Sleep Spindles as a Driver of Low Latency, Low Power ML in HLS4ML

Hardware Development: Xiaohan Liu (Speaker), Atharva Mattam, Chi-Jui Chen, Lin-Chi Yang, Yan-Lun Huang, Elham E Khoda, Scott Hauck, Shih-Chieh Hsu, Bo-Cheng Lai

Neural Interfaces: Lauren Peterson, Leo Scholl, Amy Orsborn

Neural Processing Algorithms: Trung Le, Eli Shlizerman

07/10/2023

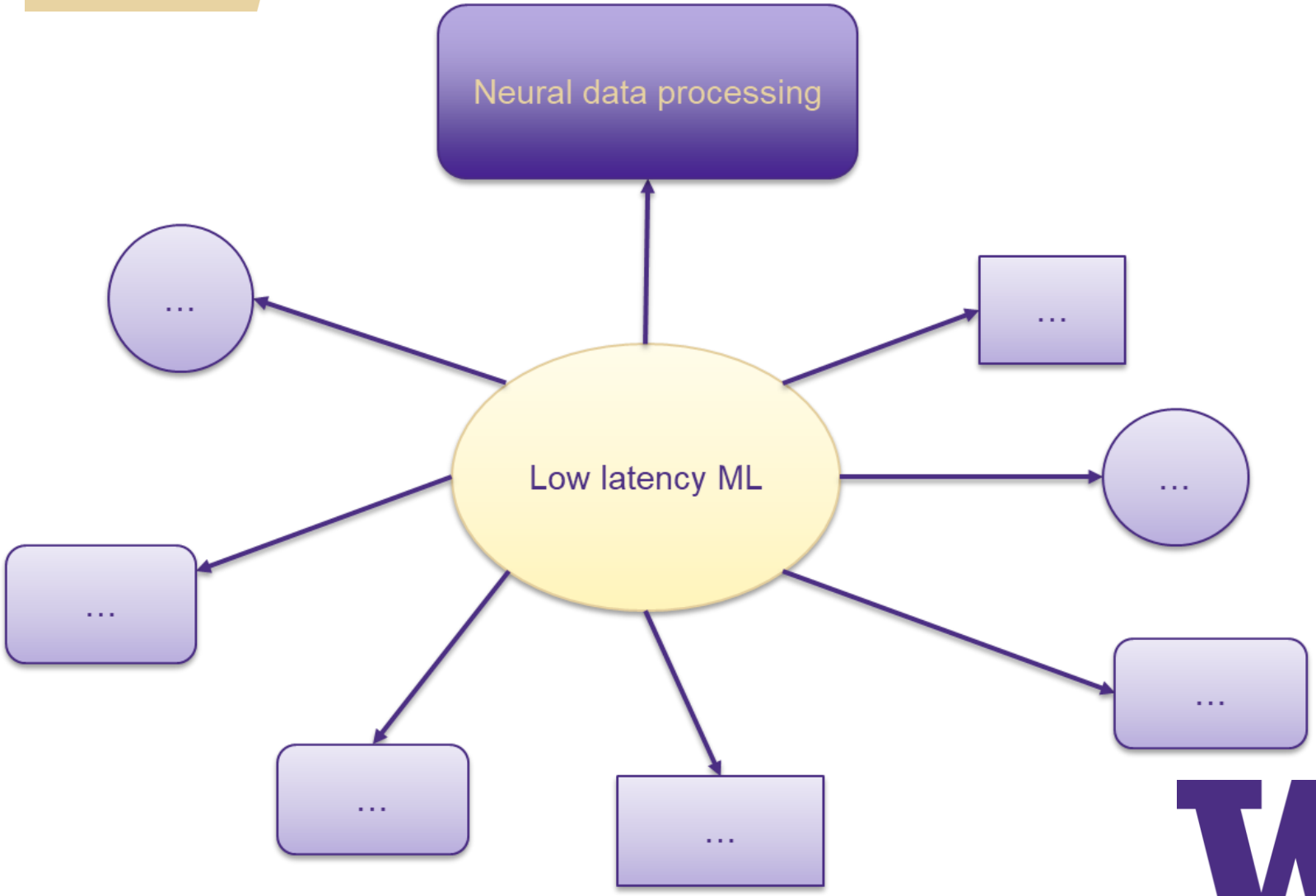
a3d3 Accelerated AI Algorithms for Data-Driven Discovery

