An IEEE Societies and Technical Councils Publication
IEEE Data Descriptions

Received 00 April, 2024; revised 11 April, 2024; accepted 00 April, 2024; Date of publication 00 May, 2024; date of current version 24 June, 2024.

Digital Object Identifier 10.1109/DD.2024.0624000

Descriptor: *Smart Event Face Dataset* (SEFD)

Riadul Islam, *Senior Member, IEEE*, Sri Ranga Sai Krishna Tummala, Joey Mulé, Rohith Kankipati, Suraj Jalapally, Dhandeep Challagundla, *Student Member, IEEE*, Chad Howard, and Ryan Robucci, *Member, IEEE*

ABSTRACT

Smart focal-plane and in-chip image processing has emerged as a crucial technology for vision-enabled embedded systems with energy efficiency and privacy. However, the lack of special datasets providing examples of the data that these neuromorphic sensors compute to convey visual information has hindered the adoption of these promising technologies. Neuromorphic imager variants, including event-based sensors, produce various representations such as streams of pixel addresses representing time and locations of intensity changes in the focal plane, temporal-difference data, data sifted/thresholded by temporal differences, image data after applying spatial transformations, optical flow data, and/or statistical representations. To address the critical barrier to entry, we provide an annotated, temporal-threshold-based vision dataset specifically designed for face detection tasks derived from the same videos used for Aff-Wild2. By offering multiple threshold levels (e.g., 4, 8, 12, and 16), this dataset allows for comprehensive evaluation and optimization of state-of-the-art neural architectures under varying conditions and settings compared to traditional methods. The accompanying tool flow for generating event data from raw videos further enhances accessibility and usability. We anticipate that this resource will significantly support the development of robust vision systems based on smart sensors that can process based on temporal-difference thresholds, enabling more accurate and efficient object detection and localization and ultimately promoting the broader adoption of low-power, neuromorphic imaging technologies. To support further research, we publicly released the dataset at https://dx.doi.org/10.21227/bw2e-dj78.

IEEE SOCIETY/COUNCIL Signal Processing Society (SPS)
DATA DOI/PID 10.21227/bw2e-dj78
DATA TYPE/LOCATION Images; Maryland, USA

INDEX TERMS Face detection, dynamic vision sensing (DVS), Aff-Wild, sparse vision, convolutional neural network (CNN), Face dataset.

Background

Algorithm development for smart sensors, those that integrate coincidental sensing and processing, is complicated by a lack of emulation models and challenges in producing synthetic data for testing. The characteristic advantage of focal-plain and on-chip processing for image sensors is that they do not transmit data irrespective of vision tasks. Instead, such sensors sift and transform signal data into representations of essential information for vision tasks. Therefore, the average sample-reporting rate is far below the Nyquist Rate, which means a loss of the ability to reconstruct all aspects of the original image by traditional standards. Furthermore, the parameters of this non-invertible signal pre-processing are often adaptive and controlled algorithmically. This presents the two-fold challenge of developing dependent algorithms for controlling the parameters of sensing and back-end algorithms for computer vision.

State-of-the-art (SOTA) approaches to supervised training for machine learning (ML) algorithms involve a prolonged iterative approach to the co-determination of salient features from source data along with other processing parameters for inference. These algorithms rely on unbiased annotation of source data [1], [2] to guide convergence on parameters of feature extraction and the structures and details of image analysis. However, the key difference is that emulation models for the influence of dynamic parameter

selection are lacking which makes exploration challenging. Even more fundamental is that the underlying characteristic frequency of smart sensors, including those based on ideas from neuromorphic and event-based image sensors [3], [4], is much higher than conventional sensors. This altogether means that the information presented to back-end algorithms is not readily synthesized from widely available public data. Approaches to temporal interpolation are sometimes required to synthesize events with a higher temporal resolution than those of the sensors used to produce traditional video datasets. Furthermore, even when existing hardware is accessible for testing, experimental parameter sweeps are non-trivial since the behavior of the subjects of observation, be it a human or cars traveling down the highway, are not typically exactly reproducible nor are the environmental conditions such as illumination. Therefore the goal of the community must be to decouple some of the challenges and provide means to step into algorithm development for these platforms without requiring researchers and developers to first address all of these issues. In this spirit, we present an annotated dataset representing a practical application of neuromorphic processing to face detection. The application is assumed to involve modest temporal frequencies and would not require temporal interpolation, but a small embedded vision system embodying efficiency and/or privacy would benefit greatly if architected around a smart/event-based camera.

Motivation

Event cameras are a promising technology increasingly used for computer vision applications because of their high temporal resolution and low power consumption [5]. In contrast to conventional synchronous energy-efficient techniques [6]–[11], event sensors introduce a paradigm shift by enabling low-power operation through distributed processing. However, one of the challenges in developing algorithms for event cameras, such as face detection, is the need for suitable datasets to undertake initial steps in algorithm development. Unlike traditional cameras that capture a sequence of frames that can be annotated later, event cameras capture a series of sparse events [12] that could be asynchronously detected or at least processed at a temporal resolution much higher than conventional low-power CMOS camera frame rates. This makes creating datasets for training and testing face detection algorithms difficult. While some datasets exist, like N-Caltech 101, they are typically small and do not represent the faces in real-world scenarios like different poses, illumination, etc. [13]. Several other face detection datasets, for example, the WIDER FACE dataset [14], the CelebA dataset [15], and the face detection dataset and benchmark (FDDB) [16]. However, none of these datasets provide motion data for faces.

