

PowerYOLO: Mixed Precision Model for Hardware Efficient Object Detection with Event Data

Dominika Przewlocka-Rus*, Tomasz Kryjak[†] *Senior Member IEEE* and Marek Gorgon[‡] *Senior Member IEEE*
Embedded Vision Systems Group, AGH University of Krakow

Krakow, Poland

Email: *dprze@agh.edu.pl, [†]tomasz.kryjak@agh.edu.pl, [‡]mago@agh.edu.pl

Abstract—The performance of object detection systems in automotive solutions must be as high as possible, with minimal response time and, due to the often battery-powered operation, low energy consumption. When designing such solutions, we therefore face challenges typical for embedded vision systems: the problem of fitting algorithms of high memory and computational complexity into small low-power devices. In this paper we propose PowerYOLO – a mixed precision solution, which targets three essential elements of such application. First, we propose a system based on a Dynamic Vision Sensor (DVS), a novel sensor, that offers low power requirements and operates well in conditions with variable illumination. It is these features that may make event cameras a preferential choice over frame cameras in some applications. Second, to ensure high accuracy and low memory and computational complexity, we propose to use 4-bit width Powers-of-Two (PoT) quantisation for convolution weights of the YOLO detector, with all other parameters quantised linearly. Finally, we embrace from PoT scheme and replace multiplication with bit-shifting to increase the efficiency of hardware acceleration of such solution, with a special convolution-batch normalisation fusion scheme. The use of specific sensor with PoT quantisation and special batch normalisation fusion leads to a unique system with almost 8x reduction in memory complexity and vast computational simplifications, with relation to a standard approach. This efficient system achieves high accuracy of mAP 0.301 on the GEN1 DVS dataset, marking the new state-of-the-art for such compressed model.

Index Terms—embedded vision systems, dynamic vision sensor, hardware-aware algorithm desing, logarithmic quantization, mixed precision, pedestrian detection, power-of-two quantization, vehicle detection,

I. INTRODUCTION

Vision systems based on machine learning algorithms are already becoming a standard in many areas of our lives: in applications and devices that provide entertainment (*intelligent* photo and video processing, augmented/virtual reality), in surveillance and security systems and autonomous vehicles, or advanced driver assistance systems.

Naturally, in most applications we consider devices with limited energy budget – mobile and battery-powered – which means that the use of power-hungry algorithms based on neural networks requires careful design of such solutions, generally

The work presented in this paper was supported by: the program "Excellence initiative — research university" for the AGH University of Krakow. We gratefully acknowledge Polish high-performance computing infrastructure PLGrid (HPC Centers: ACK Cyfronet AGH) for providing computer facilities and support within computational grant no. PLG/2023/01619

with some reduction in memory-computational complexity relative to standard approaches.

Such a reduction generally means changing the precision of computations from 32-bit floating-point numbers to 16-bit floating-point, 8-bit fixed-point or even, in special cases, 6-bit and 4-bit fixed-point numbers, but above all to 8-bit or even lower-width integers, including binary and ternary. With the right hardware platform, such simplifications can lead to increased performance (GPU, CPU), often also allowing a reduction in energy requirements with the right design of custom processor (FPGA, ASIC). In addition, they allow to address to problem of memory bandwidth gap and reduce the size of the model. It is a standard practice to use quantisation to 8-bit integers, which only slightly affects the effectiveness of the neural network, especially if appropriate quantisation aware training (QAT) is used.

The use of lower precision while retaining the accuracy of the full precision model may require specific quantisation schemes, such as the logarithmic quantisation benchmarked earlier for the classification task in [1], [2], [3].

Pedestrian and vehicle detection are tasks present in a wide range of applications: from driver assistance systems and autonomous vehicles, to security and surveillance systems. All those applications have similar requirements – high performance, real-time data processing, and rather low energy budget. Therefore, the design of such a solution must be approached comprehensively: by choosing the right algorithm, hardware platform and even sensor. A standard digital camera is a natural choice, while an event camera (dynamic vision sensor, neuromorphic camera) [4] is recently becoming an interesting alternative.

In their operation, event cameras mimic the characteristics of the human visual system, noticing only changes in brightness per pixel (and therefore events per pixel). As a result, they record and transmit data with lower latency than standard cameras and also have a higher dynamic range, making them work well in a wide variety of lighting conditions, including unevenly and poorly illuminated scenes. The latter can be particularly important in changing road conditions.

In this paper we propose to use logarithmic quantisation for a task complex and commonly considered in the context of low-power devices – pedestrian and vehicle detection. In addition to designing an algorithm with low memory and computational complexity, it was also decided to use an event

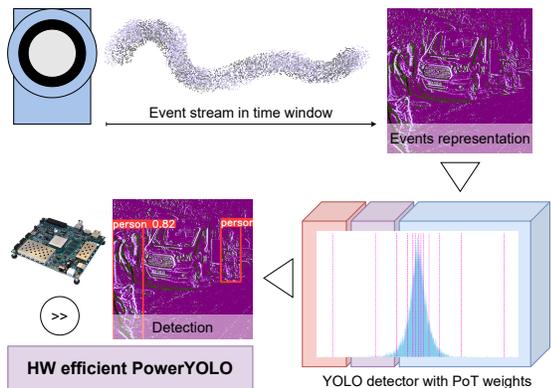


Fig. 1. Overview of the proposed system: event camera data is gathered over a predetermined time window (10 ms), and represented in the form of a pseudo-image (event frame). Pedestrians are detected using a mixed precision YOLO network, with convolution layer weights quantised logarithmically to PoT values, and the rest of parameters and activations quantised linearly.

