Spike-driven YOLO: Ultra Low-Power Object Detection with Neuromorphic Computing

Mark Barnell
Air Force Research Laboratory
Information Directorate
Rome, NY, USA
mark.barnell.1@us.af.mil

Courtney Raymond
Air Force Research Laboratory
Information Directorate
Rome, NY, USA
courtney.raymond.1@us.af.mil

Lisa Loomis
Air Force Research Laboratory
Information Directorate
Rome, NY, USA
lisa.loomis.3@us.af.mil

Francesca Vidal
SRC, Inc.
Defense Systems & Solutions
North Syracuse, NY, USA
fvidal@srcinc.com

Daniel Brown
SRC, Inc.
Defense Systems & Solutions
North Syracuse, NY, USA
dtbrown@srcinc.com

Darrek Isereau
SRC,Inc.
Defense Systems & Solutions
North Syracuse, NY, USA
isereau@srcinc.com

***Abstract* — The latest Intel neuromorphic processor, Loihi 2, provides a breakthrough in Artificial Intelligence (AI) for computing at the edge, where sensor information is collected. The computing architecture does this by leveraging computations at the transistor level in a fashion analogous to the human brain's biological neural networks (vs. a Von Neumann compute architecture). The Loihi 2 high performance, small form factor, and low-power consumption make it well-suited for a wide-range of real-time, deep learning applications such as target classification, object detection, and more. Our technical approach and findings support extreme computing needs for the internet of things (IoT) and various ground and airborne platforms' applications. The recently released Loihi 2 ecosystem and the thorough research study completed on this effort were combined to accelerate development, optimization, and demonstration of a new concept of operation for machine learning at the edge. This 2024 research included training and testing Spike-driven YOLO models on data from various sensors. Our concept uses representative sensor data to detect and classify targets of interest through a combination of image processing techniques and machine learning. Importantly, our technical approach allowed us to rapidly train and evaluate the performance of several models for benchmarking against current state-of-the-art algorithms - w/mean average precision > 93% in some cases. The use of Intel's latest *Lava* framework demonstrates the art-of-the-possible in edge computing by demonstrating capabilities on several sensor platforms with wide extensibility to other domains that can use this neuromorphic-computing hardware. In summary, this research included the use of new computing frameworks, processing algorithms, and a unique concept of operation.**

***Keywords* — Extreme Computing, Machine Learning, High Performance Embedded Computing, Neuromorphic Computing, Deep Learning, Object Detection, Intel Loihi 2, Autonomous Operation, Spiking Neural Networks.**

Received and approved for public release by the Air Force Research Laboratory (AFRL) on 12 August 2024, case number AFRL-2024-4409. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors, and do not necessarily reflect the views of AFRL or its contractors. The data analyses were partially funded under AFRL's FA8750-21-C-1511 contract, which was started in 29 April 2021 and continued through 01 Jul 2024.

I. Introduction

This research advances extreme computing technologies (computing hardware, machine learning, algorithms) through the development and demonstration of new capabilities to support several use cases and applications. The research does this by using frameworks utilized by newly invented neuromorphic computing to complete new analyses that can be directly used to inform future model architectures for object detection and classification in the spiking domain. The technical considerations and the approaches discussed in this paper provide direct insight into the value of spiking neural networks in developing more efficient models and machine learning algorithms.

Background and insight into recent research, as well as demand signals that make this research appropriate and applicable are provided in Section II. The Compute Hardware used is introduced in Section III. Section IV describes the Compute Software. The algorithms and machine learning model are described in Section V. The Data Description is introduced in Section VI. The Processing Approach, Results Conclusions and Future Research are described in Sections VII, VIII and IX, respectively.

II. Background/Significance

The Air Force Research Laboratory, Information Directorate (AFRL), High Performance Systems Branch is developing and demonstrating new computing architectures that are providing unique high-performance embedded computing (HPEC) solutions meeting the most demanding operational and tactical processing requirements for emerging and future surveillance operations.

