ENABOL - Enabling Neural Backpropagation On-chip Learning for Edge AI Systems

Manuel Blanco **Valentin**

Fast Machine Learning for Science Conference 2025

01-05 September 2025

Zürich, CH 📍

On behalf of:

PhD Seda Ogrenci

Ryan Forelli

Ethan Gindlesperger

Alan Guo

Acknowledgments

Prof. Seda Ogrenci

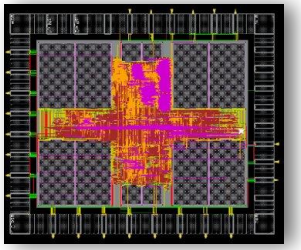
Ethan (Gindy) Gindlesperger

Houxuan (Alan) Guo

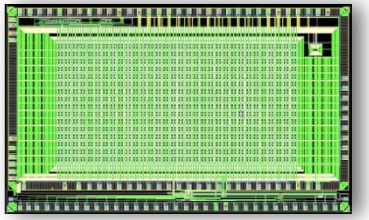
Ryan Forelli

Alexis Schuping

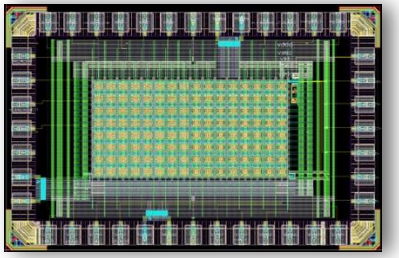
Rui Shi*



CryoAI
GF FD-SOI 22nm
FastML 2022



SPROCKET/AI-in-pixel
TSMC 65
CPAD 2022



CMS-28 / LHC
TSMC 28

ECON-T
TSMC 65

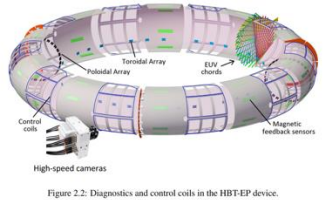


Figure 2.2: Diagnostics and control coils in the HBT-EP device.

Tokamak Fusion
Reactor Control
FPGA / ASIC

1. Why on edge training for NN accelerators?

Inference only vs Trainability:

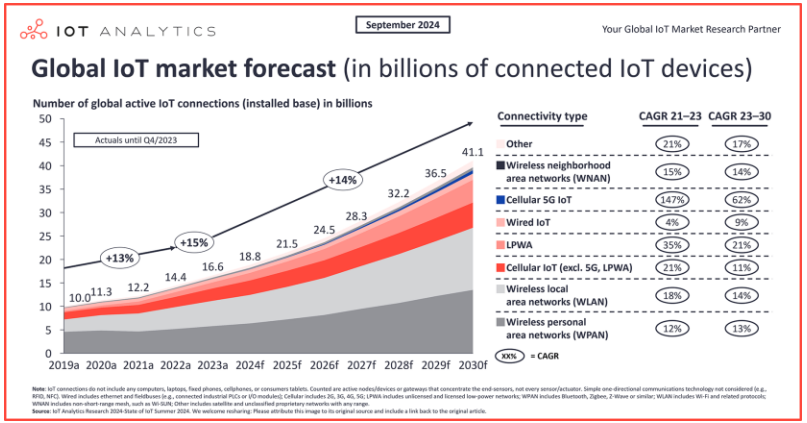
- Traditional approach: train on cloud + inference on edge
- models drift, data evolves, accuracy drops

On-edge training benefits:

- Latency → instant adaptation, no cloud delay
- Bandwidth → no raw data streaming to server
- Privacy → sensitive data never leaves device
- Personalization → user- or device-specific models

Constraints & challenges:

- Ultra-tight power & memory budgets
- No labels at the edge (unsupervised / self-supervised)



2. Canonical training algorithm → Backpropagation 101

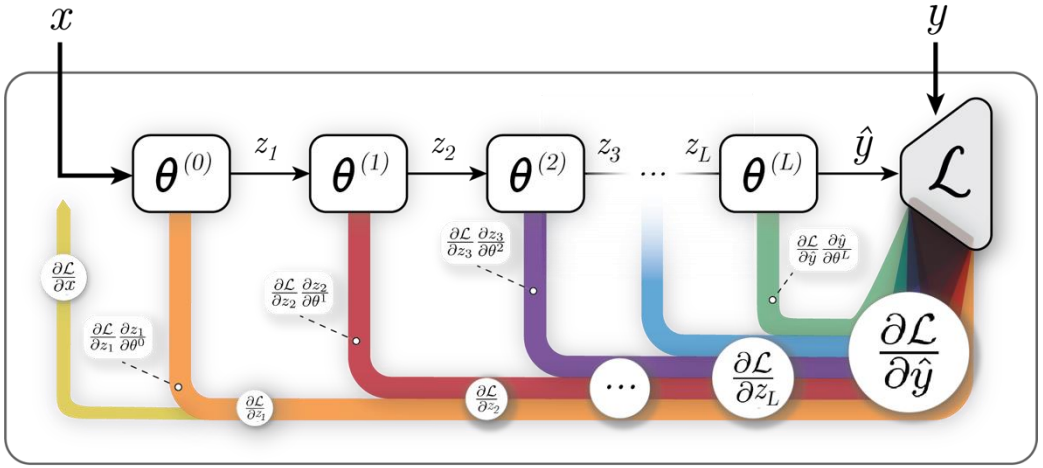
Neural Networks as Mappings

- Learnable function $f_\theta : \mathbb{R}^n \to \mathbb{R}^m$
- Loss $\mathcal{L}(f_\theta(x), y) \Rightarrow$ error
- Optimize θ to minimize total loss over data

