FPGA-Deployed Variational Autoencoder for Real-Time Soft X-Ray Electron Temperature Reconstruction in RFX-mod2

L. Orlandi, *Member, IEEE*, A. Rigoni Garola, *Member, IEEE*, L. Saccaro, *Member, IEEE*, P. Franz, M. Gobbin, L. Piron, R. Cavazzana

***Abstract*—RFX-mod2, an upgraded version of the RFX-mod Reversed-Field Pinch (RFP) device at Consorzio RFX, benefits significantly from integrating broader diagnostic data to enhance control performance. This study focuses on leveraging Soft X-Ray (SXR) measurements to characterize the internal electron temperature state in plasma discharges. A Variational Autoencoder (VAE) model was developed and trained using both synthetic and experimental datasets derived from SXR diagnostics of RFX-mod, enabling real-time denoising and low-dimensional representation of acquired signals. To deploy this model on FPGA hardware, quantization-aware training with the HGQ2 library was employed to optimize for latency and resource usage. The VAE model was then translated into an FPGA-compatible design using the HLS4ML framework. The resulting FPGA implementation achieves real-time performance with minimal loss in reconstruction accuracy compared to the floating-point baseline. Experimental results demonstrate that the VAE can consistently reconstruct electron temperature profiles, even under moderate to high levels of data loss, making it a robust solution for fusion plasma diagnostics and control systems. The model was successfully deployed on both Kria KV260 and Zynq 7020 FPGAs with efficient resource utilization and low inference latency.**

***Index Terms*—Neural networks, Machine learning, FPGA, Data imputation, Real-time, Reversed field pinch, RFX-mod2**

This paragraph of the first footnote will contain the date on which you submitted your paper for review. It will also contain support information, including sponsor and financial support acknowledgment. For example, "This work is co-funded by the European Union under the Next Generation EU initiative, ENI SpA and Università degli studi di Padova. This work has been carried out within the framework of the EUROfusion Consortium, funded by the European Union via the Euratom Research and Training Programme (Grant Agreement No. 101052200 - EUROfusion). Views and opinions expressed are however those of the authors only and do not necessarily reflect those of the European Union or the European Commission. Neither the European Union nor the European Commission can be held responsible for them."

L. Orlandi is with the Centre for Fusion Research, University of Padova, Padova, Italy and Consorzio RFX (CNR, ENEA, INFN, Università di Padova, Acciaierie Venete SpA), Corso Stati Uniti 4, 35127 Padova, Italy (e-mail: luca.orlandi.1@phd.unipd.it, luca.orlandi@igi.cnr.it).

A. Rigoni Garola is with Conzorzio RFX (CNR, ENEA, INFN, Università di Padova, Acciaierie Venete SpA), Corso Stati Uniti 4, 35127 Padova, Italy.

L. Saccaro is with the Centre for Fusion Research, University of Padova, Padova, Italy and Consorzio RFX (CNR, ENEA, INFN, Università di Padova, Acciaierie Venete SpA), Corso Stati Uniti 4, 35127 Padova, Italy.

P. Franz is with Conzorzio RFX (CNR, ENEA, INFN, Università di Padova, Acciaierie Venete SpA), Corso Stati Uniti 4, 35127 Padova, Italy.

M. Gobbin is with Conzorzio RFX (CNR, ENEA, INFN, Università di Padova, Acciaierie Venete SpA), Corso Stati Uniti 4, 35127 Padova, Italy.

L. Piron is with the Department of Physics and Astronomy, University of Padova, Padova, Italy and with Conzorzio RFX (CNR, ENEA, INFN, Università di Padova, Acciaierie Venete SpA), Corso Stati Uniti 4, 35127 Padova, Italy.

R. Cavazzana is with Conzorzio RFX (CNR, ENEA, INFN, Università di Padova, Acciaierie Venete SpA), Corso Stati Uniti 4, 35127 Padova, Italy.

I. Introduction

NUCLEAR fusion experiments, such as RFX-mod [1], generate a significant amount of data, making them ideal test beds for applying machine learning routines. Additionally, one of the critical challenges in magnetic confinement fusion is controlling plasma behavior during discharge. Historically, this has been accomplished through control loops that rely on magnetic sensors in both standard configurations [2] and more advanced setups using neural networks (NNs) [3].

The end goal of this study is to leverage the wide array of plasma diagnostics available in such experiments, including electron temperature diagnostics [4], ion temperature diagnostics [5], and density diagnostics [6], to gain a deeper understanding of plasma state and develop more effective control procedures for the experiment itself [7]. In this context, this work focuses on a particular diagnostic technique, leveraging soft X-ray (SXR) measurements [8], to determine the electron temperature. The goal is to find a way to condense the information from these diagnostics into a lower-dimensional embedding (latent space), which can then be utilized for control purposes.

The SXR diagnostic was selected due to its high time resolution, enabling the recording of numerous electron temperature profiles. This approach aims to capture a wide distribution of possible cases, given that RFX-mod discharges typically last around 0.5 seconds on average.

The paper is structured as follows: Section II provides an overview of the diagnostic setup used for measurements; Section III introduces the characteristics of the dataset employed; Section IV discusses the neural network architecture and procedures implemented for its development; Section V presents the results of the analysis; Section VI offers a discussion on the presented results; and finally, Section VII draws the conclusions.

II. Soft X-ray diagnostic

The diagnostic used for this study is called Diagnostic Soft X-ray 3-array (DSX3), which, as the name suggests, consists of three different arrays of photodiodes. A complete

description of the camera specifications can be found in [9], while the schematics for the lines of sight are reported in [10]. The electron temperature is computed from the raw signal using the *double-foil* technique. This technique involves comparing the raw brightness from two lines of sight that view nearly the same section of the plasma, after filtering them through two beryllium (Be) filters of different thicknesses, in this case 42 and 88 μm. Since the two lines of sight observe the same plasma section, the only differing parameter influencing the brightness is the Be filter thickness. Therefore, it is possible to determine the electron temperature [10].