hls4ml

OAC-2117997

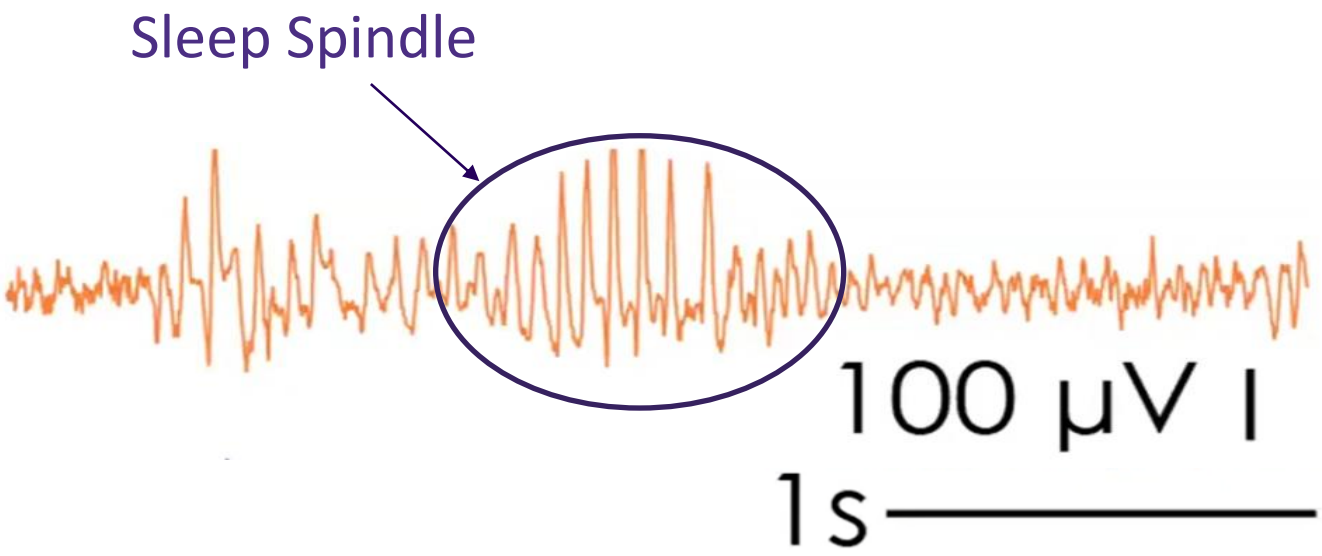
NSF National Science Foundation

What We Do



Sleep Spindles

- Oscillating signals during sleep or rest
- Believed to contribute to learn



- Goal: Predict and disrupt spindles

Orsborn A, Shlizerman E, Dadarlat M. (2021) "Understanding & Interfacing with the brain: challenges and opportunities"

