Neuromorphic AI - An Automotive Application View of Event Based Processing

K. Knobloch, P. Gerhards
Infineon Development Center Automotive Electronics & AI
2022-06-29

Outline

- Assisted/autonomous driving and electric drive impact on automotive E/E-architecture
- Automotive µC and AI – concepts, what are the key applications
- Benefits expected from neuromorphic (spiking) neural networks
- Example: neuromorphic processing of radar data
- Summary

Impact of AI compute platform for autonomous driving on power?

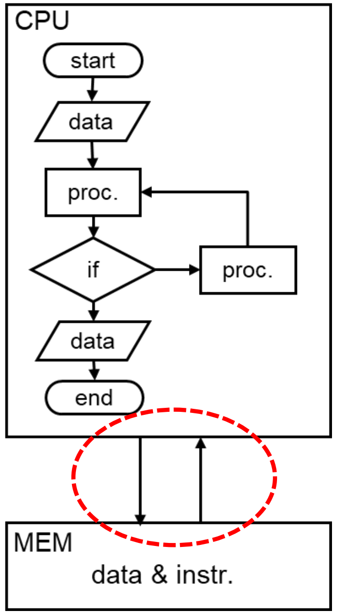
Power consumption autonomous driving

https://blogs.nvidia.com/blog/2020/05/14/drive-platform-nvidia-ampere-architecture/

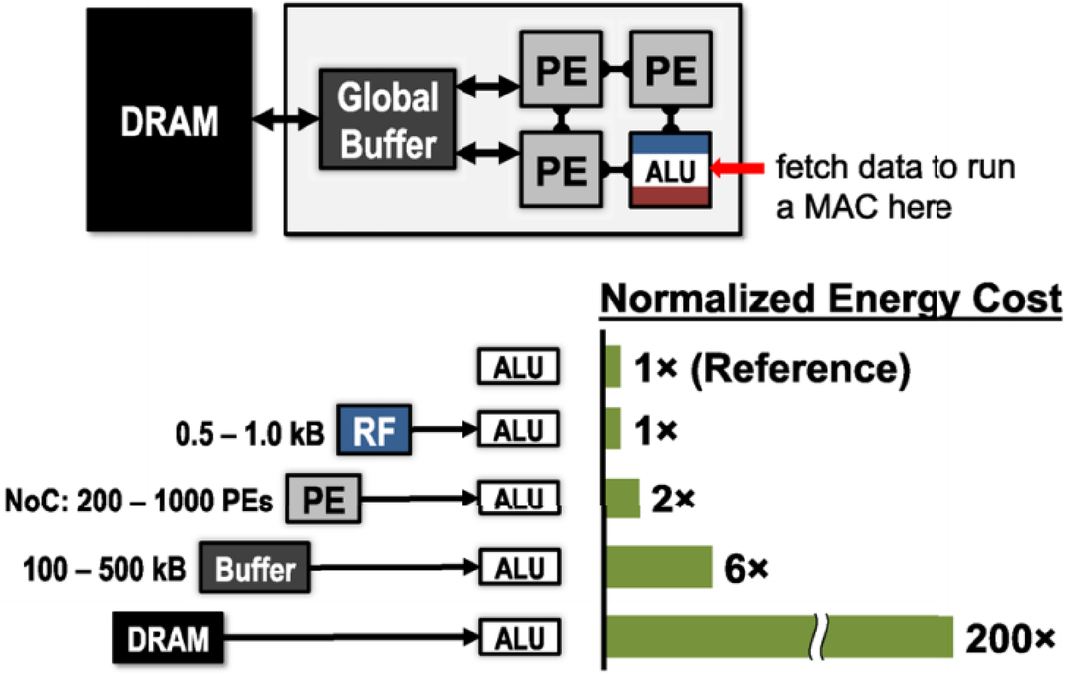
800W would add to e.g. 15kWh/100km (VW ID.4)

=> in fact ~10…30% of total power currently needed for L5 driving!

von Neumann

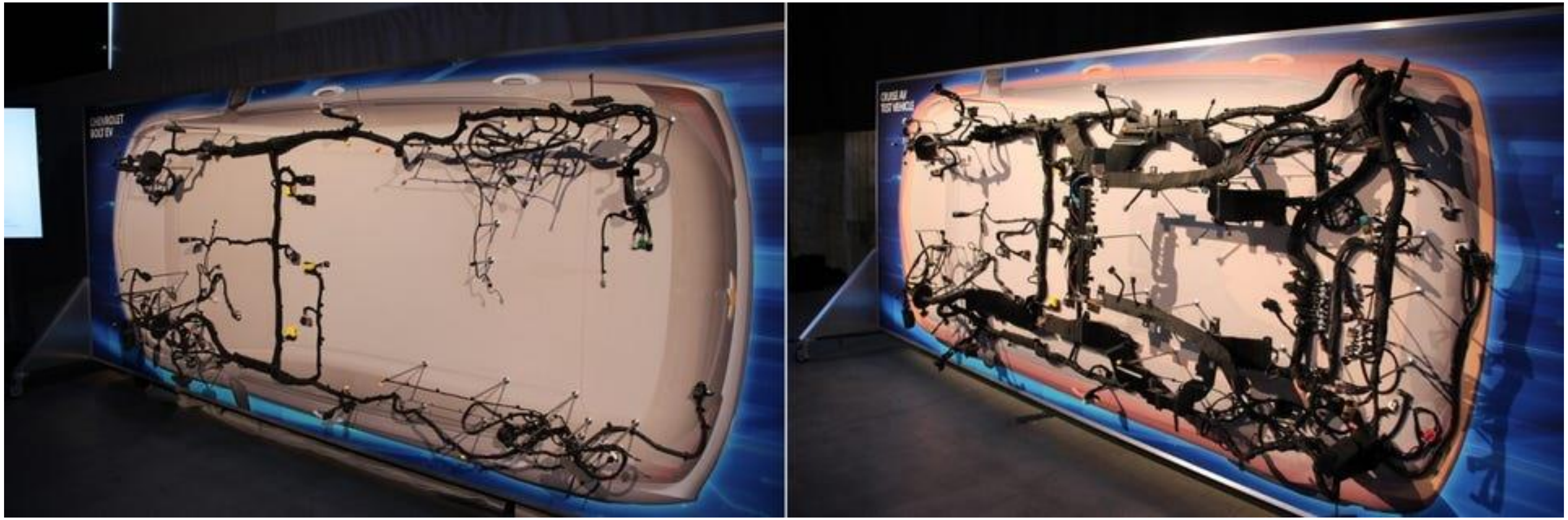


Power for memory access



Sze et al.: Efficient Processing of Deep Neural Networks: A Tutorial and Survey

Automotive trends provide severe challenge for E/E-architecture



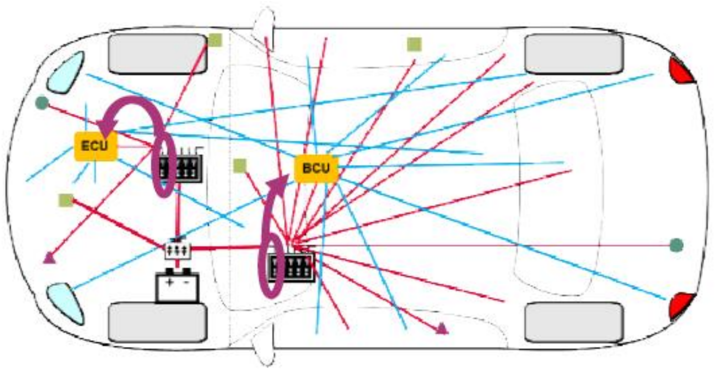
source: Forbes © 2018, Sam Abuelsamid

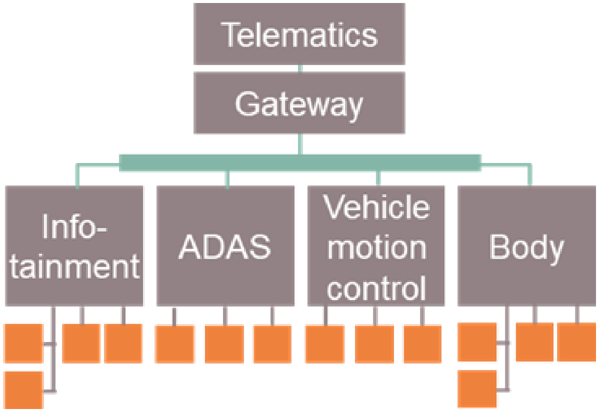
Wiring harnesses for the 2018 Chevy Bolt EV and the autonomous version

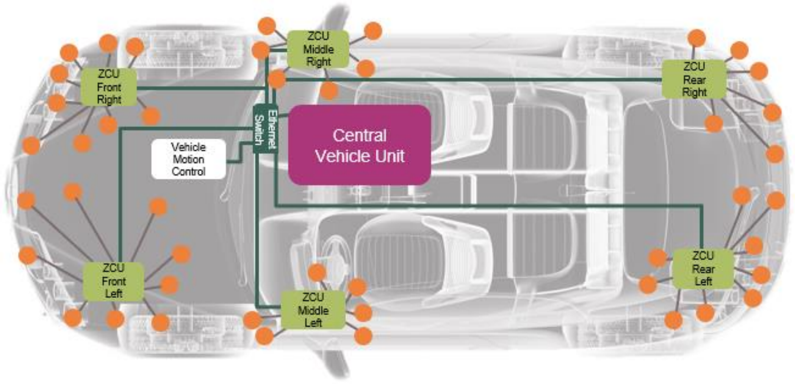
Autonomous driving requirements results in massive challenges for E/E-architecture – wiring/connections to be reduced!