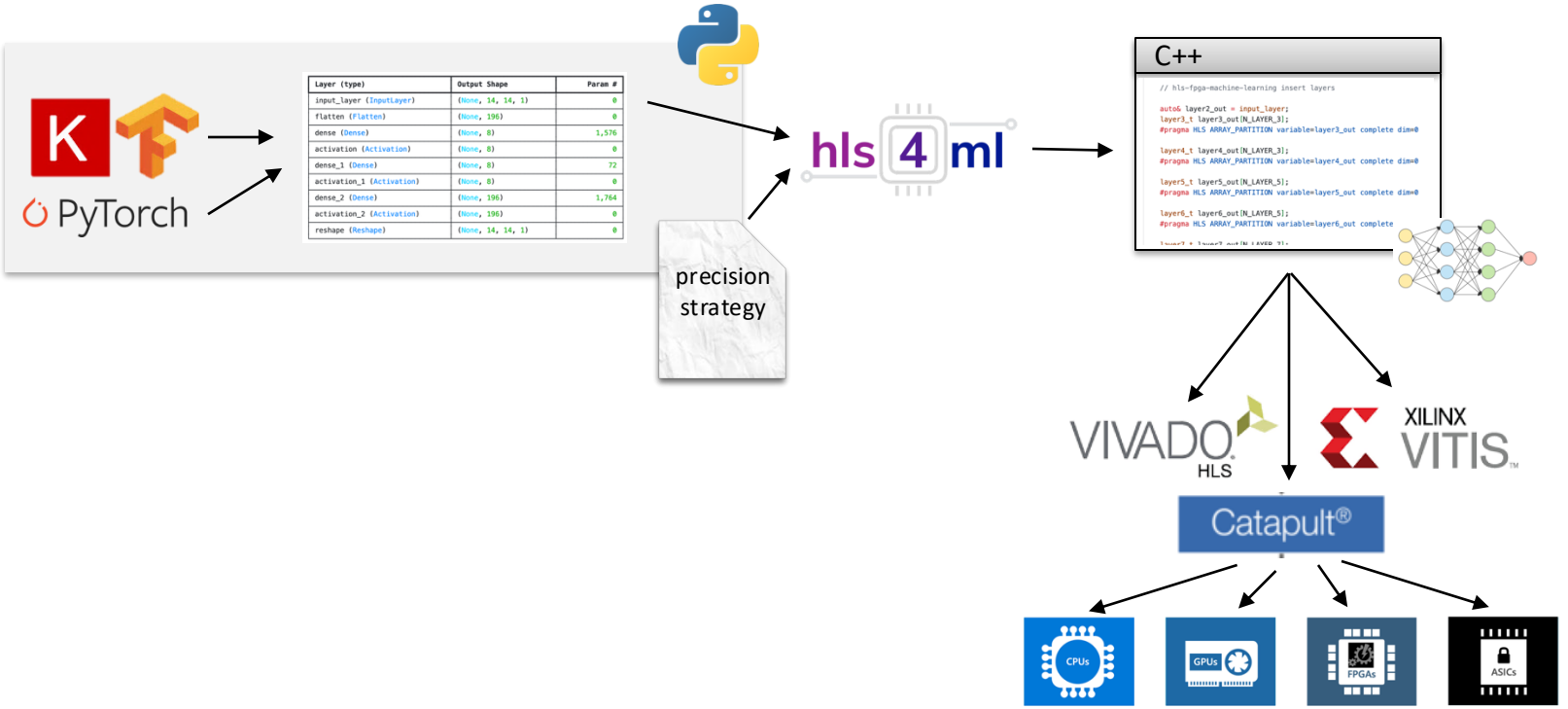
Backpropagation (BP)

- Uses chain rule to compute $\nabla_\theta \mathcal{L}$
- Propagates loss from output → input
- Updates weights: $\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}$

$$\frac{\partial \mathcal{L}}{\partial \theta^{(l)}} = \frac{\partial \mathcal{L}}{\partial \hat{y}} \cdot \prod_{k=l+1}^{L} \frac{\partial f^{(k)}}{\partial f^{(k-1)}} \cdot \frac{\partial f^{(l)}}{\partial \theta^{(l)}}$$