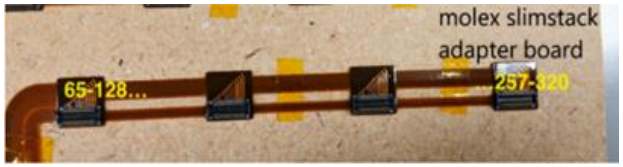
System Setup

Electrocorticography with 4 arms

+

Amplifier

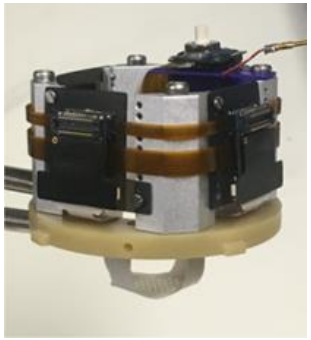
+



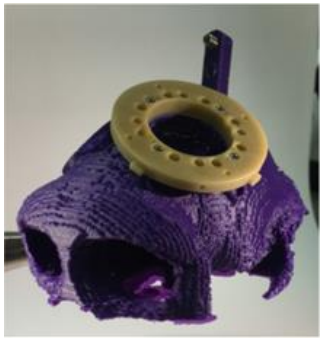
Arm adapter

+

Headstages

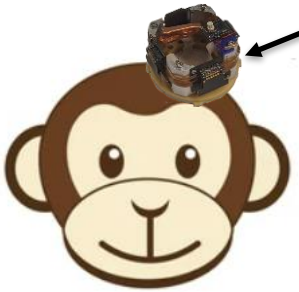


+



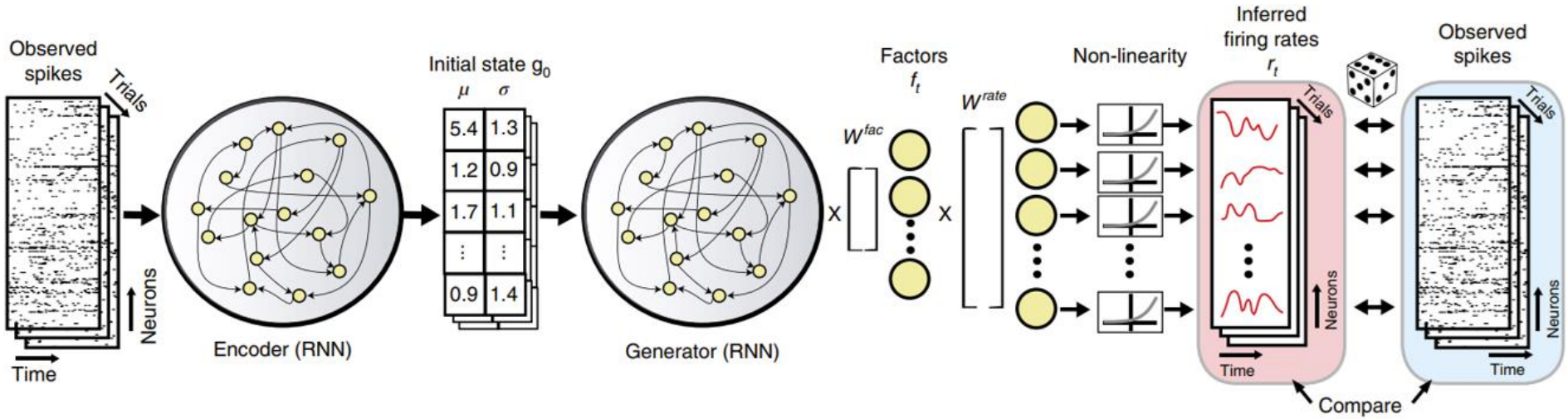
Head-mounted device

subject's skull



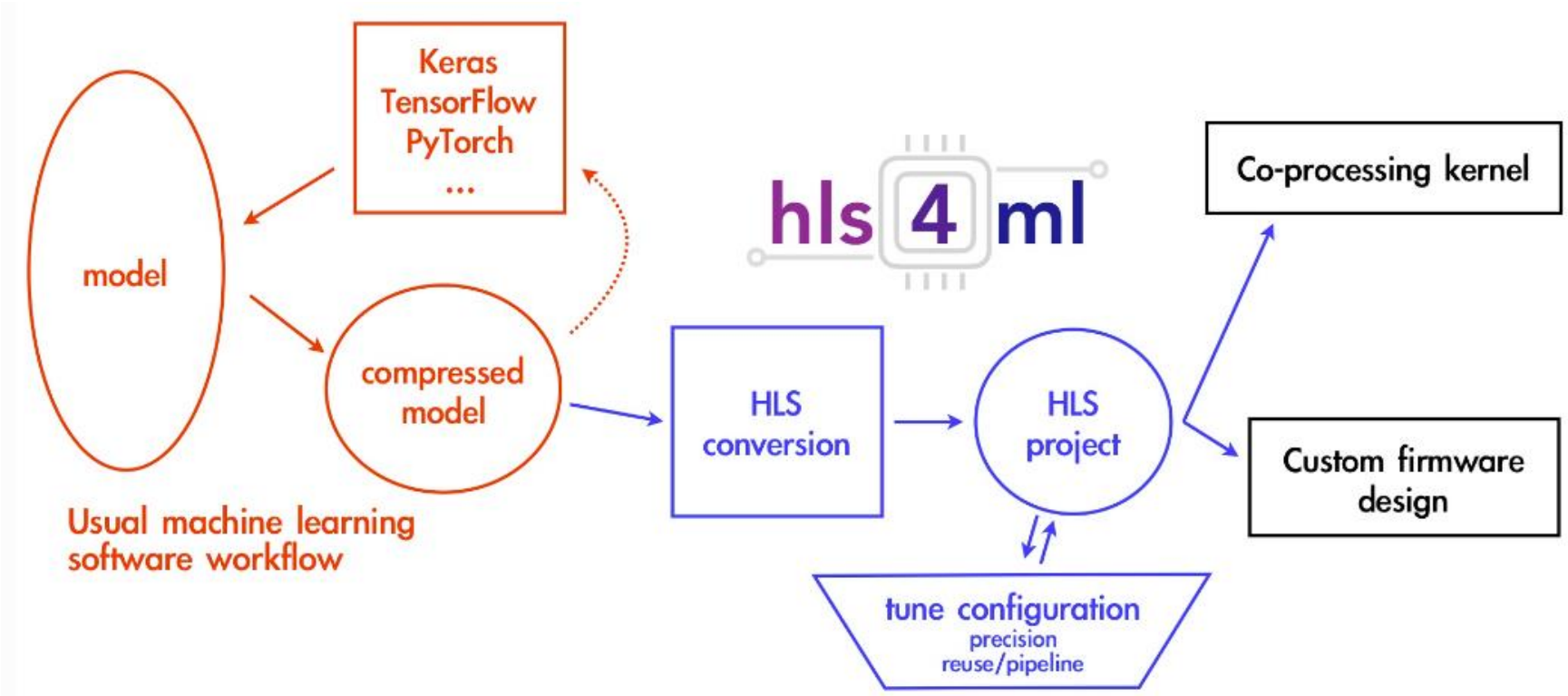
Latent Factor Analysis via Dynamical Systems (LFADs)

- RNN Variational Autoencoder (RNN VAE)



Pandarinath, C., O'Shea, D. J., Collins, J., Jozefowicz, R., Stavisky, S. D., Kao, J. C., ... & Sussillo, D. (2018). Inferring single-trial neural population dynamics using sequential auto-encoders. *Nature methods*, *15*(10), 805-815.

Tool – HLS4ML


Keras
TensorFlow
PyTorch
...
model
compressed
model
Usual machine learning
software workflow
hls 4 ml
HLS
conversion
HLS
project
Co-processing kernel
Custom firmware
design
tune configuration
precision
reuse/pipeline

LFADs Architecture - Detail

- LFADs architecture
  - Custom model
- One custom layer
  - Gaussian sampling layer
- One unsupported Keras layer
  - Bidirectional
- One unsupported arguments
  - GRU initial_state

Layer (type)	Output Shape	Param #
initial_dropout (Dropout)	multiple	0
EncoderRNN (Bidirectional)	multiple	52224
postencoder_dropout (Dropout)	multiple	0
postdecoder_dropout (Dropout)	multiple	0
DenseMean (Dense)	multiple	8256
DenseLogVar (Dense)	multiple	8256
GaussianSampling (GaussianSampling)	multiple	0
activation (Activation)	multiple	0 (unused)
DecoderGRU (GRU)	multiple	24960
Dense (Dense)	multiple	256
NeuralDense (Dense)	multiple	350