camera which characteristics are perfectly suited to the real-time and high accuracy requirements of embedded perception systems. A schematic of the system is presented in Fig. 1: the data received from the event camera is represented by a pseudo-image (i.e. event frame), which is then processed by the PowerYOLO mixed-precision network. Due to the PoT weights, a hardware implementation of the PowerYOLO network can introduce significant energy gains, resulting from the simplification of multiplication operations and therefore reducing the number of required electronic components needed to implement the corresponding accelerator. Thus, the main contribution of this work can be summarised as follows:

- 1) we propose a highly efficient INT8/LOG4 mixed precision PowerYOLO network for pedestrian and vehicle detection for GEN1 event dataset. Our model stands out for having the best ratio of efficiency to memory-computation complexity achieving mAP 0.301.
- 2) we propose a method for convolution and batch normalisation layers fusion in a way that takes into account the special form of the neural network weights when quantised to powers of two.

The remainder of this paper is organised as follows: Section II presents basic information about event cameras. Section III discusses previous work related to the optimisation of neural network architectures and an overview of object detection methods based on event data stream. Section IV presents the PowerYOLO network, which uses a mixed integer and power-of-two quantisation scheme. Its evaluation is then provided in Section V. The article concludes with a brief summary and suggestions for further research.

II. EVENT CAMERAS

Event cameras record changes in light intensity individually (asynchronously) for each pixel. A single event is represented by a vector $e = \{t, x, y, p\}$, where t is the occurrence time of the event (timestamp), x and y are the spatial coordinates of the recorded pixel, and p , called polarisation, takes the

value -1 or 1 (sometimes also 0 and 1) depending on the direction of brightness change. Therefore, data acquisition only takes place when the brightness of the pixel changes, which enables high-resolution with low-latency data flow. In addition, this results in a significant reduction in blur in the event of fast movement of objects or the camera. Moreover, the high dynamic range resolution allows events to be recorded in both bright and dark regions. This is especially important for outdoor solutions, where lighting is highly variable (e.g. in automotive: tunnel crossings, under overpasses, etc.). From the perspective of designing energy-efficient solutions, these sensors seem particularly interesting: due to their asynchronous and sparse operation (recording only changes in brightness), they are on average more energy-efficient than typical digital cameras (frame cameras).

However, the format of event data (spatio-temporal sparse cloud) makes it impossible to use explicitly standard digital image processing algorithms, including those based on neural networks, for their analysis. Therefore four possibilities are considered: designing a solution dedicated to event data (e.g. spiking neural networks – SNN) [5], processing a point cloud e.g. using graph neural networks (which can be challenging due to the different density of such a cloud in successive time segments) [6], projecting (aggregating) events onto a plane and creating so-called pseudo-images [7], or image reconstruction from such a cloud (which is computationally very complex) [8].

The use of pseudo-images best cooperates with the typical neural networks developed for images registered with frame cameras. A number of such approaches have been proposed in the literature - they differ in the way they take into account the time dimension, in particular the change in signal over time. In this work, we use the event frame method, which transforms events from a specific time slice (in our case 10ms) τ according to $f(u, t) = P_e(u)$ for $t - \sigma_e(u) \in (0; \tau)$ and $f(u, t) = 0$ otherwise, where σ_e denotes the time of the event $u(x, y)$ and P_e its polarity. This choice is dictated by two factors: firstly, the chosen object detection algorithm is dedicated to classical images, so the input format must be similar to them; secondly, due to the necessary limitations of memory and computational complexity for low-power devices, the method of converting the event cloud to a pseudo-image should not be too complex. Other pseudo-image representations worth mentioning are: simple binary frame [9], exponentially decaying time surface [10] and event frequency [11].

In this research we used the Gen1 event dataset consisting of over 39 hours of recordings of urban, highway, suburbs and countryside scenes with a Prophesee Gen1 sensor attached to a car. The collection was manually tagged based on greyscale images recorded simultaneously with the events, ultimately highlighting nearly 230000 of cars and nearly 28000 of pedestrians. For the generation of pseudo-images, we used the tools provided by the authors of the database [7].

III. RELATED WORK

In this section we present previous work on two topics closely related to our research.

A. Reducing the memory and computational complexity of neural networks

Standard methods for reducing the memory-computation complexity of neural networks include quantisation, pruning, and appropriate transformations in the form of layer fusion that do not change the inference result. The first two methods lead to a reduction in the number of bits needed for parameter storage and the number of parameters, respectively. In its simplest form, pruning, i.e. zeroing the connections or neurons with low impact on the final result, leads to a sparse network in which the number of arithmetic operations performed can be considerably reduced. However, this introduces the overhead of having to remember which weights or neurons have been zeroed (i.e. which calculations should be omitted). An alternative is to train sparse networks, i.e. networks in which the structure of the removed connections is predetermined. Since neural networks are generally redundant, appropriately performed pruning, or sparse network training, allows to achieve the performance of a dense network.

For the same reason, a reduction in computational precision (quantisation) may also not result in a significant drop in performance relative to a floating-point network, while at the same time reducing memory complexity by several times. A certain standard available in many tools, both software tools (e.g. tensorflow [12], OpenVino [13], brevitas [14]) but also in hardware implementations (e.g. FINN [15], VitisAI [16], nncf [17]) is linear quantisation, in particular to 8-bit integer values.

