Learning to Exploit Multiple Vision Modalities by Using Grafted Networks

Yuhuang Hu[0000−0002−5543−7619], Tobi Delbruck[0000−0001−5479−1141], and Shih-Chii Liu[0000−0002−7557−045X]

Institute of Neuroinformatics, University of Zürich and ETH Zürich, Switzerland
{yuhuang.hu, tobi, shih}@ini.uzh.ch

Abstract. Novel vision sensors such as thermal, hyperspectral, polarization, and event cameras provide information that is not available from conventional intensity cameras. An obstacle to using these sensors with current powerful deep neural networks is the lack of large labeled training datasets. This paper proposes a Network Grafting Algorithm (NGA), where a new front end network driven by unconventional visual inputs replaces the front end network of a pretrained deep network that processes intensity frames. The self-supervised training uses only synchronously-recorded intensity frames and novel sensor data to maximize feature similarity between the pretrained network and the grafted network. We show that the enhanced grafted network reaches competitive average precision (AP_{50}) scores to the pretrained network on an object detection task using thermal and event camera datasets, with no increase in inference costs. Particularly, the grafted network driven by thermal frames showed a relative improvement of 49.11% over the use of intensity frames. The grafted front end has only 5–8% of the total parameters and can be trained in a few hours on a single GPU equivalent to 5% of the time that would be needed to train the entire object detector from labeled data. NGA allows new vision sensors to capitalize on previously pretrained powerful deep models, saving on training cost and widening a range of applications for novel sensors.

Keywords: Network Grafting Algorithm; Self-supervised Learning; Thermal Camera; Event-based Vision; Object Detection

1 Introduction

Novel vision sensors like thermal, hyperspectral, polarization, and event cameras provide new ways of sensing the visual world and enable new or improved vision system applications. So-called *event cameras*, for example, sense normal visible light, but dramatically sparsify it to pure brightness change events, which provide sub-ms timing and HDR to offer fast vision under challenging illumination conditions [21, 11]. These novel sensors are becoming practical alternatives that complement standard cameras to improve vision systems.

Deep Learning (**DL**) with labeled data has revolutionized vision systems using conventional intensity frame-based cameras. But exploiting DL for vision

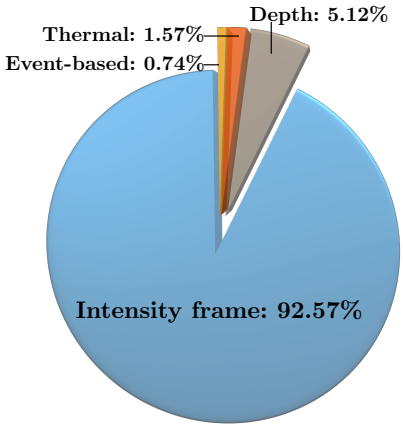

Fig. 1. Types of computer vision datasets. Data from [9].

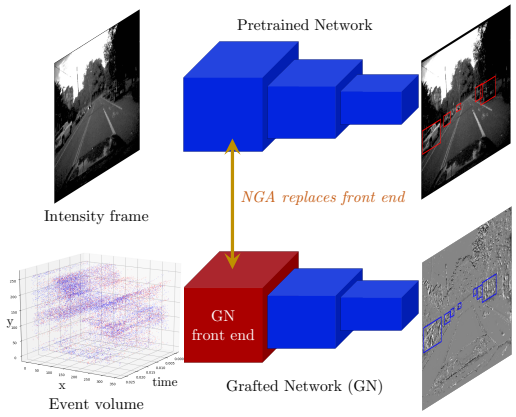

Fig. 2. A network (blue) trained on intensity frames outputs bounding boxes of detected objects. NGA trains a new GN front end (red) using a small unlabeled dataset of recordings from a DAVIS [4] event camera that concurrently outputs intensity frames and asynchronous brightness change events. The grafted network is obtained by *replacing* the original front end with the GN front end, and is used for inference with the novel camera input data.

systems based on novel cameras has been held back by the lack of large labeled datasets for these sensors. Prior work to solve high-level vision problems using inputs other than intensity frames has followed the principles of supervised Deep Neural Network (**DNN**) training algorithms, where the task-specific datasets must be labeled with a tremendous amount of manual effort [24, 2, 3, 31]. Although the community has collected many useful small datasets for novel sensors, the size, variety, and labeling quality of these datasets is far from rivaling intensity frame datasets [26, 15, 2, 10, 18, 3]. As shown in Fig. 1, among 1,212 surveyed computer vision datasets in [9], 93% are intensity frame datasets. Notably, there are only 28 event-based and thermal datasets.

Particularly for event cameras, another line of DL research employs unsupervised methods to train networks that predict pixel-level quantities such as optical flow [41], depth [40]; and that reconstruct intensity frames [28]. The information generated by these networks can be further processed by a downstream DNN trained to solve tasks such as object classification. This information is exceptionally useful in challenging scenarios such as high-speed motion under difficult lighting conditions. The additional latency introduced by running these networks might be undesirable for fast online applications. For instance, the DNNs used

for intensity reconstruction at low QVGA resolution take ∼30 ms on a dedicated GPU [28, 33].

This paper introduces a simple yet effective algorithm called the *Network Grafting Algorithm* (**NGA**) to obtain a *Grafted Network* (**GN**) that addresses both issues: 1. the lack of large labeled datasets for training a DNN from scratch, and 2. additional inference cost and latency that comes from running networks that compute pixel-level quantities. With this algorithm, we train a *GN front end* for processing unconventional visual inputs (red block in Fig. 2) to drive a network originally trained on intensity frames. We demonstrate GNs for thermal and event cameras in this paper.

