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

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: Michael Nolan, Lauren Peterson, Leo Scholl, Amy Orsborn
Neural Processing Algorithms: Trung Le, Eli Shlizerman

OAC-2117997
National Science Foundation

Neural Data – Sleep Spindles

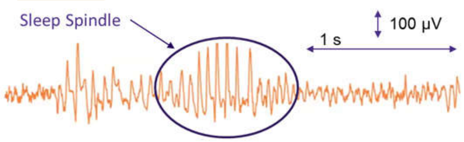


Fig. 1. Brain signal – Sleep Spindles[1]

Sleep Spindles Introduction [1]

- Rare low-frequency brain signals
- Primarily occur during sleep or rest
- Are believed to contribute to learning
- Lack of mechanistic understanding

Our goal

- Design and build a system that can help neuroscientists to understand the mechanism behind the theory

The Proposed System

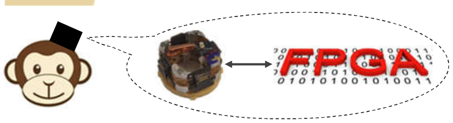


Fig. 2. Head-Mounted Device on Subject[2]

Head-Mounted Device components

- Headstage: Records brain signals from the subject
- Programmed FPGA: Processes brain signals and interacts with sleep spindles

Methods (HLS4ML & TinyML)

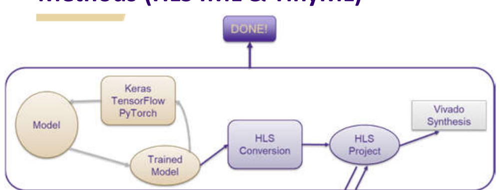
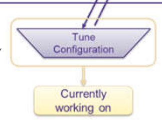


Fig. 3. HLS4ML Flow



The brain signals will be analyzed by a deep learning model, which will be pushed through the HSL4ML.

TinyML will help us to deploy the model on an ultra low power FPGA.

Baseline Deep Learning Model

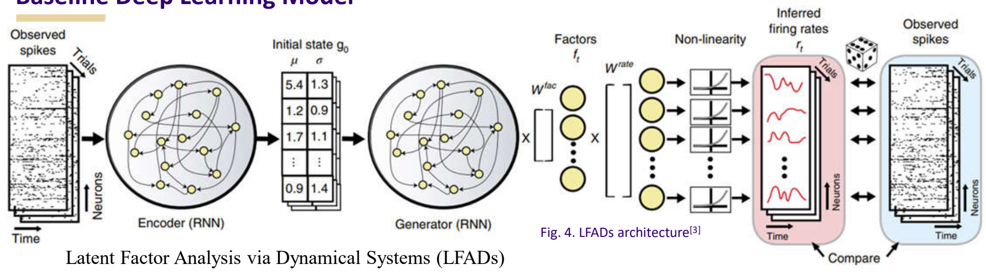


Fig. 4. LFADs architecture[3]

Latent Factor Analysis via Dynamical Systems (LFADs)

- RNN variational autoencoder (VAE) in tf.keras API
- Input: Neural spiking data
- Output: Firing Rates & LFADs Latent Factors

Modified LFADs Architecture

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

Fig. 8. Modified LFADs Model Summary

By removing the gaussian sampling layer, LFADs are converted to an autoencoder, which is easier to be pushed through HLS4ML flow.

Implementing the gaussian sampling layer in HLS4ML is also a current on-going project.

Performance Comparison per Trial

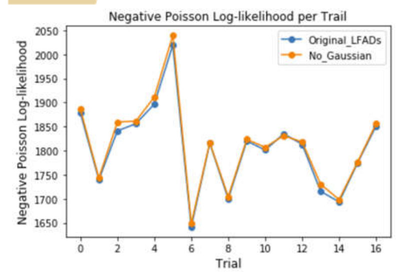


Fig. 9. Modified & Original LFADs Performance comparison

The negative poisson log-likelihood is the evaluation metric of the LFADs. Minimized negative poisson log-likelihood indicates an optimal performance.

For the same testing dataset, the numerical value of the negative poisson log-likelihood from the modified LFADs matches to the original LFADs, which indicates that removing the gaussian sampling from LFADs is acceptable.

HLS4ML Implementation

- Bidirectional layer: contains two GRU layers. One processes input data in the original sequence, the other processes input data in the reverse sequence. The output from two GRU layers will be concatenated.

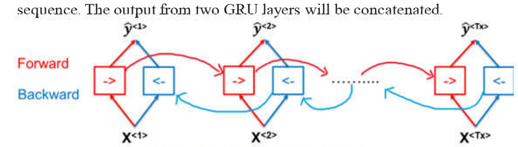


Fig. 10. Bidirectional Layer Structure

- GRU initial state: a new feature that allows the user to set specific values other than 0 to the initial state of HLS4ML GRU layer.

HLS Model Performance & FPGA Deployment

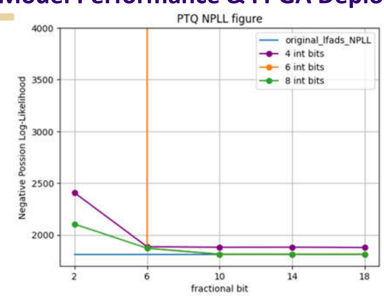


Fig. 11. HLS Model Performance

- When the integer bit is larger than 6 and the fractional bit is larger than 10, the HLS model performs the same as the floating-point model.
- Deployed onto Xilinx U55C with precision ap_fixed<16,6>, frequency at 200 MHz.
- Latency: 41.97 μs.
- Recourse utilization: 23.51% BRAM; 20.71% DSP; 5.79% FF; 12.64% LUT.

Future Work

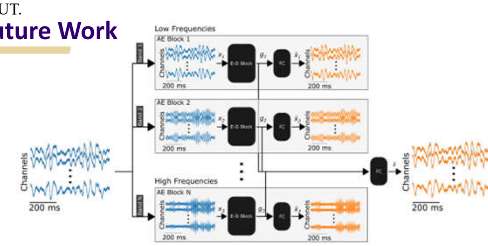


Fig. 12. Multi-block RNN Autoencoders (MRAE)[4]

- MRAE contains multiple LFADs-like models, which are separated to different sections to process neural data in different frequency.
- Deploy MRAE onto FPGA.

Acknowledgements & References

[1] Orsborn A, Shlizerman E, Dadarlat M. (2021) "Understanding & Interfacing with the brain: challenges and opportunities"
[2] Cartoon Monkey, FPGA Picture, accessed November 2021, <www.shutterstock.com>
[3] Pandarinath, Chethan, et al. "Inferring single-trial neural population dynamics using sequential auto-encoders." Nature methods 15.10 (2018): 805-815.
[4] Nolan, M., Pesaran, B., Shlizerman, E., & Orsborn, A. L. (2022). Multi-block RNN Autoencoders Enable Broadband ECoG Signal Reconstruction. bioRxiv, 2022-09.

Thanks to everyone in the HLS4ML community, Hardware team and the Neural teams.
This work is supported by National Science Foundation under OAC-2117997