E/E-Architecture needs to adopt on connectivity, e-mobility and autonomous driving

Distributed architecture → **Domain architecture** → **Zone architecture**

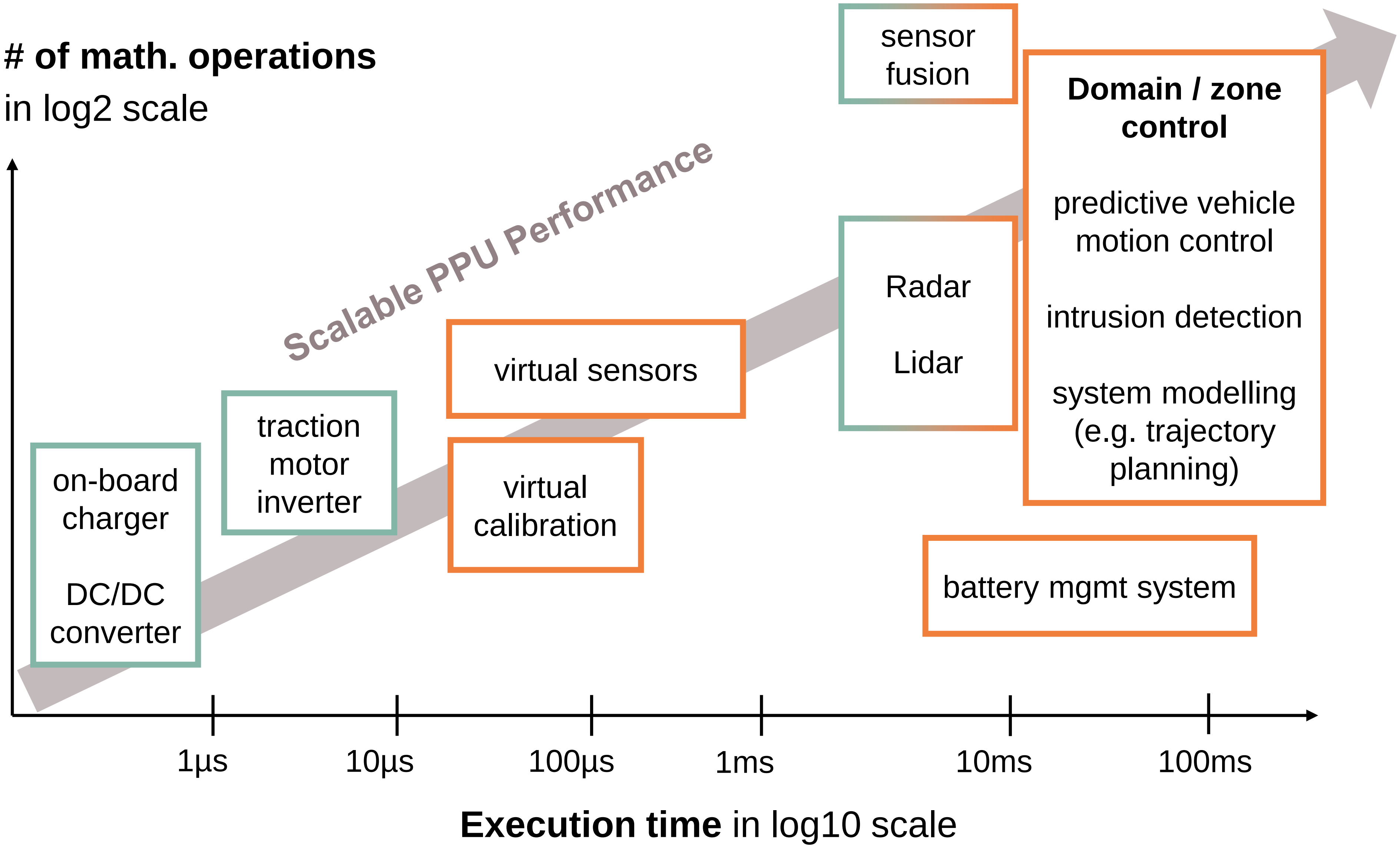






- Zonal E/E architectures enable complexity reduction in hardware (e.g. wiring) and software development
- Optimized mapping of required software functions and available hardware computing resources
- OEM objective: abstraction, scalable system (software) architecture across different vehicle types

Requirements for typical Automotive µC Application Tasks



Implemented tasks per applications

Complex data processing and observer based controlling of sensor actuator systems

Artificial neural network (MLP, RBF, RNN, CNN) based system modelling and object classification

In electrified vehicles AI can show great benefits in virtual sensor or system modelling use cases

Sensorless Induction Motor Drive

- Challenge: mismatching actual and estimated rotor flux limiting dynamic performance
- Rotor flux estimation influenced by rotor resistance (heating)
- Target: better resistance estimation

RNN MLP

Vehicle Motion Control

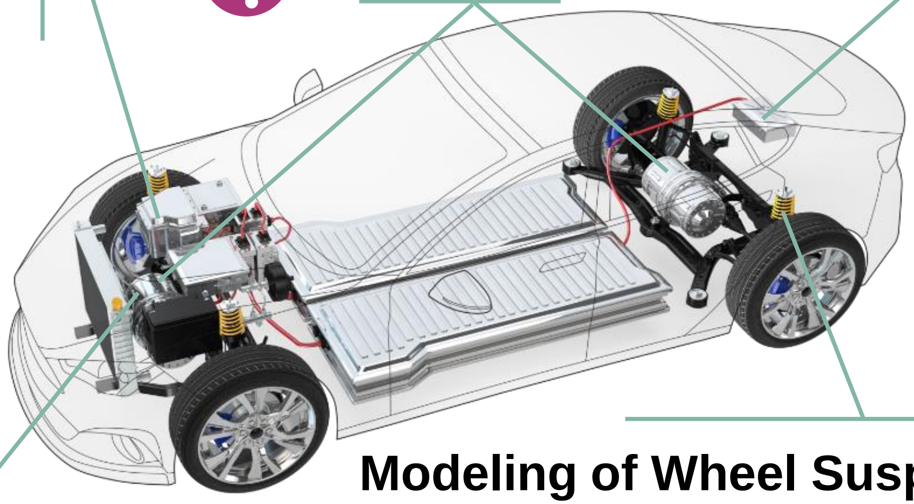
- Challenge: high number of variables for dynamics optimization
- Target: better dynamics

MLP

SoC & SoH Estimation

- Challenge: estimation of strong non linear electrochemical reactions
- Target: use known values in non-linear models: voltage, current, temperature

LSTM RNN MLP



Fault Diagnosis

- Challenge: additional sensor for vibration analysis of bearings needed (up to 50% of all faults)
- Target: Use stator current for diagnosis

LSTM

Modeling of Wheel Suspensions

- Challenge: Accurate predictions of the vehicle motion behavior and adapt it to the wishes of the targeted market segment
- Target: Modelling of wheel carrier acceleration and spring /damper force considering maneuvers and road unevenness

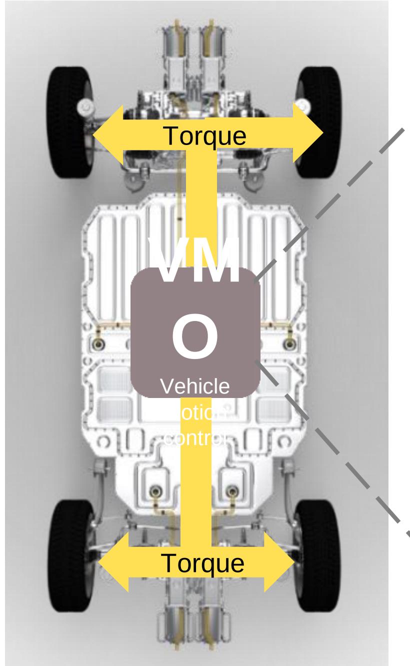
RNN MLP

Predictive neural networks can help to increase energy efficiency, thermal load & driving smoothness

M. Dendaluce Jahnke, et al., "Efficient Neural Network Implementations on Parallel Embedded Platforms Applied to Real-Time Torque-Vectoring Optimization Using Predictions for Multi-Motor Electric Vehicles," in Electronics 2019, 8, 250

Torque vectoring

- Main objective:
  - Independent torque control at each wheel
- Effect when driving a curve:
  - Provide more torque to the outside rear wheel
  - Reduce the speed of the inside wheels



Torque request

Steering angle

Car sensor data

AURIX TC4x

TriCore™	PPU
Signal Processing	Virtual Sensors (based on NN)
Default Torque Vectoring	
Control Optimization	Predictive Neural Network
Stability Functions	

- ↑ Handling
- ↑ Efficiency
- ↑ Smoothness
- ↓ Thermal stress

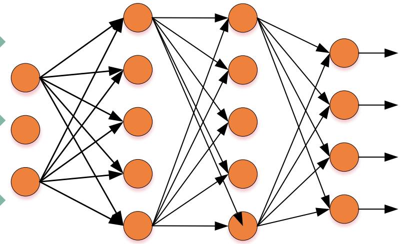
Concept

Vehicle Speed

Steering wheel

Acceleration

...



