REAL-TIME OBJECT DETECTION AND LOCALIZATION IN COMPRESSIVE SENSED VIDEO

Yeshwanth Ravi Theja Bethi, Sathyaprakash Narayanan, Venkat Rangan$^+$*, Anirban Chakraborty, Chetan Singh Thakur*

Indian Institute of Science, Bangalore, India; tinyVision.ai, San Diego, USA$^+$

ABSTRACT

Typically a 1-2MP CCTV camera generates around 7-12GB of data per day. Frame-by-frame processing of such an enormous amount of data requires hefty computational resources. In recent years, compressive sensing approaches have shown impressive compression results by reducing the sampling bandwidth. Different sampling mechanisms were developed to incorporate compressive sensing in image and video acquisition. Though all-CMOS [1, 2] sensor cameras that perform compressive sensing can help save a lot of bandwidth on sampling and minimize the memory required to store videos, the traditional signal processing, and deep learning models can realize operations only on the reconstructed data. To realize the original uncompressed domain, most reconstruction techniques are computationally expensive and time-consuming. To bridge this gap, we propose a novel task of detection and localization of objects directly on the compressed frames. Thereby mitigating the need to reconstruct the frames and reducing the search rate up to 20× (compression rate). We achieved an accuracy of 46.27% mAP with the proposed model on a GeForce GTX 1080 Ti. We were also able to show real-time inference on an NVIDIA TX2 embedded board with 45.11% mAP, thereby achieving the best balance between the accuracy, inference time, and memory constraints.

***Index Terms*—** Object Detection and Localization, Deep learning, Compressive Sensing

1. INTRODUCTION

In signal processing, compressive sensing(CS) [3, 4] is a powerful sensing paradigm to sample sparse signals with much fewer samples than demanded by the Shannon-Nyquist sampling theorem. The Nyquist theorem mandates that the number of data samples be at least as high as the dimensionality of the sampled signal. Inherent redundancy present in the real signals like images and videos allows significant compression of data. Compressive sensing exploits this inherent redundancy and enables the sampling to happen at sub-Nyquist rates. As generic video acquisition systems face a trade-off between spatial and temporal resolution due to communication bandwidth, systems that can give higher spatial resolution without compromising temporal resolution are in high demand. This makes compressive sensing extremely useful for capturing images and videos in systems that cannot afford high data bandwidth. The number of samples needed for the same video duration is much lower than that of generic imaging systems. Pixel-wise coding [3] is one among many ways of compressive sensing. To reconstruct the original frames from a CS frame, one needs to learn an over-complete dictionary or use existing dictionaries such as a 3D DCT dictionary to represent the videos in a sparse domain [5]. A patch is taken from the CS frame and the corresponding video patch in the sparse domain of the dictionaries is estimated using algorithms like Orthogonal Matching Pursuit (OMP). Typically a 100 FPS video can be reconstructed from a 5 FPS video using pixel-wise coded exposure. To overcome this issue, we propose a new approach by eliminating the need to reconstruct the original frames for object detection and localization by performing it directly on the compressed frame. For the first time in literature, we introduce the novel task of detection and localization directly on the compressed videos. Towards this, we first explore the traditional off-the-shelf detectors (for uncompressed videos) and aim to adapt them to the compressed domain. We further propose a novel object detector framework for compressed data that achieves the best balance between accuracy, runtime, and storage. To summarize the contributions: (i) We develop a compression framework that acts as a CS camera simulator. This framework can compress any video from a frame-based camera to a compressed frame (ii) We also explore the traditional off-the-shelf object detectors (for uncompressed videos) and aim to adapt them to the compressed domain. We further propose a novel object detection framework for the compressed data. (iii) To further corroborate the utility of our proposed model from a deployability perspective, we show that the model is capable of computing for real-time inference on an embedded board.