Total params: 94,315
Trainable params: 94,302
Non-trainable params: 13

Current Model Architecture

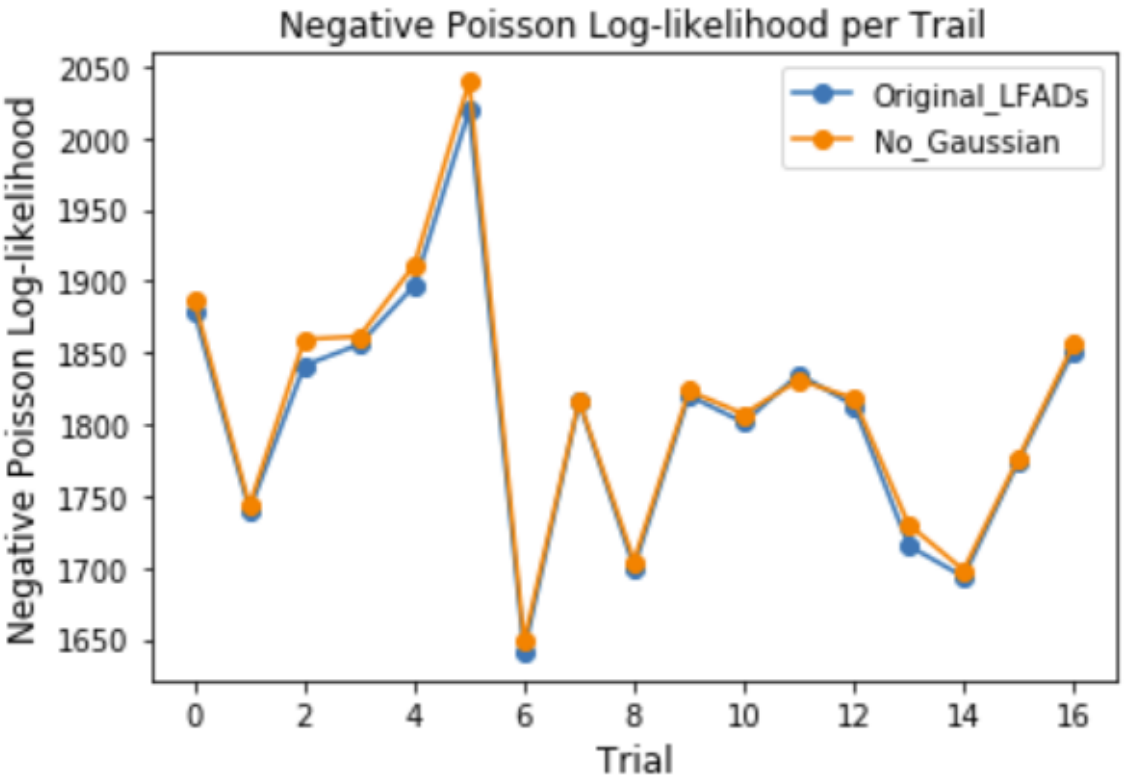
- Removed the gaussian: being developed in parallel
- Recreated LFADs in Keras functional API

Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	[(None, 73, 70)]	0	[]
initial_dropout (Dropout)	(None, 73, 70)	0	['input_1[0][0]']
Encoder_BidirectionalGRU (Bidirectional)	[(None, 128), (None, 64), (None, 64)]	52224	['initial_dropout[0][0]']
postencoder_dropout (Dropout)	(None, 128)	0	['Encoder_BidirectionalGRU[0][0]']
input_2 (InputLayer)	[(None, 73, 64)]	0	[]
dense_mean (Dense)	(None, 64)	8256	['postencoder_dropout[0][0]']
decoder_GRU (GRU)	(None, 73, 64)	24960	['input_2[0][0]', 'dense_mean[0][0]']
postdecoder_dropout (Dropout)	(None, 73, 64)	0	['decoder_GRU[0][0]']
dense (Dense)	(None, 73, 4)	256	['postdecoder_dropout[0][0]']
nerual_dense (Dense)	(None, 73, 70)	350	['dense[0][0]']

Total params: 86,046
Trainable params: 86,046
Non-trainable params: 0

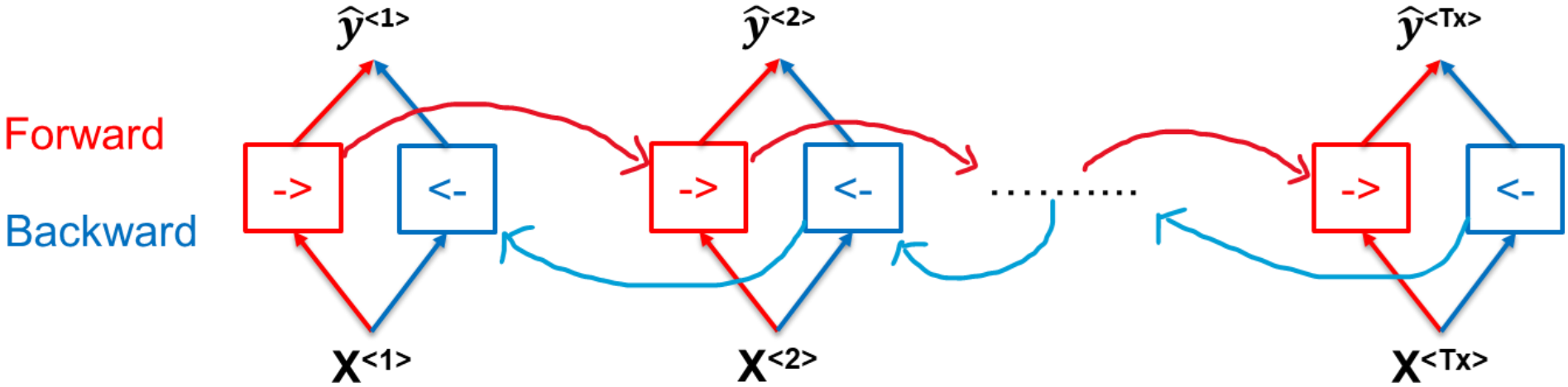
Performance Comparison

- Dataset from
  - Perich, M. G., Gallego, J. A., & Miller, L. E. (2018). A neural population mechanism for rapid learning. *Neuron*, *100*(4), 964-976.
- Evaluation metric - Negative poisson log-likelihood