Output:

Future slip values of each wheel

Mathematical implementation

$$sig\left(\begin{bmatrix} a_{00} & \cdots & a_{0m} \\ \vdots & \ddots & \vdots \\ a_{n0} & \cdots & a_{nm} \end{bmatrix} \cdot \begin{pmatrix} x_0 \\ \vdots \\ x_m \end{pmatrix}\right)$$

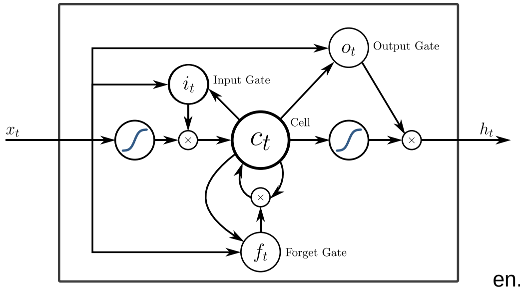
Activation function | Weight matrix | Input vector

Challenges for LSTM on MAC accelerators – google TPU (ISCA 2017)

Name	Layers				
	FC	Conv	Vector	Pool	Total
LSTM0	24		34		58
LSTM1	37		19		56
CNN0		16			16
CNN1	4	72		13	89

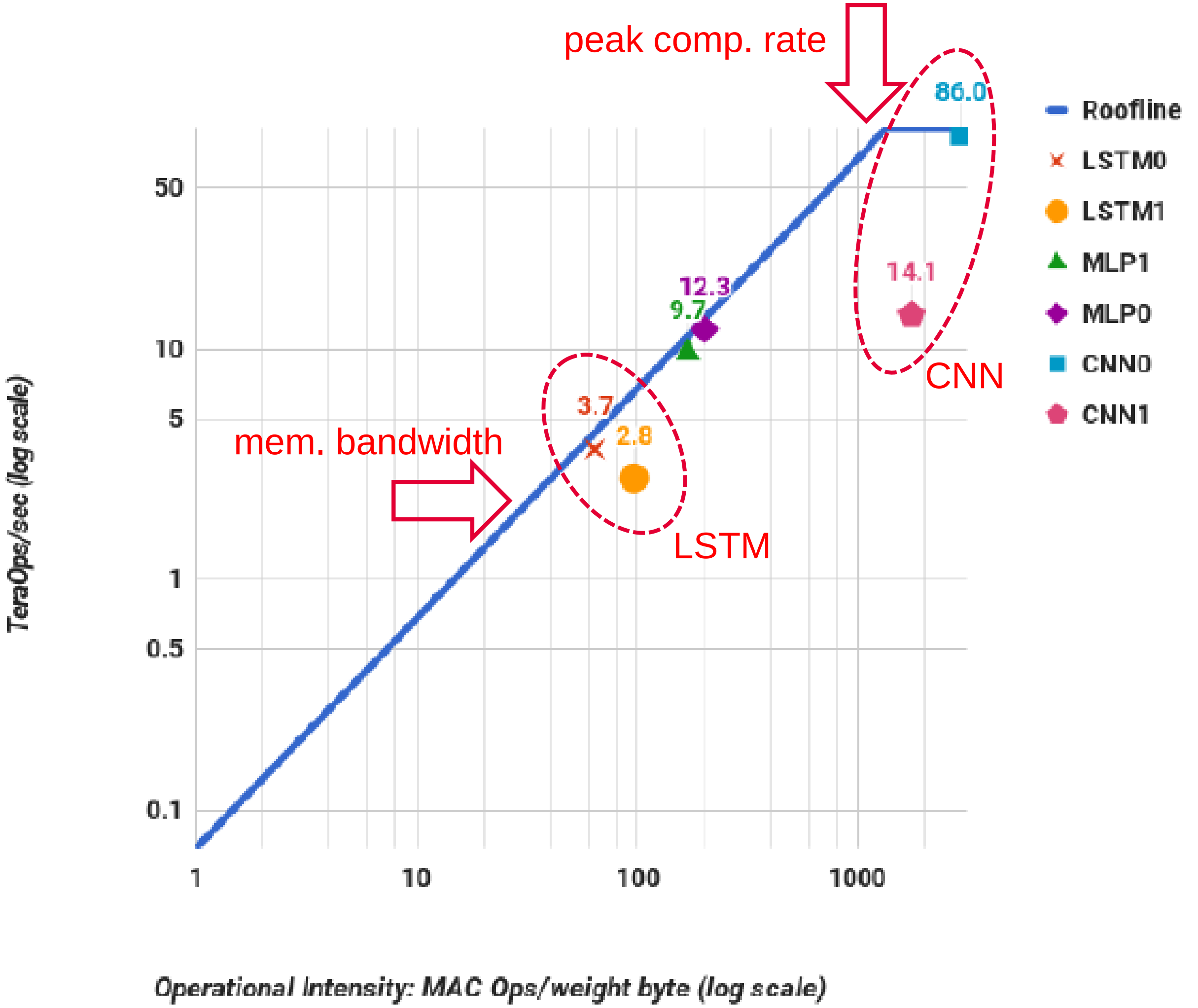
Application	*LSTM0*	*LSTM1*	*CNN0*	*CNN1*
Array active cycles	8.2%	10.5%	78.2%	46.2%
Useful MACs in 64K matrix (% peak)	8.2%	6.3%	78.2%	22.5%
Unused MACs	0.0%	4.2%	0.0%	23.7%
Weight stall cycles	58.1%	62.1%	0.0%	28.1%
Weight shift cycles	15.8%	17.1%	0.0%	7.0%
Non-matrix cycles	17.9%	10.3%	21.8%	18.7%
RAW stalls	14.6%	10.6%	3.5%	22.8%
Input data stalls	5.1%	2.4%	3.4%	0.6%
TeraOps/sec (92 Peak)	3.7	2.8	86.0	14.1

LSTM … a gated RNN



en.Wikipedia.org

MAC accelerators for LSTM have to go back from matrix-matrix to vector-matrix and typically are limited by memory bandwidth

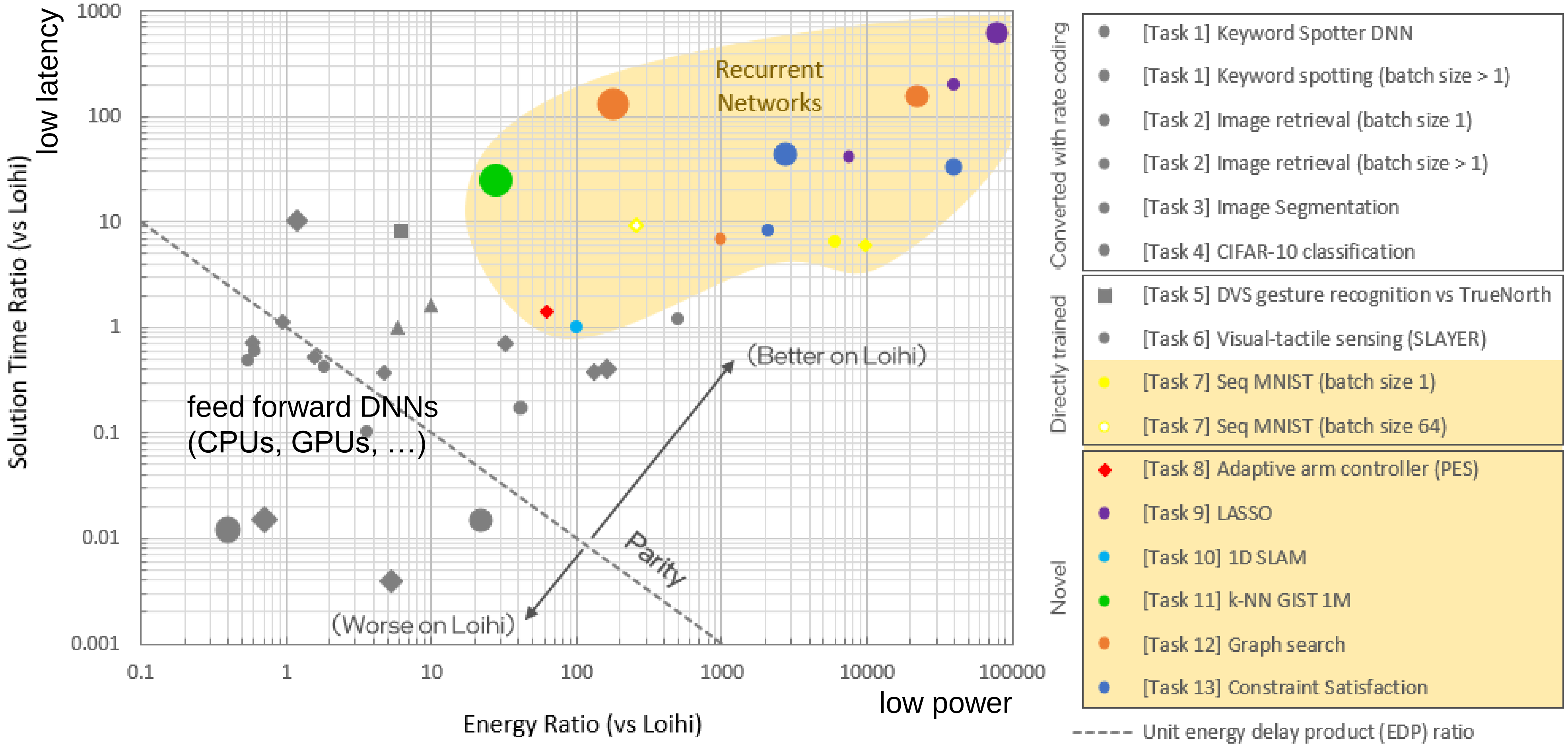


https://doi.org/10.1145/3079856.3080246

What Applications now working best on real Platforms?