3. Our contribution: Extending hls4ml for autograd



*Mia Liu, CPAD 2023 @ Stanford - Link

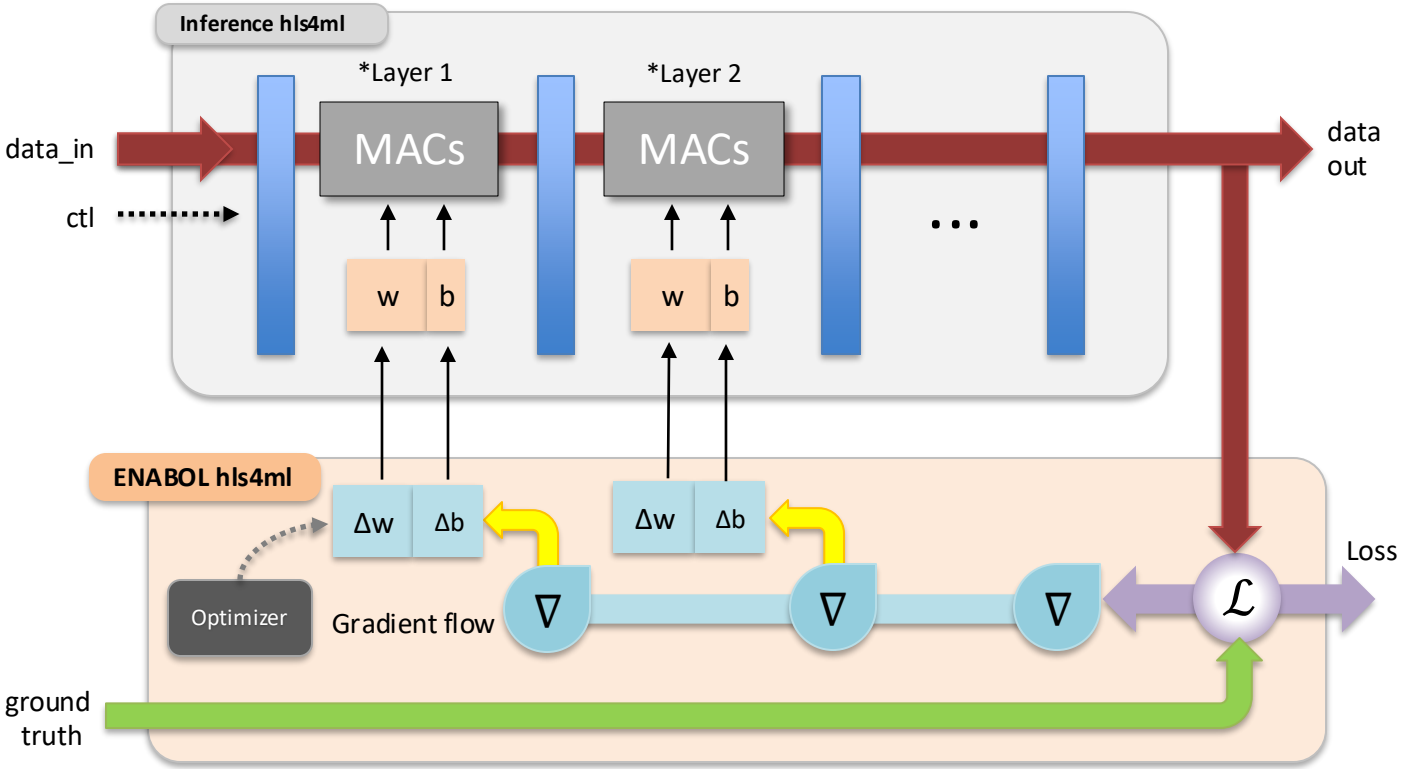
3. Our contribution: Extending hls4ml for autograd