Existing event-based vision-related benchmark datasets, for example, the N-Caltech101 [13], N-mnist, Poker-DVS [17], event data for pose, visual odometry, and SLAM [18] are generated by mounting a dynamic vision sensor (DVS) on a motorized pan-tilt unit and having the sensor move while it views. The events are generated by emulated saccades (i.e., capturing events by changing the eye's position) rather than independent object motion in the field of view. Existing datasets also use a fixed threshold, the difference in pixel intensity between consecutive image frames or the difference between pixel intensity considering a current frame and a reference frame. However, herein, a multi-threshold dataset is presented to facilitate low-bandwidth event-vision. The availability of a threshold (T_h) – based image dataset can enable fine-tuning new neural architectures for performing well on sparse images. Moreover, the T_h-based dataset will be fundamentally crucial to designing novel self-adaptive smart vision sensors.

Main Contributions

The specific contributions of this work are:

- We present a new smart event face dataset (SEFD) of multiple programmable digital thresholds to decouple the challenges of modeling smart sensors and initial algorithm development.
- We analyzed pixel activity concerning T_h values to characterize event-based object detection.
- We validate the effectiveness of the proposed dataset through training of industry-standard object detection and localization models.

Existing Event-Based Vision

Empirically, event-based vision sensors [4], [19]–[26] are promising for high-speed vehicles, robotics, drones, and moving object detection. Optical-flow features are of special interest in bio-inspired approaches, and researchers have developed an algorithm to derive optical flow features from event-based data captured by dynamic vision sensing (DVS) cameras [23]. The optical flow-based system employs a block-matching technique [27] to estimate the flow between sequential DVS event frames. This involves dividing the frames into blocks and finding matching blocks between consecutive frames to determine the motion. Another interesting event simulator (ESIM) captures event data using a rendering engine and image sensor trajectory [28]. The ESIM uses an image brightness gradient for adaptive sampling.

Other researchers have facilitated DVS using the v2e toolbox [29]. This toolbox employs various parameters, including temporal noise, leak events, pixel intensity, and the Gaussian threshold method. While proficient in generating event data, it is noteworthy that the v2e toolbox lacks explicit guidelines for creating event-based facial data, a distinctive feature of our proposed approach. Another biological retina-based tool, namely RetinoSim, was designed to synthesize event-based data for exploring neuromorphic vision architectures [30]. It addresses the need for realistic and diverse event-based data to develop and evaluate such architectures effectively. In a hybrid approach, researchers used "frame of events" to combine conventional frame scanning and DVS

approaches to improve computational latency and memory consumptions [31].

Existing Neural Network Architectures for Image Classification and Localization

The task of object detection and localization within the domain of event-vision presents a non-trivial challenge, primarily stemming from the absence of salient feature representations within the input data. In recent years, you only look once (YOLO) creates great attention from researchers for object detection and localization [32]. The initial version of the YOLO model uses 24 convolutional layers and two fully-connected layers. This approach divides an image into $S \times S$ grids and detects an object based on the location of the center of an object that falls into the grid. To improve the performance, YOLO-v4 [33] uses weighted residual connections (WRC) and cross-stage partial connections (CSPs).

In this work, we benchmarked our dataset using YOLO-v4 [33] and YOLO-v7 [34]. YOLO-v4 consists of backbone, neck, and head networks. When YOLO-v4 targets GPU, the backbone network uses CSP Network (i.e., CSPDarknet53) [35]. YOLO-v7 introduces an updated and optimized architecture compared to previous YOLO versions, with the use of Extended Efficient Layer Aggregation Network (E-ELAN) for better feature map integration compared to YOLO-v4's PANet and SPP.

Google Brain research team proposed an interesting architecture for object detection and localization called EfficientDet-b0 [36]. Another set of interesting convolutional neural network (CNN)-based architectures proposed by the Google research team, especially suitable for embedded and mobile vision applications, called MobileNets-v1 [37]. Both EfficientDet-b0 and MobileNets-v1 will be used in this research for benchmarking.

Collection Methods and Design

Description of Source Dataset

In this work, we used the Aff-Wild2 dataset as input source videos [38]. The Aff-Wild2 dataset features emotional descriptors described in terms of valence and arousal. Valence indicates the intensity of positive and negative emotions, while the latter suggests the power of triggering an emotion. The dataset contains 298 videos of 200 subjects (about 15 hours of data) annotated by seven lay experts considering valence and arousal, all captured in a natural state without any external stimulation.

Proposed Multi-Threshold Image Generation Tool Flow

The existing face detection benchmark datasets do not provide motion data to capture pixel activity [14]–[16]. Our research resolves this issue with our proposed smart event generation tool flow and datasets. The proposed binary image generation framework is shown in Figure 1. In this tool flow, we used an input video file to generate image frames considering the frame rate or temporal difference of frames. Then, we used our proposed binary event generator to create binary image frames considering a set of threshold values, as shown in Figure 1.

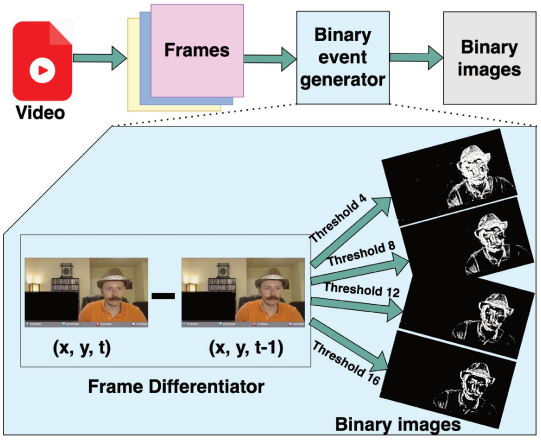

FIGURE 1: The proposed tool flow converts raw video data into image frames, and then the binary event generator differentiates temporally different image frames and a set of reference threshold values to generate thresholded images.

