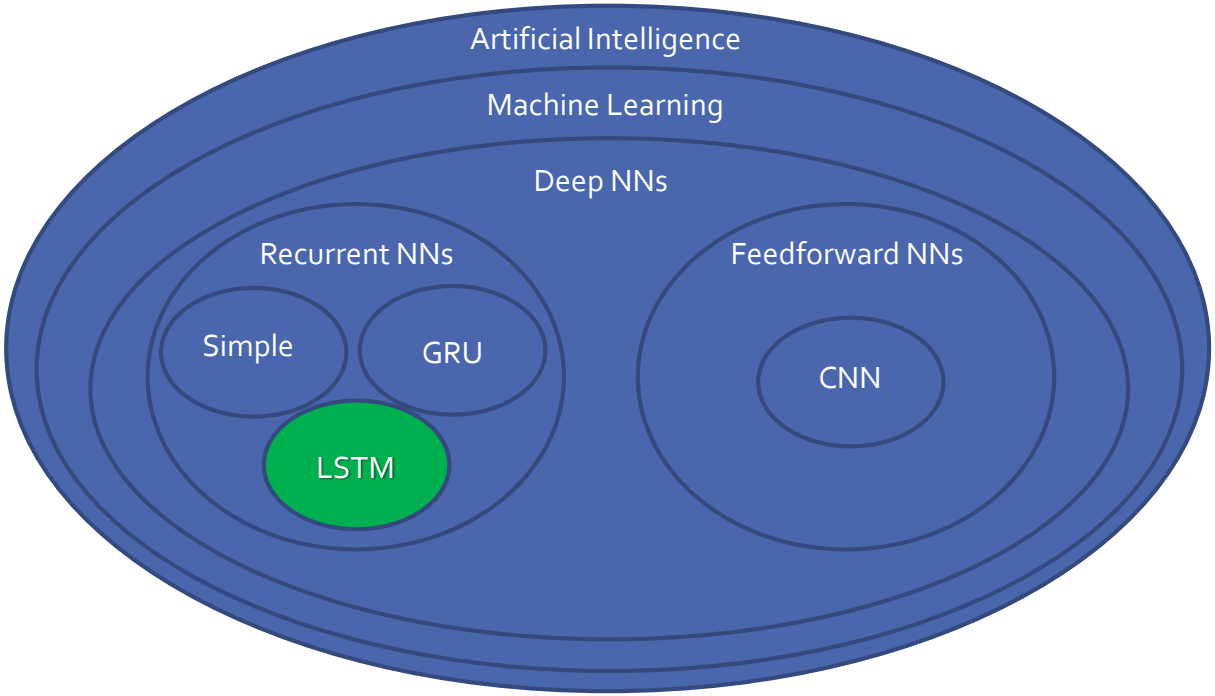


Neural Networks Hierarchy

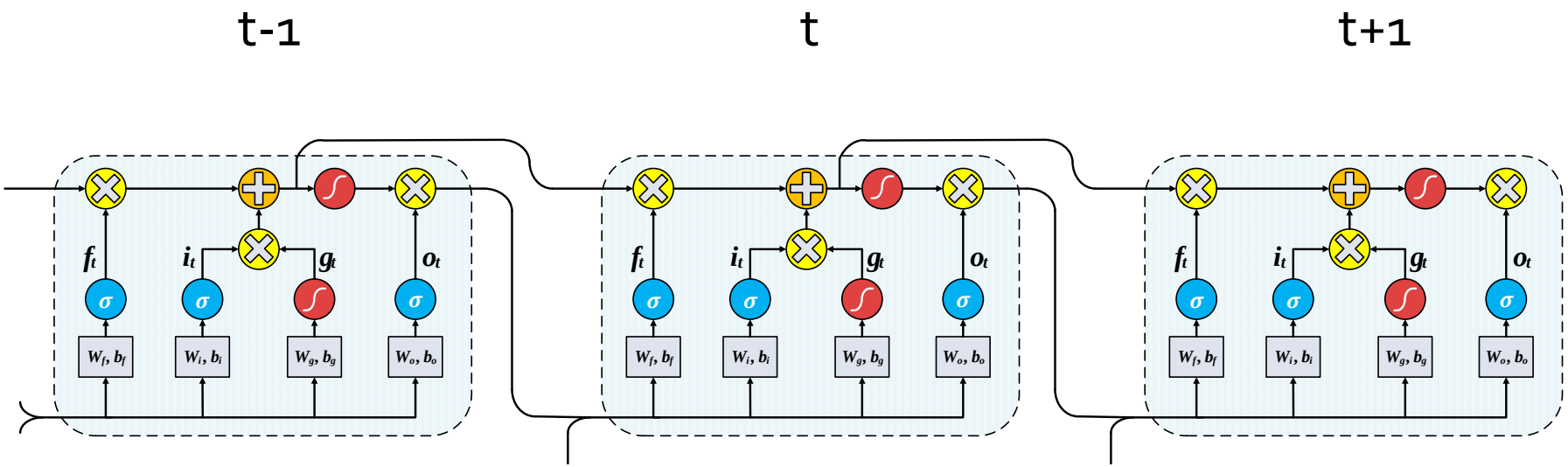


Neural Networks Hierarchy

- Natural Language Processing (NLP)
- Video Recognition
- Speech Recognition

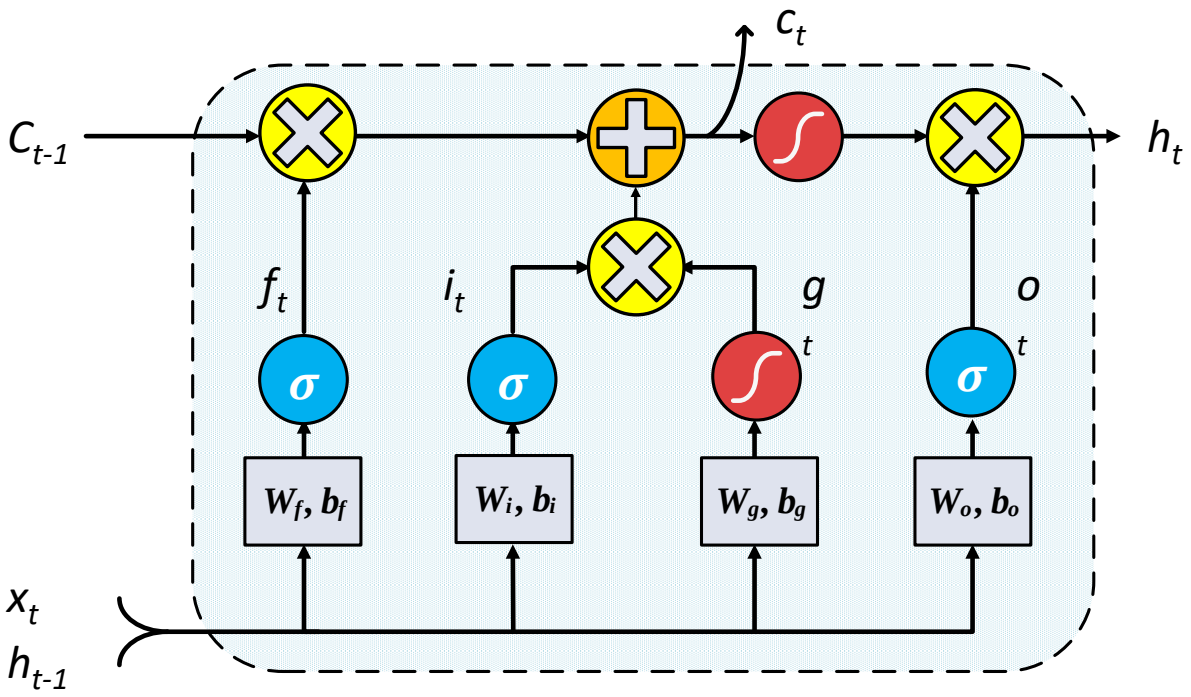


Long Short-Term Memory NNs