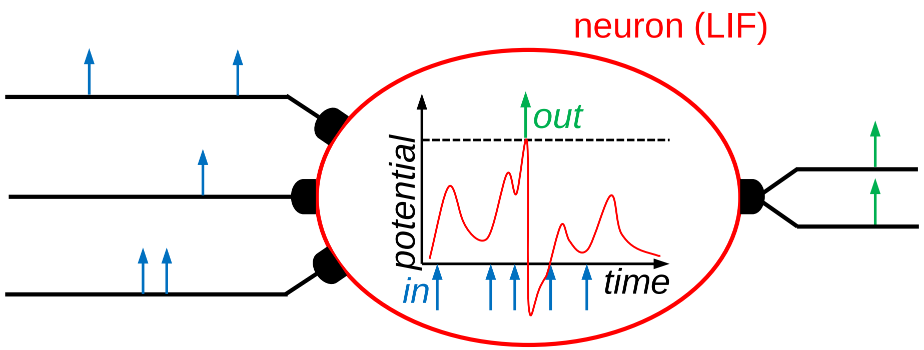
Intel Loihi:

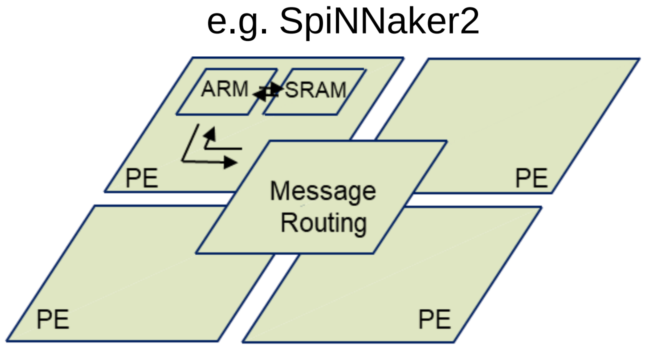
"Recurrent networks with bio-inspired properties give the best gains"



Mike Davies on Loihi app. perf., Intel @NICE2021 https://www.youtube.com/watch?v=-dl1FPrpw1A

What are Gains by Spiking Neural Networks?





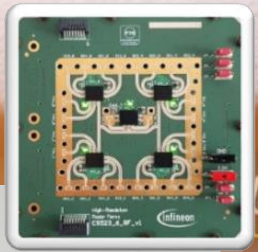
low power - sparse events, integrated memory and compute

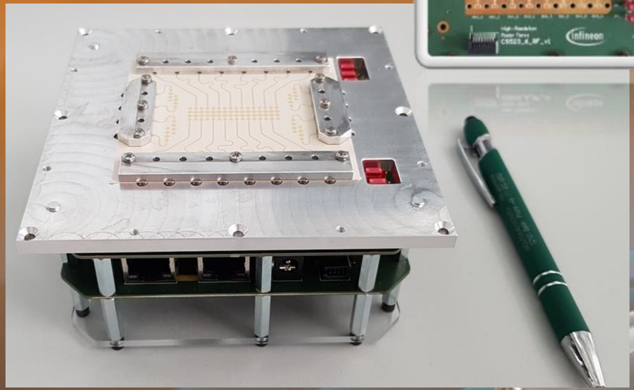
low latency - process when event occurs, #neuron connections

inherent recurrence - membrane potential

adaptive - local (un)supervised learning

KI-ASIC

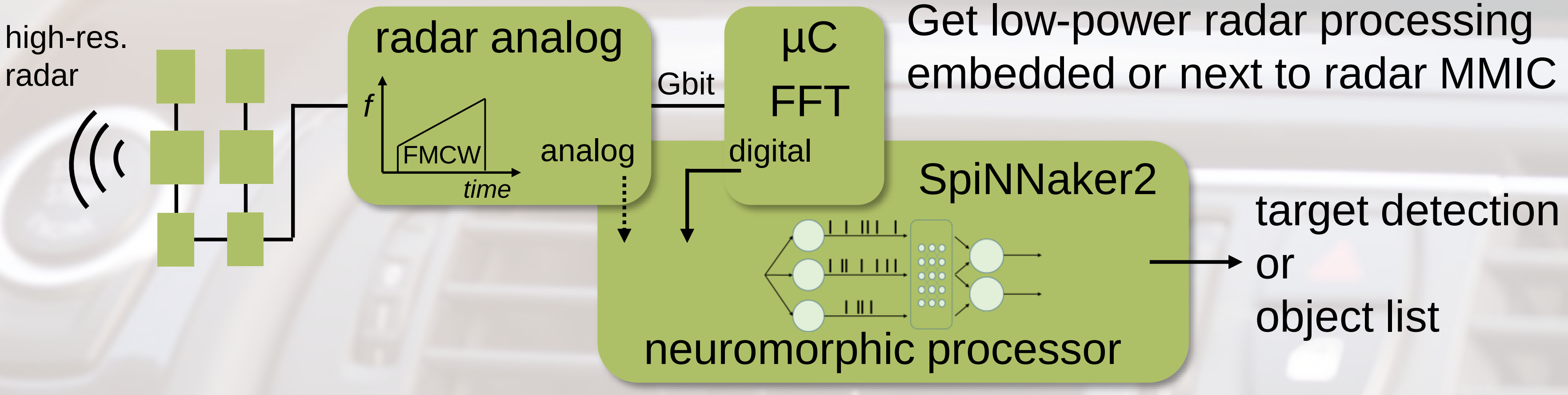




GEFÖRDERT VOM
Bundesministerium für Bildung und Forschung

Neuromorphic Signal Processing for Radar

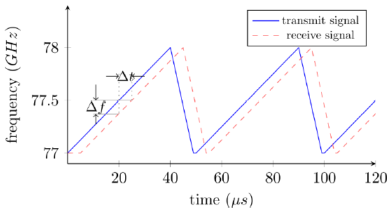
Get low-power radar processing embedded or next to radar MMIC

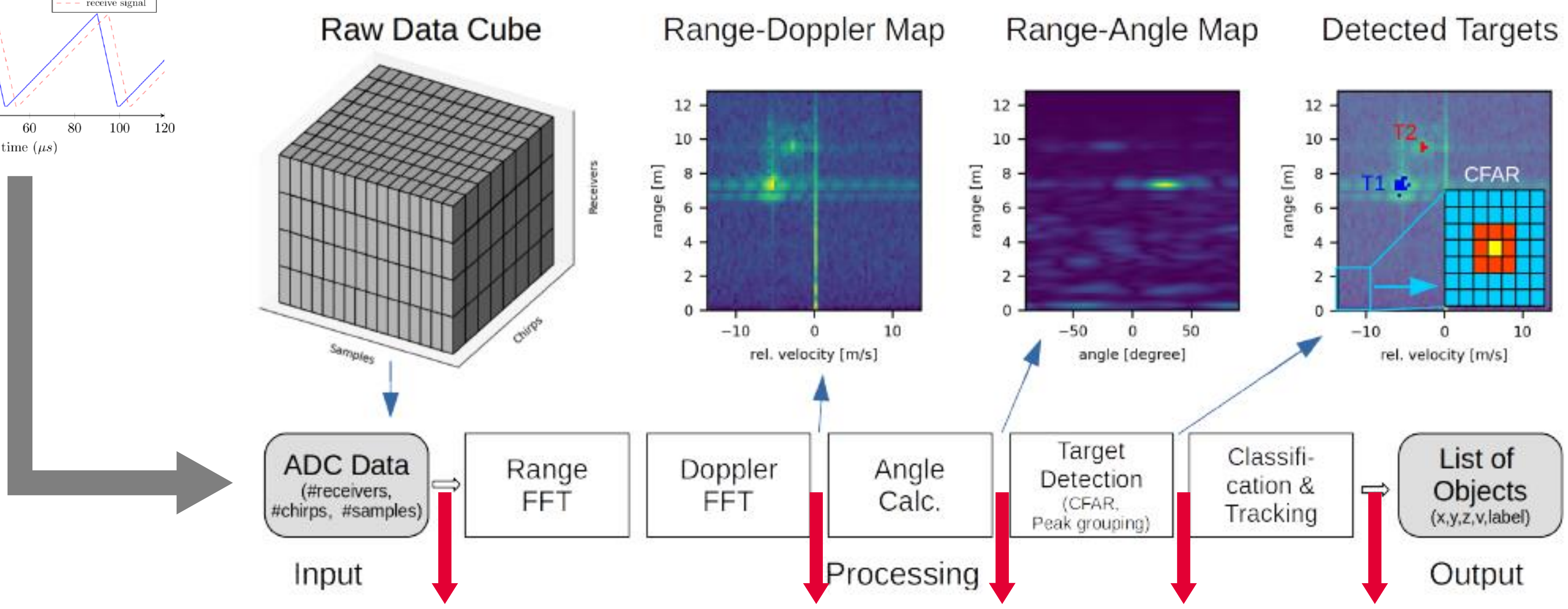


Ostbayerische Technische Hochschule Amberg-Weiden
Infineon Dresden
Technische Universität Dresden
Infineon
TUM
BMW GROUP