2. RELATED WORK

Advanced Video Coding (AVC): [6, 7, 8, 9] have shown object detection using AVC compressed videos, here the compression uses spatial compression/resolution reduction

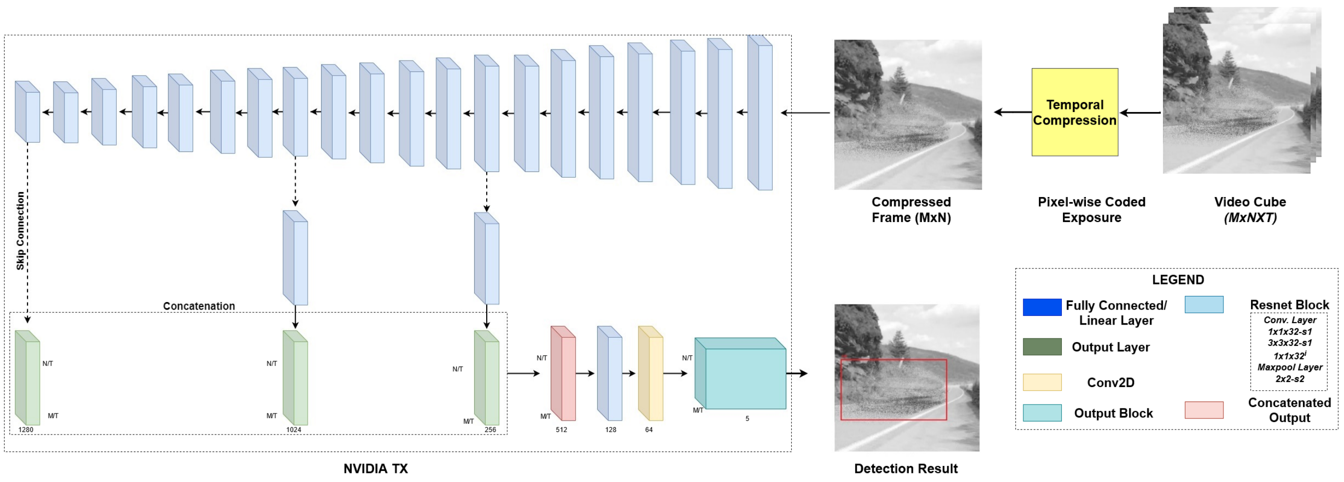

Fig. 1. Pipeline for Real-Time Object detection in CS frames

for each frame in a clip. However, we perform compression along the temporal dimension, thus reducing a manifold amount of data. Our model can perform object detection directly on this tremendously reduced dimension. **Reconstruction of CS frames:** Iliadis [10] and Xu [11] have even succeeded in reconstructing original frames from CS frames using deep learning. Though this line of work is still in progress, reconstruction of the CS frames takes a considerable amount of time, making it unsuitable for real-time processing.

3. PROPOSED METHOD

3.1. Compression Framework

3.1.1. Pixel-wise coded exposure

In conventional cameras, there is a global exposure time T_e (shutter speed) for which all the pixels in the sensor are exposed, and an image is read out from the sensor. We get $1/T_e$ frame rate from such sensors. In contrast to this, in pixel-wise coded exposure (PCE) cameras, each pixel is exposed at a random time for a 'single-on' fixed duration $T_b(> T_e)$ within the time T_v. But, only one image is read out at the end of T_v. If T_v is set to $C \times T_e$, we get a compression of C times and a frame rate of $1/T_v$. For a clip $V(M \times N \times T)$, a sensing matrix $S(M \times N \times T)$ holds the exposure control values for each pixel in the sensor. The sensing matrix S is binary, i.e., $S \in \{1, 0\}$. The value of S is 1 for each pixel for the frames for which it is on or exposed, and 0 for the rest of the frames. There is only one bump in the T_v time. The dimension T represents the compression rate (T_{cr}). The position of 1s in S is randomly decided based on a Gaussian normal distribution, which is exposed for T_b (Bump Time) number of frames. The acquired coded image $I(M \times N)$ can be denoted by the Eq.1 as represented in Fig.2.