Long Short-Term Memory Cells

- ***W***: weight matrix
  b: bias vector
- ***f***: forget gate
  i: input gate
  g: candidate cell
  o: output gate
- ***x***: input vector
  h: output vector
  c: cell vector

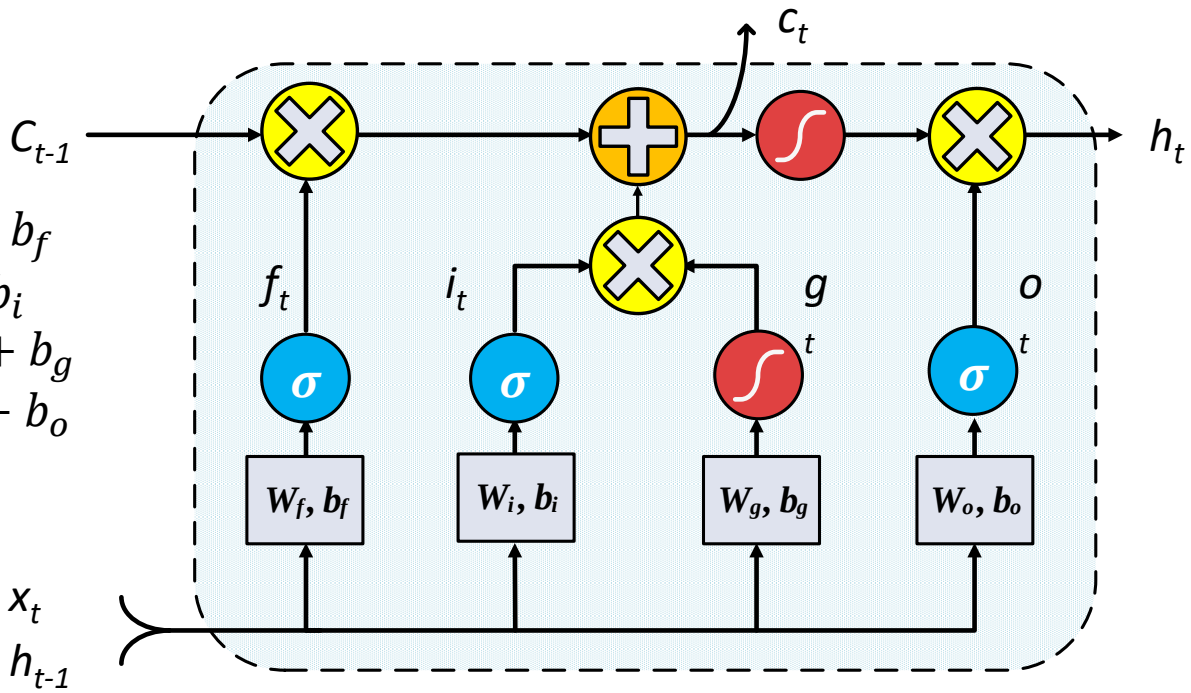


Long Short-Term Memory Cells (cont.)