This method reduces the model by a factor of four (in terms of number of bytes) and significantly simplifies the implementation of arithmetic operations, which is particularly important for low-power hardware implementations. Linear quantisation to lower bit-widths can significantly affect the accuracy of the solution, therefore logarithmic quantisation methods, first proposed in [18], are an interesting alternative. This scheme allows to preserve the accuracy of the full-precision network for parameters 4-bit and smaller (after appropriate training of the quantised network) [1], [2]. In addition, this approach allows to replace the multiplication with a bit-shift, which, using the appropriate computing platform (GPU, eGPU, and especially reconfigurable platforms like FPGAs), results in reduced latency and reduced power consumption [2], [3], [19]. Obviously, the most radical form of reduction in computational precision is binary quantisation, however it is not without an impact on the accuracy of the network [20], causing losses of several tens of per cent.

Extremely low-precision networks, or rather new quantisation methods, are most often benchmarked for classification tasks, so it is difficult to say unequivocally whether the impact of different precision reduction methods is same for each task solved by a neural network, in particular for object detection. However, it seems that due to the higher complexity

and the need for higher precision information at the output, quantisation to very low bit-widths for detection task may have a greater impact on performance: especially if we consider the coordinates of the bounding box, rather than a class probability score. In principle, the latter should simply be higher for true class objects than for others, rather than with some specific value. Several experiments with very low-precision quantisation of YOLO networks have been described in the literature.

In [21], the authors note that quantisation of YOLO networks is challenging due to the problem of oscillation of weights between quantisation thresholds when learning low-precision networks, even at final epochs. The use of two additional methods EMA (Exponentially Moving Average) and QC (Quantisation Correction) have been proposed to minimise these oscillations, in combination with quantised training. In the Quantisation Aware Training (QAT) version, EMA works similarly as in full precision network training, storing a history of both the weights and the scaling factors of the weights and activations. With the introduction of EMA into QAT, the training is finitely smoother. In addition, Quantisation Correction has been introduced after the training process, to minimise oscillations errors by linearly transforming the activations based on some pre-trained parameters. The YOLOv5s network trained with such 4-bit quantisation on the COCO dataset achieves a mAP of 0.34, with a baseline (full-precision network) mAP of 0.374 – thus a decrease of about 11% (for more results for different architectures, we refer to the article). In [22], the authors proposed a Post-Training Quantisation method that achieved a mAP of 0.196 for YOLOv5s and a COCO dataset for 4-bit weights.

It is worth noting that both of these methods propose the use of a linear quantisation scheme with additional mechanisms to prevent significant drops in efficiency, as achieved in [21]. In this paper, we propose a mixed precision solution, with the network weights quantised logarithmically to powers-of-two, and the rest of the activation and parameters quantised linearly.

Such a method allows a significant reduction in computational complexity due, firstly, to the possibility of changing the multiplication operation to a bit shift and, secondly, performing the remaining computations as integer operations.

The gains from converting multiplication operations to bit shifting have been described in several papers. In [1], the authors implemented GPU kernels for PoT-weighted computing and showed gains over standard filtering with multiplication, shortening the inference time for ResNet18 by 25%. In [23] the authors show gains in energy and logic resources used for 5-bit logarithmic quantisation relative to standard 16-bit multiplier for UMC 55nm Low Power Process. The paper [24] presents dynamic power consumption for a hardware implementation of matrix multiplication for uniformly and PoT quantised values. At 3 bits, one can assume a decrease in power requirements of about 20%, at 5 bits – 35%, in favour of PoT weights. As was previously shown in [19], the hardware convolution layer accelerator on the Zynq UltraScale+ MPSoC ZCU104 development board with a SoC FPGA chip, with 4-bit

PoT weights can be at least 1.4x more power efficient than the linearly quantised version, and the difference in power efficiency increases at higher chip clock frequencies.

In this paper, we propose a mixed-precision detector that optimises all parameters of a neural network: to our knowledge, all previous work on logarithmic quantisation has referred to bit-width reduction mainly of weights and, less often, also of activation, neglecting the issue of the batch normalisation layer. Although this layer is not a necessary part of building neural network architectures, it is a certain standard in most classical solutions. It is therefore important to propose a method that is usable for virtually any ready-made full-precision model.

B. Object detection based on event data

Work on pedestrian and vehicle detection on the GEN1 event dataset [25] can be found in the literature. The dataset itself was created from more than 39 hours of recordings using the GEN1 event sensor from the Prophesee company, collecting data at a resolution of 304x240.

In [26] authors proposed general methods for converting models trained on image-like event representations to models that operate asynchronously. Event data are represented by a Sparse Recursive Representation, which can be updated as new events arrive in a sparse manner (i.e. only where a change has occurred). This form of input allows the use of so-called sparse convolutions, recalculating only those activations which value may have changed. Using such a model, a mAP of 0.129 was obtained for the GEN1 set and event histogram representation (0.145 using a standard convolutional neural network).

The paper [27] proposes a directly trained spiking network – EMS-YOLO. The first convolution layer is trained by converting the input to spikes, the subsequent layers consist of EMS-Modules, which are fully spiking residual blocks with LIF (Leaky Integrate and Fire) activation function. The input of the network is an image-based 2D representation of events within a specific time window. On the GEN1 set, mAP of 0.267 was achieved, for inference over 4 time steps, with a Firing Rate of 21.15% neurons, and a network with 6.20M parameters.