Automotive Radar Processing with Spiking Neural Networks

https://www.frontiersin.org/articles/10.3389/fnins.2022.851774/abstract





SNN

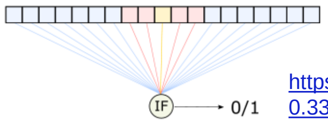
digital raw data	spiking FFT/DFT	spiking CFAR	spiking object detection	spiking object tracking

DFT as matrix multiplication

$$\begin{pmatrix} \Re(y) \\ \Im(y) \end{pmatrix} = \begin{pmatrix} \Re(\mathbf{W}_{\mathrm{DFT}}) & -\Im(\mathbf{W}_{\mathrm{DFT}}) \\ \Im(\mathbf{W}_{\mathrm{DFT}}) & \Re(\mathbf{W}_{\mathrm{DFT}}) \end{pmatrix} \begin{pmatrix} \Re(x) \\ \Im(x) \end{pmatrix}$$

http://arxiv.org/abs/2202.12650v1

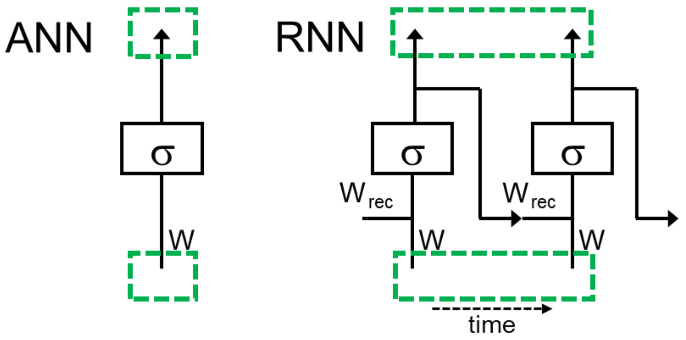
CFAR by IF neuron (time encoded)



https://www.frontiersin.org/articles/10.3389/fnbot.2021.688344/full

Non-differentiability of spiking neuron‘s activation function requires pseudo derivatives for error backpropagation

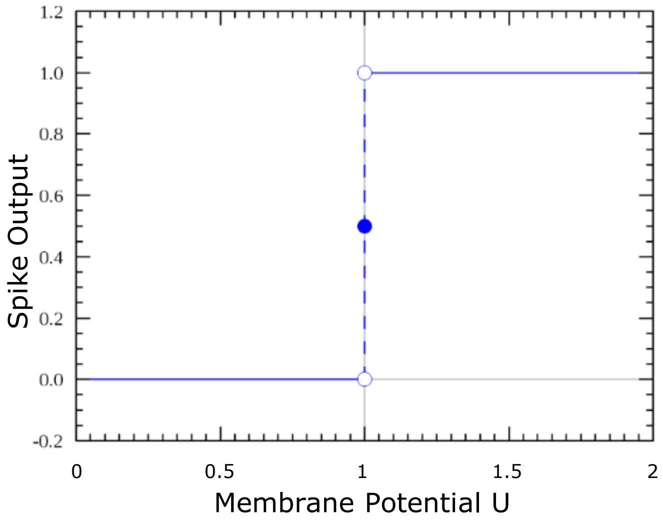
Back Propagation Through Time (BPTT)



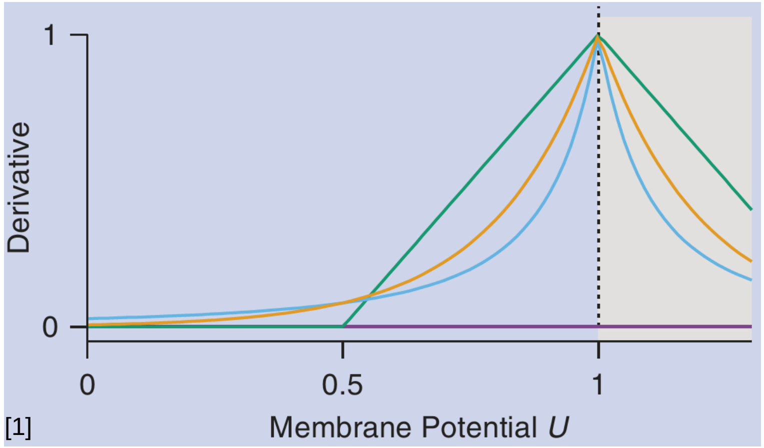
BP $\frac{\partial E}{\partial W}$

BPTT $\sum_t \frac{\partial E_t}{\partial W}$

Spike emission on threshold



BPTT with Surrogate Gradient



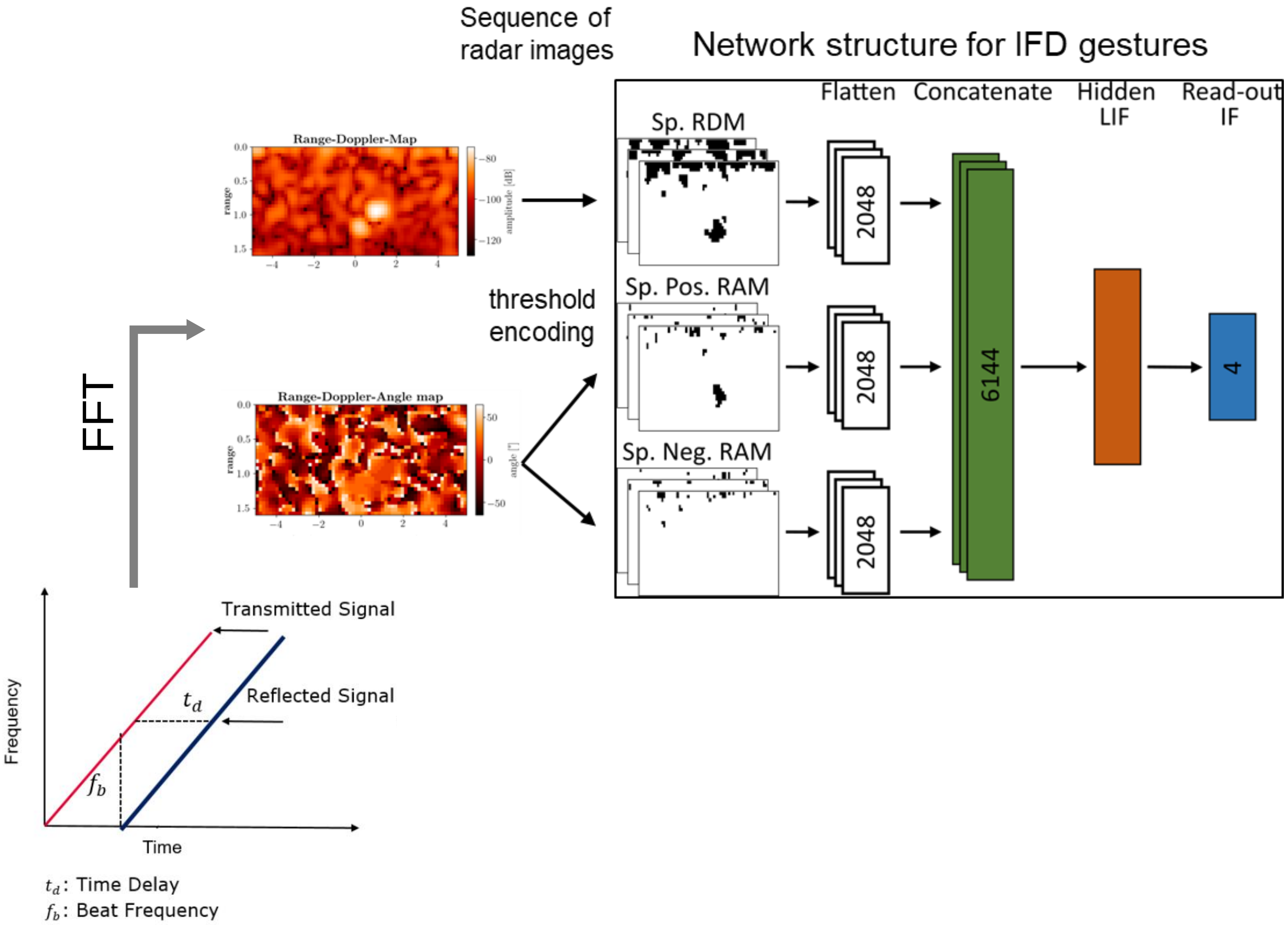
Vanishing gradient!

$$\prod_t \sigma' \ldots W_{rec} = 0; \sigma' < 1 \; (\infty; W_{rec} \gg \sigma')$$

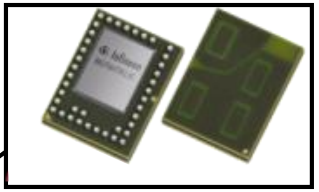
=> Simulation and training now possible in Tensorflow!