The NGA training encourages the GN front end to produce features that are similar to the features at several early layers of the pretrained network. Since the algorithm only requires pretrained hidden features as the target, the training is self-supervised, that is, no labels are needed from the novel camera data. The training method is described in Section 3.1. Furthermore, the newly trained GN has a similar inference cost to the pretrained network and does not introduce additional preprocessing latency. Because the training of a GN front end relies on the pretrained network, the NGA has similarities to Knowledge Distillation (KD) [14], Transfer Learning [27], and Domain Adaptation (DA) [12, 35, 37]. In addition, our proposed algorithm utilizes loss terms proposed for super-resolution image reconstruction and image style transfer [16, 13]. Section 2 elaborates on the similarities and differences between NGA and these related domains.

To evaluate NGA, we start with a pretrained object detection network and obtain a GN for a thermal object detection dataset (Section 4.1) to solve the same task. Then, we further demonstrate the training method on car detection using an event camera driving dataset (Section 4.2). We show that the GN achieves similar detection precision compared to the original pretrained network. We also evaluate the accuracy gap between supervised and NGA self-supervised with MNIST for event cameras (Section 4.3). Finally, we do representation analysis and ablation studies in Section 5. Our contributions are as follows:

1. We propose a novel algorithm called NGA that allows the use of networks already trained to solve a high-level vision problem but adapted to work with a new GN front end that processes inputs from thermal/event cameras.
2. The NGA algorithm does not need a labeled thermal/event dataset because the training is self-supervised.
3. The newly trained GN has an inference cost similar to the pretrained network because it directly processes the thermal/event data. Hence, the computation latency brought by *e.g.*, intensity reconstruction from events is eliminated.
4. The algorithm allows the output of these novel cameras to be exploited in situations that are difficult for standard cameras.

2 Related Work

The NGA trains a GN front end such that the hidden features at different layers of the GN are similar to respective pretrained network features on intensity

frames. From this aspect, the NGA is similar to Knowledge Distillation [14, 32, 36] where the knowledge of a teacher network is gradually distilled into a student network (usually smaller than the teacher network) via the soft labels provided by the teacher network. In KD, the teacher and student networks use the same dataset. In contrast, the NGA assumes that the inputs for the pretrained front end and the GN front end come from two *different* modalities that see the same scene concurrently, but this dataset can simply be raw unlabeled recordings. The NGA is also a form of Transfer Learning [27] and Domain Adaptation [12, 35, 37] that study how to fine-tune the knowledge of a pretrained network on a new dataset. *Our method trains a GN front end from scratch since the network has to process the data from a different sensory modality.*

Another interpretation of maximizing hidden feature similarity can be understood from the algorithms used for super-resolution (SR) image reconstruction and image style transfer. SR image reconstruction requires a network that upsamples a low-resolution image into a high-resolution image. The perceptual loss [16, 38] was used to increase the sharpness and maintain the natural image statistics of the reconstruction. Image style transfer networks often aim to transfer an image into a target artistic style where Gram loss [13] is often employed. While these networks learn to match either a high-resolution image ground truth or an artistic style, we train the GN front end to output features that match the hidden features of the pretrained network. For training the front end, we draw inspiration from these studies and propose the use of combinations of training loss metrics including perceptual loss and Gram loss.

3 Methods

We first describe the details of NGA in Section 3.1, then the the event camera and its data representation in Section 3.2. Finally in Section 3.3, we discuss the details of the thermal and event datasets.

3.1 Network Grafting Algorithm

The NGA uses a pretrained network `N` that takes an intensity frame I_t at time t, and produces a grafted network `GN` whose input is a thermal frame or an event volume V_t. I_t and V_t are synchronized during the training. The `GN` should perform with similar accuracy on the same network task, such as object detection. During inference with the thermal or event camera, I_t is not needed. The rest of this section sets up the constructions of `N` and `GN`, then the NGA is described.

Pretrained network setup. The pretrained network `N` consists of three blocks: {$\texttt{N}_\text{f}$ (Front end), $\texttt{N}_\text{mid}$ (Middle net), $\texttt{N}_\text{last}$ (Remaining layers)}. Each block is made up of several layers and the outputs of each of the three blocks are defined as

$$H_t = \texttt{N}_\text{f}(I_t), \qquad R_t = \texttt{N}_\text{mid}(H_t), \qquad Y_t = \texttt{N}_\text{last}(R_t) \tag{1}$$

where H_t is the front end features, R_t is the middle net features, and Y_t is the network prediction. The separation of the network blocks is studied in Section 5.2. The top row in Fig. 3 illustrates the three blocks of the pretrained network.

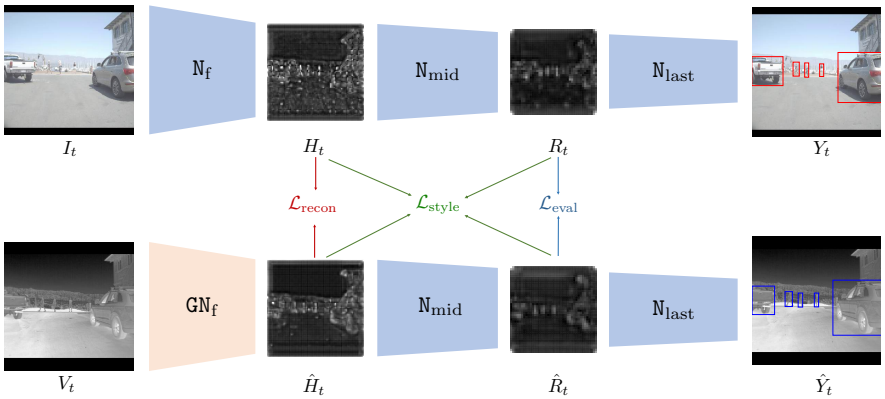

Fig. 3. NGA. (**top**) Pretrained Network. (**bottom**) Grafted Network. Arrows point from variables to relevant loss terms. I_t and V_t here are an intensity frame and a thermal frame, respectively. The intermediate features $\hat{H}_t$, H_t, $\hat{R}_t$, R_t are shown as heat maps averaged across channels. The object bounding boxes predicted by the original and the grafted network are outlined in red and blue correspondingly.