A spiking feature pyramid network (SpikeFPN), consisting of an encoder backbone spiking neural network, a spiking feature pyramid building network and a spiking multi-head-detection module, was presented in [5]. Event data are represented using Stacking Based on Time (SBT), which, with standard compression of the event stream into a frame, allows some temporal information to be retained. A mAP of 0.223 was achieved for the GEN1 dataset.

The Group Event Transformer architecture is proposed in [28], which is dedicated to process event data directly and thus without the need to convert events to a 2D representation, which results in the loss of some timing or polarity information. Events are embedded into Group Tokens based on timing and polarity information, and then processed by a transformer-based architecture using self-attention mechanisms. On the GEN1 dataset, the proposed model achieves

mAP of 0.406, and with additional memory mechanisms to support the challenges of capturing only moving objects by the event camera, 0.484.

Similarly, in [29] the authors also propose to analyse events directly, in the form of a structured (in terms of spatial dimensions) point cloud. At each time step, the model based on the Vision Transformer architecture processes a new data stream together with previous states from recursive layers. The recursive layers help to solve the problem of slow-moving objects not captured by the event camera at each time step. The described model achieves mAP of 0.472 on the GEN1 dataset, for the largest of the proposed architectures (for the 4x smaller model mAP 0.441).

A recursive architecture consisting of standard convolution layers and LSTM blocks is presented in [7]. The model directly processes the event data stream as a tensor with fixed spatial dimensions. The proposed method achieves mAP of 0.4 for GEN1 dataset, with 24.1M parameters. In [30] the authors proposed to use the YOLO (You Only Look Once) model, together with an adaptive conversion of the events to Hyper Histogram form, which allows the preservation of both temporal and polarity information. In addition, a corresponding modification of one of the key data augmentation methods for YOLO detectors in the form of Shadow Mosaic was also proposed. The model trained in such a way achieves mAP of 0.47, while the analogous solution without Hyper Histogram and Shadow Mosaic 0.394 (treated by the authors as a baseline). In [31] the authors proposed another way of representing event data - Temporal Active Focus (TAF) - along with a small Agile Event Detector (AED) neural network model with a corresponding event encoder extracting semantic information from temporal to flat vector input. The architecture of the detector itself is based on Darknet21 and head like for YOLOX. Ultimately, the model achieves mAP of 0.454, with 14.8M of parameters and a baseline (YOLOX architecture and Event Volume representation) of 0.35.

Table I summarises the discussed previous work on object detection on the GEN1 dataset. It is worth noting that these solutions are classical in the sense of computational precision, i.e. they operate on 32- or 16-bit floating point numbers, and mostly require GPU acceleration. For solutions based on spiking neural networks ([26], [27], [5]), the firing rate can be as low as 20%, which of course translates into lower computational complexity, but these solutions have much lower accuracy (and the memory complexity remains constant). Independently, due to their size, efficient use of the mentioned models in low-power devices directly is virtually impossible.

For this article, the last two methods proposed by [30] and [31] are particularly relevant: the former because of the used YOLO architecture, and the latter because of the small size of the network. To the authors' best knowledge, these models determine the current SoTA (in the sense of highest detection accuracy) on the GEN1 dataset, and it is to these models that we will compare our proposed mixed precision network. The analysis shows that, besides the choice of architecture (spiking network, classical deep detectors or transformers), the

TABLE I
OVERVIEW OF EXISTING OBJECT DETECTORS FOR THE GEN1 DATASET,
DISTINGUISHING mAP AND NUMBER OF MODEL PARAMETERS.

Method	mAP	Model size
ASCNN [26]	0.129	133M
EMS-ResNet [27]	0.267	6.2M
SpikeFPN [5]	0.223	11M
GET [28]	0.479	21.9M
RVT-S [29]	0.465	9.9M
RVT-T [29]	0.441	4.4M
Gray-RetinaNet [7]	0.44	32.8M
YOLOv5l [30]	0.47	46.5M
AED [31]	0.454	14.8M

representation of the input data itself (multidimensional tensor or 2D image), but also its augmentation, is important.

In addition, object detection on event frames faces the problem of continuity of information in case of objects moving at variable speeds (in particular low or even static). This is an issue not encountered in such form for algorithms designed for standard images. However, this paper does not aim to make further improvements in these areas, but to propose a method with the lowest possible memory-computational complexity, while maintaining the highest performance. Reducing memory-computational complexity promotes efficient information processing in embedded devices, particularly those with the potential for significant parallelisation of computation: it can provide low latency while requiring little energy.

IV. POWERYOLO NETWORK

One of the most popular algorithms for object detection are networks based on the YOLO (You Only Look Once) architecture [32]. However, due to their memory and computational complexity, proper optimisation is required for efficient inference of such architectures on low-power devices. In this section, we first discuss the complexity of the YOLO network and then present the proposed mixed-precision quantisation – power-of-two and linear.

A. YOLO computational and memory complexity

The neural networks from the YOLO “family” consist of three main blocks: backbone for image feature extraction, neck for feature pyramid extraction (by combining features from different levels of the backbone) and head as the final stage determining bounding boxes, class labels, probabilities and objectness scores. The architecture of the YOLO network has evolved over the years, and the latest version widely accepted by the community as SoTA is YOLOv8, available open source at [32]. Depending on the number of backbone layers, the network comes in several sizes (by convention: n, s, m, l, x), and for the purpose of this paper the commonly used (due to its good ratio of memory and computational complexity to achieved accuracy) YOLOv8s version was chosen.