TABLE 1: The proposed tool flow used audio video interleave (AVI) format video length ranging from 0.1 min to 14.47 min and converted to (threshold = 4, 8, 12, and 16) image frames of portable network graphic (PNG) format.

Attributes	Video Descriptions	Image Descriptions			
		$T_h = 4$	$T_h = 8$	$T_h = 12$	$T_h = 16$
Length of videos	0.10–14.47 min	–	–	–	–
Video format	AVI	PNG	PNG	PNG	PNG
Avg. image resolution (AIR)	416 × 416	416 × 416	416 × 416	416 × 416	416 × 416
Standard deviation of AIR	0	0	0	0	0
Median image resolution	416 × 416	416 × 416	416 × 416	416 × 416	416 × 416
# of Videos/ Images	100	9130	9130	9130	9130

The proposed threshold-based smart event generation methodology is shown in Algorithm 1. The algorithm uses a video file (Vi), the T_h, and a user-defined frame temporal difference (T_p) as inputs. First, the algorithm generates image frames using the $GenFrame()$ function considering frame temporal differences in Line 3. It recursively derives the previous frame from the current frame to compute the differential frame from Line 4 to Line 7. Depending on the threshold value, we update the differential frame pixel intensity from Line 8 to Line 15. Finally, the algorithm returns the binary frames in Line 16.

Image Annotation Guideline

For object detection and localization, building a bounding box (BBox) of a rectangular portion of an object is standard practice. We used four lay expert human annotators to draw rectangular boxes around objects of interest in an image (i.e., face) using Roboflow [39] to provide training data for object detection and recognition algorithms. Three additional lay experts review the annotations visually to ensure accuracy and consistency.

For consistency in the process of annotating facial regions with bounding rectangles, a set of explicit guidelines was

Algorithm 1 T_h-based smart event generation algorithm

Input: Vi, input video; T_h, threshold; T_p, temporal difference
Output: Sparse video, videos with frames featuring motion smart events
$Frames = \{frame_0, frame_1, \ldots, frame_n\} = GenFrame(Vi, T_p)$ ▷ Generate image frames from the input video file.
for all $frame_{i+1} \in Frames$ **do**
    $CurrFrame = frame_{i+1}$ ▷ Initialize current frame.
    $PrevFrame = frame_i$ ▷ Initialize previous frame.
    $DiffFrame_i = \{px_{(0,1)}, \ldots, px_{(M,N)}\} = CurrFrame - PrevFrame$ ▷ Compute a differential frame considering an image size of $M \times N$.
    for all $px_{(x,y)} \in DiffFrame_i$ **do**
        if $px_{(x,y)} > T_h$ **then**
            $px_{(x,y)} = 1$ ▷ Detecting pixel activity
        else
            $px_{(x,y)} = 0$
        end if
    end for
end for
return $DiffFrames = \{DiffFrame_1, DiffFrame_2, \ldots, DiffFrame_{n-1}\}$ ▷ Return differential frames.

used. These guidelines detail the procedure for incorporating facial landmarks to encapsulate the face within the rectangle effectively. It is of paramount importance, under all circumstances, to ensure that the bounding rectangle maintains a high degree of precision and does not deviate from an appropriate size range, avoiding both excessive enlargement and undue reduction. The size and placement of the rectangle, throughout the duration of the video or image sequence, should remain consistent with the facial features, thereby upholding a standardized and coherent representation of the face.

The dataset consists of three basic poses (i.e., frontal, profile, and angular), as shown in Figure 2. For the frontal pose, the annotators draw a rectangle around the entire face, including the brow, chin, and cheeks. The rectangle should be parallel to the face, and the diagonal should be in alignment with the horizontal plane of the face. For the profile pose, a rectangle was drawn around the face's visible area containing the brow, nose, and chin but not the hair. The rectangle's diagonal should be aligned with the vertical plane of the face. Finally, for the angular pose, the annotator drew a rectangle around the face's visible area containing the brow, nose, and chin but not the hair. The diagonal lines of the rectangle will be at an angle to the edges of the image, and the angle may vary depending on the degree of tilt of the face, as shown in Figure 2.

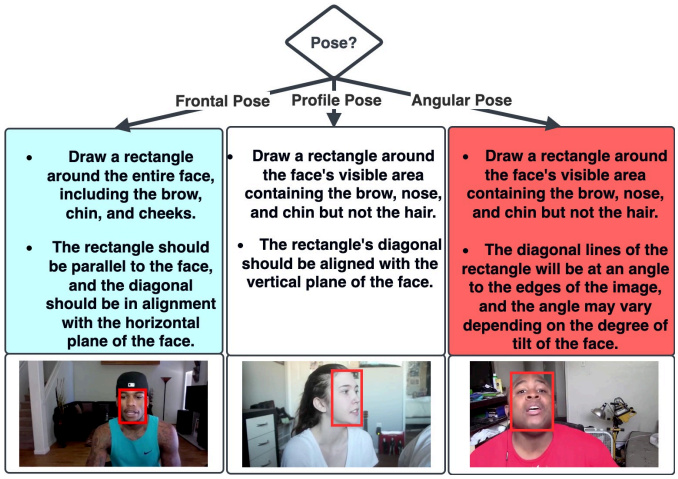

FIGURE 2: Face annotation involved creating a rectangle over the face, with the rectangle's orientation varying by pose: parallel and horizontally aligned for frontal, vertically aligned for profile, and angled for angular, depending on the face's tilt.