Grafted network setup. We define a *GN front end* $\mathtt{GN}_\mathrm{f}$ that takes V_t as the input and outputs grafted front end features, $\hat{H}_t$, of the same dimension as H_t. $\mathtt{GN}_\mathrm{f}$ combined with $\mathtt{N}_\mathrm{mid}$ and $\mathtt{N}_\mathrm{last}$ produces the predictions $\hat{Y}$:

$$\hat{H}_t = \mathtt{GN}_\mathrm{f}(V_t), \qquad \hat{Y}_t = \mathtt{N}_\mathrm{last}(\mathtt{N}_\mathrm{mid}(\hat{H}_t)) \tag{2}$$

We define $\mathtt{GN} = \{\mathtt{GN}_\mathrm{f}, \mathtt{N}_\mathrm{mid}, \mathtt{N}_\mathrm{last}\}$ as the *Grafted Network* (bottom row of Fig. 3).
Network Grafting Algorithm. The NGA trains the grafted network $\mathtt{GN}$ to reach a similar performance to that of the pretrained network $\mathtt{N}$ by increasing the representation similarity between features $H = \{H_t|\forall t\}$ and $\hat{H} = \{\hat{H}_t|\forall t\}$.

The loss function for the training of the $\mathtt{GN}_\mathrm{f}$ consists of a combination of three losses. The first loss is the Mean-Squared-Error (MSE) between H and $\hat{H}$:

$$\mathcal{L}_\mathrm{recon} = \mathrm{MSE}(H, \hat{H}) \tag{3}$$

Because this loss term captures the amount of representation similarity between the two different front ends, we call $\mathcal{L}_\mathrm{recon}$ a *Feature Reconstruction Loss* (FRL).

The second loss takes into account the output of the middle net layers in the network and draws inspiration from the Perception Loss [16]. This loss is set by the MSE between the middle net frame features $R = \{R_t|\forall t\}$ and the grafted middle net features $\hat{R} = \{\mathtt{N}_\mathrm{mid}(\hat{H}_t)|\forall t\}$:

$$\mathcal{L}_\mathrm{eval} = \mathrm{MSE}(R, \hat{R}) \tag{4}$$

Since this loss term additionally evaluates the feature similarities between front end features $\{H, \hat{H}\}$, we refer to $\mathcal{L}_\mathrm{eval}$ as the *Feature Evaluation Loss* (FEL).

Both FRL and FEL terms minimize the magnitude differences between hidden features. To further encourage the GN front end to generate intensity frame-like textures, we introduce the *Feature Style Loss* (FSL) based on the mean-subtracted Gram loss [13] that computes a Gram matrix using feature columns across channels (indexed using i, j). The Gram matrix represents image texture rather than spatial structure. This loss is defined as:

$$\text{Gram}(F)^{(i,j)} = \sum_{\forall t} \tilde{F}_t^{(i)\top} \tilde{F}_t^{(j)}, \quad \text{where } \tilde{F}_t = F_t - \text{mean}(F_t) \tag{5}$$

$$\mathcal{L}_{\text{style}} = \gamma_h \text{MSE}(\text{Gram}(H), \text{Gram}(\hat{H})) + \gamma_r \text{MSE}(\text{Gram}(R), \text{Gram}(\hat{R})) \tag{6}$$

The final loss function is a weighted sum of the three loss terms:

$$\mathcal{L}_{\text{tot}} = \alpha \mathcal{L}_{\text{recon}} + \beta \mathcal{L}_{\text{eval}} + \mathcal{L}_{\text{style}} \tag{7}$$

For all experiments in the paper, we set $\alpha = \beta = 1$, $\gamma_h \in \{10^5, 10^6, 10^7\}$, $\gamma_r = 10^7$. The loss terms and their associated variables are shown in Fig. 3. The importance of each loss term is studied in Section 5.3.

3.2 Event Camera and Feature Volume Representation

Event cameras such as the DAVIS camera [21, 4] produce a stream of asynchronous "events" triggered by local brightness (log intensity) changes at individual pixels. Each output event of the event camera is a four-element tuple $\{t, x, y, p\}$ where t is the timestamp, (x, y) is the location of the event, and p is the event polarity. The polarity is either positive (brightness increasing) or negative (brightness decreasing). To preserve both spatial and temporal information captured by the polarity events, we use the event voxel grid [41, 28]. Assuming a volume of N events $\{(t_i, x_i, y_i, p_i)\}_{i=1}^{N}$ where i is the event index, we divide this volume into D event slices of equal temporal intervals such that the d-th slice S_d is defined as follows:

$$\forall x, y; \quad S_d(x, y) = \sum_{x_i = x, y_i = y} p_i \max(0, 1 - |d - \tilde{t}_i|) \tag{8}$$

and $\tilde{t}_i = (D-1)\frac{t_i - t_1}{t_N - t_1}$ is the normalized event timestamp. The event volume is then defined as $V_t = \{S_d\}_{d=0}^{D-1}$. In Section 4, $D = 3, 10$ and $N = 25,000$. Prior work has shown that this spatio-temporal view of the input scene activity covering a constant number of brightness change events is simple but effective for optical flow computation [41] and video reconstruction [28].

3.3 Datasets

Two different vision datasets were used in the experiments in this paper and are presented in the subsections.