III. Dataset

The dataset comprises electron temperature profiles captured during RFX-mod's high plasma current campaign up to 2015. It consists of 328 different pulses, encompassing approximately 70,000 electron temperature profiles. The primary plasma parameters are as follows: the plasma current (I_p) ranges from 0.75 MA to 2 MA; the electron density, probed through the interferometer, varies between 0.1×10^{20} m^{-3} and 0.8×10^{20} m^{-3}; and the electron temperature falls within the range of 100 eV to 2000 eV. A more detailed analysis of dataset distributions and parameter correlations is provided in [11]. This dataset has been developed specifically for use with machine learning, covering a broad parameter space found in RFX-mod and anticipated in the new upgraded machine RFX-mod2. The choice of this type of data is due to the diagnostic's high temporal resolution, that can reach up to 0.1 ms [10], which allows it to provide a large amount of data necessary to properly train the machine learning model.

IV. Neural network

The neural network (NN) used for this work is a variational auto-encoder (VAE) [12], based on previous works [7] and further expanded in [11]. This time, the framework has been updated to use the `Tensorflow` Python library as a backend for `Keras`, facilitating quantization-aware training (QAT). The QAT process is essential for deploying the model on FPGA hardware to minimize latency, as it converts the model from float32 to fixed point precision. The model is then transferred from GPU architecture to FPGA using the hls4ml framework [13]. This library converts traditional machine learning models into high-level synthesis (HLS) code, simplifying the deployment of the FPGA implementation.

The implemented NN, being a VAE, has a symmetrical structure. It consists of four layers in both the encoder and decoder parts, with dimensionality reduction occurring as data is mapped to the latent space.

A. Training

As previously stated, the model is a VAE in its β variation, allowing us to tune the amount of Bayesian noise injected into the network. The basis for a β-VAE is similar to that of a normal VAE, with the loss term comprising two main components: the usual reconstruction loss (provided by the mean squared error, MSE) and the Kullback-Leibler (KL) divergence, which measures the distance between the encoded distribution and a standard normal distribution. The mathematical formalism for these components is as follows:

$$\mathcal{L}_{MSE} = \frac{1}{N}\sum_{i=1}^{N}(x_i - y_i)^2 \tag{1}$$

where the sum is over each point of the electron temperature profile, which in our case has 17 entries ($N = 17$).

$$\mathcal{L}_{KL} = -\frac{1}{2}\sum_{j=1}^{D}(1 + \log(\sigma_j^2) - \mu_j^2 - \exp(\log(\sigma_j^2))) \tag{2}$$

Here, the sum is over the dimensions of the latent space ($D = 4$). The KL divergence term forces the encoded distribution to resemble a standard normal distribution by pushing the variance (σ) close to one and the mean (μ) close to zero.

The total loss is given by:

$$\mathcal{L} = \mathcal{L}_{MSE} + \beta\mathcal{L}_{KL} \tag{3}$$

where $\beta \in [0, 1]$. When β is close to zero, the network behaves more like a standard autoencoder. When β equals one, it behaves as a standard VAE.

Two different training approaches were tried:

- **Simple Beta Annealing**: The β-VAE starts from a value of 10^{-8} and gradually increases over 230 epochs to reach a value of 1 [14].
- **Cyclical Beta Annealing**: The value of β is cyclically varied between 10^{-8} and 1, with each cycle spanning approximately 200 epochs [15]. This approach allows the VAE to learn a precise representation of the electron temperature profile while maintaining relevant variational statistics.

In Figure 1, the dynamics of the overall loss, MSE, and KL divergence are shown. The y-axis is set to a logarithmic scale for better readability. The cyclical behavior of the β-VAE parameter is evident from the loss dynamics.

B. Quantization Aware Training

Once the model has been trained with a standard routine using `tensorflow`, it is necessary to switch to another framework that supports Quantization-Aware Training (QAT). For this purpose, the `HGQ2` library [16] was utilized. This library implements an automatic bit-width optimization algorithm based on gradients, allowing for heterogeneous quantization at arbitrary granularity.

The core of the QAT procedure involves quantizing weights, biases, activations, inputs, and outputs of the model. The `QuantizerConfig` class from the library was used to manage these quantizations.

Table I summarizes the quantizer configuration for each role in the network:

- Input/Output
- Weights
- Biases

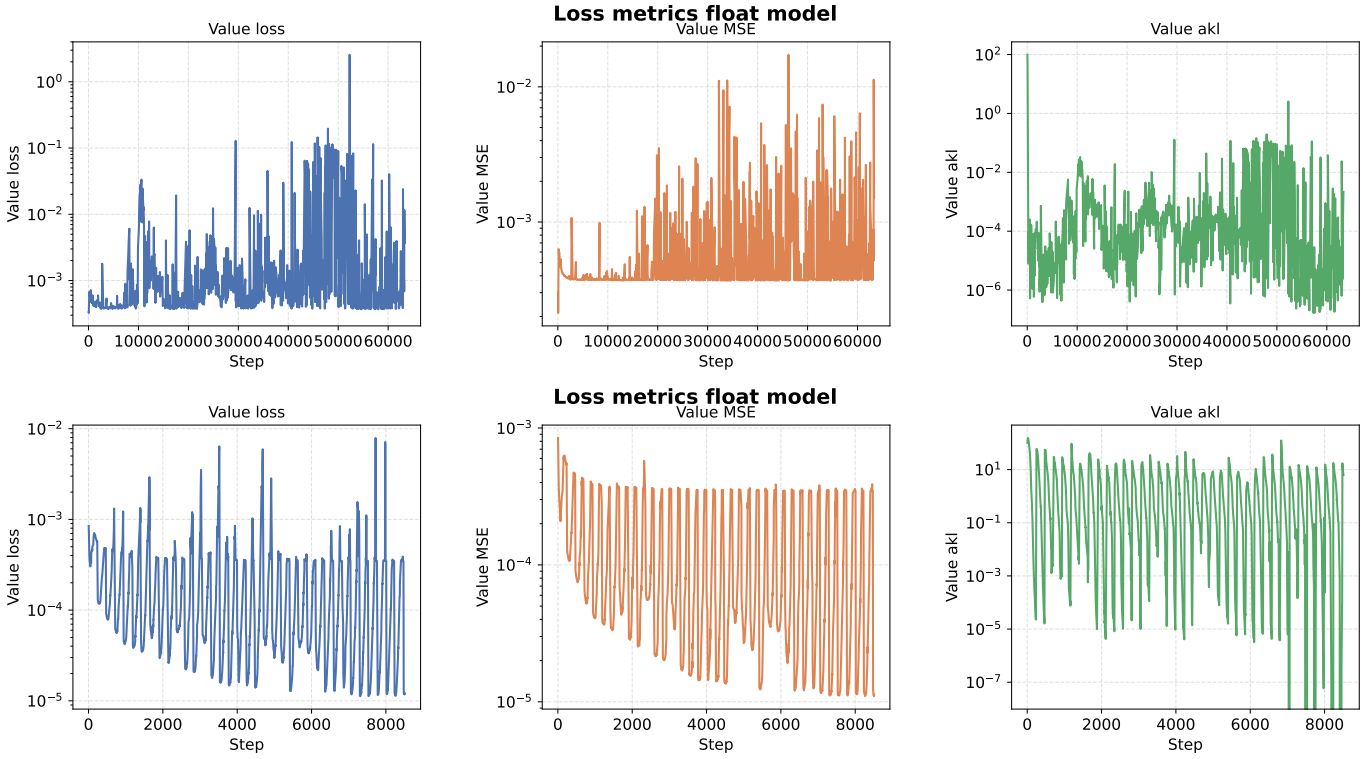

Fig. 1. Training and validation loss dynamics for the float model with two beta procedures. The top three panels correspond to the simple beta annealing while the bottom three correspond to a cyclical beta annealing. From left to right: overall loss, mean squared error (MSE), and Kullback-Leibler (KL) divergence. The y-axis is on a log scale for better readability.

- Activations

The input and output quantizers share an identical configuration and use the `KIF` parametrization, splitting the fixed-point format into integer and fractional components. Since the inputs are always non-negative (plasma temperature), the sign bit is disabled. Additionally, because the inputs are normalized between 0 and 1, the integer part is constrained to one bit (i_c=`MinMax(1,1)`). The fractional precision starts at 20 bits with a minimum constraint of 16 bits.

The weight quantizer uses `KBI` parametrization, where the total number of bits (excluding sign) is initialized to 12 with 3 integer bits. The sign bit is enabled, making it a signed 13-bit representation. Heterogeneous quantization is applied along axes `(0,1)`, meaning each combination of output channel and kernel spatial position is assigned an independently learned bit width. This allows the training process to focus bit budget on the most informative weight groups while pruning precision elsewhere.

The bias quantizer shares the same parametrization type and initial values as the weight quantizer but applies heterogeneous quantization only along axis `(0,)`, corresponding to the output channel dimension. This aligns with the standard practice of maintaining one bias value per output feature map, where per-channel precision tuning is sufficient.

The activation quantizer also uses `KIF` parametrization, similar to input and output quantizers, but enables the sign bit. It starts with 3 integer bits and 10 fractional bits. Heterogeneous quantization is applied along axis `(1,)`, enabling per-channel fractional precision adaptation across feature map channels.

All four quantizers share the same overflow mode (`SAT_SYM`) and rounding mode (`S_RND`). Symmetric saturation clips values symmetrically around zero, avoiding asymmetric clipping artifacts and simplifying hardware implementation. Stochastic rounding introduces a randomized component to the quantization error during training, preserving gradient information statistically at low bit widths [17].

The training procedure was implemented under the same conditions as the non-quantized model, using both cyclical annealing of the `beta_vae` parameter and a fixed low value. Both approaches produced similar results, with the fixed value training delivering slightly better performance.

A scheduler for the `beta_zero` parameter was also implemented to weigh the loss term related to quantization. The optimal procedure starts with `beta_zero` set to 10^{-7} and drops it to 10^{-10} when the number of Effective Bit Operations (EBOPs) reaches 5×10^5, a threshold deemed suitable for deployment on the `Kria K26` FPGA[1].

This training schedule allows for aggressive reduction of EBOPs at the beginning, without significantly impacting reconstruction capability. Once EBOPs reach the selected value,

[1]This board has been selected as the primary subject of this study due to its availability for immediate testing in the laboratory setting.

TABLE I
PER-ROLE HGQ QUANTIZER CONFIGURATION

Role	q_type	k	i_0 / f_0	Overflow	Round	Bit Constraints	Het. Axis
Input / Output	kif	0	1 / 20	SAT_SYM	S_RND	i_c:[1,1], f_c:≥16	—
Weight	kbi	1	b_0=12, i_0=3	SAT_SYM	S_RND	—	(0, 1)
Bias	kbi	1	b_0=12, i_0=3	SAT_SYM	S_RND	—	(0,)
Activation	kif	1	3 / 10	SAT_SYM	S_RND	—	(1,)

k: sign bit (0 = unsigned); i_0: initial integer bits; f_0: initial fractional bits (kif only).
b_0: total bits excl. sign (kbi only); total width = $k + b_0$ bits.
i_c: constraint on integer bits; f_c: constraint on fractional bits. "—" = default constraint.
Het. Axis: axes along which bit-widths are assigned heterogeneously. "—" = homogeneous.

both reconstruction loss and EBOPs loss are reduced to a similar level, enabling a balanced trade-off between accuracy and efficiency. The dynamics of this procedure can be observed in Figure 2. For comparison, the reconstructions for some pulses are also shown in Figure 3.

Another approach was explored using the HGQ2 library, which leverages its pruning capabilities. In this context, pruning refers to setting certain weights to zero, effectively removing unnecessary parameters from the model. A larger initial model with a layer geometry of [20, 16, 12, 10, 6] and a scale parameter of two (doubling the declared layers) was trained from the start using Quantization-Aware Training (QAT). The training had an epoch limit of 100,000 to achieve optimal quantization and pruning.

The initial model contained 210,048 parameters, while the final pruned model had only 1,123 parameters distributed across 10 available layers. This means that approximately 99.47% of the parameters were set to zero through pruning. The starting model was intentionally large to allow the training procedure to identify the most optimal configuration and resource usage via trial and error. Reconstruction distances computed on electron temperature profiles are shown in Table II and compared with previous models.

C. FPGA Deployment

Once the QAT procedure is completed, the model is almost ready to be translated into code that can be implemented on the FPGA architecture. To achieve this, we have used the HLS4ML [13] framework.