(1)
$$f_t^* = W_{fx}x_t + W_{fh}h_{t-1} + b_f$$
$$i_t^* = W_{ix}x_t + W_{ih}h_{t-1} + b_i$$
$$g_t^* = W_{gx}x_t + W_{gh}h_{t-1} + b_g$$
$$o_t^* = W_{ox}x_t + W_{oh}h_{t-1} + b_o$$

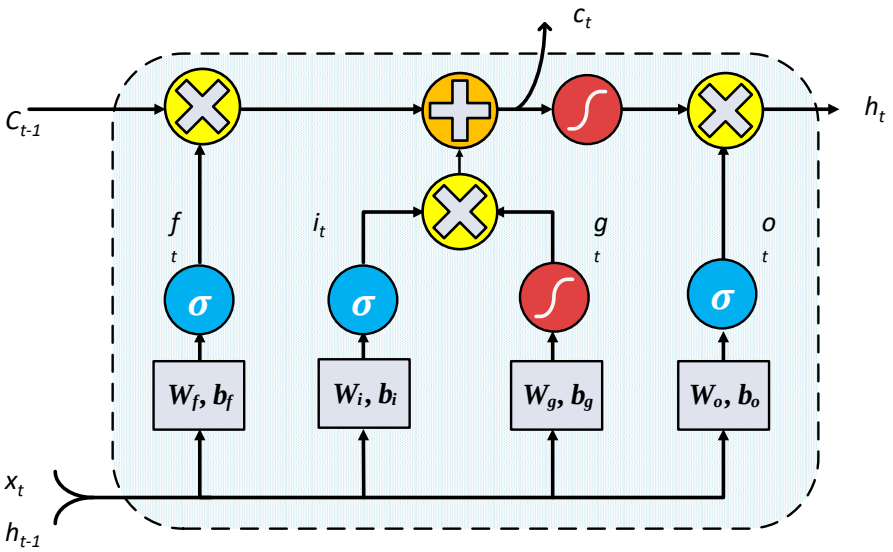
(2)
$$f_t = sig(f_t^*)$$
$$i_t = sig(i_t^*)$$
$$g_t = tanh(g_t^*)$$
$$o_t = sig(o_t^*)$$

(3)
$$c_t = f_t \odot c_{t-1} + i_t \odot g_t$$
$$h_t = o_t \odot tanh(c_t)$$

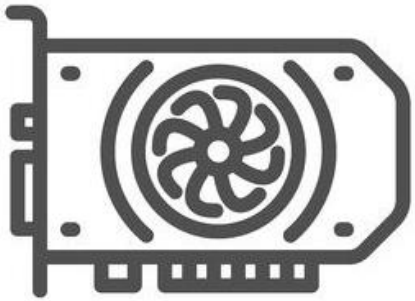


Long Short-Term Memory Cells (cont.)

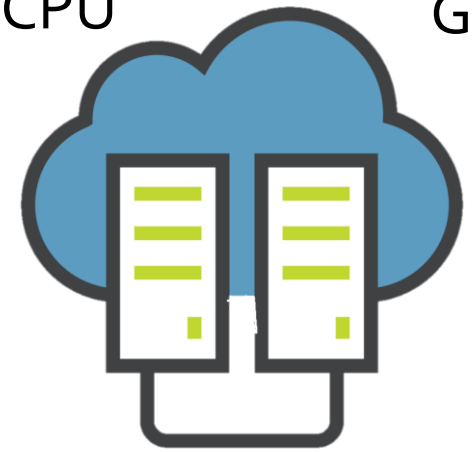
- Data Storage
- Matrix-Vector Multiplier
- Activation Function
- Point-wise Multiplier and Adder
- Timing Controller