Inference-only hls4ml

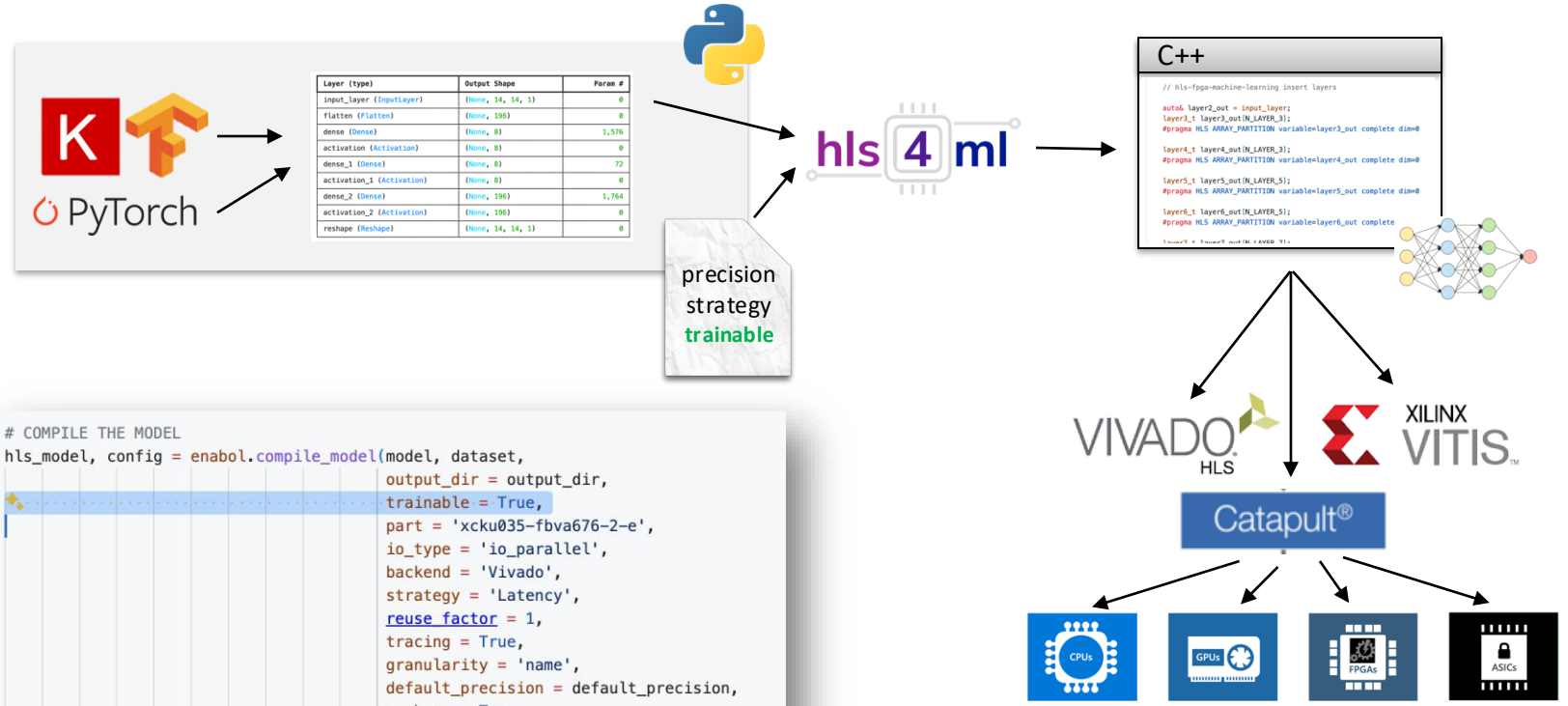
- Forward pass
- Pipelines
- Latency optimized
- DATAFLOW in → out

Enabol'd hls4ml

- Loss computation (provided ref)
- Gradients flow
- Multi-batch training
  - Accumulators
- Weight + bias updates
- Avoid overflow



3. Our contribution: Extending hls4ml for autograd



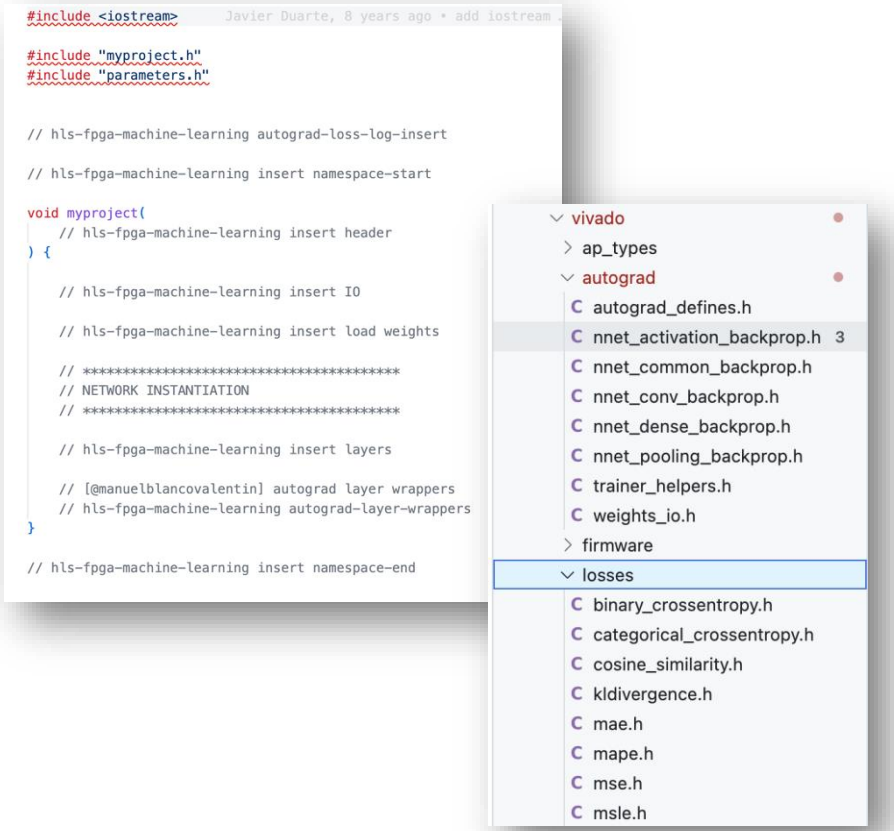

```
# COMPILE THE MODEL
hls_model, config = enabol.compile_model(model, dataset,
                                         output_dir = output_dir,
                                         trainable = True,
                                         part = 'xcku035-fbva676-2-e',
                                         io_type = 'io_parallel',
                                         backend = 'Vivado',
                                         strategy = 'Latency',
                                         reuse_factor = 1,
                                         tracing = True,
                                         granularity = 'name',
                                         default_precision = default_precision,
                                         verbose = True,
                                         overwrite = True)
```