Sensor capabilities have become less expensive and more prevalent; this has resulted in vast quantities of valuable and high-quality data becoming more accessible to both commercial and government consumers. While this has greatly contributed to the rapid advancement in many areas of technology, the demands for more advanced computing capabilities also continue to grow at an outstanding pace [1]. One of the areas that drive these demands is machine learning for sensing and robotics. Today's technology consumers desire to process and use more and more data efficiently and effectively to develop and/or improve various machine learning applications including the detection, classification, and identification of features and objects in imagery or other sensor information/data [2].

These applications often utilize deep learning networks. Deep learning networks are generally incredibly complex, consisting of multiple layers and millions of parameters which allow the network to learn incredibly abstract feature information from a dataset and achieve high network performance. However, as the scale of these networks increases, so do the resources required to successfully train models to achieve state-of-the-art results and performance [3].

Additionally, deploying a deep network application can be computationally expensive in terms of memory (large models may not fit on smaller, embedded processors), bandwidth (loading and transferring models), power (larger models require more power), and compute and datatype constraints (float32 may not be supported).

Spiking neural networks (SNNs) and neuromorphic hardware provide an alternative to these large models that commonly use conventional hardware, such as large banks of Graphics Processing Units (GPUs).

SNNs learn new information through discrete spatiotemporal events (or spikes), which can reduce the computational demand for model training and inference [4]. The focus of this research was to investigate how to leverage the use of SNNs to develop faster, more energy-efficient, deep learning models for object detection and target classification.

These models were developed and implemented using Intel's latest neuromorphic ecosystem, which includes *Lava*, a custom software framework that provides capabilities such as rapid development and optimization of various SNN models, and an automated process to deploy trained models on Intel's ultra-low powered neuromorphic hardware, Loihi 2.

III. Compute Hardware

In late 2021, Intel released an advanced neuromorphic processor called Loihi 2. The Loihi 2 has over 2.3 billion transistors with over a million neurons per chip, which contain state variable allocation between 0 to 4096. This makes Loihi 2 outperform its predecessor by 10x [5].

The Loihi 2 supports low-power applications, below 1 Watt with a die area of only 31 mm2. Loihi 2 is an advancement to its predecessor, which is a 60-mm2 chip [6]. In addition, its intuitive Python-based API for specifying SNNs, a compiler and runtime library for building and executing SNNs make it a practical solution [7].

This makes it a favorable compute asset for extreme edge computing research and development. Additional details on this chip architecture are shown in Fig. 1.

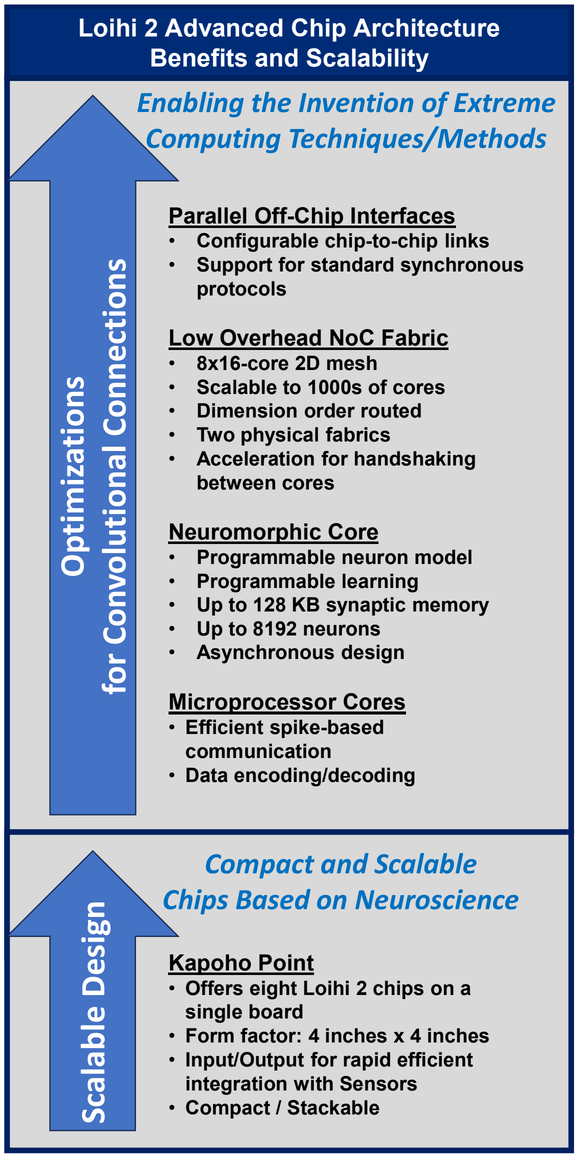
Loihi 2 Advanced Chip Architecture Benefits and Scalability
Enabling the Invention of Extreme Computing Techniques/Methods
Optimizations for Convolutional Connections
Parallel Off-Chip Interfaces
• Configurable chip-to-chip links
• Support for standard synchronous protocols
Low Overhead NoC Fabric
• 8x16-core 2D mesh
• Scalable to 1000s of cores
• Dimension order routed
• Two physical fabrics
• Acceleration for handshaking between cores
Neuromorphic Core
• Programmable neuron model
• Programmable learning
• Up to 128 KB synaptic memory
• Up to 8192 neurons
• Asynchronous design
Microprocessor Cores
• Efficient spike-based communication
• Data encoding/decoding
Compact and Scalable Chips Based on Neuroscience
Scalable Design
Kapoho Point
• Offers eight Loihi 2 chips on a single board
• Form factor: 4 inches x 4 inches
• Input/Output for rapid efficient integration with Sensors
• Compact / Stackable