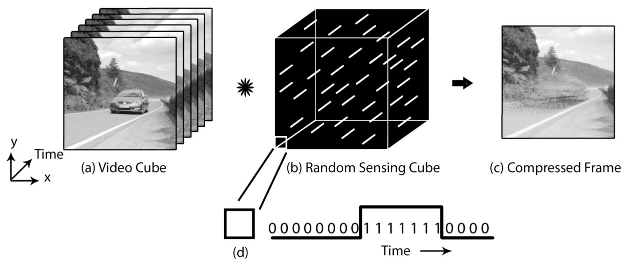

Fig. 2. Pixel-wise coded exposure

$$I(m,n) = \sum_{t=1}^{T} S(m,n,t).V(m,n,t) \qquad (1)$$

3.1.2. Enveloped Boundary Boxes for Object labeling on Compressed Image

We can observe from Fig.2 that manual labeling of the CS frames is difficult as the edges of moving objects are often indeterminable even to the human eye. Conventional Localization labeling would involve marking the bounding boxes in one individual frame. In a CS frame, the bounding box would have to enclose all the individual bounding boxes of the original constituent frames. Hence, we merge the bounding boxes of each object across the constituent frames. We used the VATIC [12] annotation module, where each frame was manually labeled for the boundary boxes to fit the object more compactly. Then with these boundary box coordinates, we find the minimum and maximum of its X and Y coordinates of the entire set of T_{cr} number of frames and envelope into a single boundary box for the CS frame. This dataset is now openly available to the community [13].

4. DATASET

For training, we used a subset of Youtube 8M [14], raw video clips were considered over artificially edited ones because, during compression, the localization of the object would be lost during the transition effects produced by the alteration of the video during editing. To incorporate more motion and object variance, we collected more videos on our own for more training data to incorporate more motion and object variance. We captured videos on multiple days and during different parts of the day and tried to maintain as much variance as possible. This dataset comprises a combination of motion, both with respect to the camera and the object, thus covering almost all the scene dynamics that can occur in the original domain (Fig.2). These carefully selected clips were temporally compressed labeled using the compression framework, as shown in Fig.2. Here T_{cr} was set to 13, while T_b was set to 3. Hence, for every T_{cr} frames in the original video, a single compressed frame is generated using Equation 1. The random sensing matrix has been varied for each set of T_{cr} number of frames, to generalize compressed sensed frames and not fit over a particular sensing matrix. Testing our model's performance is essential for evaluating the model's generalizing capability. Hence for validation and experimentation of our network, we choose KITTI Multi-Object Tracking dataset [15]. We used the ground truth labels and generated the enveloped boundary boxes as discussed in the compression framework in Sec. 3.1. The dataset comprises of ≈180K samples (compressed frames and corresponding boundary boxes of objects in the compressed domain) for the classes Cars and Person.

5. MODEL AND TRAINING

The input to the network is a $M \times N$ frame which is actually temporally compressed from a $M \times N \times T$ dimensional video segment (T denotes the temporal depth for compression). The input segment is passed to a fully convolutional network that learns space-time features. The dimensions of the output features are $\frac{M}{T} \times \frac{N}{T} \times 5$, here the 5D associates the variables that represent the objects presence and localization information namely (p, w, h, x, y). The spatial resolution is reduced by a factor of T, while the existence of the objects along the temporal dimension is preserved. Our network is largely based on Yolo. There is three major difference between our model and the original Yolo model: (i) We employ residual connection in our model, which has been shown to preserve the low-level features (ii) We used resnet blocks as opposed to the original Yolo's vanilla convolutions, which has shown to preserve the low-level features in the model. The training converges within 25 epochs, using Adam optimizer for parameter updates with learning rate = 1e-5, beta_1 = 0.9, beta_2 = 0.999, and epsilon = 1e-08. The mini-batch size was 16 for the training. We trained this network on the dataset with a train and test split ratio of 7:3. Our model is trained using mean-squared error on (w, h, x, y) and a multi-class cross-entropy loss on for classification probability (p).