The data source is the original data from the Aff-Wild2 database. The video lengths ranged from 0.1 min to 14.47 min [38], as shown in Table 1. The image data was resized so that the average image resolution (AIR) was converted from the original 607×359 to 416×416. On average 100 images were extracted from each video resulting in 10k full-frame images. From each of these these $\sim 9.1k$ images were produced considering each threshold (i.e., 4, 8, 12, and 16) using Algorithm 1, resulting in additional $\sim 36.5k$ images.

Validation and Quality

We implement the proposed threshold-based smart event generation Algorithm 1 using Python programming language. All the computation was performed on an Intel Xeon Gold processor with 64 GB RAM, equipped with an NVIDIA Quadro P4000 GPU using Ubuntu 22.04. We used GPU to train, validate, and test SOTA neural network architectures using the proposed dataset. The Algorithm 1 used Aff-Wild2 video dataset [38] for threshold-based event-frame generation. For analysis, we used widely used anchor boxes-based highly accurate YOLO-v4 and YOLO-v7 models. We also used EfficientDet-b0 [36], which is designed to be a lightweight yet effective model for object detection tasks. In addition, we used MobileNets-v1 [37], emphasizing computational efficiency and adaptability, which is generally used in scenarios where real-time, on-device object detection and localization are required.

Pixel Activity Computation

In general, event-based vision sensing is based on *pixel activity*— a term used herein to refer to pixels associated with a change in sensed light intensity that surpasses a given T_h and triggers communication. The sensors detect spatial motion of objects through the change of pixel intensities.

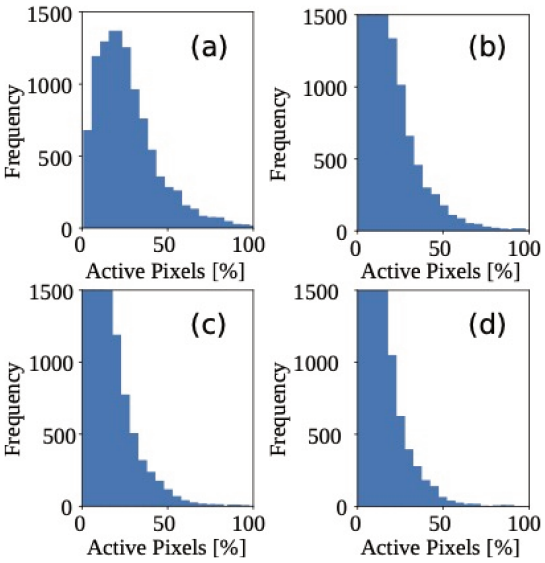

FIGURE 3: For a 333 ms frame time difference, lower thresholds yield higher average active pixel rates, with (a) $T_h = 4$ yielding 25.39% activity and (b) $T_h = 8$ yielding 17.86%, (c) $T_h = 12$ yielding 14.27% and (d) $T_h = 16$ yielding 11.94%.

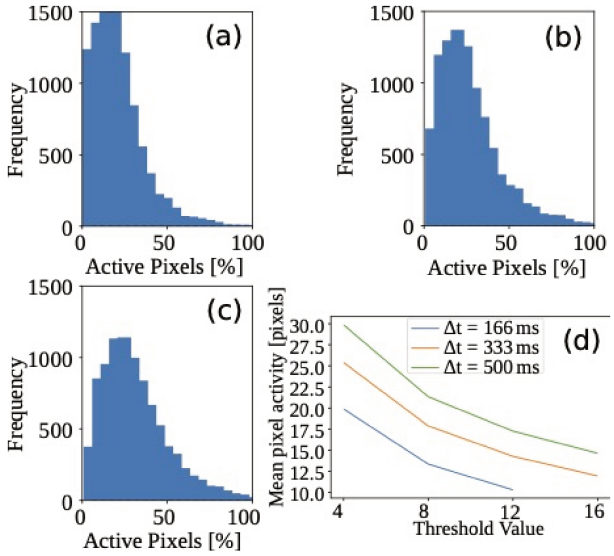

FIGURE 4: For a constant T_h of 4, the average active pixel count increases with the larger interframe time difference, (a) 19.88% average pixel activity at 166 ms, (b) 25.40% at 333 ms, and (c) 29.86% at 500 ms. Conversely, (d) the average active pixel count decreases as T_h increases.

Pixel activity shows which pixels detect motion at various thresholds. Analyzing pixel activity allows for the optimization of threshold parameters, enhancing accuracy and efficiency in event detection. Hence, we performed analysis on the *pixel activity* of the proposed dataset considering both the temporal and the intensity difference as parameters.

In order to compute the *pixel activity*, we considered intensity threshold values of 4, 8, 12, and 16. Figure 3 shows the analysis regarding the temporal difference of frames of 333 ms. According to our analysis on $\sim 10k$ images, each image contains 25.39% active pixels on average, considering a T_h of 4, as shown in Figure 3(a). For T_h of 8, each image contains 17.86% active pixels on average, as shown in Figure 3(b). Using $\sim 10k$ images and considering a T_h of 12, we observed 14.27% of average *pixel activity*, as shown in Figure 3(c). Using the same amount of images, when we considered a T_h value of 16, we observed 11.94% average *pixel activity*, as shown in Figure 3(d).

Besides, we computed the *pixel activity* inside the BBox, which is critically important for object detection and localization. For a T_h value of 16, the average number of active pixels in the BBox is 1.13%. As expected, by decreasing the intensity T_h value, the average *pixel activity* in the bounding boxes increased. When considering T_h values of 8 and 4, the average active pixels in bounding boxes are 1.68% and 2.34%, respectively.