Fig. 1: Loihi 2 Chip Architecture, Benefits and Scalability

IV. Compute Software

With Loihi 2 came the release of Lava, a suite of libraries designed by Intel's Neuromorphic Research Community (INRC) to optimize models to run on neuromorphic hardware [8]. For our application, we used Lava-dl [9] a complete framework that enables rapid end-to-end development of spiking neural networks. With Lava-dl, we were able to complete the following research tasks listed below:

1. Define a deep convolutional spike-driven model architecture similar to the state-of-the-art You Only Look Once (YOLO) model using predefined neuron models available in Lava-dl.
2. Direct training of SNNs using Lava-dl Slayer. Direct training is often the preferred training method to get the best performance from SNNs compared to hybrid ANN-SNN models (i.e. a pretrained ANN model converted to an SNN using python libraries such as Intel's

snntoolbox) or rate-coded SNNs [10, 11]. The workflow when using Lava-dl Slayer is illustrated in Fig. 2.

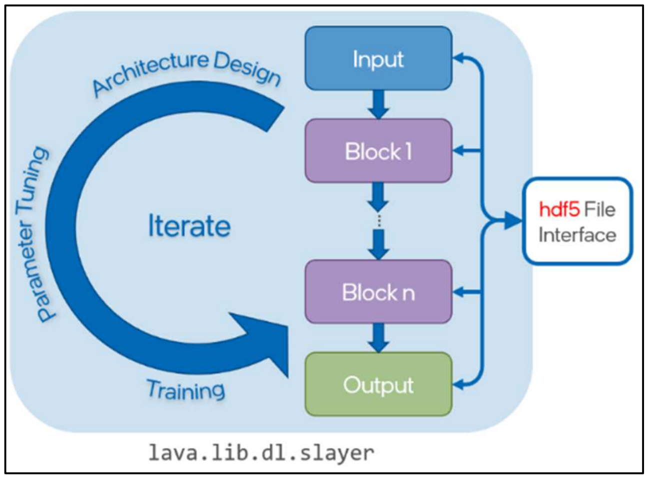

Fig. 2: Workflow Illustration of Direct Training with Slayer [8]

3. Compile and deploy the trained models for inferencing. With Lava-dl NetX, the hdf5 model file is converted into a Lava process which enables inferencing on both simulation and hardware environment. This workflow is illustrated in Fig 3.