Model	mAP%		Inference Time(ms)	Model Size(MB)
	Pre-Trained only	Fine Tuned on CS Dataset		
YOLOv3 320[16]	8.53	41.66	22	237
Tiny YOLO [17]	2.74	10.12	25	92
YOLOv2 608x608 [18]	5.01	34.47	40	169
SSD300 [19]	3.22	33.22	61	208
SNIPER [20]	7.61	**49.16**	66	536
SSD512 [19]	5.59	36.75	156	208
Proposed Model (Trained-CS Dataset)	**46.27**		**28**	**241.9**

Table 1. Comparative study of accuracy and performance, as obtained by different state-of-the-art object detection methods(both pretrained only on uncompressed data and after fine-tuning on CS dataset)

6. EVALUATION

6.1. Mean Average Precision

We have followed the COCO dataset [21] method of evaluating the Mean Average Precision (mAP), by averaging the Average Precision (AP) over multiple IOU (Intersection over Union ratio over predicted box and ground truth) thresholds from 0.50 to 0.95. The 'Car' and 'Person' classes got an mAP of 42.42% and 50.12%, respectively. Comparison of mAP values of existing object detection models in comparison to our model Table 1. The mAP of our proposed model suggests that it performs far better with that of the generic models that work on the original domain, even though the boundaries and texture of the objects are not so well defined in the CS domain. The proposed model could detect and localize all the classes it has been trained in the compressed domain.

7. EXPERIMENTS

Compression rates and bump times play a vital role that explains how well the model has learned to detect objects in the compressed domain. The AP values at different IOUs for different bump times and compression rates are shown in Fig.4. The results plotted by the mean mAP (black color-coded), show no conclusive proof of any trend of decline or ascent in the mAP with respect to the bump time or the compression unless they are the extremes. This shows that our model is generalized and can work with other compression rates and bump times. Our model was able to detect other compression standards is because CS frames have high variance in the appearance of the objects, based on the amount of motion in the constituent frames.

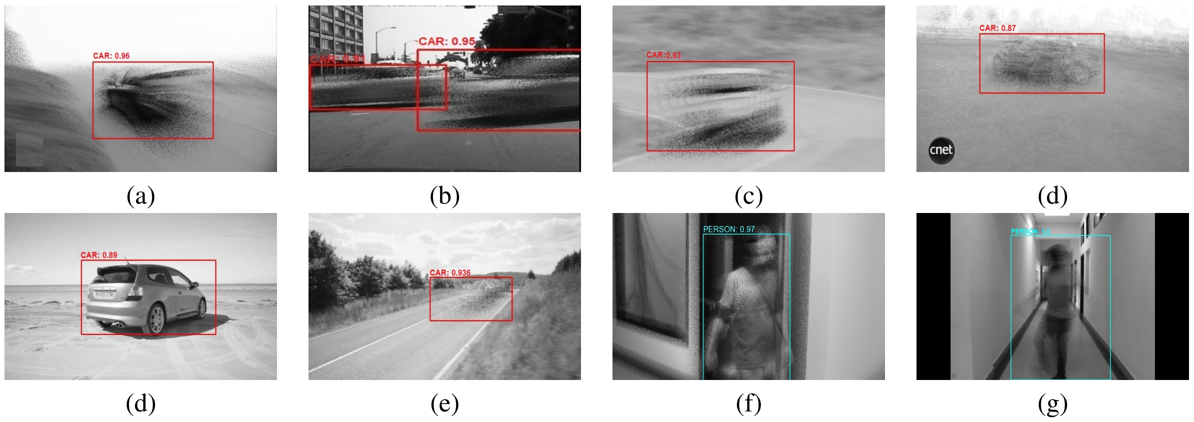

Fig. 3. Results on CS Frames. (a),(f) Both object and camera in motion; (b),(e) Object in motion and Stationary camera; (c),(d) Stationary object and camera in motion. (b) Multiple objects detected in a CS frame. Highlighted with bounding boxes in Red, Cyan for class 'CAR' and 'PERSON' respectively.