In addition, we traced the pixel activities due to variations in the frame's temporal differences. The average *pixel activity* is 19.88%, 25.40%, and 29.86% considering temporal frame difference of 5 (i.e., 166 ms), 10 (i.e., 333 ms), and 15 (i.e., 500 ms), as shown in Figure 4(a), Figure 4(b), and Figure 4(c), respectively. Figure 4(d) exhibits the overall *pixel activity* considering the variation of T_h values and image frames temporal differences. Clearly, there is a positive correlation between temporal difference and *pixel activity*, which implies that when the time delay between two frames rises, so does pixel activation. The T_h, on the other hand, has an anticorrelation with *pixel activity*. *Pixel activity* reduces as the T_h value increases, which indicates that an imager with a higher T_h would produce fewer pixel activations per frame on average. In contrast, a lower T_h would create more pixel activations per frame.

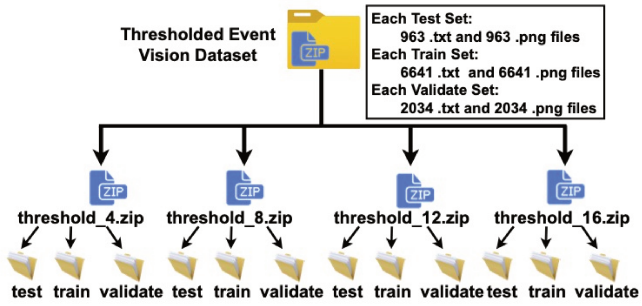

FIGURE 5: Outline of dataset directory structure

Records and Storage

The dataset is organized into multiple ZIP files (Figure 5), each representing a different threshold level, specifically named threshold_4.zip, threshold_8.zip, threshold_12.zip, and threshold_16.zip. Within each ZIP file, the data is structured into three sub-directories: test, train, and validate, following common conventions for NN implementations. Each sub-directory contains a set of text (annotation) and image files. The test, train, and validate sub-directories hold 904 text and 904 image files, 6392 text and 6392 image files,

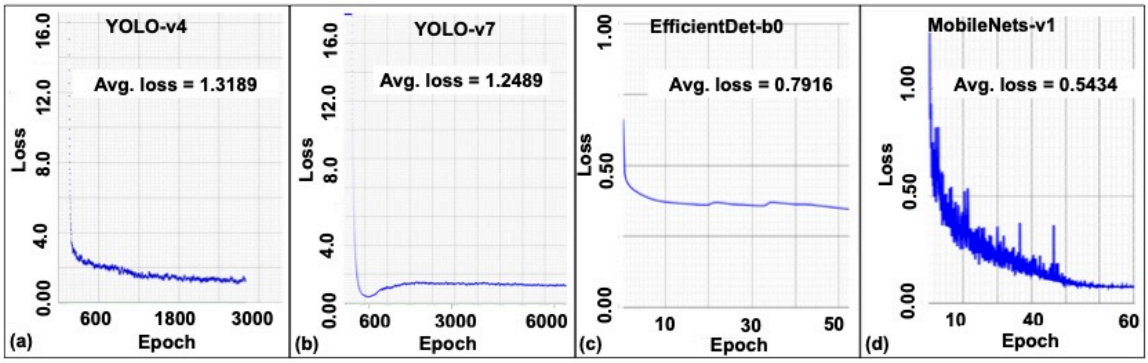

FIGURE 6: All benchmark models achieved low training losses: (a) YOLO-v4 reached an average loss of 1.3189 after 3,000 epochs, (b) YOLO-v7 recorded an average loss of 1.2489 after 6,000 epochs, (c) EfficientDet-b0 converged quickly, achieving an average loss of 0.7916 within 50 epochs, and (d) MobileNets-v1 achieved the lowest average loss of 0.5434 in just 60 epochs.

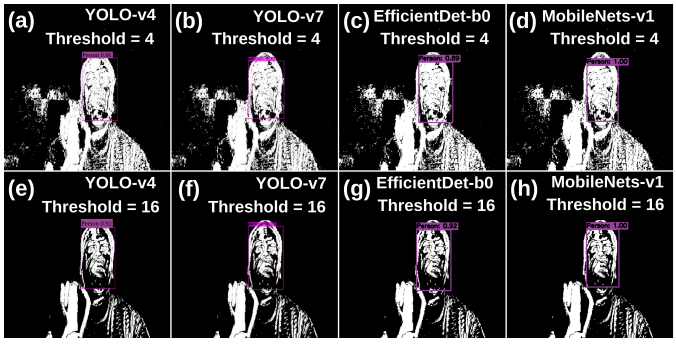

FIGURE 7: A comprehensive assessment of the T_h-based dataset using SOTA neural networks demonstrated efficient object detection and localization. (a-d) Models YOLO-v4, YOLO-v7, EfficientDet-b0, and MobileNets-v1 applied to $T_h = 4$ images showed high detection performance, with YOLO-v7 achieving a confidence level of 94.1%. (e-h) For $T_h = 16$ images, all models maintained strong performance, although YOLO-v7 exhibited a slight drop in confidence level to 91.7%.

and 1834 text and 1834 image files, respectively. The txt files correspond to labels of the png files and follow the standard bounding box format used in YOLO and Darknet models.