HLS4ML is designed to accept a trained model, in this case, a Keras one, and parse its architecture layer by layer. Each layer is then mapped to an internal HLS4ML representation. At this stage, the quantization procedure performed earlier becomes critical because HLS4ML can read the per-weight bit widths set via HGQ directly from the model, propagating them into the generated firmware.

HLS4ML generates a self-contained HLS C++ project consisting of a set of `.cpp` and `.h` files that implement the network using arbitrary-precision types. The code extensively uses HLS pragmas to provide precise micro-architectural directives for the downstream synthesis tool. Each layer becomes a templated C++ function, and the entire forward pass is a composition of these functions with explicit dataflow.

The generated project is then handed off to Vitis HLS, which performs the following operations:

- C Simulation: Functional verification of the HLS model against the original.
- C Synthesis: Translation of the HLS C++ into RTL, estimating latency, initiation interval (II), and resource usage such as Look Up Tables, Flip Flops, Digital Signal Processors, and Block RAMs (LUTs, FFs, DSPs, BRAMs).
- Co-simulation: RTL-level verification against the C testbench.
- Export: Packaging the RTL as an IP block.

HLS4ML can invoke these steps programmatically via its `build()` method.

V. RESULTS

The floating-point baseline VAE was trained on the high plasma current dataset of RFX-mod for $\sim$ 8000 epochs, achieving a reconstruction loss of 1.10×10^{-5} and a KL-divergence term of 8.78 with a β_{VAE} of 8×10^{-8} on the validation set. After applying quantization-aware training via HGQ2, the model converged in $\sim$ 26k epochs with a final total loss of 6.91×10^{-5}. A complete summary of the NN structure with the bit width of the various layers is provided in Table III.

To make the interpretation of the data clearer, the average bit width for the various layers has also been reported in Figure 4.

The quantized Keras model was exported to an HLS design using hls4ml. Co-simulation confirmed functional equivalence between the HLS model and its Keras counterpart, with no appreciable difference between the two models on the test samples, as shown in Figure 5.

A. Implementation on Kria K26

Synthesis and implementation were carried out in Vivado 2024.1 targeting the FPGA `Kria K26` at a clock frequency of 100 MHz. The implemented design consumed 33,323 LUTs (28.45%), 12,778 flip-flops (5.46%), and 168 DSPs (13.46%), with no BRAMs utilized, meaning that the entire design fits in LUT-based logic without needing memory tiles.

End-to-end inference latency for a single input sample of size (17,) was measured at 0.140 µs, yielding a theoretical throughput of approximately 7.1 M samples/s. For comparison, a GPU baseline running the floating-point model on an

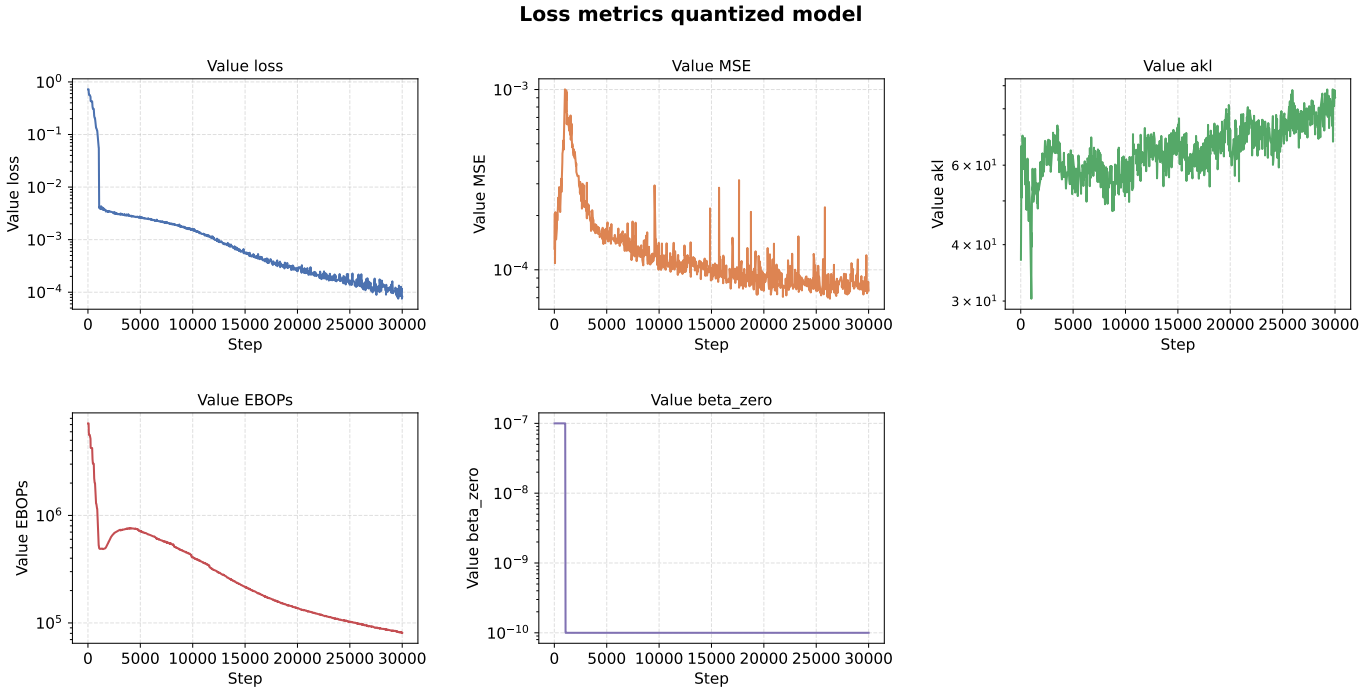

Fig. 2. Dynamics of the training and validation loss for the quantization-aware training procedure. From the upper left corner, the total loss (sum of standard VAE loss and EBOPs term) is shown, followed by the mean squared error as a metric of reconstruction accuracy, and then the Kullback-Leibler divergence. On the second row, the dynamics of EBOPs are depicted along with the point where `beta_zero` drops to prioritize model reconstruction ability when EBOPs reach 5×10^5.