HLS4ML Implementation – Bidirectional

- Implemented Bidirectional GRU layer



HLS4ML Implementation – GRU Initial State

- Implemented initial_state
  - Initial_state is hidden state of the unit

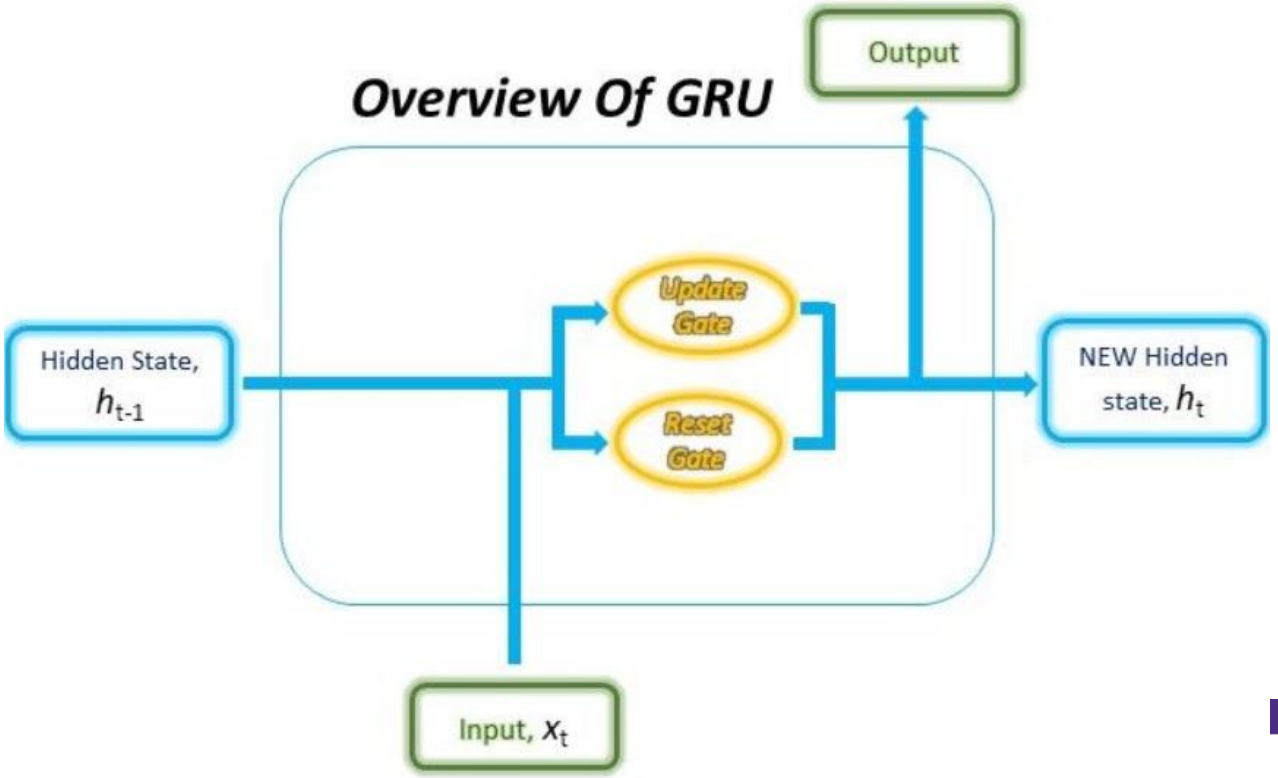
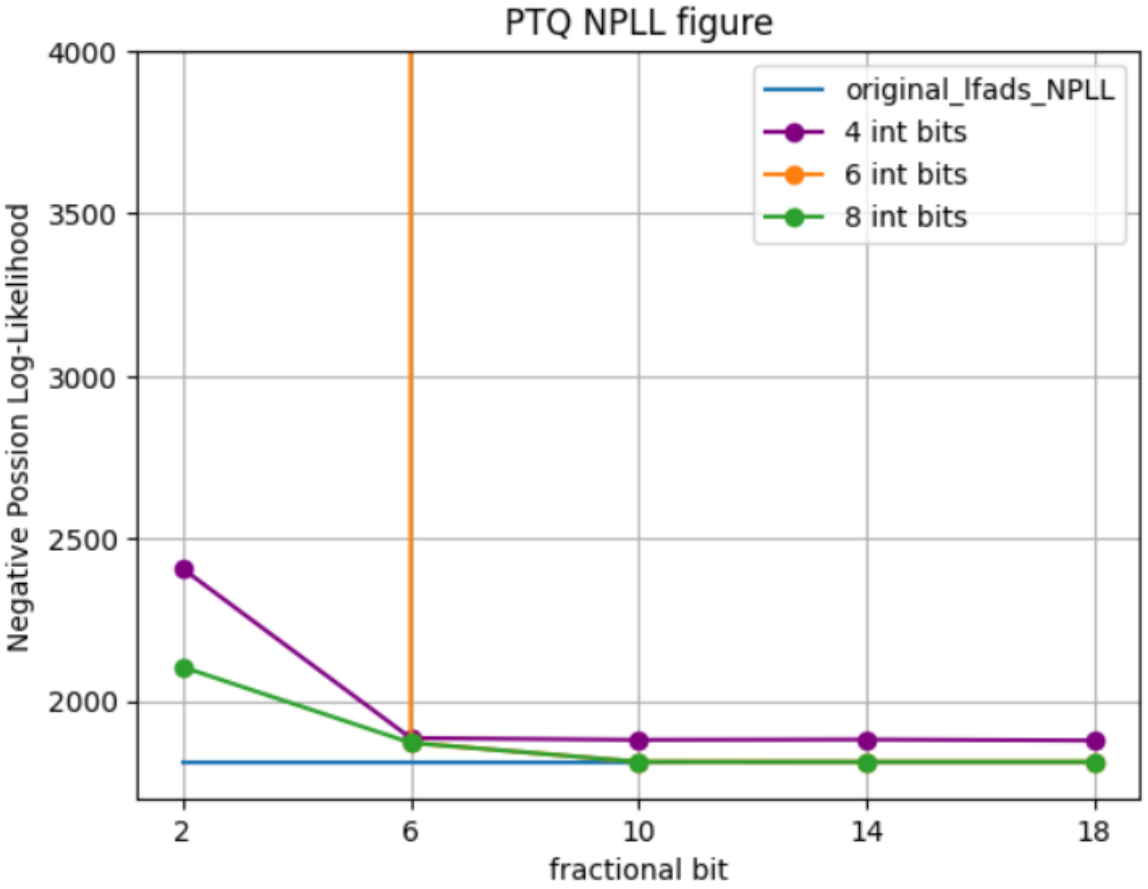


