An Efficient Compressive Convolutional Network for Unified Object Detection and Image Compression

Xichuan Zhou*[1], Lang Xu[1], Shujun Liu[1], Yingcheng Lin[1], Lei Zhang[1], Cheng Zhuo[2]
[1]College of Microelectronics and Communication Engineering, Chongqing University, Chongqing, China, 400044,
[2]College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou, Zhejiang, China 310058,
* Indicates the corresponding author, Email: zxc@cqu.edu.cn

Abstract

This paper addresses the challenge of designing efficient framework for real-time object detection and image compression. The proposed Compressive Convolutional Network (CCN) is basically a compressive-sensing-enabled convolutional neural network. Instead of designing different components for compressive sensing and object detection, the CCN optimizes and reuses the convolution operation for recoverable data embedding and image compression. Technically, the incoherence condition, which is the sufficient condition for recoverable data embedding, is incorporated in the first convolutional layer of the CCN model as regularization; Therefore, the CCN convolution kernels learned by training over the VOC and COCO image set can be used for data embedding and image compression. By reusing the convolution operation, no extra computational overhead is required for image compression. As a result, the CCN is 3.1 to 5.0 fold more efficient than the conventional approaches. In our experiments, the CCN achieved 78.1 mAP for object detection and 3.0 dB to 5.2 dB higher PSNR for image compression than the examined compressive sensing approaches.

Introduction

Since the last a few years, the approach of convolutional neural network (CNN) has been proven to be successful for various computer vision applications (Sun et al. 2014; Amato et al. 2016; Loquercio et al. 2018). For example, one application scenario is the smart wireless surveillance camera, which is a new type of device that performs object detection using embedded system. However, due to the constraints of computation resource and power budget, it is still a challenge to implement real-time CNN-based computer vision using wireless embedded systems (Sze et al. 2017).

Early CNN based object detection approaches, such as the well-known Regional CNN (R-CNN) (Girshick et al. 2014), consisted of a region-proposal stage to select thousands of regions from the target image for object recognition, which resulted in high computational overhead. Recently, a group of more efficient CNN frameworks, including the Fast R-CNN (Girshick 2015) , Faster R-CNN (Ren et al. 2015), the Single Short Detection (SSD) (Liu et al. 2016) and the YOLO models (Redmon and Farhadi 2017) were proposed for efficient object detection. Thanks to these inspiring works, the complexity of the CNN based object detection has been significantly reduced, which makes it possible to implement CNN-based computer vision applications on wireless embedded devices.

Besides high computational complexity, another challenge of designing embedded computer vision systems is limited power budget. Since a major proportion of the energy is consumed by wireless communication (SanMiguel and Cavallaro 2016), it is desired that data should be compressed before wireless transmission. One conventional way to achieve that is to use an extra device for image compression, which will potentially increase the cost and energy consumption on the system level. To address that challenge, this paper proposes an efficient method for unified object detection and image compression. The basic idea is to optimize and reuse the CNN convolution operation for both feature extraction and image compression; therefore, no extra computational overhead is required compared to the original neural network built for object detection.

The proposed method is based on the modern compressive sensing technique. It has been proven that, due to the sparse nature of the image signal, one may recover it from far fewer samples by solving undetermined linear systems (Candès, Romberg, and Tao 2006). To recover a compressed image, the compressive sensing theory requires the sufficient condition of incoherence, which is applied through the isometric property (Donoho 2006; Candès and Wakin 2008). At the front end, the conventional compressive sensing approach uses random matrices for data embedding, which has been proven to satisfy the incoherence condition with high probability (Candes and Tao 2006). It is worth noting that, since the *convolution operation* is also linear, it can be used for data embedding. Romberg found that, the convolution between the data and a random embedding matrix is an efficient compressive sensing strategy, and the image compressed by random convolution can be recovered via L1-norm optimization (Romberg 2009).