[1] E. O. Neftci, H. Mostafa, und F. Zenke, „Surrogate Gradient Learning in Spiking Neural Networks: Bringing the Power of Gradient-Based Optimization to Spiking Neural Networks“, IEEE Signal Processing Magazine, Nov. 2019, doi: 10.1109/MSP.2019.2931595.

2D-FFT algorithm extracts range and velocity of targets from time delay and doppler shift of reflected signal

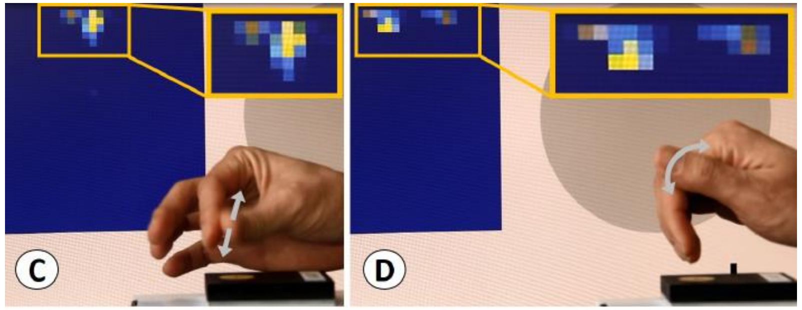


IFD hand gestures
60 GHz radar



4 gestures

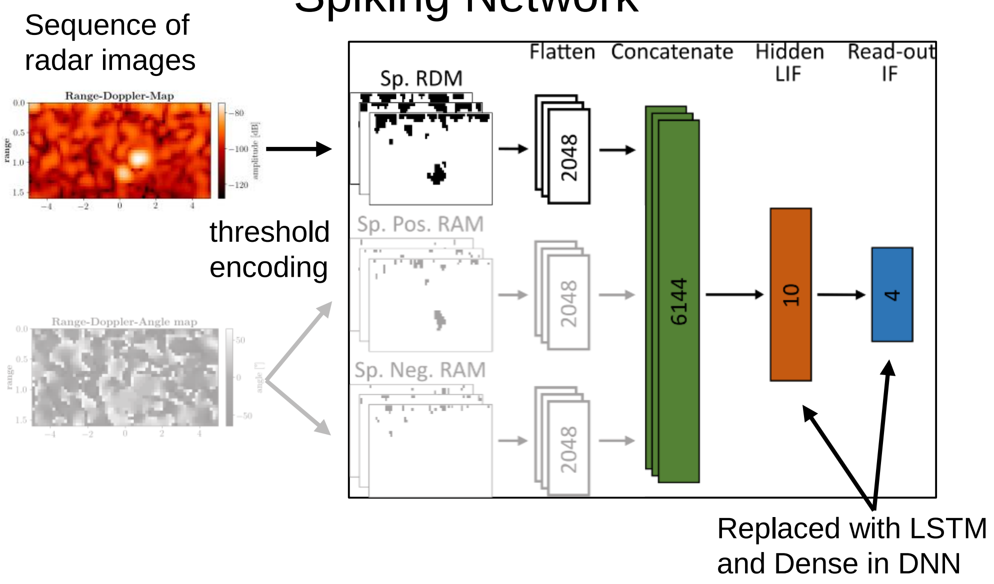
Google Soli hand gestures



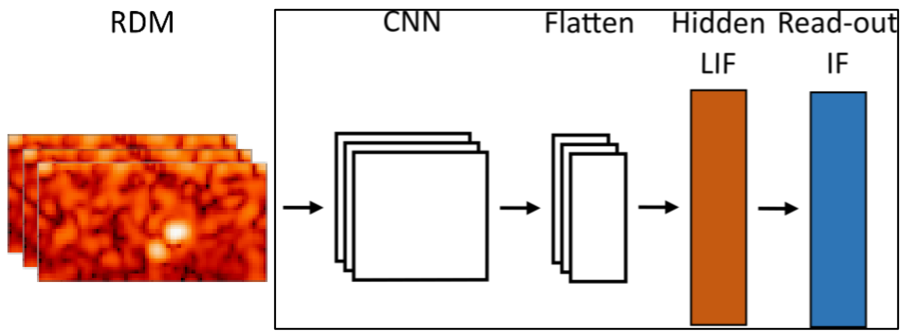
12 fine grained gestures

Hybrid and spiking NNs promise significant gains in energy consumption compared to LSTM networks without loss of accuracy

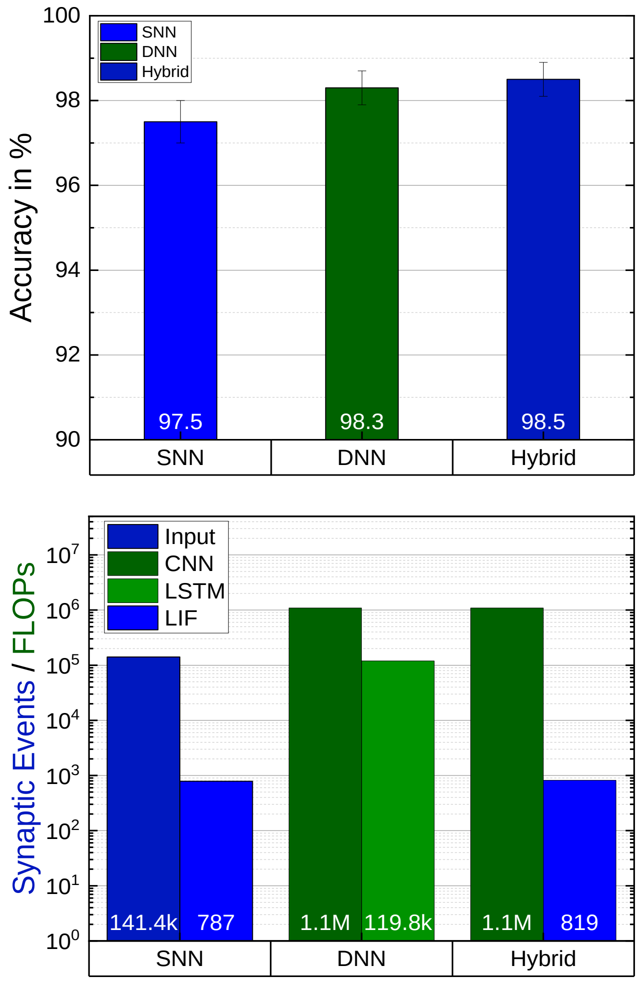
Spiking Network



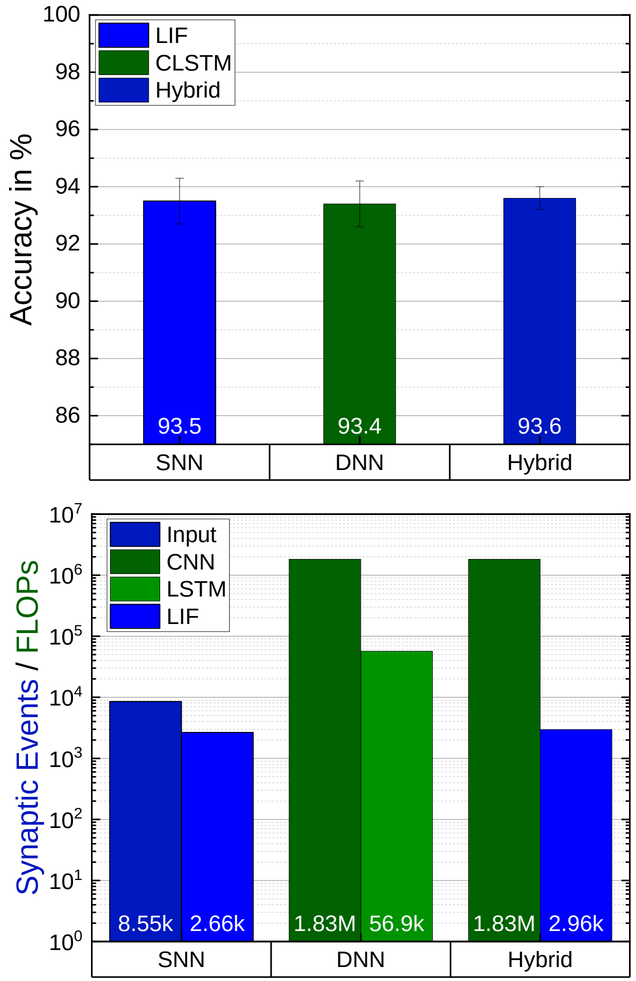
Hybrid Network



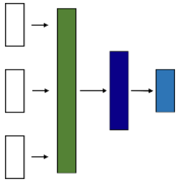
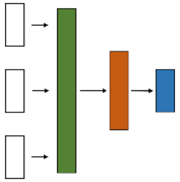
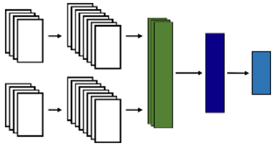
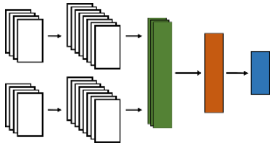
IFD Gesture Set