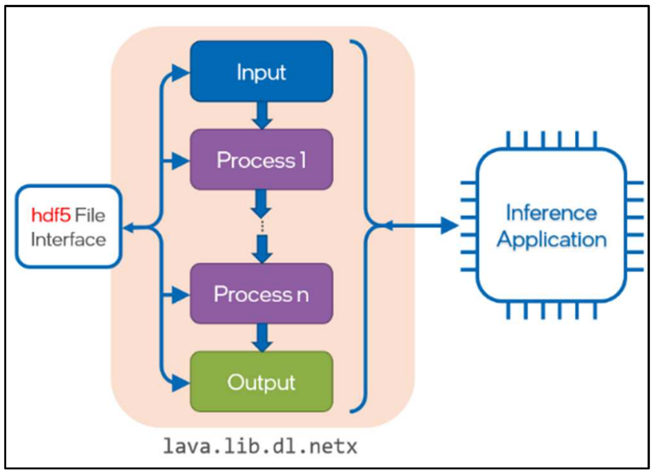

Fig 3: Workflow Illustration of Inferencing with NetX [8]

V. Spike-driven object detection Algorithm

Neuromorphic computing aims at a paradigm shift from Von Neumann-based architectures to distributed and co-integrated memory, the granularity at which this paradigm shift is achieved in digital implementations strongly varies between a distributed Von Neumann or full custom approaches [12, 13,14]. These custom chip approaches enable the implementation of various algorithms/methods.

Neuromorphic systems hold a critical position in the investigation of novel architectures, as the brain exemplifies an exceptional model for accomplishing scalable, energy-efficient, and real-time embodied computation [15]. It promises to realize artificial intelligence while reducing the energy requirements of computing platforms [16].

The object-detection algorithm utilized for this research was a Yolo-Kapoho-Point (Yolo-KP) model defined with Lava-dl Slayer. This model is based on the Yolov3-Tiny architecture [17] and optimized to run on both CPU simulation and Intel's 8-chip Kapoho Point System. For our benchmark comparison, we also trained a YOLOv8s model on the same datasets trained with Yolo-KP.

When pretrained deep learning models are deployed for real-time testing/inferencing, they are often used to process temporal/sequential data (e.g. video or input stream from a sensor). Convolutional neural networks (CNNs) such as YOLOv8 process video with a fixed amount of computation for each individual frame without consideration to the relationship/similarities from one frame to the next. But because some degree of spatial and temporal redundancy is expected in sequential inputs, performing the same computation for each frame can become computationally wasteful.

Yolo-KP provides a more efficient workflow using the following methods: taking in a sequence of images as a single input; and using Sigma-Delta neurons to define the layers in the model.

With Sigma-Delta networks (SDNN), for each new input, each layer in this network sends a discretized form of its change in activation to the next layer. Thus the amount of computation that the network does scales with the amount of change in the input and layer activations, rather than the size of the network [18]. Since Yolo-KP takes in temporal input, the activations do not change much. Therefore, the message between the layers are reduced which in turn reduces the synaptic computation in the next layer. In addition, with Lava-dl Slayer, the graded event values can encode the change in magnitude in one time-step which means there is no increase in latency at the cost of time-steps unlike the rate-coded SNNs.

VI. Data Description

The datasets used for analyses in this paper were comprised of images from various sensors including Electro-optical (EO), Infrared (IR), and Event-based (EBS) imagery. These images were taken from public datasets such as DSIAC [19], ThermalUAV [20] and M3ED [21]. Samples of targets are shown in **Error! Reference source not found.**.

VII. Technical and processing Approaches

One of the early challenges we faced during the preliminary training and testing of Yolo-KP models was small-object detection – or when the targets of interest are much smaller relative to the rest of the image or relative to other classes. One example is shown in Fig. 5 from ThermalUAV dataset.

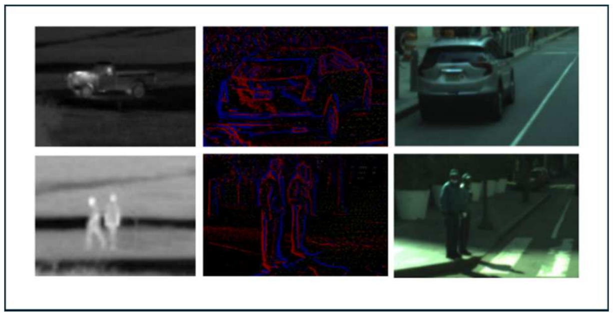

Fig. 4: 'Car' (top row) and 'Person' (bottom row) classes from different sensors. From the left column: IR, EBS, and EO.