The entire network has 7.2 M parameters (and thus, in the classical approach, 32- or 16-bit floating-point values), of which almost 79% are the weights of the convolution

layers. Linear quantisation of the weights and activations to 8-bit numbers generally preserves the accuracy of the full-precision network, while reducing the memory complexity by 4x (relative to 32-bit numbers) and increasing the throughput of the mathematical operations by 16x [33]. If a further reduction in the precision of the convolution layer weights to 4-bit values is made, a very compact architecture both in terms of memory and computation is obtained. In such a case, according to [33], the throughput of mathematical operations on the GPU increases up to x32. Implementing such a mixed-precision network (with 4-bit weights and 8-bit activations) in both eGPUs and dedicated processors therefore allows for significant improvements in inference time, while reducing memory complexity by almost x8. Using a logarithmic quantisation scheme of weights to powers of two, we will additionally enable the conversion of multiply-accumulate (MAC – Multiply ACcumulate) operations to shift-accumulate (BAC – Bitshift ACcumulate) operations, ultimately introducing a radical simplification in neural network computation.

B. Mixed Precision Quantization

In this work, we propose to use mixed quantisation, with convolution layer weights quantised logarithmically to 4-bit powers of two (so each weight can be represented by 2^n where n is integer) and activations quantised linearly to 8-bit values. Bias is customarily quantised to a 32-bit number to avoid overflow during the accumulation of convolution results. Linear quantisation of 8-bit activations allows the network to maintain high performance, without the need to re-train the quantised network, but only by calibrating the values of the scaling factors and activation offset on some representative set of input data (customarily a subset of the training data).

Using PoT weights quantisation, which concentrates more quantisation intervals around zero thus mimics the shape of the distribution of weight values in the convolution layer, we can achieve better accuracy results with lower than standard 8-bit bit widths.

PoT quantisation is used in the so-called quantisation aware training (QAT). The weights of the quantised network are initialised with the trained full-precision network and then the quantised network is trained (fine-tuned) for a relatively small number of epochs. Due to the discontinuity of the quantisation function, the gradient method cannot be used explicitly. The forward pass is done using quantised weights, and during backpropagation the gradients are computed using full precision equivalents (after the weights are updated, the network is re-quantised).

Another method of reducing memory and computational complexity in neural network inference is the fusion of convolution layers with batch normalisation layers according to Eq.(1):

$$y_{bn} = \frac{x_{conv} - \mu}{\sqrt{\sigma^2 - \epsilon}} \gamma + \beta = \frac{\gamma}{\sqrt{\sigma^2 - \epsilon}} x_{conv} - \mu \frac{\gamma}{\sqrt{\sigma^2 - \epsilon}} + \beta \quad (1)$$

where x_{conv} is the output of the convolution layer. Thus, the weights w and the bias b of the filters for inference are

modified accordingly: $w_{fused} = \frac{\gamma}{\sqrt{\sigma^2 - \epsilon}} * w$ and $b_{fused} = b - \mu \frac{\gamma}{\sqrt{\sigma^2 - \epsilon}} + \beta$.

Naturally, such a modification does not affect the accuracy of the neural network, but it reduces the required number of multiplication and addition operations, and the number of parameters. Introducing exactly such a fusion to a power-of-two network would result in the weights no longer being in the form of powers of two, and therefore losing an important property of the described method. In order to reduce the number of operations, we propose a small yet powerful modification. First, since the bias is quantised linearly, it can be successfully modified according to the standard fusion scheme (Eq. (1)). Next, we propose to combine the multiplier usually associated with weights not with weights, but with the scaling factor of the quantisation operation. In this way, we reduce as much as possible all the additional computations introduced by batch normalisation layer, while only slightly increasing the memory complexity relative to the model after standard fusion - instead of one scaling factor for the whole layer, we get scaling factors for each output feature map separately, and the batch normalisation completes during re-quantisation between layers.

Ultimately, we obtain a mixed-precision model, efficient in terms of memory and computational complexity, for which the calculations in the standard convolution layer – batch normalisation block can be written with the simplified equation (2), where $\phi = \sqrt{\sigma^2 - \epsilon}$:

$$y_q = \frac{\gamma}{\phi} \frac{s_x s_w}{s_y} \left(\sum_{i=0}^k \sum_{j=0}^k x_q^{ij} \gg w_{pot}^{ij} + B_q \right) \quad (2)$$

where s_w, s_x, s_y are the scaling factors for quantising the weights, block inputs and block outputs, k is the dimensions of the convolution filter, x_q is the activation from the previous layer quantised to INT8, B_q is the modified bias $B = \frac{\gamma(b-\mu)}{\phi} + \beta$ quantised to INT16, and w_{pot} are weights in the INT4 format, in powers of two.

For re-quantisation between successive layers of the neural network during inference, for PoT quantisation the factor $\frac{\gamma}{\phi} \frac{s_x s_w}{s_y}$ is in a form of vector. In comparison, in case of linear quantisation with standard convolution and batch normalisation fusion, it's a scalar (with value $\frac{s_x s_w}{s_y}$). For simplicity, the indices indicating the output feature map are omitted in the equation.

Although the granularity of quantisation can be chosen in many ways, in this paper we chose the layer-wise quantisation. This decision is based on the following facts: firstly, the shape of the distribution of weights in the convolution layer of the network is logarithmic; secondly, other schemes are characterised by higher computational complexity. For example in the case of the channel-wise scheme, quantisation is performed for each filter separately, which significantly increases the training time. For the network in question, experiments indicated as much as a 25 times increase in the duration of one epoch. However, due to the proposed convolution and batch normalisation fusion scheme, channel-wise quantisation would not

TABLE II
OBJECT DETECTION PERFORMANCE FOR NETWORKS: FULL-PRECISION (BASELINE), WITH UNIFORM (INT*) AND PoT QUANTISATION (LOG*).