*Mia Liu, CPAD 2023 @ Stanford - Link

4. Implementation details (II)

Under the hood

- **Autograd definitions**
  - Back pass for layers
  - Loss definitions
  - Helper functions for grads, batching, etc.
- **Vitis/Vivado Templates**
  - Modified to add autograd (if model is trainable)
  - Modified testbenching for training + logging
- **Extended vivado_writer.py to generate**:
  - Config structs w/ accumulators
  - Backward layer wrappers
  - Loss modules with correct grad scaling
- **Batch-training support**:
  - apply_update, batch_size, learning_rate



4. Implementation details (III)

What's supported?

Layers	Dense	Conv1d	Conv2d	Batch norm	Pooling1d	Pooling2d	Inverse Conv / Upsampling /	RNN / LSTM / GRU
	✅	⚠	✅	❌	⚠	✅	❌	❌

Losses	MSE	MAE	MAPE	MSLE	Crossentropy	KLDiv	Cosine-sim
	✅	✅	✅	✅	✅	⚠	⚠

Actv.	Linear	ReLU	Sigmoid	Tanh	Softmax	SeLU	leakyReLU
	✅	✅	✅	⚠	✅	⚠	⚠

Backend	Vivado/Vitis	Catapult	Oneapi	Quartus	Symbolic
	✅	❌	❌	❌	❌

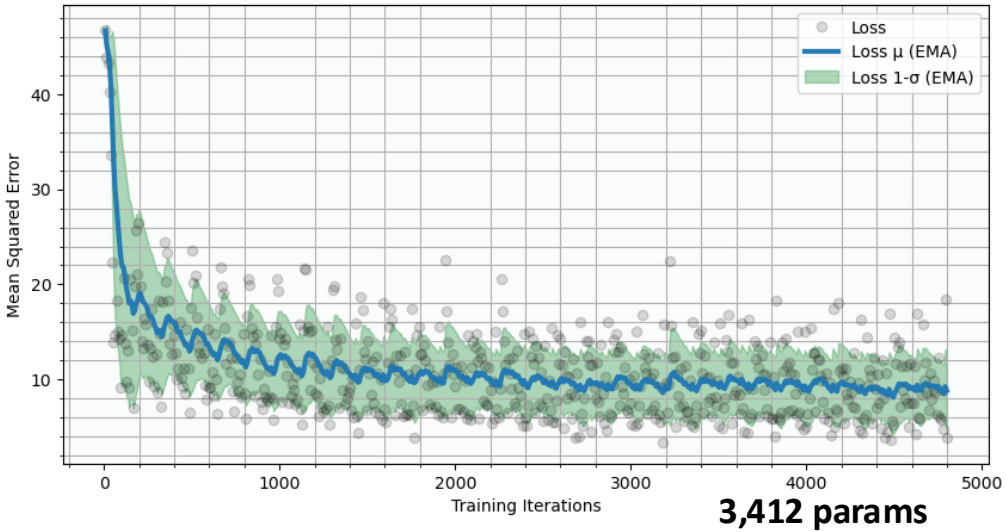
Optimz.	SGD	Adam	AdaDelta	AdaGrad	RMSprop
	✅	❌	❌	❌	❌

✅ Fully working & tested
⚠ Implementing but debugging/testing
❌ Not implemented yet