Insights and Notes

Benchmarking with SOTA NN Architectures

In order to enable event-vision using the proposed event-based face dataset, we used SOTA neural network architectures. Among different existing architectures, we used large YOLO-v4 model [35], YOLO-v7 [34], and two smaller sized high-performance models, EfficientDet-b0 [36] and MobileNets-v1 [37]. In this analysis, we used all four proposed thresholds (i.e., 4, 8, 12, and 16) datasets, as shown in Table 2.

For the YOLO-v4 and YOLO-v7 models, we used 6359 images of each threshold dataset. Once trained, we used 915 images from each of the different T_h value datasets for testing. The YOLO-v4 and YOLO-v7 models have an average training loss of 1.3189 and 1.2489, as shown in Figure 6(a) and Figure 6(b), respectively.

Like YOLO models, the EfficientDet-b0 and the MobileNets-v1 models used all four threshold-valued datasets for training and testing. The EfficientDet-b0 and the MobileNets-v1 models have an average training loss of 0.7916 (Figure 6(c)) and 0.5434 (Figure 6(d)).

We measured the performance of each neural network architecture considering Precision, Recall, F1-score, intersection over union (IoU), average Precision at 50% IoU (AP50), and average Precision at 75% IoU (AP75). The results demonstrate evidence of object detection with a confidence level of 86% for $T_h = 4$ and 92% for $T_h = 16$ when utilizing the YOLO-v4 model, as presented in Figure 7(a) and Figure 7(e), respectively. Predictively, using $T_h = 4$ dataset, the YOLO-v7 model has a better average confidence level of 94.1% compared to $T_h = 16$ dataset, which has a confidence level of 91.7%, as shown in Figure 7(b) and Figure 7(f), respectively. Meanwhile, Figure 7(c) and Figure 7(g) showcase the outcomes of object detection with confidence levels of 89% for $T_h = 4$ and 92% for $T_h = 16$, and localization using the EfficientDet-b0.

Remarkably, for the same sample, the MobileNets-v1 model exhibited superior performance, achieving object detection with a confidence level of 100% for T_h = 4 and 100% for T_h = 16, and successfully localizing the object, as illustrated in Figure 7(d) and Figure 7(h), respectively.

Considering all four proposed T_h values, the average test Precision, Recall, and F1-score are 94.30%, 90.50%, and 92.50%, respectively. Besides, the YOLO-v4 model achieves, on average, 72.54%, 92.96%, and 40.17% IoU, AP50, and AP75, respectively, considering equal amounts of test data for each kind of T_h, as shown in Table 2. Besides, the YOLO-v4 model achieves the best IoU compared to other higher threshold values using $T_h = 4$. The YOLO-v7 has

TABLE 2: The YOLO-v4 model exhibits a parameter count that is notably greater, approximately 15.46× more than that of the EfficientDet-b0 model and about 9.97× more than that of the MobileNets-v1 model; furthermore, in the context of the proposed T_h = 8 datasets, the MobileNets-v1 model attains the highest IoU of 82.53% among the models considered, and the EfficientDet-b0 model demonstrates the highest average Precision at 50% (AP50) of 98.08% within the same T_h = 8 valued dataset, surpassing the performance of other architectures and datasets with different thresholds.

Models	# of Parameters	T_h	Dataset split		Metrics			IoU (%)	AP50	AP75
			Training	Testing	Precision	Recall	F1-score			
YOLO-v4 [33]	60.3M	4	6392	904	94.00	89.00	91.00	72.77	90.93	45.85
		8	6392	904	89.00	89.00	89.00	69.04	90.37	44.57
		12	6392	904	91.00	86.00	89.00	70.57	90.32	43.89
		16	6392	904	93.00	79.00	85.00	71.36	86.54	38.74
YOLO-v7 [34]	25.2M	4	6392	904	94.00	92.00	93.00	75.25	94.55	60.80
		8	6392	904	92.00	90.00	91.00	73.62	91.82	64.31
		12	6392	904	95.00	87.00	91.00	76.40	92.45	58.12
		16	6392	904	95.00	95.00	90.00	76.11	90.73	56.45
EfficientDet-b0 [36]	3.9M	4	6392	904	97.78	99.34	98.56	80.45	97.78	76.38
		8	6392	904	98.08	99.45	98.76	80.53	98.08	78.77
		12	6392	904	97.63	99.34	98.48	80.06	97.63	75.72
		16	6392	904	97.29	99.34	98.30	78.69	97.29	70.91
MobileNets-v1 [37]	6.05M	4	6392	904	95.90	97.92	96.90	80.92	95.90	68.77
		8	6392	904	96.69	98.36	97.52	82.53	96.69	70.92
		12	6392	904	96.61	98.36	97.48	81.36	96.61	66.52
		16	6392	904	96.01	98.14	97.06	80.13	96.01	66.58

similar test Precision, Recall, F1-score, IoU, and AP50; however, 32.96% better AP75 compared to the YOLO-v4.

Considering all four threshold values, the EfficientDet-b0 architecture achieved an average test Precision, Recall, and F1-score are 97.70%, 99.37%, and 98.53%, respectively. In addition, the EfficientDet-b0 model achieves, on average, 79.93%, 97.70%, and 75.45% IoU, AP50, and AP75, respectively, considering equal amounts of test data for each kind of T_h, as shown in Table 2. Among all the models, MobileNets-v1 has highest average Precision, Recall, F1-score, and IoU of 96.30%, 98.19%, 97.24%, and 81.24%, respectively.

The average inference time was derived using 10 trials each involving 100 unique image inferences. The YOLO-v4 requires 30.83% and 27.99% more inference time compared to EfficientDet-b0 and MobileNets-v1 models, respectively, as shown in Figure 8. The YOLO-v4 has 114.70× and 1.37× more FLOPS/MACs compared to EfficientDet-b0 and MobileNets-v1 models, respectively, as shown in Figure 8.