TABLE II
DISTANCE WITH STANDARD DEVIATION FOR ELECTRON TEMPERATURE PROFILES ACROSS THREE MODELS

Coordinate	**Model 1**[a] $d \pm \sigma$ [eV]	**Model 2**[b] $d \pm \sigma$ [eV]	**Model 3**[c] $d \pm \sigma$ [eV]
1	43.65 ± 26.53	60.07 ± 65.01	49.68 ± 56.97
2	59.22 ± 31.35	73.19 ± 59.95	64.89 ± 54.44
3	74.13 ± 36.76	86.76 ± 61.44	81.61 ± 60.17
4	88.14 ± 41.69	98.17 ± 60.51	93.15 ± 64.02
5	102.70 ± 47.41	107.51 ± 61.77	108.47 ± 69.71
6	119.69 ± 53.18	121.44 ± 64.94	125.43 ± 76.93
7	140.88 ± 61.87	131.55 ± 68.71	145.02 ± 85.24

[a] float32 model.
[b] quantization aware training of the float32 model.
[c] quantization aware training from scratch with larger model.

TABLE III
HGQ2 BIT-WIDTH REPORT

Layer	**Shape**	**W bits**	**W prune%**	**Bias bits**	**EBOPs**
enc_dense_0	17→120	3.77	71.2%	5.99	37,260
enc_dense_1	120→96	2.17	93.1%	4.03	12,145
enc_dense_2	96→72	2.39	93.3%	2.58	7,399
enc_dense_3	72→24	2.83	0.0%	2.72	4,444
enc_mu_logvar	24→24	1.95	81.8%	1.10	1,905
dec_dense_0	12→24	4.94	74.7%	1.96	3,095
dec_dense_1	24→72	3.37	91.7%	5.99	4,204
dec_dense_2	72→96	2.66	94.8%	5.00	7,106
dec_dense_3	96→120	2.29	94.9%	5.14	9,643
dec_output	120→17	2.96	76.9%	7.61	9,689

W bits: average number of bits assigned to the active weights of that layer.
W prune%: fraction of weights in the layer that are zero.
Bias bits: average number of bits assigned to the active bias.

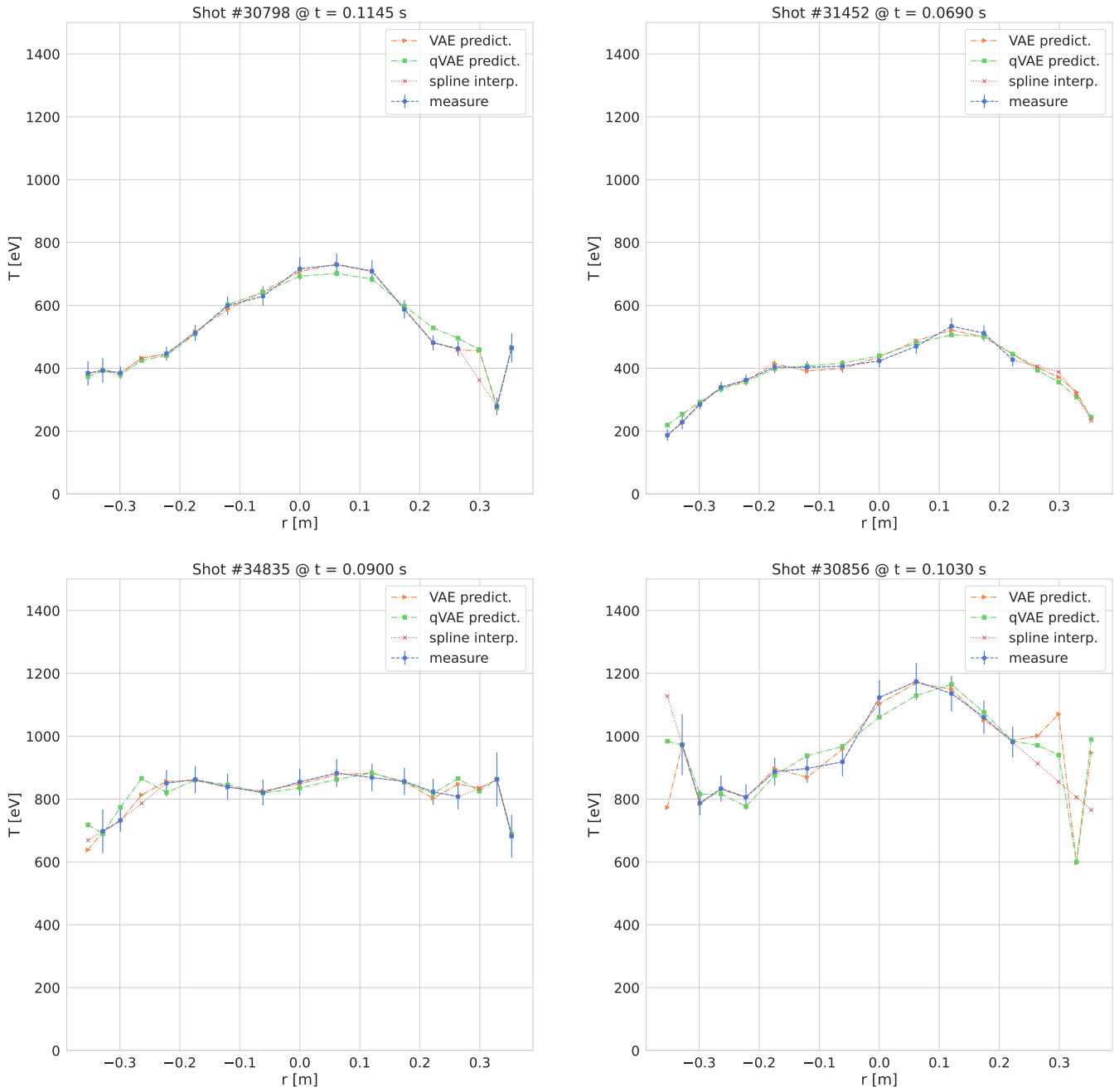

Fig. 3. Different profiles taken at different time instants for various RFX-mod shots. The measured temperature profiles are shown with blue circles, while the models' reconstructions are shown with orange triangles (float32) and green squares (quantized version). The red dotted line with crosses represents an interpolation via a B-spline.