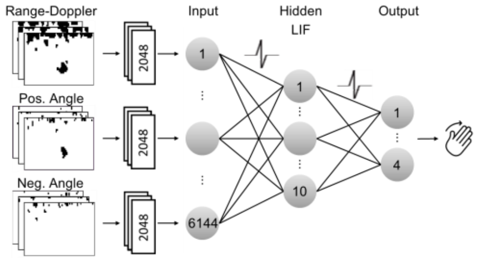


Google Soli



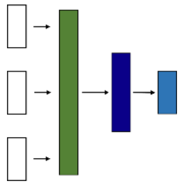
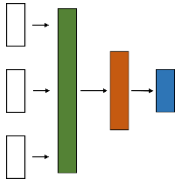
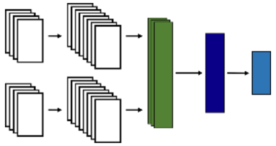
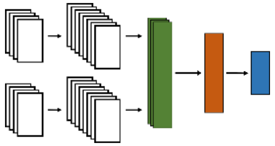
Radar Gesture Recognition – CNN – LSTM – SNN Comparison

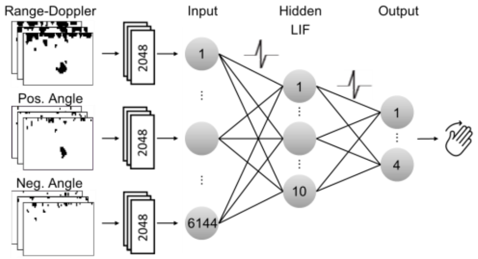
Network	Layer Architecture	With Angle Input	Neuron Type	#Parameters	Flops per inference (CNN/LSTM)	Synaptic events per inference (inp/hidden)	Accuracy
3D-CNN	8C(1,3,6)K – (1,2,4)P – 12C(1,3,3)K – (1,2,2)P – 64 – 4	Yes	CNN & FC	59.7k	24.0M CNN 0.11M Dense		95.6
LSTM 	2048 – **8** – 4	No	LSTM	65.9k	179k		95.4±1.5
	6144 – **4** – 4	Yes	LSTM	65.6k	226k		40.9±2.3
SNN 	2048 – **30** – 4	No	LIF	61.6k		141k/787	97.5±0.5
	6144 – **10** – 4	Yes	LIF	61.5k		60k/335	99.2±0.3
CNN-LSTM 	4C3K(2,4)S – 8C3K2S – **35** – 4	No	CNN & LSTM	60.4k	1.09M/120k		98.3±0.4
	2x[4C3K(2,4)S – 4C3K2S] – **19** – 4	Yes	CNN & LSTM	61.8k	2.17M/122k		98.7±0.6
CNN-SNN 	4C3K(2,4)S – 8C3K2S – **117** – 4	No	CNN & LIF	60.5k	1.09M/0	0/819	98.5±0.4
	4C3K(2,4)S – 8C3K2S – **70** – 4	Yes	CNN & LIF	60.8k	2.17M/0	0/2.5k	97.9±0.8



data: P. Gerhards

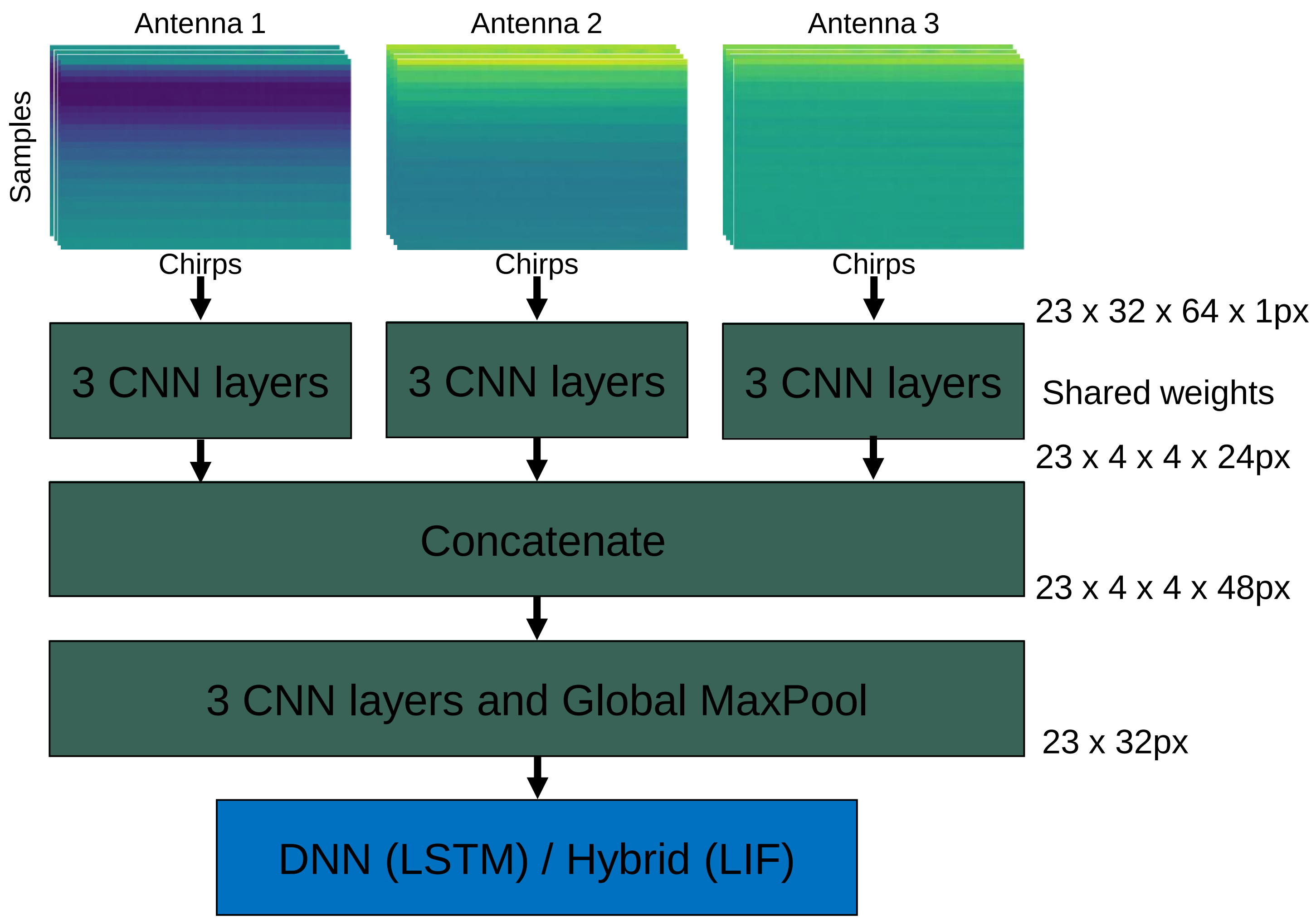
Radar Gesture Recognition – CNN – LSTM – SNN Comparison

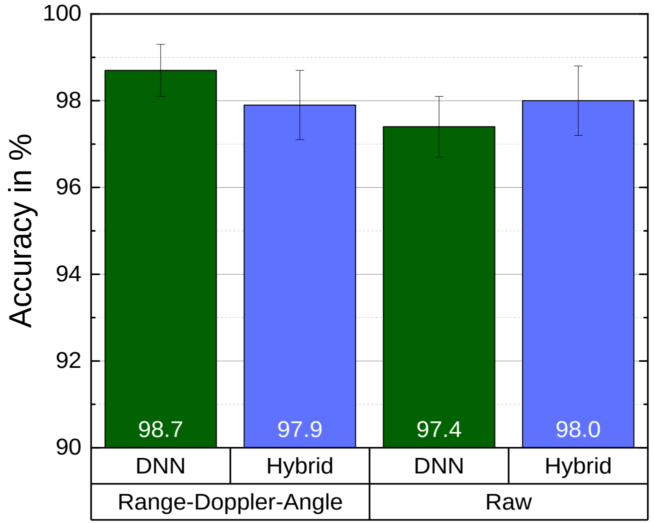
Network	Layer Architecture	With Angle Input	Neuron Type	#Parameters	Flops per inference (CNN/LSTM)	Synaptic events per inference (inp/hidden)	Accuracy
3D-CNN	8C(1,3,6)K – (1,2,4)P – 12C(1,3,3)K – (1,2,2)P – 64 – 4	Yes	CNN & FC	59.7k	24.0M CNN 0.11M Dense		95.6
LSTM 	2048 – **8** – 4	No	LSTM	65.9k	179k		95.4±1.5
	6144 – **4** – 4	Yes	LSTM	65.6k	226k		40.9±2.3
SNN 	2048 – **30** – 4	No	LIF	61.6k		141k/787	97.5±0.5
	6144 – **10** – 4	Yes	LIF	61.5k		60k/335	99.2±0.3
CNN-LSTM 	4C3K(2,4)S – 8C3K2S – **35** – 4	No	CNN & LSTM	60.4k	1.09M/120k		98.3±0.4
	2x[4C3K(2,4)S – 4C3K2S] – **19** – 4	Yes	CNN & LSTM	61.8k	2.17M/122k		98.7±0.6
CNN-SNN 	4C3K(2,4)S – 8C3K2S – **117** – 4	No	CNN & LIF	60.5k	1.09M/0	0/819	98.5±0.4
	4C3K(2,4)S – 8C3K2S – **70** – 4	Yes	CNN & LIF	60.8k	2.17M/0	0/2.5k	97.9±0.8

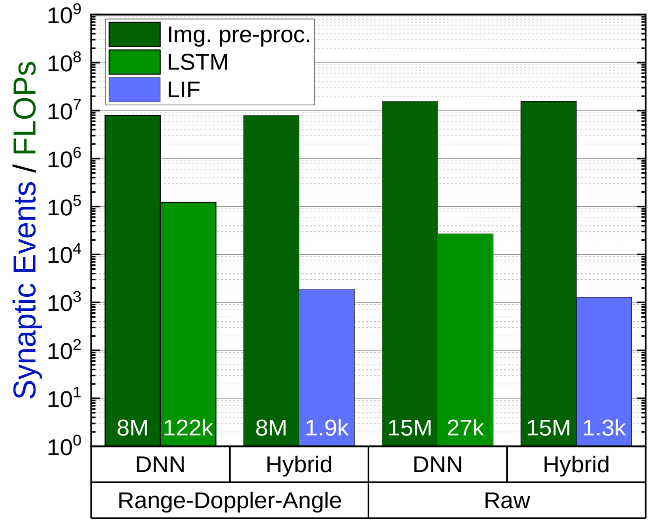


- Parameter count constant
- All ~95-99% at 60k param.
- 3D-CNN way more flops
- ~100k inp. syn. eq. 1M CNN
- ~120k LSTM eq. ~1k syn.