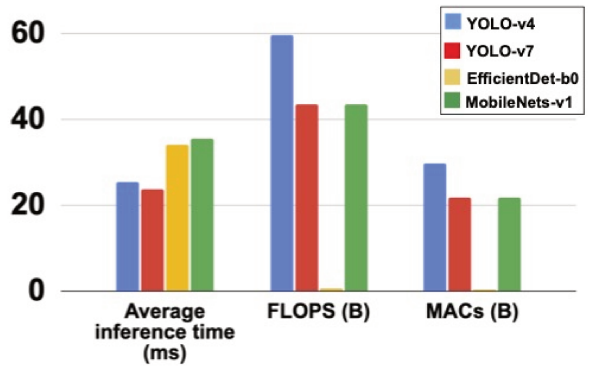

FIGURE 8: YOLO-v4 has the longest inference time and highest FLOPS/ MACs demand, requiring 30.83% more time than EfficientDet-b0 and 27.99% more than MobileNets-v1. In contrast, MobileNets-v1 has the shortest inference time and the lowest computational demand.

Source Code and Scripts

In this study, we utilized the Aff-Wild2 video dataset [38] and conducted analysis using open-source SOTA NN models. The dataset analysis scripts are available at the following GitHub repository: https://github.com/riaduli/Thresholded_event_vision_face_dataset.

Acknowledgment and Interests

Conceptualization, R. Islam and S.R.S.K. Tummala; dataset and analysis, R. Islam, J. Mulé, R. Kankipati, R. Robucci, S. Jalapally, and S.R.S.K Tummala; original draft preparation, R. Islam; Review and editing, R. Robucci, C. Howard, and D. Challagundla; funding acquisition, R.Islam and R. Robucci.

This research was funded by the National Science Foundation (NSF) under award number 2138253, the Maryland Industrial Partnerships (MIPS) program under award number MIPS0012, and the UMBC Startup grant. Special thanks to Raiyan Zaman and Rishi Mulchandani for assisting in annotating part of the dataset.

The article authors have declared no conflicts of interest.

REFERENCES

[1] R. Padilla, C. Costa Filho, and M. Costa, "Evaluation of haar cascade classifiers designed for face detection," *World Academy of Science, Engineering and Technology*, vol. 64, pp. 362–365, 2012.

[2] W. Yuan and S. Ramalingam, "Fast localization and tracking using event sensors," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 4564–4571.

[3] A. Amir, B. Taba, D. Berg, T. Melano, J. McKinstry, C. Di Nolfo, T. Nayak, A. Andreopoulos, G. Garreau, M. Mendoza, J. Kusnitz, M. Debole, S. Esser, T. Delbruck, M. Flickner, and D. Modha, "A low power, fully event-based gesture recognition system," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 7388–7397.

[4] G. Gallego, T. Delbrück, G. Orchard, C. Bartolozzi, B. Taba, A. Censi, S. Leutenegger, A. J. Davison, J. Conradt, K. Daniilidis, and D. Scaramuzza, "Event-based vision: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 1, pp. 154–180, 2022.

[5] E. Mueggler, G. Gallego, H. Rebecq, and D. Scaramuzza, "Continuous-time visual-inertial odometry for event cameras," *IEEE Transactions on Robotics*, vol. PP, pp. 1–16, 08 2018.

[6] R. Islam, "Negative capacitance clock distribution," *IEEE Transactions on Emerging Topics in Computing*, vol. 9, no. 1, pp. 547–553, 2021.

[7] J.-S. Hyeon, S.-H. Kim, and H.-J. Kim, "A low-power CMOS image sensor with multiple-column-parallel readout structure," *IEEE Journal of the Electron Devices Society*, vol. 10, pp. 180–187, 2022.

[8] R. Islam, H. A. Fahmy, P. Y. Lin, and M. R. Guthaus, "DCMCS: Highly robust low-power differential current-mode clocking and synthesis," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 26, no. 10, pp. 2108–2117, 2018.

[9] K. Park, S. Yeom, and S. Y. Kim, "Ultra-low power CMOS image sensor with two-step logical shift algorithm-based correlated double sampling scheme," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 67, no. 11, pp. 3718–3727, 2020.

[10] R. Islam, B. Saha, and I. Bezzam, "Resonant energy recycling SRAM architecture," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 68, no. 4, pp. 1383–1387, 2021.

[11] R. Islam, "Low-Power Highly Reliable SET-Induced Dual-Node Upset-Hardened Latch and Flip-Flop," *IEEE Canadian Journal of Electrical and Computer Engineering*, vol. 42, no. 2, pp. 93–101, 2019.

[12] G. Gallego, T. Delbrück, G. Orchard, C. Bartolozzi, B. Taba, A. Censi, S. Leutenegger, A. J. Davison, J. Conradt, K. Daniilidis, and D. Scaramuzza, "Event-based vision: A survey," *CoRR*, vol. abs/1904.08405, 2019. [Online]. Available: http://arxiv.org/abs/1904.08405

[13] G. Orchard, A. Jayawant, G. K. Cohen, and N. Thakor, "Converting static image datasets to spiking neuromorphic datasets using saccades," *Frontiers in Neuroscience*, vol. 9, 2015. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fnins.2015.00437

[14] S. Yang, P. Luo, C. C. Loy, and X. Tang, "WIDER FACE: A face detection benchmark," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[15] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[16] V. Jain and E. Learned-Miller, "FDDB: A benchmark for face detection in unconstrained settings," University of Massachusetts, Amherst, Tech. Rep. UM-CS-2010-009, 2010.