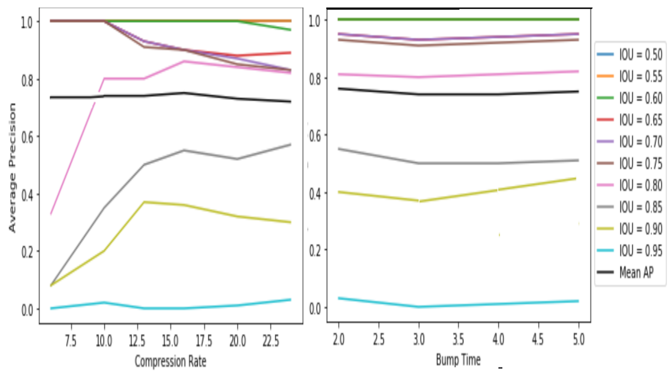

Fig. 4. Average Precision (AP) at Different IOU and mAP with change in Compression Rates T_{cr} and Bump Time T_b

8. RESULTS AND DISCUSSION

We compared with a few state-of-the-art object detectors on the compressed frames, which are listed in Table 1. It is observed that our model displays relatively superior results. We can observe that our proposed method achieves substantially better performance from our above ablation study (%mAP with fine-tuning) than all other SOTA competitors performed on the CS dataset except SNIPER with a negligible difference of 2.9%. However, considering the real-time deployment aspect, SNIPER being a heavier architecture consuming 536MB makes it resource intensive compared to our proposed model, which only consumes 242MB. We can observe the same by comparing inference time for the efficacy between these two models. Hence, our solution offers the best of both worlds, which allows an efficient real-time framework on a low-power device like TX2 and a respectable performing architecture. Most of the state of art architectures failed to detect even the presence of an object in the compressed domain, as shown in Table.1 (%mAP without fine-tuning). We also observed that the model was capable of detection with different combinations of background and foreground movements Fig.3. Baraunink explains the connection between compressive sensing and Johnson-Lindenstrauss lemma [22]. The preservation of distances between the higher-dimensional space to the lower-dimensional spaces can be why object detection in CS frames is possible using convolutional neural networks. The model was also deployed on NVIDIA TX2 and inferred using TensorRT with a run time of 34ms with mAP of 45.11, 0.025% loss from the original model. We can observe that the compressed frame contains hazy information, due to which the privacy concerns raising due to surveillance can be mitigated. The compressed frame preserves all the information intact required for the reconstruction of the frames while protecting privacy. This proves that our model has learned an essential latent representation of the objects in the compressed domain rather than to over-fit onto the hazy information on the compressed domain. On the contrary, the model's performance cannot be evaluated on the original domain as it is not the claim made in this paper.

9. CONCLUSION

In this work, we show that object detection and localization are possible directly at the compressed frame level in pixel-wise coded images in real-time. We also show that our model generalizes and works with varying T_{cr} and T_b. Thus generalizing over the entire natural set of the compressed domain. This in turn helps to reduce the time of object searching in a video by order of up to 20x(T_{cr}). We envisage that this will be the beginning of object detection and other computer vision tasks in the CS domain, making a significant improvement in surveillance as it embraces privacy at its core.

10. REFERENCES

[1] Jie Zhang, Tao Xiong, Trac Tran, Sang Chin, and Ralph Etienne-Cummings, "Compact all-cmos spatiotemporal compressive sensing video camera with pixel-wise coded exposure," *Optics express*, vol. 24, no. 8, pp. 9013–9024, 2016.

[2] Jie Zhang, Jonathan P. Newman, Xiao Wang, Chetan Singh Thakur, John Rattray, Ralph Etienne-Cummings, and Matthew A. Wilson, "A closed-loop, all-electronic pixel-wise adaptive imaging system for high dynamic range videography," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 67, no. 6, pp. 1803–1814, 2020.