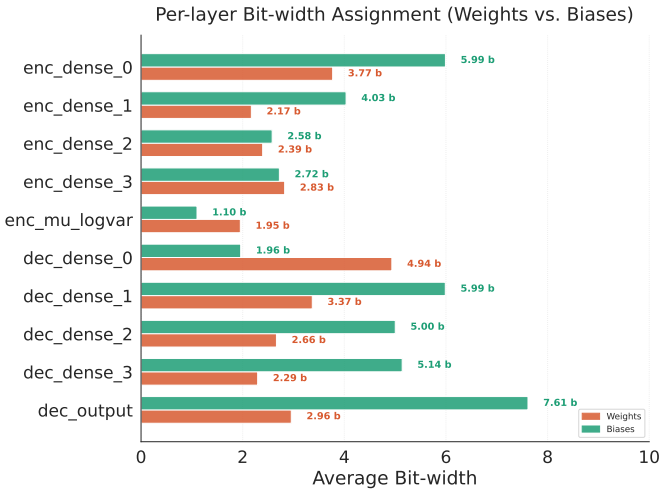

Fig. 4. Histogram of the bit width per layer in the QAT network.

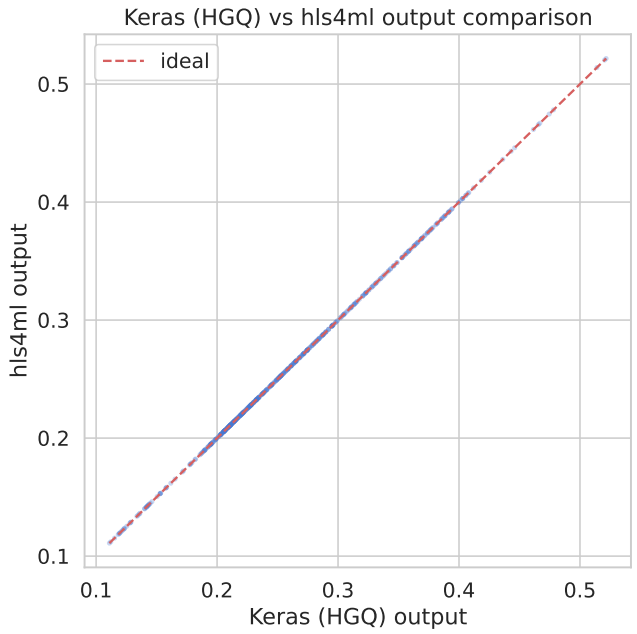

Fig. 5. Scatter plot of the reconstructions made on the same samples by the quantized model via HGQ and its synthesis produced by hls4ml.

NVIDIA^{TM} Tesla P40 has an end-to-end inference latency for a single input of 310 µs, corresponding to a throughput of approximately 3.2 k samples/s. It is important to note that the estimates for the GPU figures could be influenced by the time it takes for communication with the CPU.

It is evident that the `Kria K26` can easily handle this model, so an additional implementation targeting the smaller `Zynq 7000` board was also performed, as explained in the following subsection.

B. Implementation on Zynq 7000

A second implementation of the model was performed on a smaller board, specifically targeting the `Zynq 7000` FPGA. The synthesis followed the same steps as those used for the `Kria K26` board. Below are the results of this additional implementation.

The design consumed 39,144 LUTs (73.58%) in total and also used 58,117 flip-flops (54.62%), 188 DSPs (85.45%), and none of the available BRAM tiles. This is a good result, indicating that even with a smaller device, the model fits perfectly. The end-to-end inference latency was measured at 0.792 µs, yielding a theoretical throughput of approximately 1.3 M samples/s. The comparison with the GPU remains the same as in the previous section.

For completeness, a graph summarizing the resource usage of the models is provided in Figure 6.

VI. Discussion

The results demonstrate that quantization-aware training with HGQ2 introduces only a modest accuracy penalty compared to the floating-point baseline. This confirms that the heterogeneous per-layer bit-width strategy is well-suited for VAE architectures, where the bottleneck latent space typically demands higher precision than the outer layers of the encoder and decoder. The automatic bit-width selection provided by HGQ2 avoids manual tuning and yields a compact model.

The computed latency and throughput appear sufficient to implement this reconstruction in real time and also to inject information in the control loop, which operates at a frequency of 10 kHz. This suggests that the GPU could also be a viable candidate for the implementation. However, the figures presented in Fig 7 do not account for the data transfer time.

Neither FPGA is fully saturated by the model, leaving room for the boards to perform diagnostic merging, one of the foreseen future steps of this work. The low DSP utilization — only 13.46% — indicates that the current implementation fits well within the chosen FPGA (`Kria K26`) without overusing arithmetic resources. Furthermore, the relatively low number of utilized LUTs suggests efficient use of logic resources. Both of these results are a consequence of the high quantization applied to weights and biases, which also explains why the model occupies less than 15% of the overall resources of the `Kria K26` board and, notably, why it can be deployed on a considerably smaller board such as the `Zynq 7000`.

VII. Conclusions

In this work, a comprehensive design flow for deploying a quantization-aware trained Variational Autoencoder (VAE) on an FPGA was presented. The process spans from model training in Keras to heterogeneous quantization with HGQ2, high-level synthesis via hls4ml, and physical implementation in Vivado.

The resulting hardware design achieves a latency of 0.140 µs per inference on the `Kria K26` board and 0.792 µs on the `Zynq 7000` board, with resource utilization well within the capacity of the target devices. The absolute reconstruction