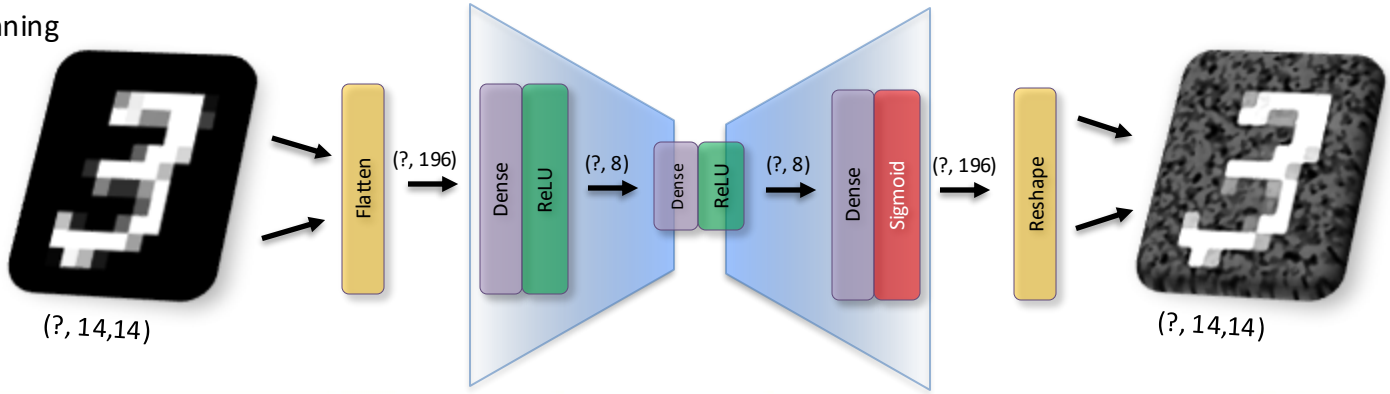
5. MNIST AE – Unsupervised

- **Dataset:**
  - 2x Downsampled images
  - Batch-size of 8
  - Not randomized
  - Only ~160 samples out of 70k (speed)

- **Training**
  - Vanilla SGD
  - Constant learning rate
  - No hyperparameter tunning
  - No batchnorm



3,412 params

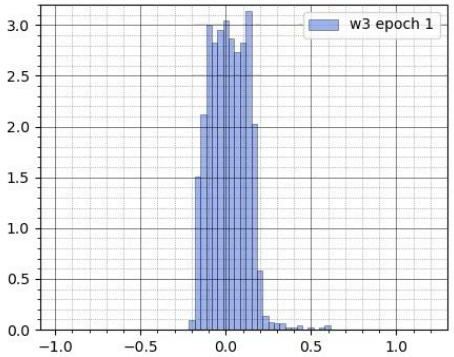


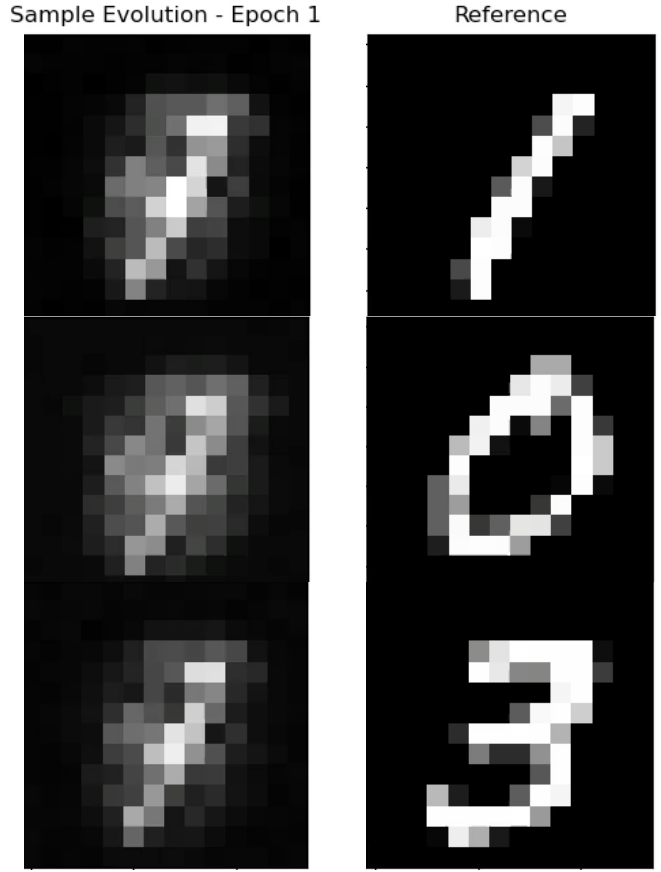
*Forward-stage: latency-opt
*Backward-stage: resource-opt

5. MNIST AE – Unsupervised

- **Dataset:**
  - 2x Downsampled images
  - Batch-size of 8
  - Not randomized
  - Only ~160 samples out of 70k (speed)

- **Training**
  - Vanilla SGD
  - Constant learning rate
  - No hyperparameter tunning
  - No batchnorm