Thermal dataset for object detection. The FLIR Thermal Dataset [10] includes labeled recordings from a thermal camera for driving on Santa Barbara,

CA area streets and highways for both day and night. The thermal frames were captured using a FLIR IR Tau2 thermal camera with an image resolution of 640×512. The dataset has parallel RGB intensity frames and thermal frames in an 8-bit JPEG format with AGC. Since the standard camera is placed alongside the thermal camera, a constant spatial displacement is expected, and this shift is corrected for the training samples. The dataset has 4,855 training intensity-thermal pairs, and 1,256 testing pairs, of which 60% are daytime and 40% are nighttime driving samples. We excluded samples where the intensity frames are corrupted. The annotated object classes are `car`, `person`, and `bicycle`.

Event camera dataset. The Multi Vehicle Stereo Event Camera Dataset (MVSEC) [39] is a collection of event camera recordings for studying 3D perception and optical flow estimation. The `outdoor_day2` recording is carried out in an urban area of West Philadelphia. This recording was selected for the car detection experiment because of its better quality compared to other recordings, and it has a large number of cars in the scenes distributed throughout the entire recording. We generated in total 7,000 intensity frames and event volume pairs from this recording. Each event volume contains $N = 25,000$ events. The first 5,000 pairs are used as the training dataset, and the last 2,000 pairs are used as the testing dataset. There are no temporally overlapping pairs between the training and testing datasets.

Because MVSEC does not provide ground truth bounding boxes for cars, we pseudo-labeled data pairs of the testing dataset for intensity frames that contain at least one car detected by the Hybrid Task Cascade (HTC) Network [6], which provides state-of-the-art results in object detection. We only use the bounding boxes with 80% or higher confidence to obtain high-quality bounding boxes. To compare the effect of using different numbers of event slices D in an event volume on the detection results, we additionally created two versions of this dataset: DVS-3 where $D = 3$ and DVS-10 where $D = 10$.

4 Experiments

We use the NGA to train a GN front end for a pretrained object detection network. In this case, we use the YOLOv3 network [29] that was trained using the COCO dataset [22] with 80 objects. This network was chosen because it still provides good detection accuracy and could be deployed on a low-cost embedded real-time platform. The pretrained network is referred to as YOLOv3-`N` and the grafted thermal/event-driven networks as YOLOv3-`GN` in the rest of the paper. The training inputs consist of 224×224 image patches randomly cropped from the training pairs. No other data augmentation is performed. All networks are trained for 100 epochs with the Adam optimizer [17], a learning rate of 10^{-4}, and a mini-batch size of 8. Each model training takes ∼1.5 hours using an NVIDIA RTX 2080 Ti, which is only 5% of the 2 days it typically requires to train one of the object detectors used in this paper on standard labeled datasets. More results from the experiments on the different vision datasets are presented in the supplementary material.

4.1 Object Detection on Thermal Driving Dataset

This section presents the experimental results of using the NGA to train an object detector for the thermal driving dataset.

Fig. 4. Examples of six testing pairs from the thermal driving dataset. The red boxes are objects detected by the original intensity-driven YOLOv3 network and the blue boxes show the objects detected by the thermal-driven network. The magenta box shows cars detected by the thermal-driven GN that are missed by the intensity-driven network when the intensity frame is underexposed. Best viewed in color.

Fig. 4 shows six examples of object detection results from the original intensity-driven YOLOv3 network and the thermal-driven network. These examples show that when the intensity frame is well-exposed, the prediction difference between YOLOv3-`N` and YOLOv3-`GN` appears to be small. However, when the intensity frame is either underexposed or noisy, the thermal-driven network detects many more objects than the pretrained network. For instance, in the magenta box of Fig. 4, most cars are not detected by the intensity-driven network but they are detected by the thermal-driven network.

The detection precision (AP_{50}) results over the entire test set (Table 1) show that the accuracy of our pretrained YOLOv3-`N` on the intensity frames (30.36) is worse than on thermal frames (39.92) because 40% of the intensity night frames look noisy and are underexposed. The YOLOv3-`GN` thermal-driven network achieved the highest AP_{50} detection precision (45.27) among all our YOLOv3 variants while requiring training of only 5.17% (3.2M) parameters with NGA. A baseline Faster R-CNN which was trained on the same labeled thermal dataset [10] achieved a higher precision of 53.97. However, it required training of 47M parameters which is 15X more than the YOLOv3-`GN`. Overall, the results show that the self-supervised GN front end significantly improves the accuracy results of the original network on the thermal dataset.

For comparison with other object detectors, we also use the `mmdetection` framework [7] to process the intensity frames using pretrained SSD [23], Faster R-CNN [30] and Cascade R-CNN [5] detectors. All have worse AP_{50} scores than any of the YOLOv3 networks, so YOLOv3 was a good choice for evaluating the effectiveness of NGA.

Table 1. Object detection AP_{50} scores on the FLIR driving dataset. The training of YOLOv3-GN repeats five times.

Network	**Modality**	AP_{50}	# Trained Params
This work			
YOLOv3-GN	Thermal	**45.27±1.14**	**3.2M**
YOLOv3-N	Intensity	30.36	62M
YOLOv3-N	Thermal	39.92	62M
SSD	Intensity	8.00	36M
Faster R-CNN	Intensity	23.82	42M
Cascade R-CNN	Intensity	27.10	127M
Baseline supervised thermal object detector			
Faster R-CNN [8]	Thermal	53.97	47M

4.2 Car Detection on Event Camera Driving Dataset

To study if the NGA is also effective for exploiting another visual sensor, *e.g.*, an event camera, we evaluated car detection results using the pretrained network YOLOv3-N and a grafted network YOLOv3-GN using the MVSEC dataset.