As the pioneering work to use the convolution operation for compressive sensing, research (Romberg 2009) still adopted random embedding matrices to ensure the incoherence condition. Recently, a group of learning based approaches were proposed to estimate deterministic embedding matrices using a set of training samples. As an

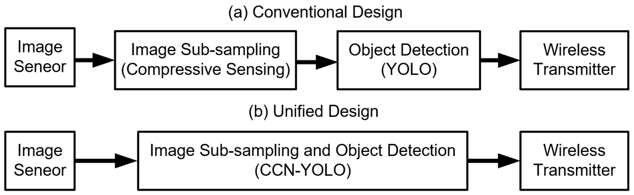

Figure 1: The CCN-YOLO method designed for wireless computer vision applications. With no extra computational overhead, the unified design is over three fold more efficient than the conventional design that contains separate modules for image sub-sampling and object detection.

early attempt, the NuMax approach estimated the linear near-isometric embedding matrix via solving a semi-definite program (SDP) problem (Hegde et al. 2015). Baldassarre proposed a learning based sub-sampling approach, which formulated combinatorial optimization problems to estimate the embedding matrices (Baldassarre et al. 2016). Blasiok relaxed the NuMax SDP problem to an eigenvalue problem, thus it can be applied to larger data set that contained a few thousand samples (Błasiok and Tsourakakis 2016).

Our method is also a learning based approach for compressive sensing and object detection. Fig. 1 compares the proposed method and the conventional implementation, which uses separate modules to perform image compression and object detection. As a unified approach, the CCN reformulates and reuses the convolution operation for both feature extraction and recoverable data embedding; therefore, no extra computational overhead or latency is required for image compression at the front end, which saves cost and energy for wireless computer vision implementations.

The incorporation of compressive sensing in the deep learning framework is not trivial. Different from work (Romberg 2009) which studied random convolution operations, the CCN convolution operation is deterministic and learned from samples. To satisfy the incoherence condition for accurate image recovery, a relaxed measurement of mutual coherence between the embedding matrix and the basis matrix is defined. And the coherence measurement is used as regularization for the modified convolutional layer. By minimizing the coherence measurement through training, the compressed output features of the modified convolutional layer can be used to reconstruct the original image with higher quality.

From a system point of view, there are several advantages of adopting the incoherence regularization. First, by learning near-isometric embedding matrices from large-scale training set, the CCN-compressed images have 3.0-5.2 dB higher P-SNR than the conventional compressive sensing approaches. Secondly, since there is no computational overhead or extra latency at the inference stage, the CCN is over three-fold more efficient than the conventional implementations shown in Fig. 1. Thirdly, since the CCN training process is based on back propagation, it is significantly more efficient than the NuMax and ADAGIO approaches of estimating near-isometric data embedding (Hegde et al. 2015; Błasiok and Tsourakakis 2016). The CCN extends the approach of data-driven compressive sensing to large-scale data sets.

Recently, several researches attempted to address different problems of compressive sensing using convolutional neural networks. Mousavi proposed a CNN based approach to recover the original signal from random undersampled measurements (Mousavi and Baraniuk 2017). Later, Mousavi improved his approach by adopting the CNN for both signal compression and recovery (Mousavi, Dasarathy, and Baraniuk 2017). Similar idea was explored by Lu for image processing (Lu et al. 2018). Iliadis proposed a deep fully-connected networks for recovering video images (Iliadis, Spinoulas, and Katsaggelos 2018). The main difference between these researches and the proposed method is that, instead of building a network for image compression or recovery, the CCN is a network enhancement approach, which enables object-detection networks to gain the ability of image compression without degrading performance.

Background

The proposed method attempts to add the function of recoverable data embedding in the CNN framework. Compressive sensing is a sampling approach for the signals that are known to be sparse. Suppose the signal $\mathbf{x} \in \mathbf{R}^{\mathrm{N}}$ is an N dimensional vector, which can be sampled using an *embedding matrix* $\mathbf{\Phi} \in \mathbf{R}^{\mathrm{M}\times\mathrm{N}}$ as