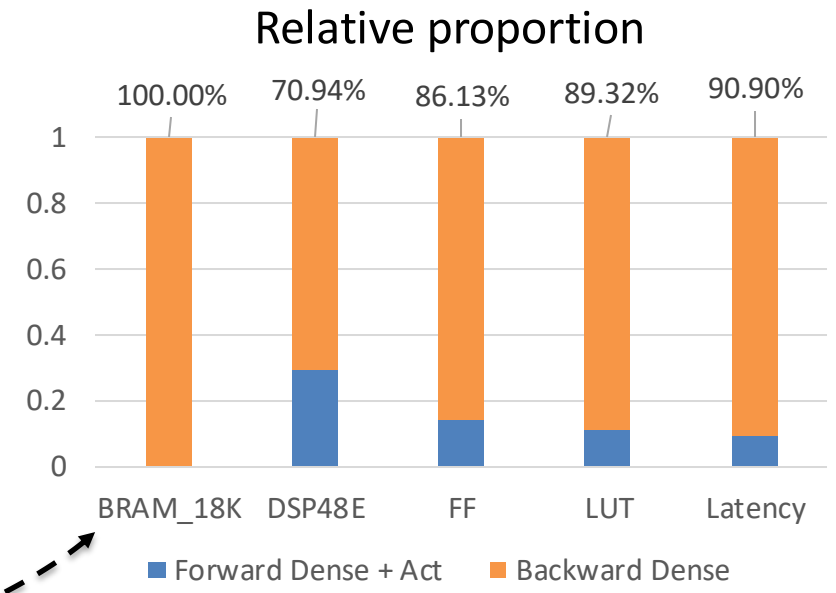


*Forward-stage: latency-opt
*Backward-stage: resource-opt

5. MNIST AE – Unsupervised (II)

	BRAM18K	DSP48E	FF	LUT	Reported Latency (cycles)*
Forward Dense (Inference)	0	13,110	84,717	86,234	55,512*
Forward Activation	0	0	0	320	320*
Backward Dense (Training)	12	**32,000**	**526,101**	**723,679**	**557,665***
Top (end-to-end)	115	18,858	908,901	1,475,998	788*

RF=1



- **Backprop overhead breakdown:**
  - ~4.1× DSPs
  - ~6.2× FFs
  - ~8.4× LUTs
  - BRAM shows up only in backprop

*Latency from csynth sub-reports; not equal to top due to pipelining.

6. Tokamak Fusion application – Unsupervised

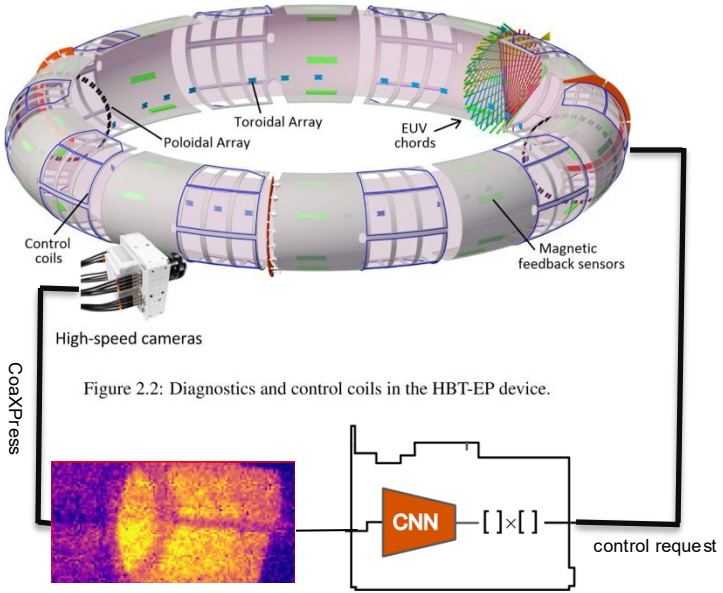
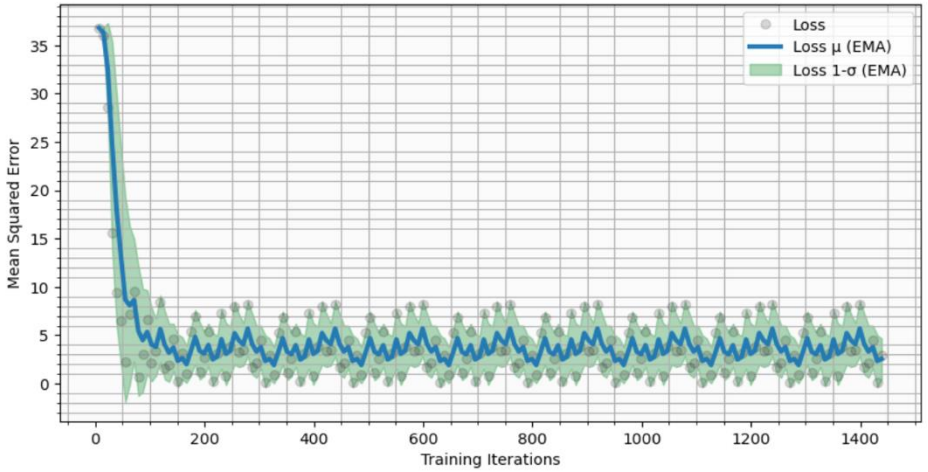


Figure 2.2: Diagnostics and control coils in the HBT-EP device.

- **Fusion Model for real-time MHD instability suppression**
  - Real-time feedback is critical to stabilize fusion plasmas where instabilities evolve on **microsecond timescales**.
  - Plasma conditions and diagnostic signals change over time. **Continuous retraining** directly on FPGA/edge hardware is needed to keep pace with changing data distributions.
- **Training results** **(tiny version, compressed input)**