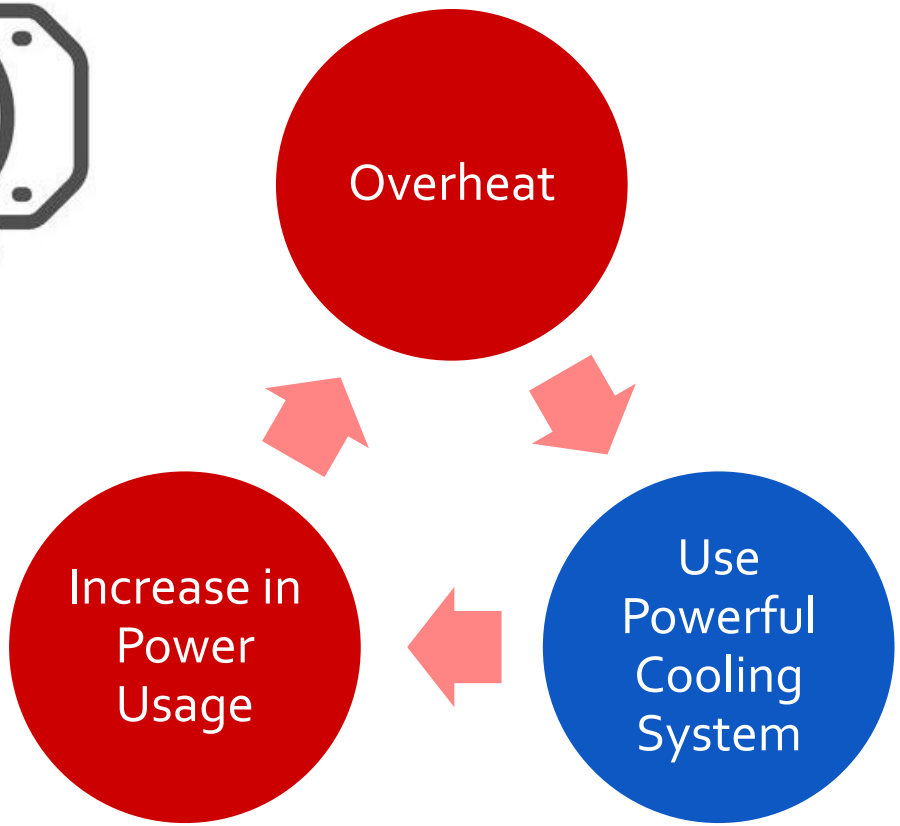


Hardware Realization



CPU GPU



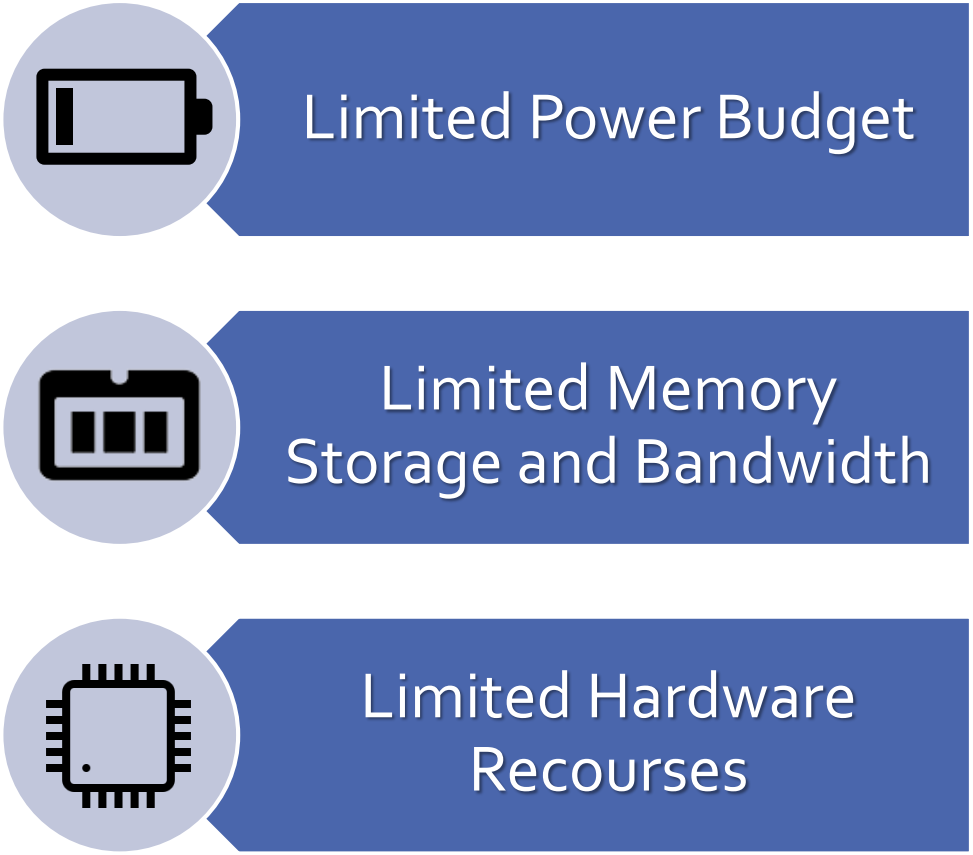


Hardware Realization

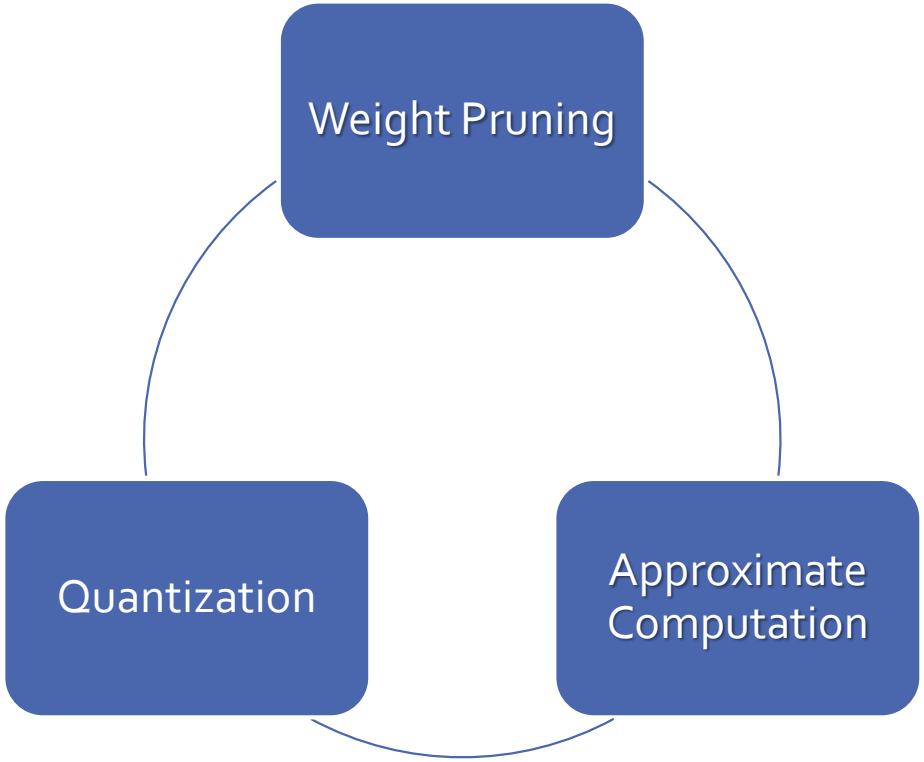
Hardware Accelerator

Decrease in Power Usage

Challenges in Design



Approximate Computing

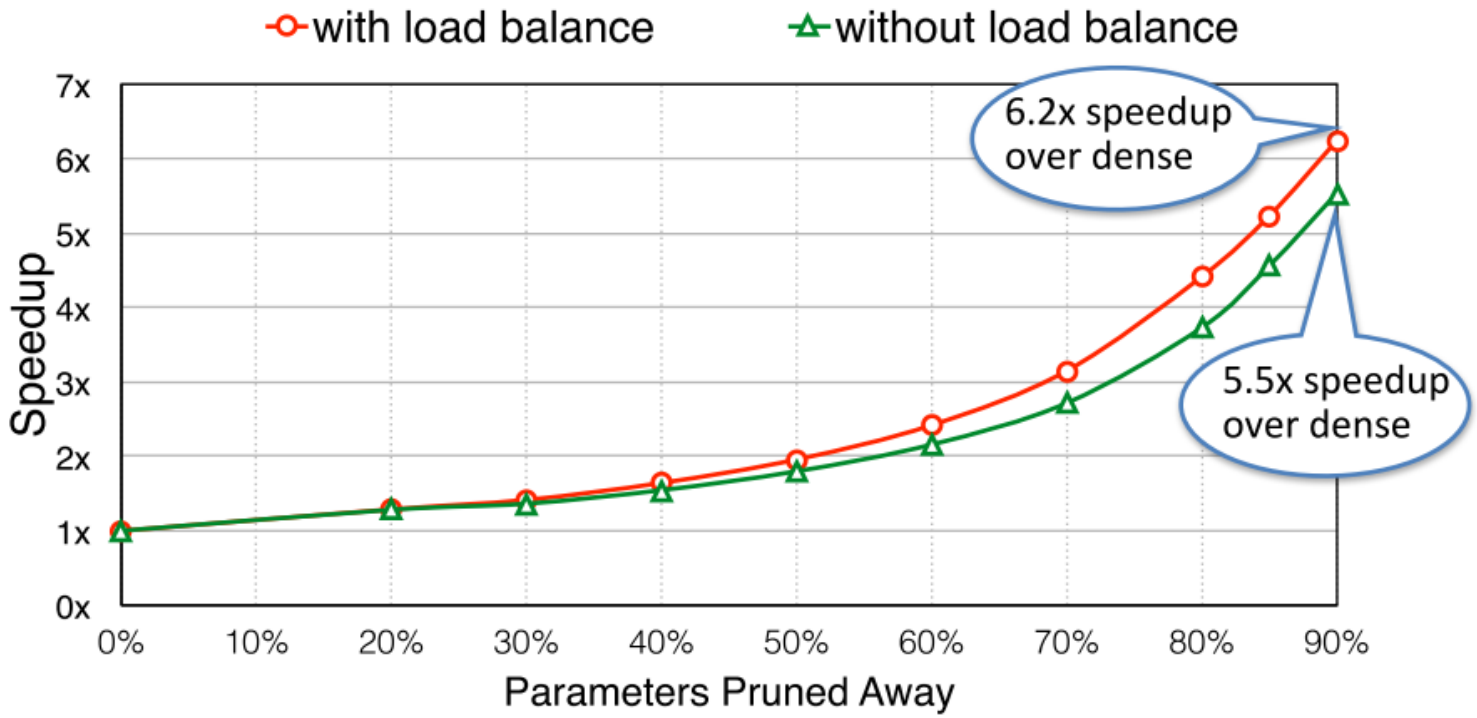


Approximate Computing Efficiency



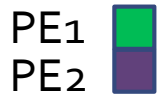

	DeltaRNN	ESE	FP-DNN	CPU	GPU
Hardware Model	XC7Z100	XCKU060	GSMD5	i7-8700K	GTX 1080 Ti
Quantization	Fixed16	Fixed12	Fixed16	Float32	Float32
Effective Throughput [G(FL)Op/s]	1198	**2520**	315.85	18.78	123.34