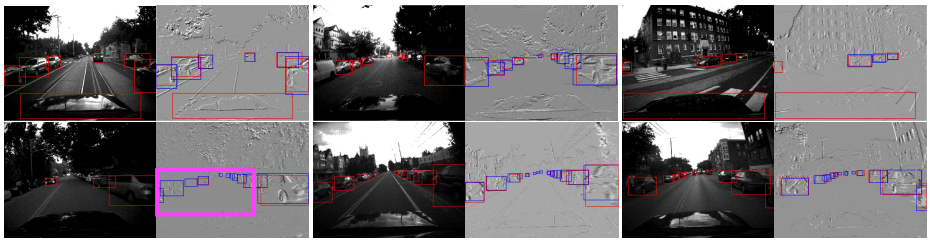

Fig. 5. Examples of testing pairs from the MVSEC dataset. The event volume is visualized after averaging across slices. The predicted bounding boxes (in red) from the intensity-driven network can be compared with the predicted bounding boxes (in blue) from the event-driven network. The magenta box shows cars detected by the event-driven network that are missed by the intensity-driven network. Best viewed in color.

The event camera operates over a larger dynamic range of lighting than an intensity frame camera and therefore will detect moving objects even in poorly lighted scenes. From the six different data pairs in the MVSEC testing dataset (Fig. 5), we see that the event-driven YOLOv3-GN network detects most of the cars found in the intensity frames and additional cars not detected in the intensity frame (see the magenta box in the figure). These examples help illustrate how event cameras and the event-driven network can complement the pretrained network in challenging situations.

Table 2 compares the accuracy of the intensity and event camera detection networks on the testing set. As might be expected for these well-exposed and sharp daytime intensity frames, the YOLOv3-N produces the highest average

precision (AP). Surprisingly, the YOLOv3-GN with DVS-3 input achieves close to the same accuracy, although it was never explicitly trained to detect objects on this type of data. We also tested if the pretrained network would perform poorly on the DVS-3 event dataset. The AP_{50} is almost 0 (not reported in the table) and confirms that the intensity-driven front end fails at processing the event volume and that using a GN front end is essential for acceptable accuracy.

We also compare the performances of the event-driven networks that receive as input, the two datasets with different numbers of event slices for the event volume, *i.e.*, DVS-3, and DVS-10. The network trained on DVS-10 shows a better score of $AP_{50} = 70.35$, which is only 3.18 lower than the original YOLOv3 network accuracy. Table 2 also shows the effect on accuracy when varying the number of training samples. Even when trained using only 40% of training data (2k samples), the YOLOv3-GN still shows strong detection precision at 66.75. But when the NGA has access to only 10% of the data (500 samples) during training, the detection performance drops by 22.47% compared to the best event-driven network. Although the NGA requires far less data compared to standard supervised training, training with only a few hundreds of samples remains challenging and could benefit from data augmentation to improve performance.

Table 2. AP_{50} scores for car detection on the MVSEC driving dataset (five runs).

Network	**Modality**	AP_{50}	# Trained Params
YOLOv3-N	Intensity	73.53	62M
YOLOv3-GN	DVS-3	70.14±0.36	3.2M
YOLOv3-GN	DVS-10	70.35±0.51	3.2M
YOLOv3-GN	DVS-10 (40% samples)	66.75±0.30	3.2M
YOLOv3-GN	DVS-10 (10% samples)	47.88±1.86	3.2M
Combined	Intensity+DVS-10	**75.45**	N/A
SSD	Intensity	36.17	36M
Faster R-CNN	Intensity	71.89	42M
Cascade R-CNN	Intensity	85.16	127M

To study the benefit of using the event camera brightness change events to complement its intensity frame output, we combined the detection results from both the pretrained network and event-driven network (Row **Combined** in Table 2). After removing duplicated bounding boxes through non-maximum suppression, the AP_{50} score of the combined prediction is higher by 1.92 than the prediction of the pretrained network using intensity frames.

Reference AP_{50} scores from three additional intensity frame detectors implemented using the `mmdetection` toolbox are also reported in the table for comparison.

4.3 Comparing NGA and Standard Supervised Learning

Intuitively, a network trained in a supervised manner should perform better than a network trained through self-supervision. To study this, we evaluate the accuracy gap between classification networks trained with supervised learning, and the NGA using event recordings of the MNIST handwritten digit recognition dataset, also called N-MNIST dataset [26]. Each event volume is prepared by setting $D = 3$. The training uses the Adam optimizer, a learning rate of 10^{-3} and a batch size of 256.

First, we train the LeNet-`N` network with the standard LeNet-5 architecture [20] using the intensity samples in the MNIST dataset. Next, we train LeNet-`GN` with the NGA by using parallel MNIST and N-MNIST sample pairs. We also train an event-driven LeNet-`supervised` network from scratch on N-MNIST using standard supervised learning with the labeled digits. The results in Table 3 show that the accuracy of the LeNet-`GN` network is only 0.36% lower than that of the event-driven LeNet-`supervised` network even with the training of a front end which has only 8% of the total network parameters, and without the availability of labeled training data. The LeNet-`GN` also performed better or on par with other models that have been tested on the N-MNIST dataset [19, 25, 34].

Table 3. Classification results on MNIST and N-MNIST datasets.

Network	**Dataset**	Error Rate (%)	# Trained Params
LeNet-`N`	MNIST	0.92	64k
LeNet-`GN`	N-MNIST	1.47±0.05	5k
LeNet-`supervised`	N-MNIST	1.11±0.06	64k
HFirst [25]	N-MNIST	28.80	-
HOTS [19]	N-MNIST	19.20	-
HATS [34]	N-MNIST	0.90	-

5 Network Analysis

To understand the representational power of the GN features, Section 5.1 presents a qualitative study that shows how the grafted front end features represent useful visual input under difficult lighting conditions. To design an effective GN, it is important to select what parts of the network to graft. Sections 5.2 and 5.3 describe studies on the network variants and the importance of the loss terms.