[3] Richard G. Baraniuk, Thomas Goldstein, Aswin C. Sankaranarayanan, Christoph Studer, Ashok Veeraraghavan, and Michael B. Wakin, "Compressive video sensing: Algorithms, architectures, and applications," *IEEE Signal Processing Magazine*, vol. 34, no. 1, pp. 52–66, 2017.

[4] Volkan Cevher, Aswin Sankaranarayanan, Marco F Duarte, Dikpal Reddy, Richard G Baraniuk, and Rama Chellappa, "Compressive sensing for background subtraction," in *European Conference on Computer Vision*. Springer, 2008, pp. 155–168.

[5] Abin Bassam Ayub, Pallab Kumar Nath, Venkat Rangan, and Chetan Singh Thakur, "Fpga based compressive sensing framework for video compression on edge devices," in *2020 24th International Symposium on VLSI Design and Test (VDAT)*, 2020, pp. 1–5.

[6] Shiyao Wang, Hongchao Lu, and Zhidong Deng, "Fast object detection in compressed video," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 7104–7113.

[7] B. Deguerre, C. Chatelain, and G. Gasso, "Fast object detection in compressed jpeg images," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, 2019, pp. 333–338.

[8] L. Kong, R. Dai, and Y. Zhang, "A new quality model for object detection using compressed videos," in *2016 IEEE International Conference on Image Processing (ICIP)*, 2016, pp. 3797–3801.

[9] Q. Liu, B. Liu, Y. Wu, W. Li, and N. Yu, "Real-time online multi-object tracking in compressed domain," *IEEE Access*, vol. 7, pp. 76489–76499, 2019.

[10] Michael Iliadis, Leonidas Spinoulas, and Aggelos K. Katsaggelos, "DeepBinaryMask: Learning a Binary Mask for Video Compressive Sensing," pp. 1–13, 2016.

[11] Kai Xu and Fengbo Ren, "Csvideonet: A real-time end-to-end learning framework for high-frame-rate video compressive sensing," in *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2018, pp. 1680–1688.

[12] Carl Vondrick, Donald Patterson, and Deva Ramanan, "Efficiently scaling up crowdsourced video annotation," *International Journal of Computer Vision*, vol. 101, no. 1, pp. 184–204, 2013.

[13] Sathyaprakash Narayanan, Yeshwanth Bethi, and Chetan Singh Thakur, "A compressive sensing video dataset using pixel-wise coded exposure," *arXiv preprint arXiv:1905.10054*, 2019.

[14] Sami Abu-El-Haija, Nisarg Kothari, Joonseok Lee, Paul Natsev, George Toderici, Balakrishnan Varadarajan, and Sudheendra Vijayanarasimhan, "YouTube-8M: A Large-Scale Video Classification Benchmark," 2016.

[15] Andreas Geiger, Philip Lenz, and Raquel Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[16] Joseph Redmon and Ali Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[17] Joseph Redmon and Ali Farhadi, "YOLO9000: Better, Faster, Stronger," *CVPR 2017*, 2016.

[18] Joseph Redmon and Ali Farhadi, "Yolo9000: Better, faster, stronger," 2016.

[19] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg, "Ssd: Single shot multibox detector," in *Computer Vision – ECCV 2016*, Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, Eds., Cham, 2016, pp. 21–37, Springer International Publishing.

[20] Bharat Singh, Mahyar Najibi, and Larry S Davis, "Sniper: Efficient multi-scale training," in *Advances in Neural Information Processing Systems*, 2018, pp. 9310–9320.

[21] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár, "Microsoft coco: Common objects in context," 2014.

[22] Richard Baraniuk, Mark Davenport, Ronald Devore, and Michael Wakin, "The Johnson-Lindenstrauss Lemma Meets Compressed Sensing," pp. 1–9, 2006.