Power [W]	**7.3**	41	25	35.6	95.9
Power Efficiency [G(FL)Op/s/W]	**164.11**	61.46	12.63	0.53	1.29

Approximate Computing Efficiency (cont.)



[ESE - S. Han's ACM/SIGDA 2016]

Matrix Vector Multiplication



0.3	0.1	0.2	-0.1	0.4	-0.5	0.2	0.6
0.3	0.4	0.6	0.1	0.3	0.2	0.5	-0.5
0.7	0.8	-0.6	0.6	-0.2	0.5	0.3	0.1
0.2	-0.6	0.6	0.5	0.1	0.2	0.3	0.7

X

0.6
-0.6
0.1
0.7
0.9
-0.9
-0.8
0.1

Pruning Patterns

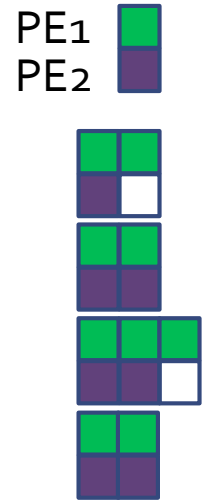
Unstructured Sparse

0.3	0.1	0.2	-0.1	0.4	-0.5	0.2	0.6
0.3	0.4	0.6	0.1	0.3	0.2	0.5	-0.5
0.7	0.8	-0.6	0.6	-0.2	0.5	0.3	0.1
0.2	-0.6	0.6	0.5	0.1	0.2	0.3	0.7