5.1 Decoding Grafted Front End Features

Previous experiments show that the grafted front end features provide useful information for the GN in the object detection tasks. In this section, we provide

qualitative evidence that the grafted features often faithfully represent the input scene. Specifically, we decode the grafted features by optimizing a decoded intensity frame I_t^{d} that produces features through the intensity-driven network best matching the grafted features $\hat{H}_t$, by minimizing:

$$\arg\min_{I_t^{\mathrm{d}}} \mathrm{MSE}(\mathtt{N}_{\mathrm{f}}(I_t^{\mathrm{d}}), \hat{H}_t) + 5 \times \mathrm{TV}(I_t^{\mathrm{d}}) \tag{9}$$

where $\mathrm{TV}(\cdot)$ is a total variation regularizer for encouraging spatial smoothness [1]. The decoded intensity frame I_t^{d} is initialized randomly and has the same spatial dimension as the intensity frame, then the pixel values of I_t^{d} are optimized for 1k iterations using an Adam optimizer with learning rate of 10^{-2}.

Figure 6 shows four examples from the thermal dataset and the event dataset. Under extreme lighting conditions, the intensity frames are often under/over-exposed while the decoded intensity frames show that the thermal/event front end features can represent the same scene better (see the labeled regions).

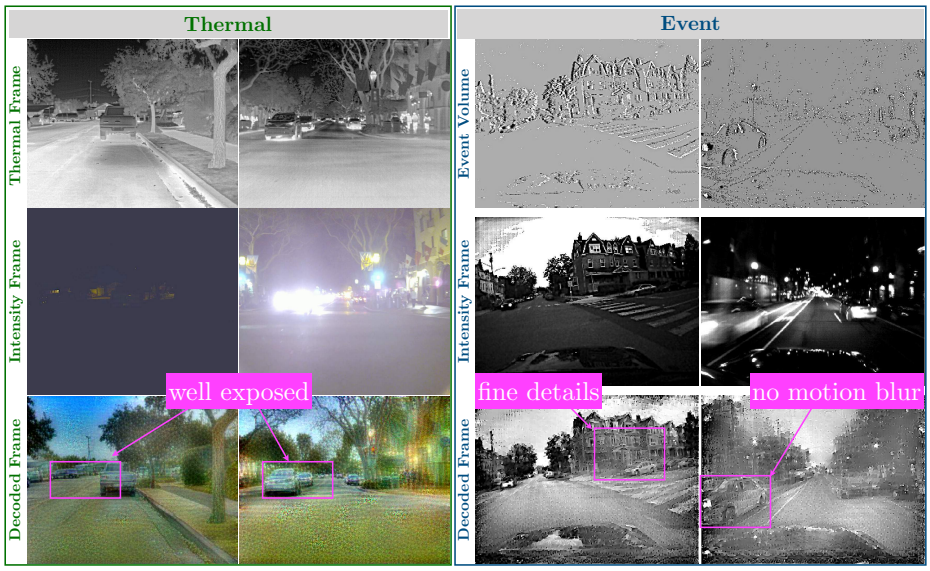

Fig. 6. Decoded frames of image pairs taken from both the thermal and event datasets. Each column represents an example image from either the thermal dataset (the leftmost two columns) or the event dataset (the rightmost two columns). The top panel of each column shows either the thermal frame or the event volume. The middle panel shows the raw intensity frames. The bottom panel shows the decoded intensity frames (see main text). Labeled regions in the decoded frames show details that are not visible in the four original intensity frames. The figure is best viewed in color.

5.2 Design of Grafted Network

The backbone network of YOLOv3 is called Darknet-53, and consists of five residual blocks (Fig. 7). Selecting the correct set of residual blocks used for the NGA front end is important. Six combinations of the front end and middle net by using different numbers of residual blocks: {S1, S4}, {S1, S5}, {S2, S4}, {S2, S5}, {S3, S4} and {S3, S5} are tested. S1, S2, S3 indicate front end variants with different number of residual blocks that uses 0.06% (40k), 0.45% (279k), and 5.17% (3.2M) of total parameters (62M) respectively. The number of blocks for S4 and S5 vary depending on the chosen variant. Figure 8 shows the AP_{50} scores for different combinations of front end and middle net variants. The best separation of the network blocks is {S3, S4}. In the YOLOv3 network, the detection results improve sharply when the front end includes more layers. On the other hand, the difference in AP_{50} between using S4 or S5 for the middle net is not significant. These results suggest that using a deeper front end is better than a shallow front end, especially when training resources are not a constraint.

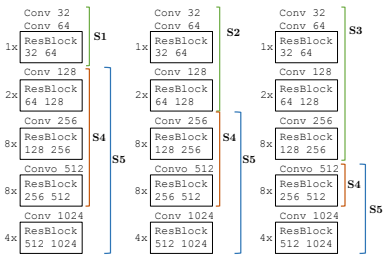

Fig. 7. YOLOv3 backbone: Darknet-53 [29]. The front end variants are S1, S2 and S3. The middle net variants are S4 and S5. `Conv` represents a convolution layer, `ResBlock` represents a residual block.

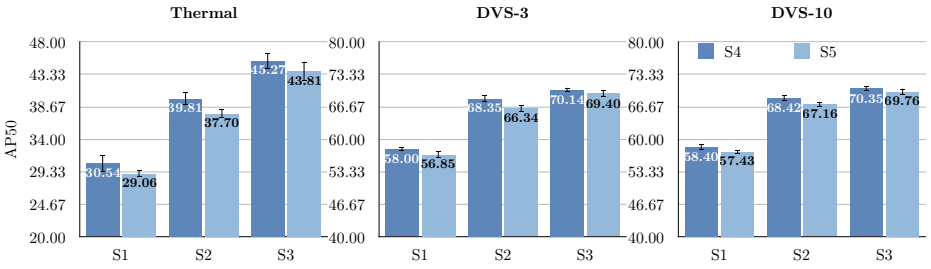

Fig. 8. Results of different front end and middle net variants in Fig. 7 for both thermal and event datasets in AP_{50}. Experiments for each variant are repeated five times.