Image recourse: https://blog.floydhub.com

HLS (LFADs) Model Performance

- Post training quantization (PTQ) with different precision



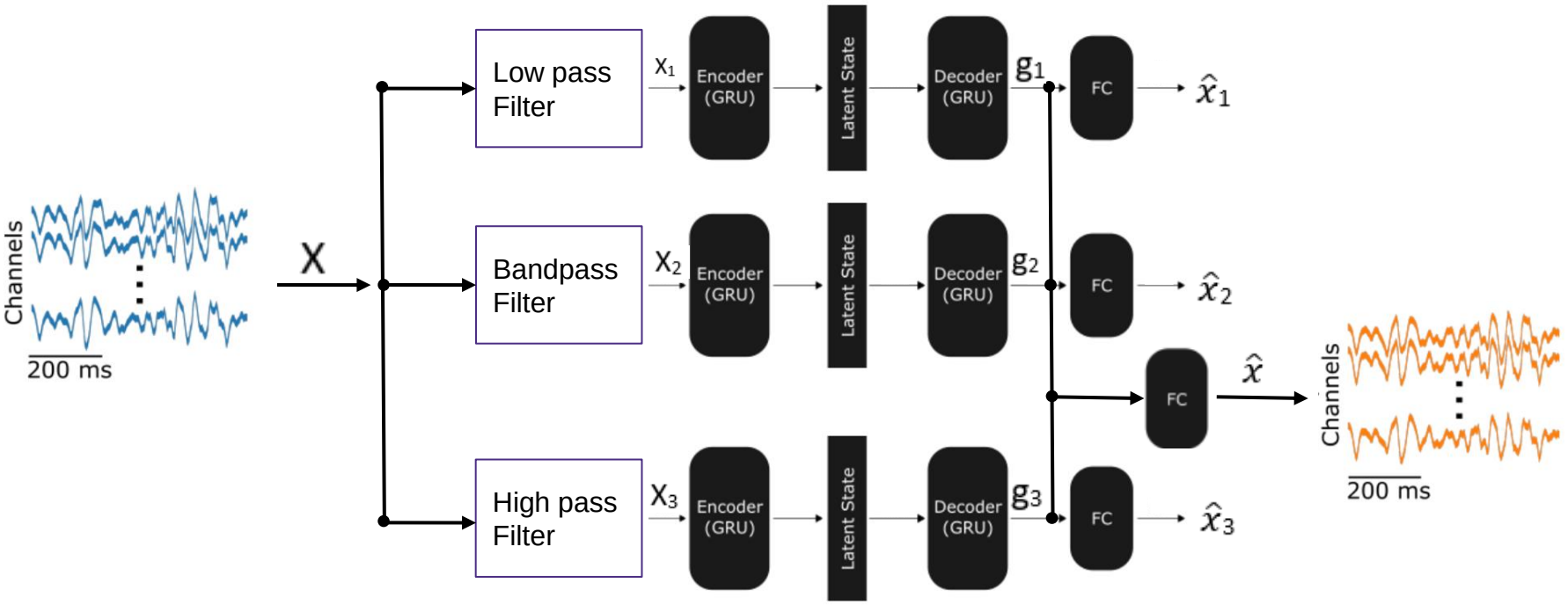
FPGA Deployment Result

- Current board: Xilinx U55C
- Precision: ap_fixed<16,6>
- Frequency: 200 MHz
- Latency: 41.97 µs

V synthesis	U55C
HLS version	2022
BRAM	474 (23.51%)
DSP	1,869 (20.71%)
FF	150,882 (5.79%)
LUT	164.726 (12.64%)