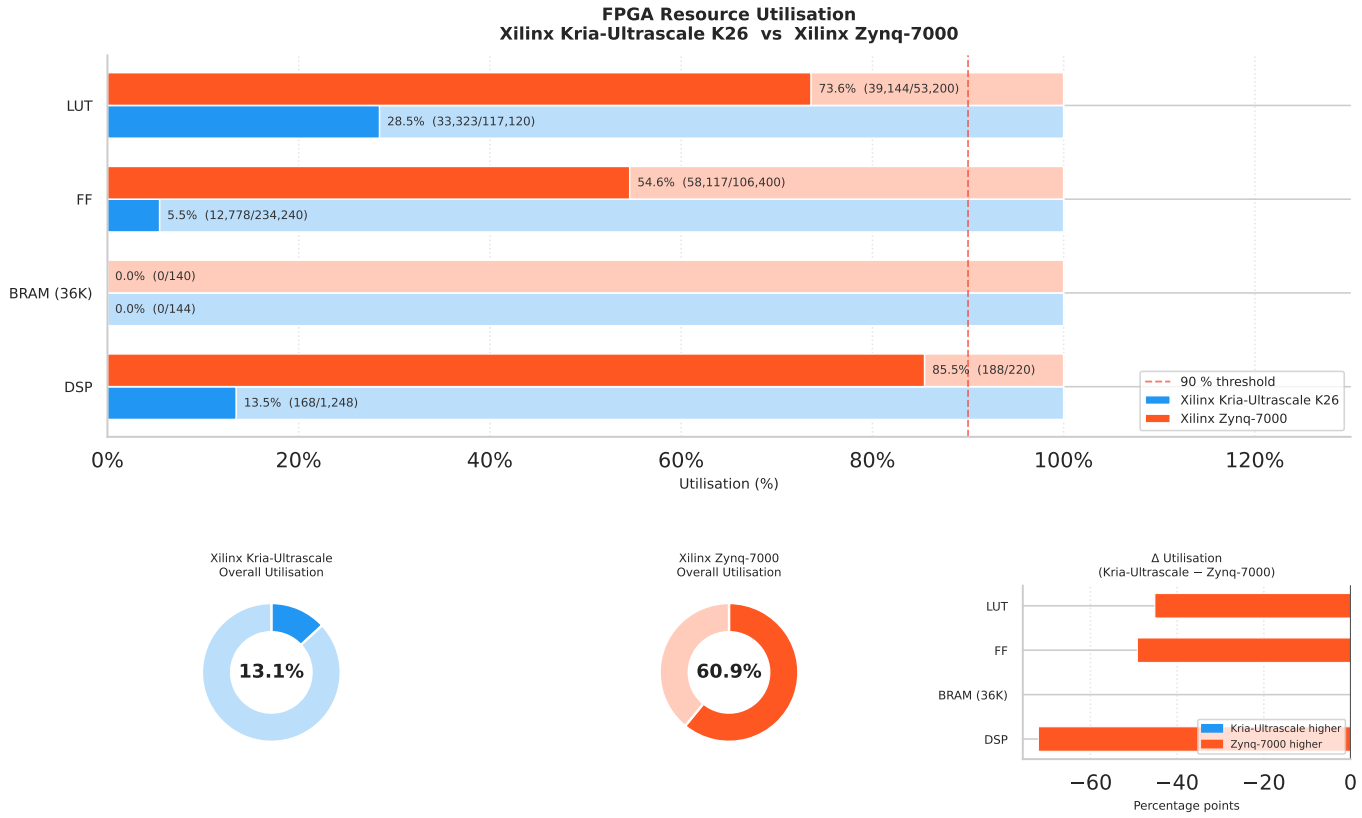

Fig. 6. Resource usage of the quantized model when performing Vivado synthesis with Kria and Zynq FPGAs as targets.

quality loss relative to the floating-point reference model is approximately 5×10^{-5}. These findings are summarized in Figure 7.

These figures make the proposed implementation suitable for real-time extrapolation of electron temperature data for the RFX-mod2 device, which could then be used as additional information to supply to the control system. This would leverage both internal state information and boundary conditions on the magnetic configuration, enhancing current capabilities.

The tight numerical agreement observed between the Keras and HLS models confirms that the HGQ2–hls4ml toolchain preserves model fidelity throughout the compilation chain, with co-simulation providing a reliable gate before committing to resource-intensive synthesis. The heterogeneous bit-width allocation produced by HGQ2 proved particularly valuable in this context, concentrating numerical precision where the model is most sensitive (e.g., in the latent space as visible from Figure 4), while aggressively compressing the outer layers. This results in a favorable accuracy-to-resource trade-off that uniform fixed-point quantization could not achieve.

Future work will focus on connecting the model to actual experiments, such as RFX-mod2, to further validate its applicability and effectiveness in fusion experiments.

References

[1] S. Peruzzo *et al.*, "Design concepts of machine upgrades for the rfx-mod experiment," *Fusion Engineering and Design*, vol. 123, pp. 59–62, 2017, proceedings of the 29th Symposium on Fusion Technology (SOFT-29) Prague, Czech Republic, September 5-9, 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0920379617302788

[2] M. Cavinato, G. Manduchi, A. Luchetta, and C. Taliercio, "General-purpose framework for real time control in nuclear fusion experiments," *IEEE Transactions on Nuclear Science*, vol. 53, no. 3, p. 1002–1008, Jun. 2006. [Online]. Available: http://dx.doi.org/10.1109/TNS.2006.873002

[3] J. Degrave *et al.*, "Magnetic control of tokamak plasmas through deep reinforcement learning," *Nature*, vol. 602, no. 7897, p. 414–419, Feb. 2022. [Online]. Available: http://dx.doi.org/10.1038/s41586-021-04301-9

[4] A. Murari *et al.*, "An optimized multifoil soft x-ray spectrometer for the determination of the electron temperature with high time resolution," *Review of Scientific Instruments*, vol. 70, no. 1, p. 581–585, Jan. 1999. [Online]. Available: http://dx.doi.org/10.1063/1.1149342

[5] L. Carraro, M. E. Puiatti, F. Sattin, P. Scarin, and M. Valisa, "Requirements for an active spectroscopy diagnostic with neutral beams on the rfx reversed field pinch," *Review of Scientific Instruments*, vol. 70, no. 1, p. 861–864, Jan. 1999. [Online]. Available: http://dx.doi.org/10.1063/1.1149518

[6] G. De Masi *et al.*, "Design of a new reflectometric system for real time plasma position control on the rfx-mod2 device," *Journal of Instrumentation*, vol. 17, no. 01, p. C01071, Jan. 2022. [Online]. Available: http://dx.doi.org/10.1088/1748-0221/17/01/C01071

[7] A. Rigoni Garola *et al.*, "Diagnostic data integration using deep neural networks for real-time plasma analysis," *IEEE Transactions on Nuclear Science*, vol. 68, no. 8, pp. 2165–2172, 2021.