Pruning Patterns

Unstructured Sparse

								# of non-zeros
				0.4	-0.5		0.6	3
	0.4	0.6				0.5	-0.5	4
0.7	0.8	-0.6	0.6		0.5			5
	-0.6	0.6	0.5				0.7	4



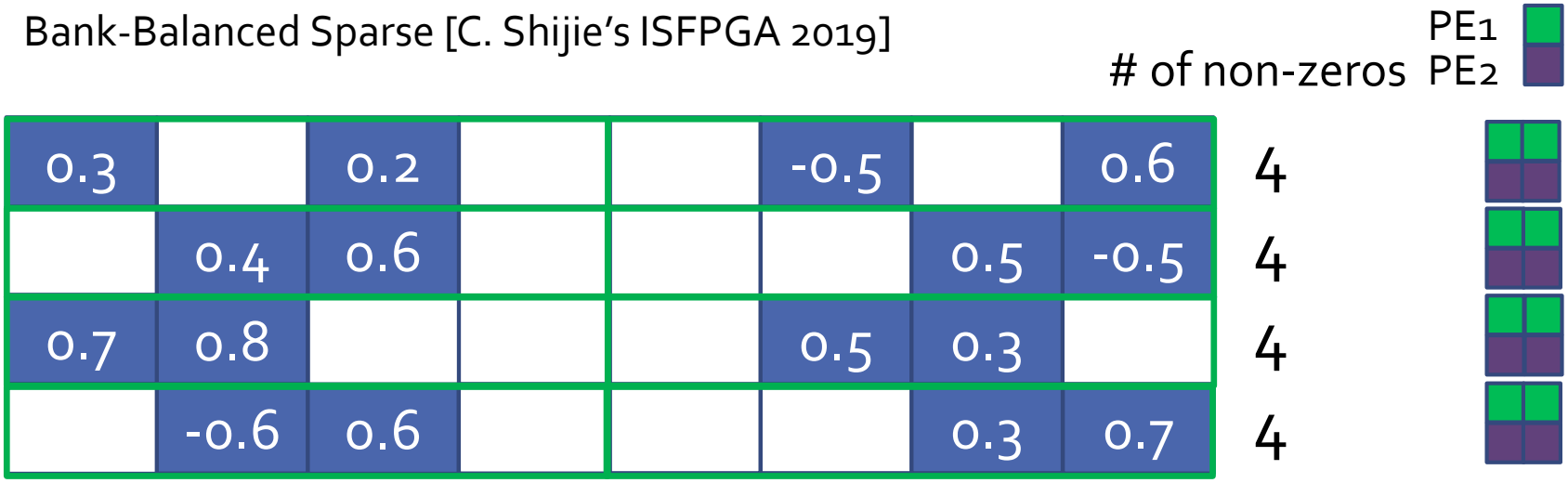
Pruning Patterns

Bank-Balanced Sparse [C. Shijie's ISFPGA 2019]

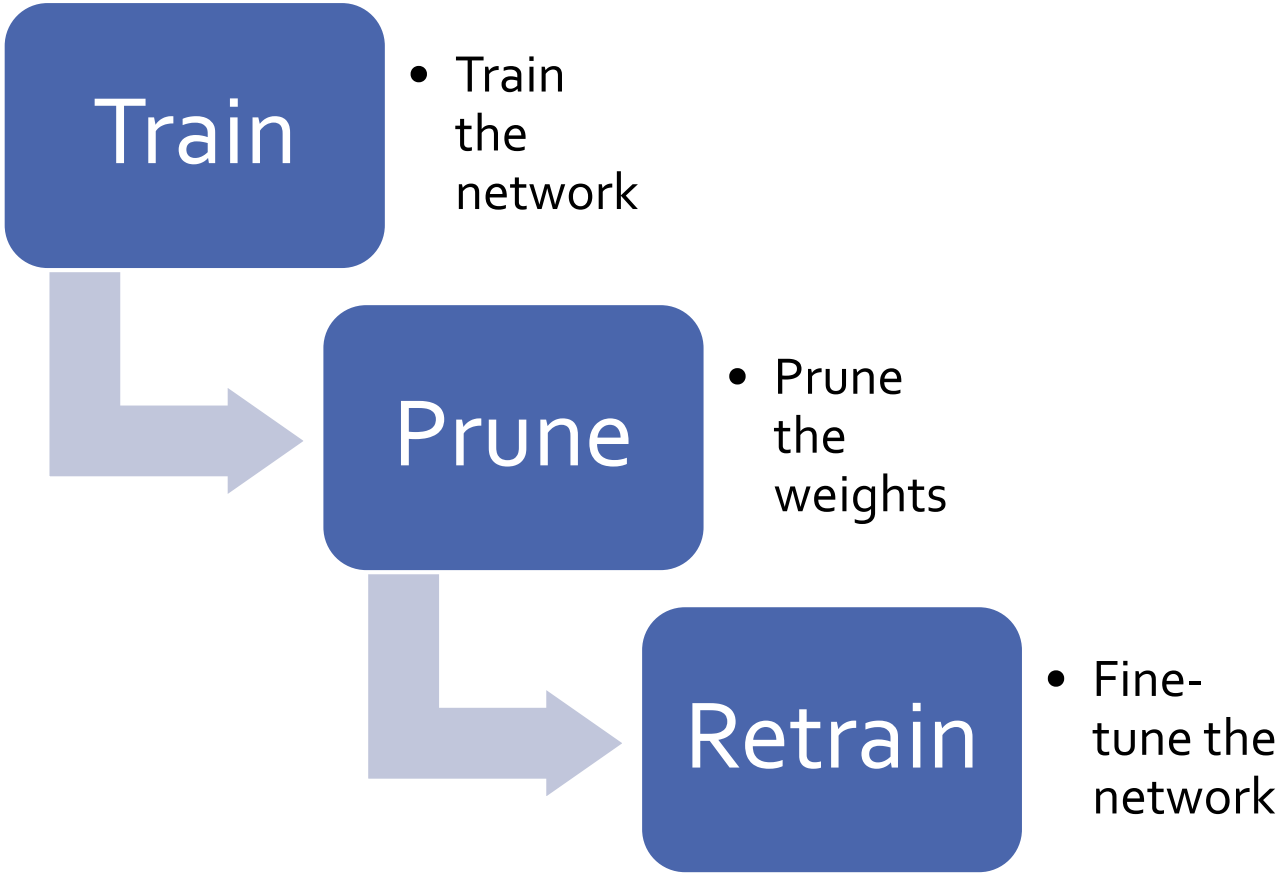
0.3	0.1	0.2	-0.1	0.4	-0.5	0.2	0.6
0.3	0.4	0.6	0.1	0.3	0.2	0.5	-0.5
0.7	0.8	-0.6	0.6	-0.2	0.5	0.3	0.1
0.2	-0.6	0.6	0.5	0.1	0.2	0.3	0.7

Pruning Patterns

Bank-Balanced Sparse [C. Shijie's ISFPGA 2019]