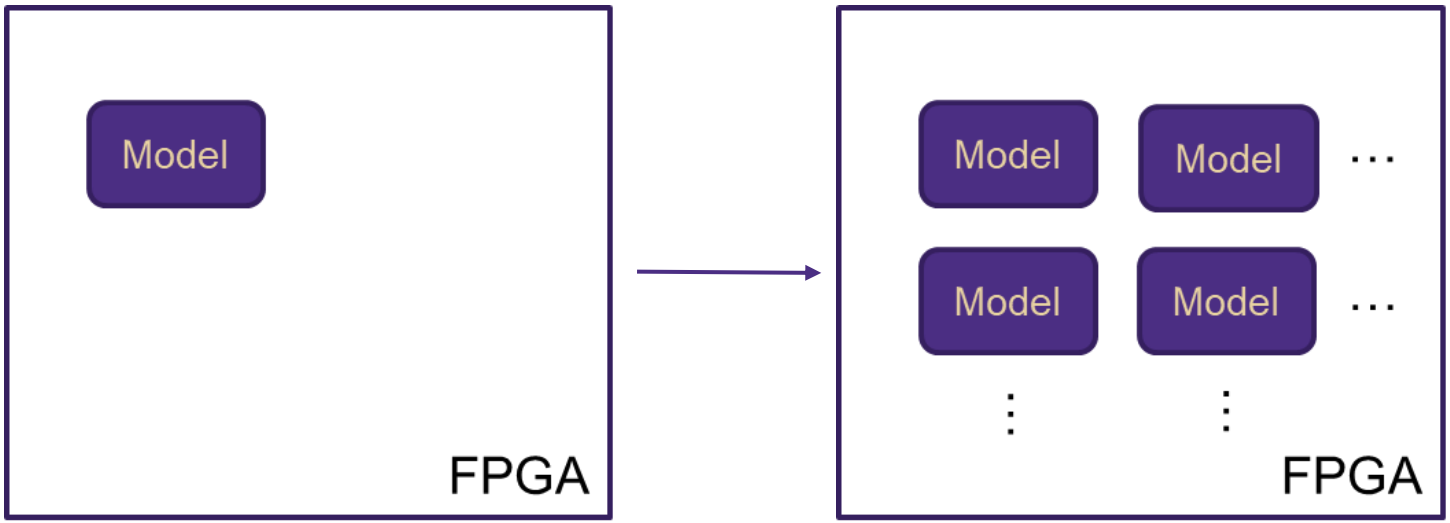
Multi-block RNN Autoencoders (MRAE)

- *Nolan, M., Pesaran, B., Shlizerman, E., & Orsborn, A. L. (2022). Multi-block RNN Autoencoders Enable Broadband ECoG Signal Reconstruction. bioRxiv, 2022-09.*



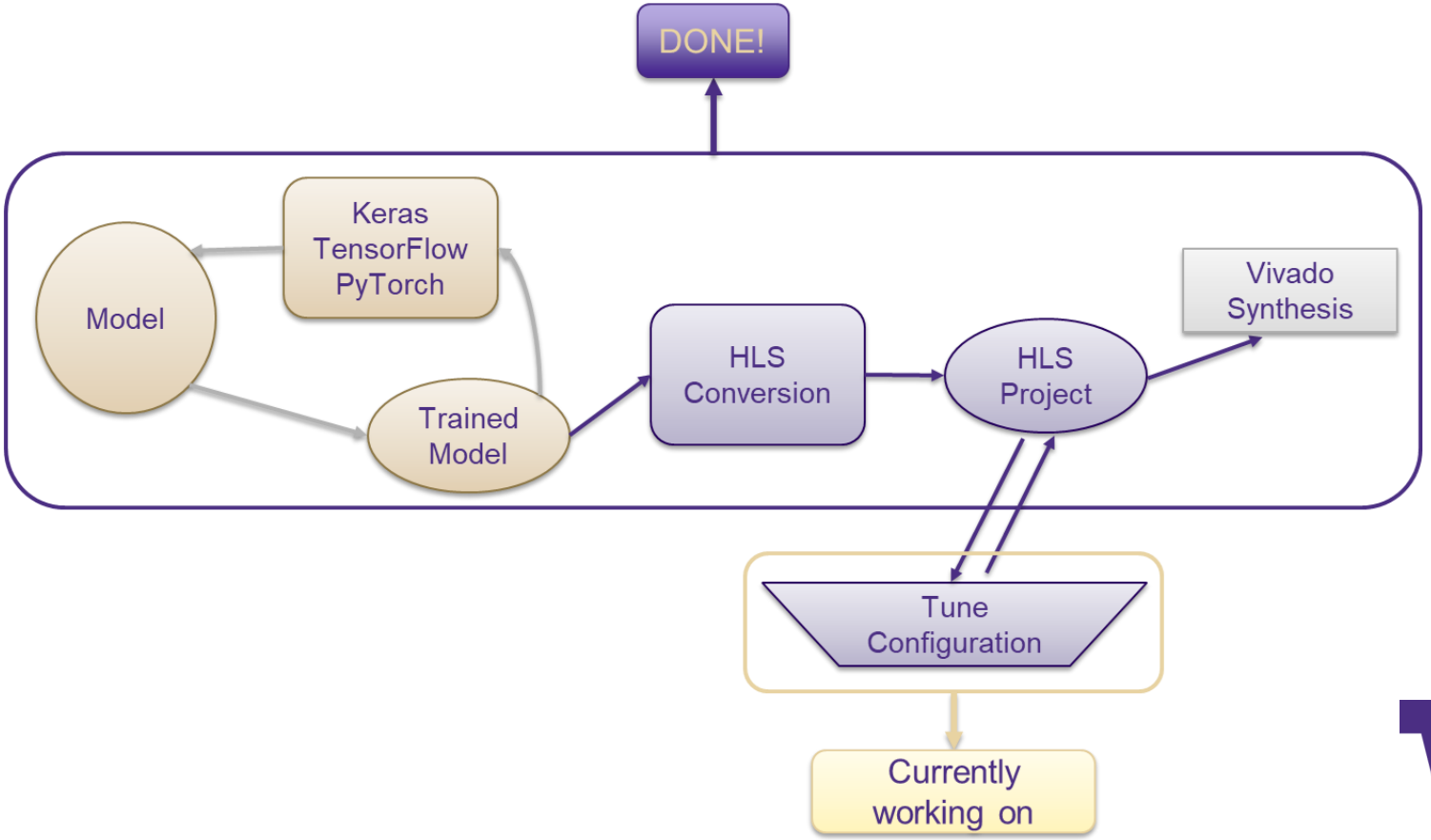
Multi-block RNN Autoencoders (MRAE)