[17] T. Serrano-Gotarredona and B. Linares-Barranco, "Poker-DVS and MNIST-DVS. their history, how they were made, and other details," *Frontiers in Neuroscience*, vol. 9, Dec 2015.

[18] E. Mueggler, H. Rebecq, G. Gallego, T. Delbruck, and D. Scaramuzza, "The event-camera dataset and simulator: Event-based data for pose estimation, visual odometry, and SLAM," *The International Journal of Robotics Research*, vol. 36, no. 2, pp. 142–149, 2017.

[19] P. Lichtsteiner, C. Posch, and T. Delbruck, "A 128× 128 120 db 15 μs latency asynchronous temporal contrast vision sensor," *IEEE Journal of Solid-State Circuits*, vol. 43, no. 2, pp. 566–576, 2008.

[20] C. Rizk, J. Lin, S. Kennerly, P. Pouliquen, A. Goldberg, and A. Andreou, "High-performance, event-driven, low-cost, and swap imaging sensor for hostile fire detection, homeland protection, and border security," *Proceedings of SPIE - The International Society for Optical Engineering*, vol. 8359, pp. 18–, 05 2012.

[21] J. H. Lin, P. O. Pouliquen, A. G. Andreou, A. C. Goldberg, and C. G. Rizk, "Flexible readout and integration sensor (FRIS): a bio-inspired, system-on-chip, event-based readout architecture," in *Infrared Technology and Applications XXXVIII*, B. F. Andresen, G. F. Fulop, and P. R. Norton, Eds., vol. 8353, International Society for Optics and Photonics. SPIE, 2012, p. 83531N. [Online]. Available: https://doi.org/10.1117/12.919584

[22] A. Niwa, F. Mochizuki, R. Berner, T. Maruyarma, T. Terano, K. Takamiya, Y. Kimura, K. Mizoguchi, T. Miyazaki, S. Kaizu, H. Takahashi, A. Suzuki, C. Brandli, H. Wakabayashi, and Y. Oike, "A 2.97μm-pitch event-based vision sensor with shared pixel front-end circuitry and low-noise intensity readout mode," in *IEEE International Solid-State Circuits Conference (ISSCC)*, 2023, pp. 4–6.

[23] M. Liu and T. Delbruck, "Adaptive time-slice block-matching optical flow algorithm for dynamic vision sensors." BMVC, 2018.

[24] C. G. Rizk, J. P. Wilson, and P. Pouliquen, "Advanced computational sensors technology: testing and evaluation in visible, SWIR, and LWIR imaging," in *Image Sensing Technologies: Materials, Devices, Systems, and Applications II*, N. K. Dhar and A. K. Dutta, Eds., vol. 9481, International Society for Optics and Photonics. SPIE, 2015, p. 94810E. [Online]. Available: https://doi.org/10.1117/12.2177350

[25] S. Klenk, L. Koestler, D. Scaramuzza, and D. Cremers, "E-NeRF: Neural radiance fields from a moving event camera," *IEEE Robotics and Automation Letters*, vol. 8, no. 3, pp. 1587–1594, 2023.

[26] V. Rudnev, M. Elgharib, C. Theobalt, and V. Golyanik, "EventNeRF: Neural radiance fields from a single colour event camera," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2023, pp. 4992–5002.

[27] M. Liu and T. Delbruck, "Block-matching optical flow for dynamic vision sensors: Algorithm and fpga implementation," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, 2017, pp. 1–4.

[28] H. Rebecq, D. Gehrig, and D. Scaramuzza, "Esim: an open event camera simulator," in *Conference on robot learning*. PMLR, 2018, pp. 969–982.

[29] Y. Hu, S. C. Liu, and T. Delbruck, "v2e: From video frames to realistic DVS events," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. IEEE, 2021. [Online]. Available: http://arxiv.org/abs/2006.07722

[30] J. Sengupta, S. Liu, and A. Andreou, "Retinosim: An event-based data synthesis tool for neuromorphic vision architecture exploration," in *Proceedings of the International Conference on Neuromorphic Systems 2022*, ser. ICONS '22. New York, NY, USA: Association for Computing Machinery, 2022. [Online]. Available: https://doi.org/10.1145/3546790.3546805

[31] S. Venkatachalam, V. S. Vivekanand, and R. Kubendran, "Frame of events: A low-latency resource-efficient approach for stereo depth maps," in *International Conference on Automation, Robotics and Applications (ICARA)*, 2023, pp. 324–328.

[32] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[33] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," *arXiv preprint arXiv:2004.10934*, pp. 1–17, 2020.

[34] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2023, pp. 7464–7475.

[35] C.-Y. Wang, H.-Y. M. Liao, Y.-H. Wu, P.-Y. Chen, J.-W. Hsieh, and I.-H. Yeh, "CSPNet: A new backbone that can enhance learning capability of CNN," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, 2020, pp. 390–391.

[36] M. Tan, R. Pang, and Q. V. Le, "Efficientdet: Scalable and efficient object detection," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 10 781–10 790.

[37] A. G. Howard, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, pp. 1–9, 2017.

[38] D. Kollias and S. Zafeiriou, "Aff-wild2: Extending the aff-wild database for affect recognition," *CoRR*, vol. abs/1811.07770, 2018. [Online]. Available: http://arxiv.org/abs/1811.07770

[39] B. Dwyer, J. Nelson, J. Solawetz, and et al., "Roboflow (version 1.0) [software]," https://roboflow.com, 2022, accessed: November, 2023.