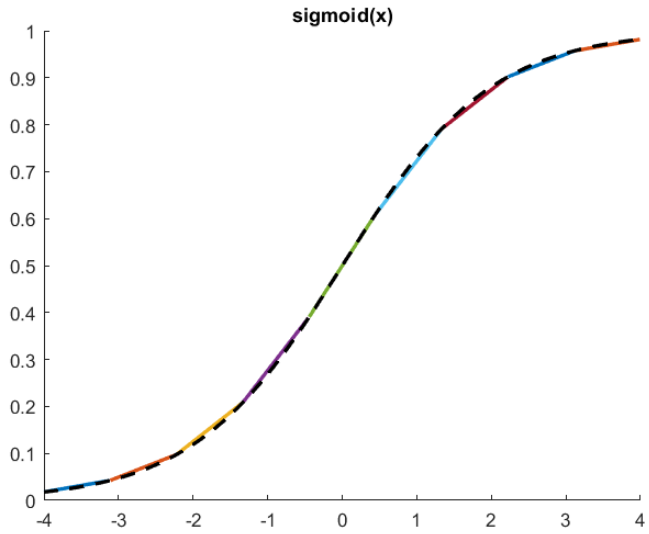


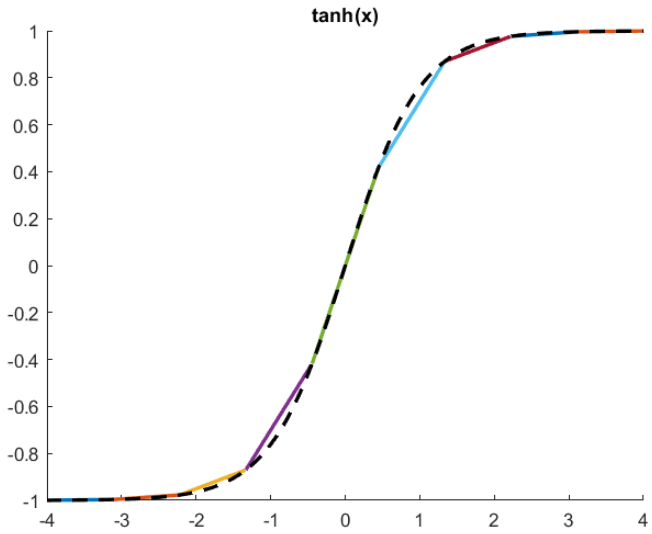
Pruning Algorithm



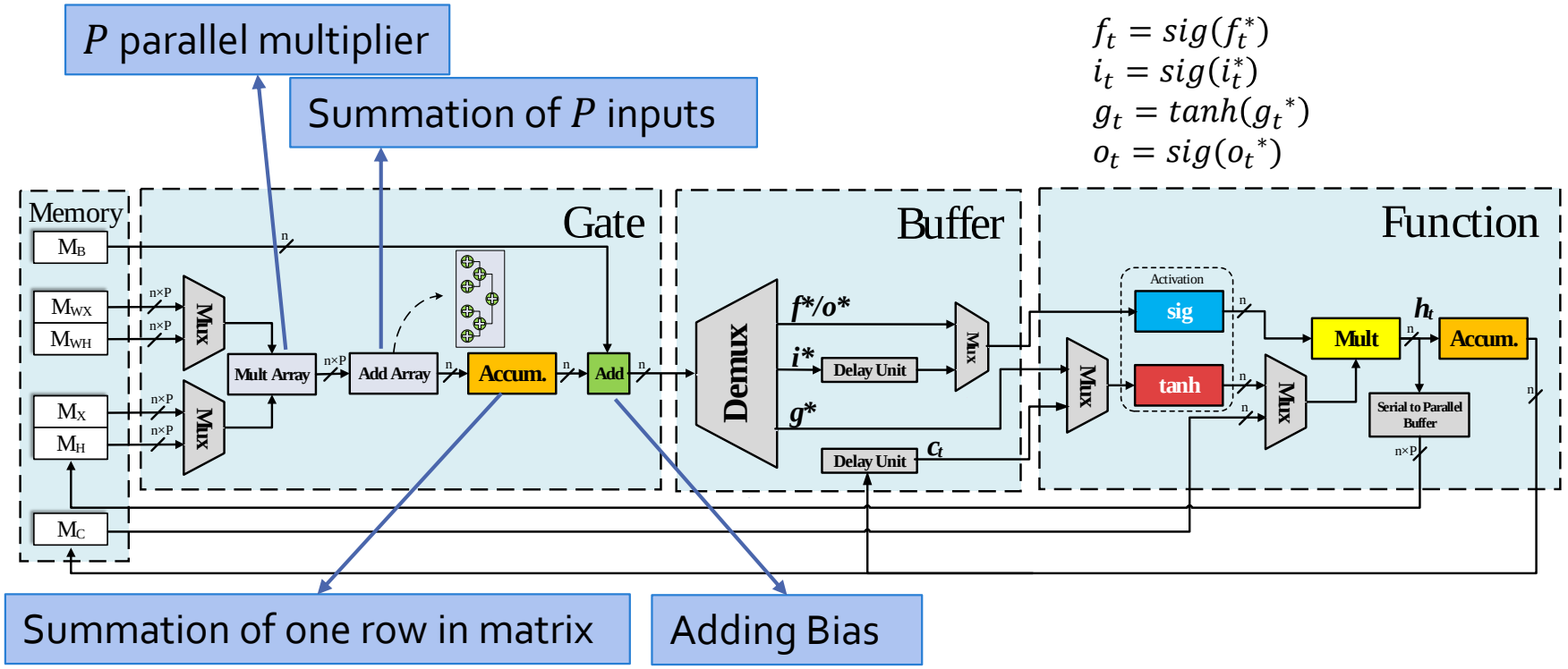
Activation Functions Approximation

- Linear $(ax + b)$
- Second-order $(ax^2 + bx + c)$
- Taylor Series





An LSTM Accelerator

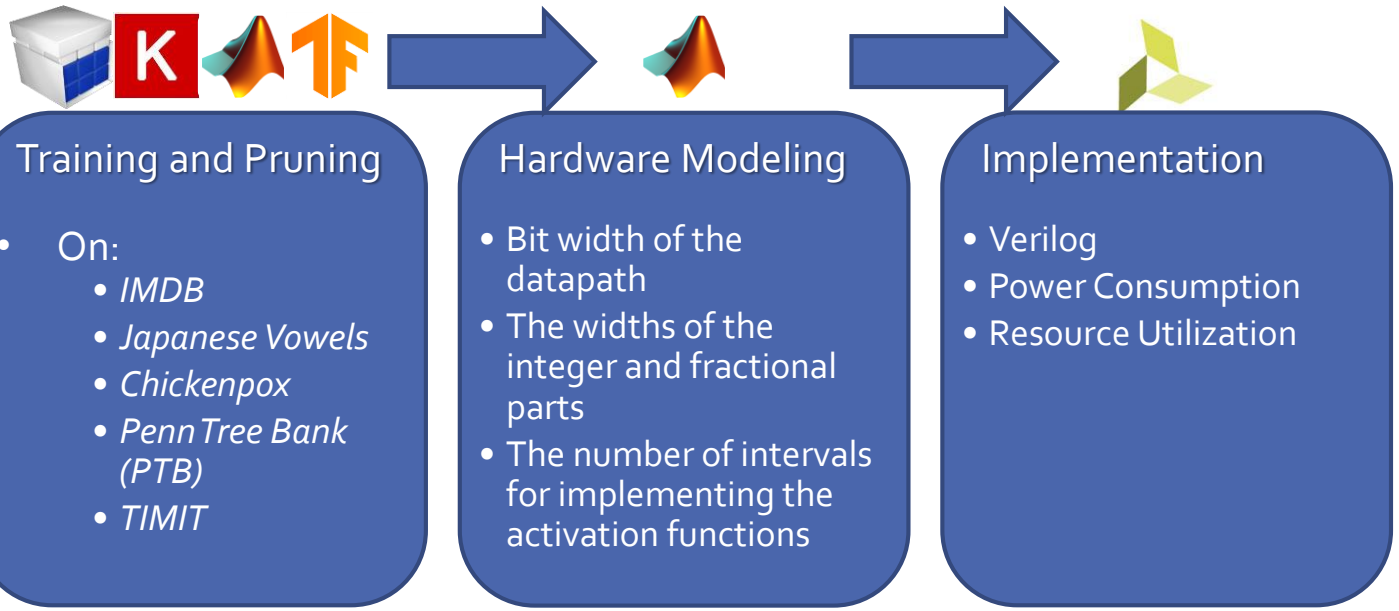


$$f_t = sig(f_t^*)$$
$$i_t = sig(i_t^*)$$
$$g_t = tanh({g_t}^*)$$
$$o_t = sig({o_t}^*)$$

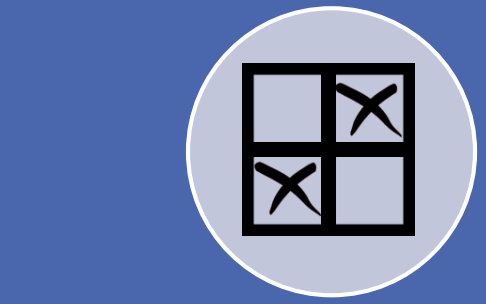
$$f_t^* = W_{fx}x_t + W_{fh}h_{t-1} + b_f$$
$$i_t^* = W_{ix}x_t + W_{ih}h_{t-1} + b_i$$
$${g_t}^* = W_{gx}x_t + W_{gh}h_{t-1} + b_g$$
$${o_t}^* = W_{ox}x_t + W_{oh}h_{t-1} + b_o$$

$$c_t = f_t \odot c_{t-1} + i_t \odot g_t$$
$$h_t = o_t \odot tanh(c_t)$$

Experimental Setup



Conclusion



Approximation

- To reduce the power consumption of accelerator

Hardware Accelerator

- For customizing the implementation of neural networks

References

- A. X. M. Chang, B. Martini, E. Culurciello, “Recurrent Neural Networks Hardware Implementation on FPGA,” CoRR, vol. abs/1511.05552, Mar. 2015.
- A. X. M. Chang, E. Culurciello, “Hardware Accelerators for Recurrent Neural Networks on FPGA,” *in Proceeding of the IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1-4, May 2017.
- J.C. Ferreira and J. Fonseca, “An FPGA Implementation of a Long Short-Term Memory Neural Network,” *in Proceeding of the IEEE International Conference on ReConFigurable Computing and FPGAs*, pp. 1– 8, 2017.