[8] F. Bonomo, P. Franz, A. Murari, G. Gadani, A. Alfier, and R. Pasqualotto, "A multichord soft x-ray diagnostic for electron temperature profile measurements in RFX-mod," *Review of Scientific Instruments*, vol. 77, no. 10, p. 10F313, 09 2006. [Online]. Available: https://doi.org/10.1063/1.2219400

[9] P. Martin, A. Murari, and L. Marrelli, "Electron temperature measurements with high time resolution in rfx," *Plasma Physics and*

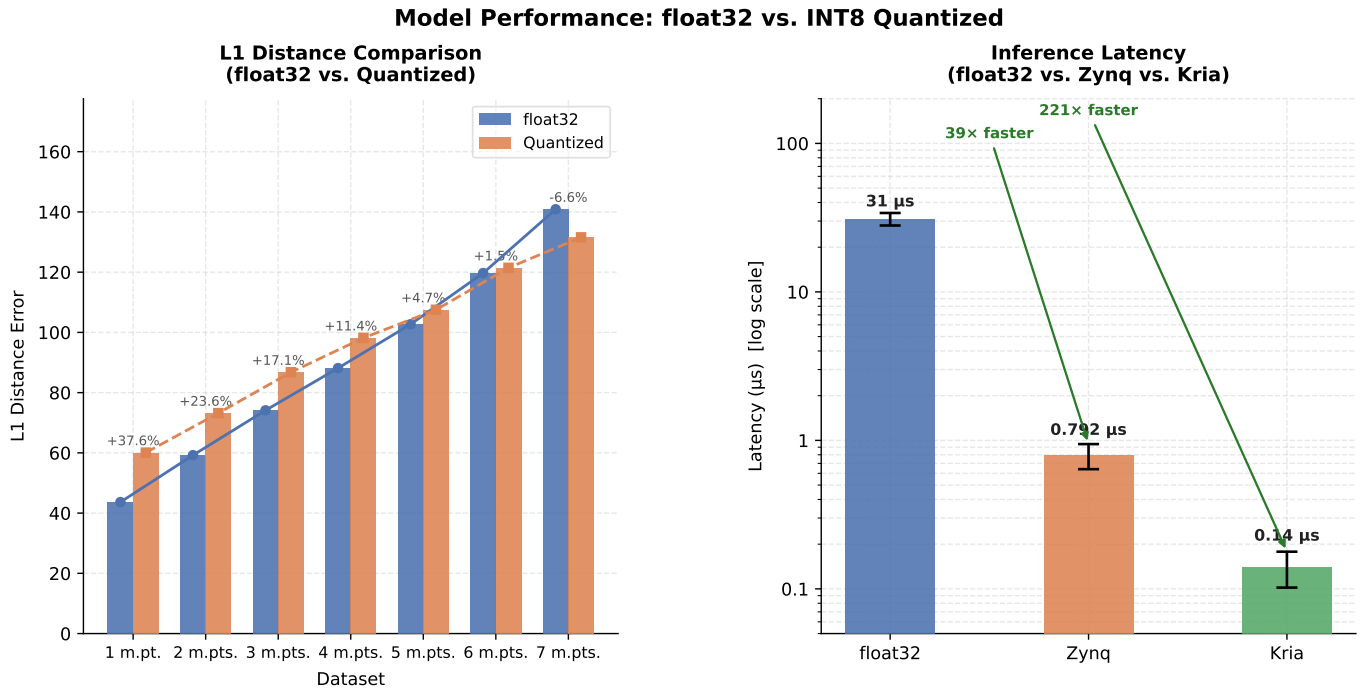

Fig. 7. On the left panel, the error for the float32 and the quantized model is shown. On the right-hand side, the latency on GPU for the float32 model and the latency for the quantized model implemented on the Kria and Zynq FPGAs is shown.

Controlled Fusion, vol. 38, no. 7, p. 1023–1031, Jul. 1996. [Online]. Available: http://dx.doi.org/10.1088/0741-3335/38/7/007

[10] P. Franz *et al.*, "Experimental investigation of electron temperature dynamics of helical states in the rfx-mod reversed field pinch," *Nuclear Fusion*, vol. 53, no. 5, p. 053011, 4 2013. [Online]. Available: https://dx.doi.org/10.1088/0029-5515/53/5/053011

[11] L. Orlandi, A. Rigoni Garola, M. Gobbin, P. Franz, and L. Piron, "Data reconstruction using variational autoencoders and error analysis compared to b-spline interpolation," *Plasma Physics and Controlled Fusion*, vol. 68, no. 2, p. 025028, Feb. 2026. [Online]. Available: http://dx.doi.org/10.1088/1361-6587/ae4716

[12] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," 2022. [Online]. Available: https://arxiv.org/abs/1312.6114

[13] F. Fahim *et al.*, "hls4ml: An open-source codesign workflow to empower scientific low-power machine learning devices," 2021. [Online]. Available: https://arxiv.org/abs/2103.05579

[14] S. Bowman, L. Vilnis, O. Vinyals, A. Dai, R. Jozefowicz, and S. Bengio, "Generating sentences from a continuous space," in *Proceedings of the 20th SIGNLL conference on computational natural language learning*, 2016, pp. 10–21.

[15] H. Fu, C. Li, X. Liu, J. Gao, A. Celikyilmaz, and L. Carin, "Cyclical annealing schedule: A simple approach to mitigating KL vanishing," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, J. Burstein, C. Doran, and T. Solorio, Eds. Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 240–250. [Online]. Available: https://aclanthology.org/N19-1021/

[16] C. Sun *et al.*, "Hgq: High granularity quantization for real-time neural networks on fpgas," in *Proceedings of the 2026 ACM/SIGDA International Symposium on Field Programmable Gate Arrays*. ACM, Feb. 2026, p. 79–91. [Online]. Available: http://dx.doi.org/10.1145/3748173.3779200

[17] S. Gupta, A. Agrawal, K. Gopalakrishnan, and P. Narayanan, "Deep learning with limited numerical precision," in *Proceedings of the 32nd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, F. Bach and D. Blei, Eds., vol. 37. Lille, France: PMLR, 07–09 Jul 2015, pp. 1737–1746. [Online]. Available: https://proceedings.mlr.press/v37/gupta15.html