6. Tokamak Fusion application – Unsupervised

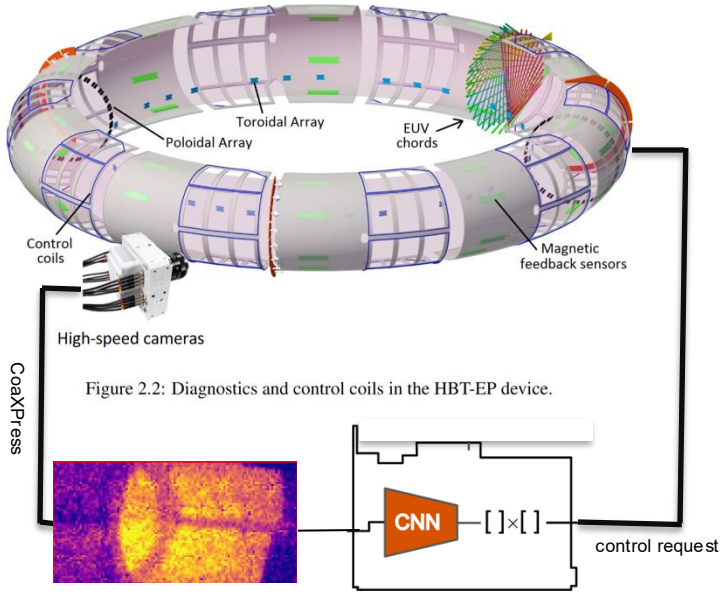


Figure 2.2: Diagnostics and control coils in the HBT-EP device.

- **Fusion Model for real-time MHD instability suppression**
  - Real-time feedback is critical to stabilize fusion plasmas where instabilities evolve on **microsecond timescales**.
  - Plasma conditions and diagnostic signals change over time. **Continuous retraining** directly on FPGA/edge hardware is needed to keep pace with changing data distributions.
- **Synthesis results** **(tiny version, compressed input)**

	BRAM1 8K	DSP48E	FF	LUT	Reported Latency (cycles)
Forward Dense (Inference)	0	129	19,428	47,769	1,442
Forward Activation	0	0	0	17,678	3
Backward Dense (Training)	40	2,260	198,884	481,940	30,136
Top (end-to-end)	40	2389	218,312	547387	31581

7. Conclusions

Takeouts

- First extension of **hls4ml** to support **on-chip backpropagation & training**
- Modular **autograd framework**: backward pass for layers, loss modules, gradient flow
- Preserves **usability**: minimal user change (trainable=True)
- Demonstrated **proof-of-concept training**
- Preliminary hardware results: clear resource/latency trade-offs between forward and backward passes
- Established **foundation for future hardware training research**

Future directions

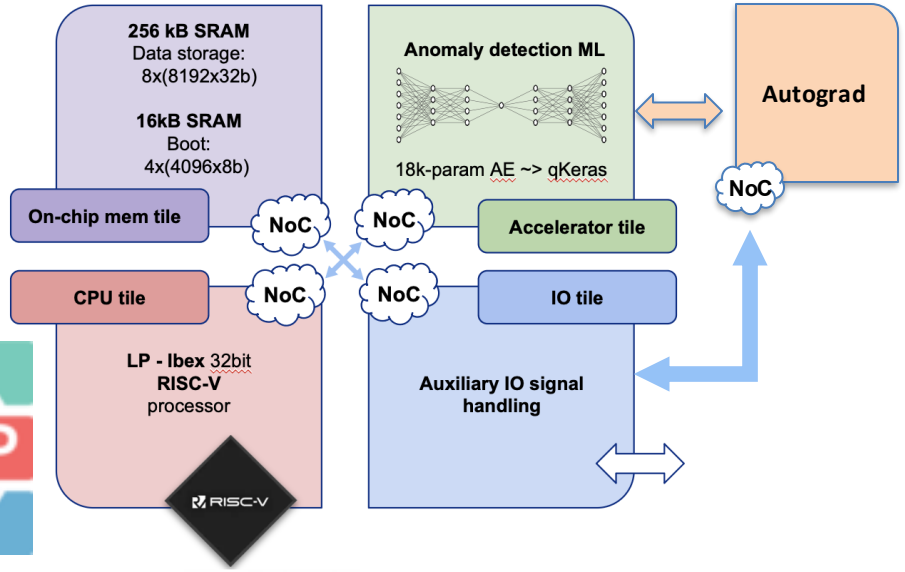
- **Next steps**
  - Add support for optimizers, layers, losses, activations, backends…
  - Extend more complex models (GANs, Attention, Transformers…)
  - Support inner-layer losses
- **Opens door for:**
  - Continual learning
  - Adaptive detectors/classifiers on edge
  - AE for anomaly detection on sensors (cryoAI) Encoding
  - Push toward **ASIC implementation** for **energy efficiency**
  - Integration with **ESP** platform for deployment

7. Conclusions

Takeouts

- First extension of **hls4ml** to support **on-chip backpropagation & training**
- Modular **autograd framework**: backward pass for layers, loss modules, gradient flow
- Preserves **usability**: minimal user change (trainable=True)
- Demonstrated **proof-of-concept training**
- Preliminary hardware results: clear resource/power/latency trade-offs between forward and backward passes
- Established **foundation for future hardware trai research**

Future directions



Thank you for your attention!
Questions?

PhD Cand. Manuel B Valentin


manuel.valentin@northwestern.edu


github.com/manuelblancovalentin