Architecture	Quantisation	mAP50	mAP50-95
YOLO8s	Baseline	0.586	0.340
	INT8 W/A	0.582	0.330
	INT4W	0.523	0.288
	LOG4W	0.566	0.312
	LOG4W/INT8A	0.554	0.301

introduce additional computation during inference, and the impact on accuracy remains an open topic. Nonetheless, as shown in section V, the layer-wise scheme allows for very high accuracy.

V. RESULTS

In this section, we present the results of the proposed PowerYOLO network for the task of object detection on event camera GEN1 dataset. We present comparison both with other solutions for the same task, and with other networks belonging to the TinyML group. In addition, we show the applicability of the obtained results for building a hardware accelerator.

A. Mixed Precision YOLO

Neural network training was performed based on YOLO8s code available in [32], suitably modified to perform quantised training. The full precision network was trained for 100 epochs with 4 A100 NVIDIA GPUs, using the SGD method, with a momentum of 0.9 and an initial learning rate of 0.01 reduced linearly to 0.0001. The weights of the full precision network were then used to initialise the quantised model. The quantised network was trained for 20 epochs, using single A100 NVIDIA GPU, also using SGD and with a low initial learning rate value of 0.0001. The value of the learning rate was reduced at epochs 5, 8 and 15 according to $lr = \gamma lr$, where $\gamma = 0.1$. The Exponentially Moving Average (EMA) model was used during training – the final model parameters were a weighted average of the parameters from each training iteration (this average is quantised logarithmically, and the EMA model is quantised after each update). In the conducted experiments, it was found that disabling EMA in the quantised learning process results in a decrease in the performance of the network in terms of mAP by several percents. Table II shows the training results of the Yolov8s network on the Gen1 dataset. The mAP50 and mAP50-95 are averaged over the two object classes specified: vehicles and pedestrians. In general, the metric is defined as area under the precision-recall curve and is widely used for evaluation of detection algorithms. Quantisation to 8-bit integer values after custom QAT PoT training was performed using the OpenVino library (Fig. 2 demonstrates the detector operation).

A decrease in detection performance is evident for both networks quantised to INT8 and LOG4, with the former being approximately 1% (mAP50), 3% (mAP50-95) and the latter 3.5% (mAP50), 8.3% (mAP50-95) and 5.5% (mAP50), 11.5%

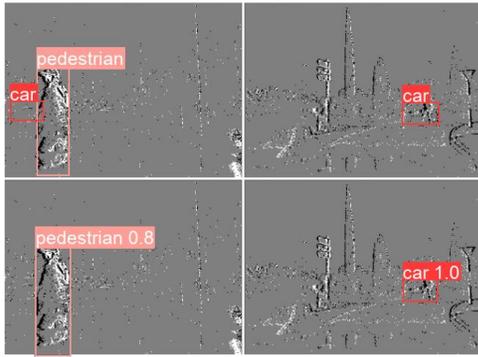


Fig. 2. Example results of object detection using PowerYOLO mixed precision network (bottom) compared to labels (top)

(mAP50-95) depending on whether other parameters and activations are also quantised. Using the same QAT approach, we also trained a model with 4-bit width uniformly quantised weights. The gap to baseline model is almost 2 times higher than in case of LOG4 model, which shows the superiority of logarithmic quantisation. However, it's worth noting that there are efforts in the literature to minimise the gap between full precision and low-bitwidth uniformly quantised models by introducing additional mechanisms, like in [34].

The comparison with SoTA has to be done in two steps, as to the authors' knowledge there are no other solutions that process event data using YOLO networks with very low computational precision. Thus, it is first necessary to refer to other object detectors developed on the Gen1 dataset. A comparison with the full-precision SoTA for the Gen1 dataset is shown in Table III. To enable the best possible comparison, we have trained the YOLOv8l network in addition to the YOLOv8s network. The solution proposed in [30] uses the YOLOv5l network with a slightly different representation of the event data. Without the additional mechanisms proposed by the authors, and therefore with the difference only due to the way the data are processed before and after the neural network, and the way the event data are represented, the difference is small, less than 2% for the mAP50-95 index in favour of [30]. On the other hand, considering a model with Hyper Histogram event representation and with data augmentation adapted to the event representation, the difference is more than 20%.

Secondly, low-precision YOLO networks should also be analysed. Table IV compares our solution with other 4-bit detectors in which all convolution layer weights except the first and last are quantised. It is worth noting that in [22] the solution is based on Post Training Quantisation, which, as a rule, with such low precision computatnois, does not allow to maintain satisfactory performance. In [34], the authors also quantised the activations, so a direct comparison to our model in any version is not possible – even comparing to the mixed-precision version, for which in our case the difference to the floating-point version is about 11.5%, it is hard to say unequivocally which model performs better. In the case of the mixed-precision version, we additionally quantise the bias and

the batch normalisation parameters, which has, as the results presented in Table II indicate, a major impact.

Nevertheless, our solution is comprehensive, i.e. it reduces computation on floating-point numbers to a minimum (actually limiting it only to re-quantisation operations, which is unavoidable with any quantisation) and targets all elements (all layers) of the information flow in the neural network. For this reason, we believe that, in terms of ultimate efficiency with respect to memory-computational complexity, our solution is at least as good as [34], especially given the potential for reducing computational complexity introduced by using quantisation for weights with powers of two (as we show in section ??), and thus we set a new SoTA in this category.