- Y. Guan, Z. Yuan, G. Sun, J. Cong, “FPGA-based Accelerator for Long Short-Term Memory Recurrent Neural Networks,” *in Proc. of Asia and South Pacific Design Automation Conference*, pp. 629-634, 2017.
- S. Han et al., “ESE: Efficient Speech Recognition Engine with Sparse LSTM on FPGA,” Dec. 2016, [online] Available: https://arxiv.org/abs/1612.00694.
- K. Chen, L. Huang, M. Li, X. Zeng, and Y. Fan, “A Compact and Configurable Long Short-Term Memory Neural Network Hardware Architecture,” *in Proc. of the 25th IEEE International Conference on Image Processing (ICIP)*, 2018.

References (cont.)

- S. Wang, Z. Li, C. Ding, B. Yuan, Q. Qiu, Y. Wang, and Y. Liang, “C-LSTM: Enabling Efficient LSTM using Structured Compression Techniques on FPGAs,” *in Proc. of the 2018 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, 11–20, 2018.
- S. Narang, E. Undersander, G. Diamos, "Block-Sparse Recurrent Neural Networks", CoRR, 2017.
- C. Shijie et al., "Efficient and effective sparse LSTM on FPGA with Bank-Balanced Sparsity", Proc. Int. Symp. Field-Program. Gate Arrays, pp. 63-72, 2019.
- Xilinx. 2017. Xilinx 7 Series DSP48E1 Slice User Guide. https://www.xilinx.com/support/documentation/user_guides/ug479_7Series_DSP48E1.pdf
- S. Han, X. Liu, H. Mao, J. Pu, A. Pedram, M. Horowitz, W. Dally, "Eie: Efficient inference engine on compressed deep neural network", *ISCA*, 2016.

THANKS FOR YOUR ATTENTION!

Any Questions?