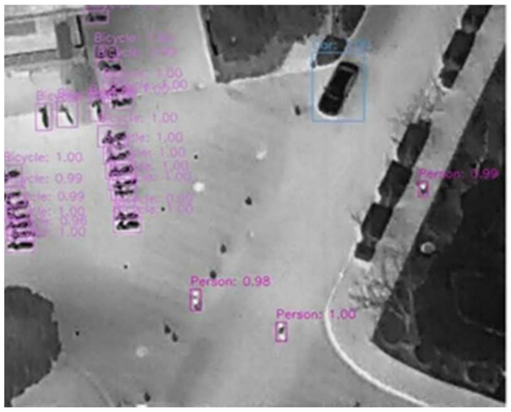

Fig. 5: Sample Training Image from ThermalUAV Dataset showing small targets (person, bicycle) and large targets (car) in a single image.

To improve training and inferencing results, datasets with small targets and/or mixture of large and small targets were tiled down to smaller subsets. This allows the model to 'zoom in' on smaller sections of the images and improve learning the features of smaller targets. The ThermalUAV dataset, for example, was tiled from the original size of 640x512 down to 448x448 and the network input size was also set to 448x448.

VIII. Results

Results of training Yolo-KP on tiled images are shown on TABLE I. below. We observed that Yolo-KP performed close to the baseline YOLOv8s models scores in terms of mAP50 scores.

TABLE I. Performance Comparison Between Yolo-KP and YOLOv8s

Dataset	Sensor	Yolo-KP mAP50	Yolo-KP Model Size (MB)	Yolov8s mAP50	Yolov8s Model Size
DSIAC	IR	87.5	13	91.3	87
M3ED	EO	85.8	12.7	92.0	85
M3ED	EBS	85.0	13.4	89.6	85
ThermalUAV	IR	84.7	14	90.5	82

To further optimize the models and get mAP scores closer to the baseline, we implemented an auto-anchor function based on the functionality used in YOLOv4/v5/v7 models. Prior to training, the bounding boxes for the training set is analyzed against the default anchor boxes. If the anchor boxes are larger or smaller by no more than 25% of the bounding boxes from the training set, then we leave the anchor boxes as is. However, if the difference is larger than the threshold, k-means clustering is used to set new anchor box values. The results of using auto-anchor are shown in TABLE II.

TABLE II. Performance Comparison Between Yolo-KP and YOLOv8s

Dataset	Sensor	Yolo-KP mAP50	Yolo-KP Model Size (MB)	Yolov8s mAP50	Yolov8s Model Size
DSIAC	IR	93.6	13	91.3	87
M3ED	EO	88.2	12.7	92.0	85
M3ED	EBS	86.3	13.4	89.6	85
ThermalUAV	IR	87.2	14	90.5	82

IX. Conclusion and Future Research

The analyses performed for this research helped provide insight and emphasis on the value and potential of using SNNs to train, develop, and deploy fast and efficient machine learning models. These results also serve as a baseline to help inform the next steps in both current and future research, which includes: exploring other preprocessing methods (augmentations) in order to improve the performance of Yolo-KP models for small object detection [22], implementing an anchorless version of Yolo-KP, similar to the latest versions of YOLO (v8 onwards), which may better suit Yolo-KP's architecture of only one output layer.

Lastly, the findings and lessons learned in this research illustrates the potential of neuromorphic models in other machine learning applications such as object tracking, pose/depth estimation, and segmentation and classification.

Acknowledgment

The authors would like to thank Dan Sherlock for his guidance, expertise, and technical contributions.

References

[1] Han Hu, Yonggang Wen, "Toward Scalable Systems for Big Data Analytics: A Technology Tutorial", 2014, 652-687. 10.1109/ACCESS.2014.2332453.

[2] Mark D. Barnell, Courtney Raymond, Matthew Wilson, Darrek Isereau, Chris Cicotta, Published 2020, Computer Science, 2020 IEEE High Performance Extreme Computing Conference (HPEC) .