Moreover, as was already established in [3] and [19], the hardware implementation of neurons and layers based on BAC (Bitshift Accumulate) units leads to significant area and power reductions. For a single processing element with weights of bit width 4 and 8-bit activations, using PoT weights allows to increase the power efficiency by a factor of 2. Using this PE for a complete convolution layer leads to 1.6 increase of power efficiency and allows for higher frequency operation, as shown for ZCU 104 platform.

VI. CONCLUSION

In this work we presented an extremely efficient, mixed precision PowerYOLO detector achieving mAP of 0.301 for the GEN1 event dataset. By using both logarithmic quantisation of weights and linear quantisation of activations and other parameters, we eliminated most of floating point operations in the inference flow, allowed replacing multiplication with efficient bit-shifting, and reduced the model size almost 8x while still proposing an accurate detector. In order to close the gap between floating point solutions, for future work we consider extending the training flow with shadow mosaic augmentation (proper for event data) and using other event representation techniques, possibly allowing to deal with slowly moving and static objects. Our initial thesis stated that for embedded applications one should address the algorithm, hardware and sensor (data) at once, and with our solution we show that comprehensive analysis of all those elements leads to truly tiny, yet powerful, machine learning solution.

REFERENCES

- [1] Elhoushi, M., Chen, Z., Shafiq, F., Tian, Y. H., and Li, J. Y. (2021). DeepShift: Towards Multiplication-Less Neural Networks. 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 2359-2368.
- [2] Li, Y., Dong, X., and Wang, W. (2019). Additive Powers-of-Two Quantization: A Non-uniform Discretization for Neural Networks. *CoRR*, volume abs/1909.13144, 2019. Available at: <http://arxiv.org/abs/1909.13144>.
- [3] Przewlocka-Rus, D., Sarwar, S. S., Sumbul, H. E., Li, Y., and De Salvo, B. (2022). Power-of-Two Quantization for Low Bitwidth and Hardware Compliant Neural Networks. 2022. *arXiv preprint* <https://arxiv.org/abs/2203.05025> [cs.LG].
- [4] Lichtsteiner, P., Posch, C. & Delbruck, T., A 128× 128 120 dB 15 μ s Latency Asynchronous Temporal Contrast Vision Sensor in IEEE Journal of Solid-State Circuits, vol. 43, no. 2, pp. 566-576, Feb. 2008, doi: 10.1109/JSSC.2007.914337.

TABLE III
COMPARISON WITH SOTA FULL PRECISION FOR GEN1 DATASET AGAINST MAP AND MODEL SIZE.

Solution	Event representation	Architecture	mAP50-95	No Parameters
Ours	Event frame	YOLOv8s	0.347	11.2M
Baseline [30]	ITS	YOLOv8l	0.375	43.7M
AEC [30]	HH + SM	YOLOv5l	0.382	46.5M
			0.470	

TABLE IV
COMPARISON OF DIFFERENT SOTAs OF LOW-PRECISION YOLO DETECTORS. IT IS WORTH NOTING THAT THE OTHER METHODS LISTED IN THE TABLE ARE BASED ON PROCESSING CLASSICAL IMAGES AND ONLY OURS TREATS EVENT DATA.

Solution	Architecture	Dataset	Method	Bitwidth	mAP50-95 gap
Ours	Yolov8s	Gen1	PoT QAT, layerwise	4bits W	8.3%
EMA+QC [34]	Yolov5s	COCO	Uni QAT, layerwise	4bits W&A	10.4%
Q-Yolo [22]	Yolov5s	COCO	Uni PTQ, channelwise	4bits W	52.4%