- *Reduce FPGA resource utilization*
- *Increase throughput*



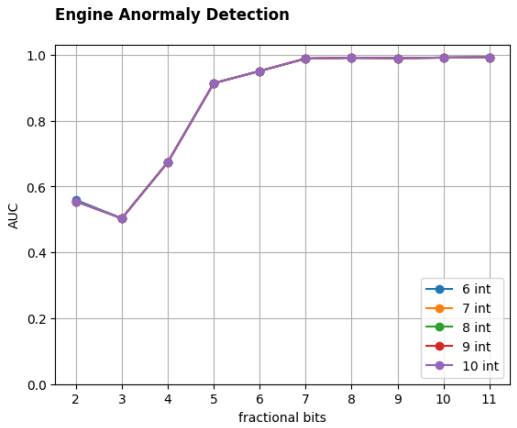
Current progress & future work

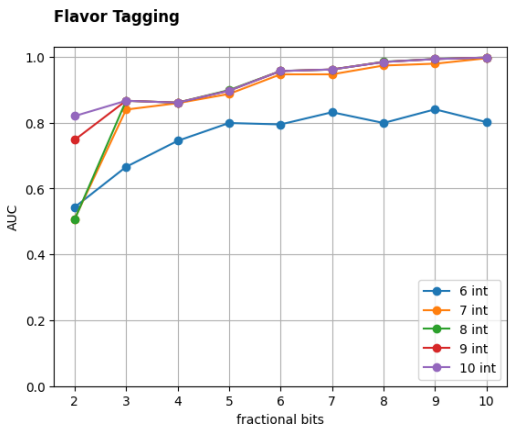
- Add Gaussian Sampling layer back
- Extend to MRAE

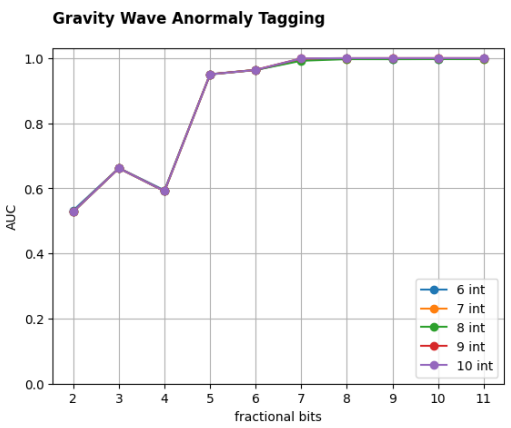


Projects Related to A3D3 in Our Group

- Transformer model for Neural signals
  - Trung Le, Ethan Jiang, Dennis Yin, Vidhi Desai, Elham E Khoda
  - Model done; HLS4ML Transformer layer done

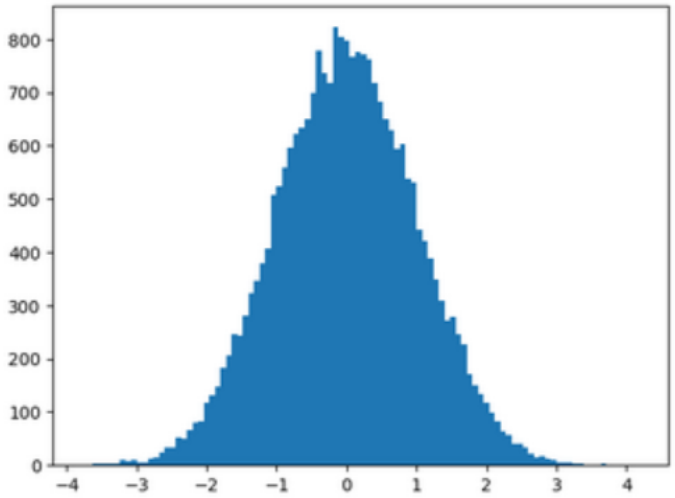






Projects related to A3D3 in Our Group

- Gaussian sampling in HLS4ML
  - Atharva Mattam
  - HLS code done
  - HLS4ML PR528



- Benchmarking HLS4ML vs SystemVerilog
  - Caroline Johnson & Waiz Khan
- Algorithm Processing Unit (APU)
  - Ethan Jiang, Bowen Zuo

Thank you for listening!

Questions?