[3] Chen, Chunlei, Zhang, Peng, Zhang, Huixiang, Dai, "Deep Learning on Computational-Resource-Limited Platforms: A Survey", *Mobile Information Systems*, 2020, 8454327, 19 pages, 2020. doi: 10.1155/2020/8454327

[4] J. D. Nunes, M. Carvalho, D. Carneiro and J. S. Cardoso, "Spiking Neural Networks: A Survey," in *IEEE Access*, vol. 10, pp. 60738-60764, 2022, doi: 10.1109/ACCESS.2022.3179968.

[5] Intel Technology Brief, "Taking Neuromorphic Computing to the Next Level with Loihi 2", https://download.intel.com/newsroom/2021/new-technologies/neuromorphic-computing-loihi-2-brief.pdf

[6] M. Davies et al., "Loihi: A Neuromorphic Manycore Processor with On-Chip Learning," in IEEE Micro, vol. 38, no. 1, pp. 82-99, January/February 2018, doi: 10.1109/MM.2018.112130359.

[7] C. -K. Lin et al., "Programming Spiking Neural Networks on Intel's Loihi," in Computer, vol. 51, no. 3, pp. 52-61, March 2018, doi: 10.1109/MC.2018.157113521.

[8] Lava: A software framework for neuromorphic computing. https://github.com/lava-nc/lava, 2021.

[9] Lava DL: https://github.com/lava-nc/lava-dl, 2021.

[10] Yujie Wu et al., “Direct training for spiking neural networks: Faster, larger, better,” in Proceedings of the AAAI Conference on Artificial Intelligence, 2019, vol. 33, pp. 1311–1318.

[11] Wenzhe Guo et al., “Neural coding in spiking neural networks: A comparative study for robust neuromorphic systems,” Frontiers in Neuroscience, vol. 15, pp. 212, 2021.

[12] C. Frenkel, D. Bol, and G. Indiveri, Bottom-Up and Top-Down Neural Processing Systems Design: Neuromorphic Intelligence as the Convergence of Natural and Artificial Intelligence. CoRR abs/2106.01288, 2021.

[13] Cristina Silvano, Daniele Ielmini, Fabrizio Ferrandi, Leandro Fiorin, Serena Curzel, Luca Benini, et al. "A Survey on Deep Learning Hardware Accelerators for Heterogeneous HPC Platforms." arXiv preprint arXiv:2306.15552, 2023.

[14] Y. S. Yang and Y. Kim, "Recent Trend of Neuromorphic Computing Hardware: Intel's Neuromorphic System Perspective," 2020 International SoC Design Conference (ISOCC), Yeosu, Korea (South), 2020, pp. 218-219, doi: 10.1109/ISOCC50952.2020.9332961.

[15] J. Yik, et al. "NeuroBench: Advancing neuromorphic computing through collaborative, fair and representative benchmarking." arXiv preprint arXiv:2304.04640, 2023.

[16] K. Roy, A. Jaiswal and P. Panda, "Towards spike-based machine intelligence with neuromorphic computing", *Nature*, vol. 575, pp. 607-617, 2019.

[17] Adarsh, Pranav & Rathi, Pratibha & Kumar, Manoj. (2020). YOLO v3-Tiny: Object Detection and Recognition using one stage improved model. 687-694. 10.1109/ICACCS48705.2020.9074315.

[18] Peter O’Connor and Max Welling, “Sigma Delta Quantized Networks”, 2017 International Conference on Learning Representations (ICLR), arXiv:1611.02024.

[19] Defense Systems Information Analysis Center (DSIAC) Automated Target Recognition (ATR) dataset

[20] K. Chaney *et al.*, "M3ED: Multi-Robot, Multi-Sensor, Multi-Environment Event Dataset," *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Vancouver, BC, Canada, 2023, pp. 4016-4023, doi: 10.1109/CVPRW59228.2023.00419.

[21] Jiashun Suo, Tianyi Wang, Xingzhou Zhang, Haiyang Chen, Wei Zhou and Weisong Shi, “HIT-UAV: A high-altitude infrared thermal dataset for Unmanned Aerial Vehicle-based object detection”, Scientific Data 10, 227 (2023).

[22] Kisantal, M., Wojna, Z., Murawski, J., Naruniec, J. and Cho, K., 2019. “Augmentation for small object detection”, *arXiv:1902.07296*.