- [5] Zhang, H., Li, Y., Leng, L., Che, K., Liu, Q., Guo, Q., Liao, J., & Cheng, R. (2023). Automotive Object Detection via Learning Sparse Events by Spiking Neurons. <https://doi.org/10.48550/arXiv.2307.12900>
- [6] Gehrig, D., & Scaramuzza, D. (2022). Pushing the limits of asynchronous graph-based object detection with event cameras. *arXiv preprint* <https://arxiv.org/abs/2211.12324> [cs.CV]
- [7] Perot, E., de Tournemire, P., Nitti, D., Masci, J., & Sironi, A. (2020). Learning to detect objects with a 1 megapixel event camera. NIPS'20: Proceedings of the 34th International Conference on Neural Information Processing Systems, pages 16639–16652.
- [8] Rebecq, H., Ranftl, R., Koltun, V., and Scaramuzza, D., High Speed and High Dynamic Range Video with an Event Camera in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 43, no. 6, pp. 1964–1980, 1 June 2021, doi: 10.1109/TPAMI.2019.2963386.
- [9] Liu, M. et al. (2017). Block-matching optical flow for dynamic vision sensors: Algorithm and FPGA implementation. 2017 IEEE ISCAS, 1–4. <https://doi.org/10.1109/ISCAS.2017.8050295>.
- [10] Afshar, S. et al. Event-Based Feature Extraction Using Adaptive Selection Thresholds. *Sensors* 2020, 20, 1600. <https://doi.org/10.3390/s20061600>.
- [11] Chen, N. Y. (2018). Pseudo-Labels for Supervised Learning on Dynamic Vision Sensor Data, Applied to Object Detection Under Ego-Motion. 2018 IEEE/CVF Conf. CVPR Workshops, 757–75709. <https://doi.org/10.1109/CVPRW.2018.00107>.
- [12] Abadi, M., et al. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org. <https://www.tensorflow.org/>
- [13] OpenVINO toolkit. Available at: <https://github.com/openvinotoolkit/openvino>.
- [14] Pappalardo, A., (2023). Xilinx/brevitas. Zenodo. <https://doi.org/10.5281/zenodo.3333552>
- [15] Umuroglu, Y., Fraser, N.J., Gambardella, G., Blott, M., Leong, P., Jahre, M., Vissers, K. (2017) Finn: A framework for fast, scalable binarized neural network inference. In: Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays. pp. 65–74. FPGA '17, ACM. <https://doi.org/10.48550/arXiv.1612.07119>
- [16] Xilinx, VitisAI deveop environment, <https://www.xilinx.com/products/design-tools/vitis/vitis-ai.html>
- [17] Kozlov, A., Lazarevich, I., Shamporov, V., Lyalyushkin, N. & Gorbachev, Y., (2020). Neural network compression framework for fast model inference. *arXiv preprint* arXiv:2002.08679
- [18] Miyashita, D., Lee, E. H., and Murmann, B. (2016). Convolutional Neural Networks using Logarithmic Data Representation. *CoRR*, volume abs/1603.01025, 2016. Available at: <http://arxiv.org/abs/1603.01025>
- [19] Przewlocka-Rus, D., and Kryjak, T. (2023). Energy Efficient Hardware Acceleration of Neural Networks with Power-of-Two Quantisation. In: Chmielewski, L.J., Orłowski, A. (eds) Computer Vision and Graphics. ICCVG 2022. Lecture Notes in Networks and Systems, vol 598. Springer, Cham. https://doi.org/10.1007/978-3-031-22025-8_1]
- [20] Wang, Z., Wu, Z., Lu, J., and Zhou, J. (2020). BiDet: An Efficient Binarized Object Detector. *CoRR*, volume abs/2003.03961. Available at: <https://arxiv.org/abs/2003.03961>.
- [21] Gupta, K., and Asthana, A. (2023). Reducing the Side-Effects of Oscillations in Training of Quantized YOLO Networks. arXiv:2311.05109.
- [22] Wang, M. et al. (2023). Q-YOLO: Efficient Inference for Real-Time Object Detection. In: Lu, H., Blumenstein, M., Cho, SB., Liu, CL., Yagi, Y., Kamiya, T. (eds) Pattern Recognition. ACPR 2023. Lecture Notes in Computer Science, vol 14408. Springer, Cham. https://doi.org/10.1007/978-3-031-47665-5_25
- [23] Xu, J., Huan, Y., Jin, Y. et al. (2020). Base-Reconfigurable Segmented Logarithmic Quantization and Hardware Design for Deep Neural Networks. Springer, Journal of Signal Processing Systems, vol. 92, 2020. <https://doi.org/10.1007/s11265-020-01557-8>
- [24] Lee, E. H., iyashita, D., Chai, E., Murmann, B. and Wong, S. S. (2017). LogNet: Energy-efficient neural networks using logarithmic computation. In: 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). DOI: 10.1109/ICASSP.2017.7953288.
- [25] de Tournemire, P., Nitti, D., Perot, E., Migliore, D., & Sironi, A. (2020). A Large Scale Event-based Detection Dataset for Automotive. arXiv: 2001.08499.
- [26] Messikommer, N., Gehrig, D., Loquercio, A., & Scaramuzza, D. (2020). Event-based Asynchronous Sparse Convolutional Networks. European Conference on Computer Vision (ECCV).
- [27] Su, Q., Chou, Y., Hu, Y., Li, J., Mei, S., Zhang, Z., & Li, G. (2023). Deep Directly-Trained Spiking Neural Networks for Object Detection. Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 6555-6565
- [28] Peng, Y., Zhang, Y., Xiong, Z., Sun, X., Wu, F. (2023). GET: Group Event Transformer for Event-Based Vision. International Conference on Computer Vision (ICCV) 2023.
- [29] Gehrig, M., & Scaramuzza, D. (2023). Recurrent Vision Transformers for Object Detection with Event Cameras. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) 2023.
- [30] Peng, Y., Zhang, Y., Xiao, P., Sun, X., & Wu, F. (2023). Better and Faster: Adaptive Event Conversion for Event-Based Object Detection. Proceedings of the AAAI Conference on Artificial Intelligence, 37(2), 2056-2064. <https://doi.org/10.1609/aaai.v37i2.25298>
- [31] B. Liu, C. Xu, W. Yang, H. Yu and L. Yu, "Motion Robust High-Speed Light-Weighted Object Detection With Event Camera," in IEEE Transactions on Instrumentation and Measurement, vol. 72, pp. 1-13, 2023, Art no. 5013113, doi: 10.1109/TIM.2023.3269780.
- [32] Ultralytics. *Ultralytics YOLOv8*. Available at: <https://github.com/ultralytics/ultralytics>. Accessed: 3rd February, 2024
- [33] Wu, H., Judd, P., Zhang, X., Isaev, M., & Micikevicius, P. (2020). Integer Quantization for Deep Learning Inference: Principles and Empirical Evaluation. ArXiv, abs/2004.09602.
- [34] Gupta, K., & Asthana, A. (2023). Reducing the Side-Effects of Oscillations in Training of Quantized YOLO Networks. IEEE/CVF Winter Conference on Applications of Computer Vision (WACV) 2024