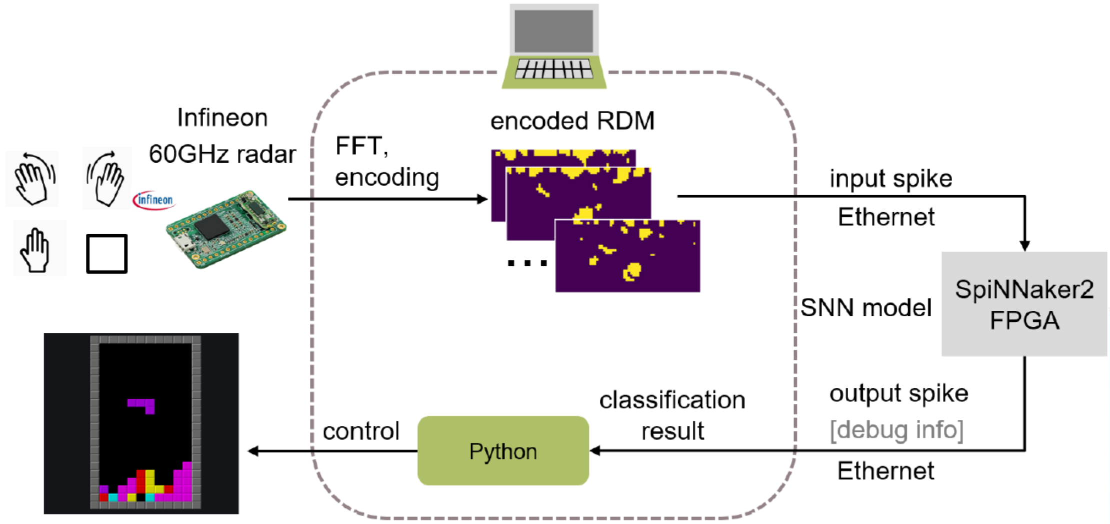
Do we need FFT-preprocessing or can we use Neural networks to extract the relevant information directly from raw radar data?





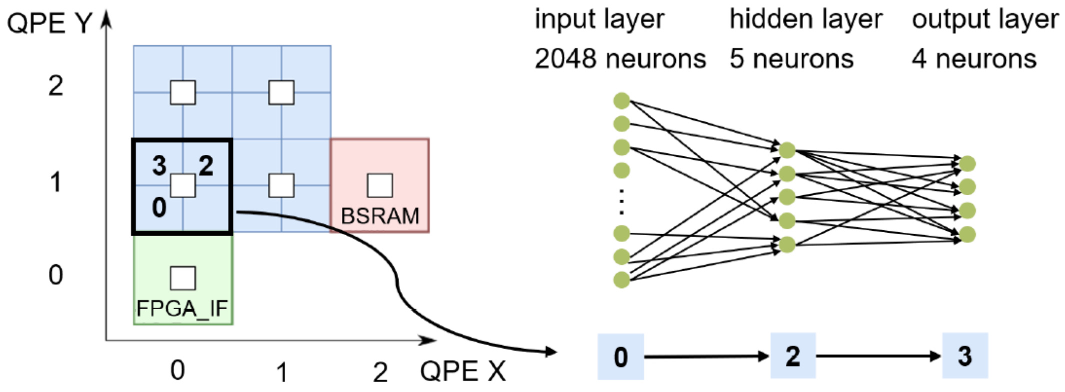


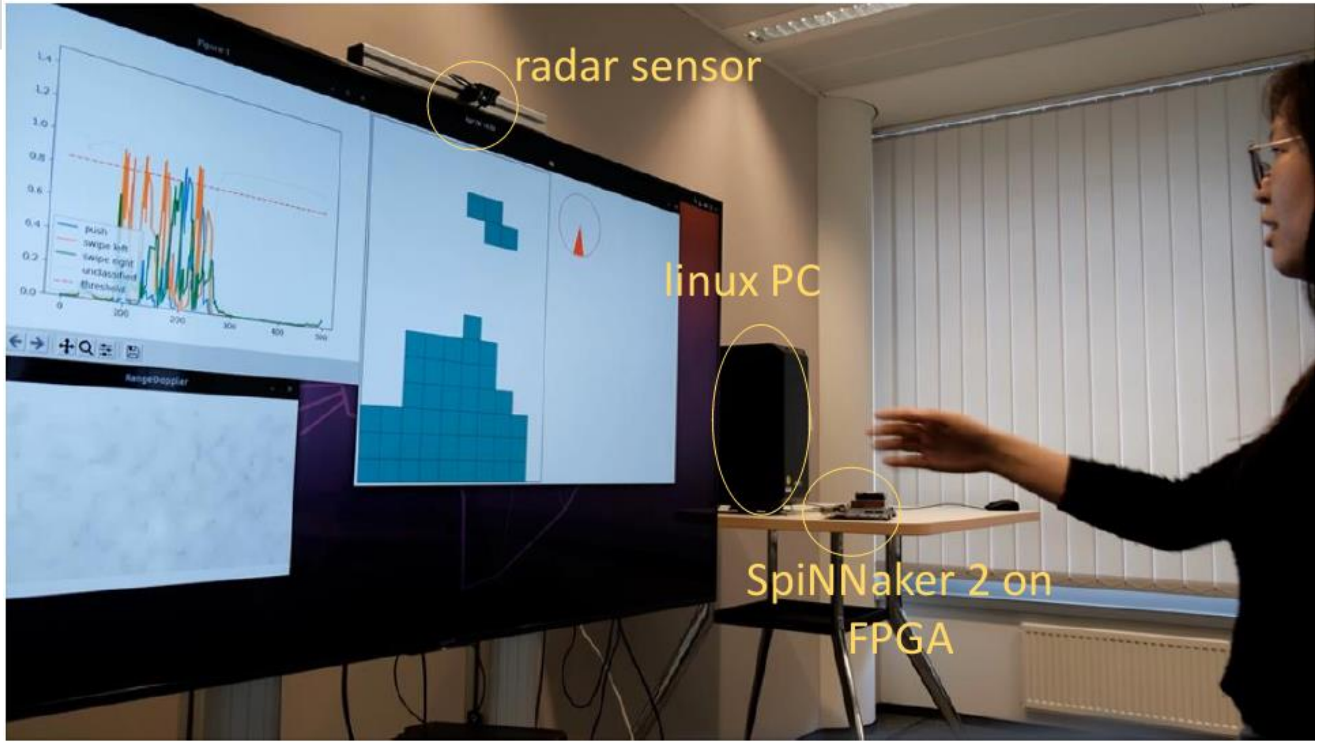
Radar Gesture SNN implemented on SpiNNaker2 FPGA



Radar frequency	60 GHz
Radar frame rate	33 ms
Delay from PC sending input data to receiving classificiation output	35 ms per frame
Neuron update timestep (systick)	1 ms
SpiNNaker 2 FPGA frequency	10 MHz
Number of gesture	3 (left swipe, right swipe, push) + 1 (none)
Gesture trigger threshold of game control	Softmax 90%
Insensitive classification duration	0.5 s

	PE memory	Operation cycle	Energy cost
PE 0	39.47%	593	avg.3.29 µJ/frame
PE 2	44.53%	6 k-8 k	
PE 3	20.87%	~300	





Presentation at AICAS 2022, Jiaxin Huang

Summary

- Automotive trends like electric drive and autonomous driving push for AI control and prediction applications and other time series data like radar
- E/E-architectures will move from domain to zone architecture to enable hardware complexity reduction and allow for abstraction and scalable system architectures (software)
- Control & prediction, as well as radar processing, demanding use of recurrent AI architectures in zone controllers – resource and power efficient processing is key
- Applications with spatio-temporal stream and high data rates (radar) could benefit from (sparse) spiking neural network processing
- SNN model architecture and training to be co-developed with (generalized) hardware
- SNN benefits have to be demonstrated in practice. Hard- and software concepts to run generalized algorithms are to be developed. Standardized frameworks for network architecture and training are to be established.