5.3 Ablation Study on Loss Terms

The NGA training includes three loss terms: FRL, FEL, and FSL. We studied the importance of these loss terms by performing an ablation study using both the thermal dataset and the event dataset. These experiments are done on the network configuration {S3, S4} that gave the best accuracy (see Fig. 8). The

detection precision scores are shown in Fig. 9 for different loss configurations. The FRL and the FEL are the most critical loss terms, while the role of the FSL is less significant. The effectiveness of different loss combinations seems task-dependent and sometimes fluctuates, *e.g.*, FRL+FEL for thermal and FEL+FSL for DVS-10. The trend lines indicate that using a combination of loss terms is most likely to produce better detection scores.

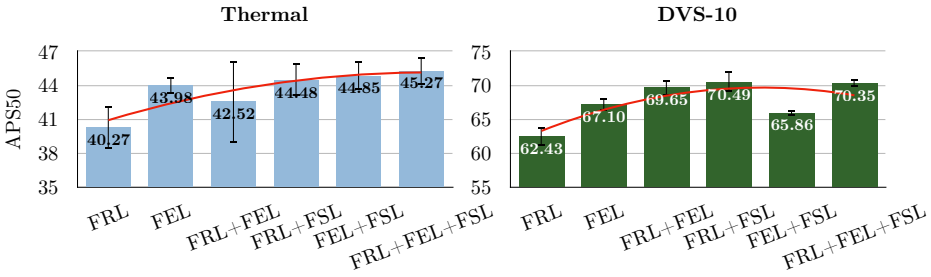

Fig. 9. GN performance (AP_{50}) trained with different loss configurations. Results are from five repeats of each loss configuration.

6 Conclusion

This paper proposes the Network Grafting Algorithm (NGA) that replaces the front end of a network that is pretrained on a large labelled dataset so that the new grafted network (GN) also works well with a different sensor modality. Training the GN front end for a different modality, in this case, a thermal camera or an event camera, requires only a reasonably small unlabeled dataset ($\sim$5k samples) that has spatio-temporally synchronized data from both modalities. By comparison, the COCO dataset on which many object detection networks are trained has 330k images. Ordinarily, training a network with a new sensor type and limited labeled data requires a lot of careful data augmentation. NGA avoids this by exploiting the new sensor data even if unlabeled because the pretrained network already has informative features.

The NGA was applied on an object detection network that was pretrained on a big image dataset. The NGA training was conducted using the FLIR thermal dataset [10] and the MVSEC driving dataset [39]. After training, the GN reached a similar or higher average precision (AP_{50}) score compared to the precision achieved by the original network. Furthermore, the inference cost of the GN is similar to that of the pretrained network, which eliminates the latency cost for computing low-level quantities, particularly for event cameras. This newly proposed NGA widens the use of these unconventional cameras to a broader range of computer vision applications.

Acknowledgements. This work was funded by the Swiss National Competence Center in Robotics (NCCR Robotics).

References

1. Aly, H.A., Dubois, E.: Image up-sampling using total-variation regularization with a new observation model. IEEE Transactions on Image Processing **14**(10), 1647–1659 (Oct 2005)
2. Anumula, J., Neil, D., Delbruck, T., Liu, S.C.: Feature representations for neuromorphic audio spike streams. Frontiers in Neuroscience **12**, 23 (2018). https://doi.org/10.3389/fnins.2018.00023
3. Bahnsen, C.H., Moeslund, T.B.: Rain removal in traffic surveillance: Does it matter? IEEE Transactions on Intelligent Transportation Systems pp. 1–18 (2018)
4. Brandli, C., Berner, R., Yang, M., Liu, S.C., Delbruck, T.: A 240×180 130 dB 3 μs latency global shutter spatiotemporal vision sensor. IEEE Journal of Solid-State Circuits **49**(10), 2333–2341 (2014)
5. Cai, Z., Vasconcelos, N.: Cascade R-CNN: Delving into high quality object detection. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2018)
6. Chen, K., Pang, J., Wang, J., Xiong, Y., Li, X., Sun, S., Feng, W., Liu, Z., Shi, J., Ouyang, W., Loy, C.C., Lin, D.: Hybrid task cascade for instance segmentation. In: 2019 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2019)
7. Chen, K., Wang, J., Pang, J., Cao, Y., Xiong, Y., Li, X., Sun, S., Feng, W., Liu, Z., Xu, J., Zhang, Z., Cheng, D., Zhu, C., Cheng, T., Zhao, Q., Li, B., Lu, X., Zhu, R., Wu, Y., Dai, J., Wang, J., Shi, J., Ouyang, W., Loy, C.C., Lin, D.: MMDetection: Open MMLab Detection Toolbox and Benchmark. CoRR **abs/1906.07155** (2019)
8. Devaguptapu, C., Akolekar, N., M Sharma, M., N Balasubramanian, V.: Borrow from anywhere: Pseudo multi-modal object detection in thermal imagery. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops (June 2019)
9. Fisher, R.: CVonline: Image Databases (2020), http://homepages.inf.ed.ac.uk/rbf/CVonline/Imagedbase.htm
10. FLIR: Free FLIR thermal dataset for algorithm training (2018), https://www.flir.com/oem/adas/adas-dataset-form/
11. Gallego, G., Delbrück, T., Orchard, G., Bartolozzi, C., Taba, B., Censi, A., Leutenegger, S., Davison, A.J., Conradt, J., Daniilidis, K., Scaramuzza, D.: Event-based vision: A survey. CoRR **abs/1904.08405** (2019)
12. Ganin, Y., Lempitsky, V.: Unsupervised domain adaptation by backpropagation. In: Bach, F., Blei, D. (eds.) Proceedings of the 32nd International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 37, pp. 1180–1189. PMLR, Lille, France (07–09 Jul 2015)
13. Gatys, L.A., Ecker, A.S., Bethge, M.: Image style transfer using convolutional neural networks. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2414–2423 (June 2016)
14. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. In: NIPS Deep Learning and Representation Learning Workshop (2015)
15. Hu, Y., Liu, H., Pfeiffer, M., Delbruck, T.: DVS benchmark datasets for object tracking, action recognition, and object recognition. Frontiers in Neuroscience **10**, 405 (2016)
16. Johnson, J., Alahi, A., Fei-Fei, L.: Perceptual losses for real-time style transfer and super-resolution. In: 2016 European Conference on Computer Vision (2016)

17. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: Proceedings of the 3rd International Conference on Learning Representations (ICLR) (2014)
18. Krišto, M., Ivašić-Kos, M.: Thermal imaging dataset for person detection. In: 2019 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO). pp. 1126–1131 (May 2019)
19. Lagorce, X., Orchard, G., Galluppi, F., Shi, B.E., Benosman, R.B.: Hots: A hierarchy of event-based time-surfaces for pattern recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence **39**(7), 1346–1359 (July 2017)
20. Lecun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE **86**(11), 2278–2324 (Nov 1998)
21. Lichtsteiner, P., Posch, C., Delbruck, T.: A 128x128 120 dB 15 μs latency asynchronous temporal contrast vision sensor. IEEE Journal of Solid-State Circuits **43**(2), 566–576 (Feb 2008)
22. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft COCO: Common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) Computer Vision – ECCV 2014. pp. 740–755. Springer International Publishing, Cham (2014)
23. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: Ssd: Single shot multibox detector. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) Computer Vision – ECCV 2016. pp. 21–37. Springer International Publishing, Cham (2016)
24. Moeys, D.P., Corradi, F., Kerr, E., Vance, P., Das, G., Neil, D., Kerr, D., Delbrück, T.: Steering a predator robot using a mixed frame/event-driven convolutional neural network. In: 2016 Second International Conference on Event-based Control, Communication, and Signal Processing (EBCCSP). pp. 1–8 (June 2016)
25. Orchard, G., Meyer, C., Etienne-Cummings, R., Posch, C., Thakor, N., Benosman, R.: Hfirst: A temporal approach to object recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence **37**(10), 2028–2040 (Oct 2015)
26. Orchard, G., Jayawant, A., Cohen, G.K., Thakor, N.: Converting static image datasets to spiking neuromorphic datasets using saccades. Frontiers in Neuroscience **9**, 437 (2015)
27. Pan, S.J., Yang, Q.: A survey on transfer learning. IEEE Trans. on Knowl. and Data Eng. **22**(10), 1345–1359 (Oct 2010)
28. Rebecq, H., Ranftl, R., Koltun, V., Scaramuzza, D.: Events-To-Video: Bringing modern computer vision to event cameras. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2019)
29. Redmon, J., Farhadi, A.: YOLOv3: An incremental improvement. arXiv (2018)
30. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: Towards real-time object detection with region proposal networks. In: Cortes, C., Lawrence, N.D., Lee, D.D., Sugiyama, M., Garnett, R. (eds.) Advances in Neural Information Processing Systems 28, pp. 91–99. Curran Associates, Inc. (2015)
31. Rodin, C.D., de Lima, L.N., de Alcantara Andrade, F.A., Haddad, D.B., Johansen, T.A., Storvold, R.: Object classification in thermal images using convolutional neural networks for search and rescue missions with unmanned aerial systems. In: 2018 International Joint Conference on Neural Networks (IJCNN). pp. 1–8 (July 2018)
32. Romero, A., Ballas, N., Kahou, S.E., Chassang, A., Gatta, C., Bengio, Y.: FitNets: hints for thin deep nets. In: International Conference on Laerning Representations (ICLR) 2015 (2015)

33. Scheerlinck, C., Rebecq, H., Gehrig, D., Barnes, N., Mahony, R.E., Scaramuzza, D.: Fast image reconstruction with an event camera. In: 2020 IEEE Winter Conference on Applications of Computer Vision (WACV). pp. 156–163 (2020)
34. Sironi, A., Brambilla, M., Bourdis, N., Lagorce, X., Benosman, R.: Hats: Histograms of averaged time surfaces for robust event-based object classification. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2018)
35. Sun, Y., Tzeng, E., Darrell, T., Efros, A.A.: Unsupervised domain adaptation through self-supervision (2019)
36. Yim, J., Joo, D., Bae, J., Kim, J.: A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (July 2017)
37. You, K., Long, M., Cao, Z., Wang, J., Jordan, M.I.: Universal domain adaptation. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2019)
38. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2018)
39. Zhu, A.Z., Thakur, D., Özaslan, T., Pfrommer, B., Kumar, V., Daniilidis, K.: The multivehicle stereo event camera dataset: An event camera dataset for 3d perception. IEEE Robotics and Automation Letters **3**(3), 2032–2039 (July 2018)
40. Zhu, A.Z., Yuan, L., Chaney, K., Daniilidis, K.: Unsupervised event-based learning of optical flow, depth, and egomotion. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2019)
41. Zhu, A.Z., Yuan, L., Chaney, K., Daniilidis, K.: Unsupervised event-based optical flow using motion compensation. In: Leal-Taixé, L., Roth, S. (eds.) Computer Vision – ECCV 2018 Workshops. pp. 711–714. Springer International Publishing, Cham (2019)