$$\mathbf{y} = \mathbf{\Phi x} \tag{1}$$

The constant M is defined as the sampling rate. Because M is smaller than N in compressive sensing, Eq. 1 is underdetermined, the signal **x** cannot be uniquely recovered from $\mathbf{\Phi}$ and **y**. However, the assumption of sparsity allows the signal **x** to be represented using a set of sparse coefficients $\mathbf{s} \in \mathbf{R}^{\mathrm{N}}$ and a matrix of basis $\mathbf{\Psi} \in \mathbf{R}^{\mathrm{N}\times\mathrm{N}}$ as $\mathbf{x} = \mathbf{\Psi s}$. Then we have $\mathbf{y} = \mathbf{\Phi\Psi s} = \mathbf{\Theta s}$ where $\mathbf{\Theta} = \mathbf{\Phi\Psi}$ is an $\mathrm{M}\times\mathrm{N}$ measurement matrix. Since **s** is sparse, it is possible to retrieve the value of **s** by solving the L1 norm minimization problem as

$$\min_{\mathbf{s}} \|\mathbf{s}\|_1, \;\; \text{s.t.} \|\mathbf{y} - \mathbf{\Theta s}\|_2 < \xi \tag{2}$$

After solving Eq. 2, one could approximate the original signal **x** by $\hat{\mathbf{x}} = \mathbf{\Psi s}$. To find the unique sparest solution of Eq. 2, $\mathbf{\Phi}$ should be built to satisfy the *incoherence condition*, i.e. the matrices $\mathbf{\Phi}$ and $\mathbf{\Psi}$ should be incoherent.

In practice, $\mathbf{\Psi}$ is usually a predefined constant matrix, and one can either use random or deterministic embedding matrix $\mathbf{\Phi}$ to fulfill the incoherence condition. For the random approach, it is proven that, the matrix with Gaussian entries satisfies the incoherence condition (Candès, Romberg, and Tao 2006). On the other hand, recent researches showed that, by learning deterministic embedding matrix from data, the compressed images can be reconstructed with higher quality. The proposed method attempts to calculate the optimal deterministic embedding matrix by deep learning approach.

Proposed Method

This section presents the proposed method of Compressive Convolutional Network (CCN) for efficient object detection and image compression.

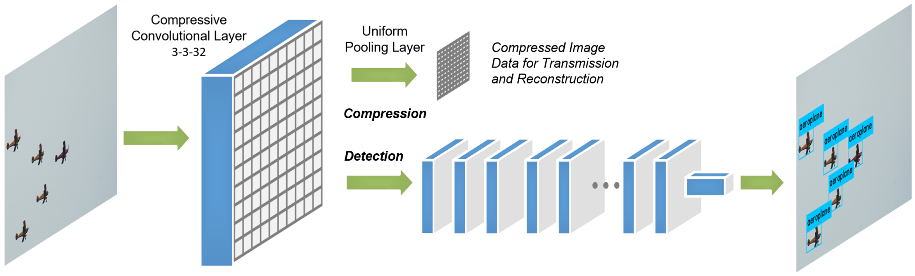

Figure 2: The prototyping Compressive Convolutional Network built for unified object detection and image compression. Compared to the standard YOLOv2 model, the first convolutional layer and pooling layer are modified and optimized for recoverable data embedding and image compression.

Network Design

The goal of object detection is to recognize multiple objects in a single image. The class confidence and the bounding box for each object are returned. So far, many CNN based frameworks have been proposed for object detection (Ren et al. 2015; Liu et al. 2016; Redmon and Farhadi 2017). Different frameworks usually have different types of layers and different parameters, yet a typical CNN framework uses a pipeline of modules for classification or object detection.

Fig. 2 shows the prototyping CCN model trained over the VOC and COCO data sets. Though the proposed approach can be applied to different CNN based frameworks, this paper uses the YOLOv2 model as the basic architecture, which contains 1 *compressive convolutional layer*, 22 conventional convolutional layers, 1 *uniform pooling layer*, 1 reorg layer, 5 max-pooling layers and 1 soft-max layer. The YOLO model was earlier invented for efficient object detection. We modified the first two layers to perform unified feature extraction and image compression. The network configuration parameters such as layer type, kernel number and kernel size are compatible with the YOLOv2 model (Redmon and Farhadi 2017).

From a functional point of view, the CCN can perform image compression and object detection simultaneously after training. Specifically, given a test image, the uniform pooling layer, built after the first convolutional layer, subsamples the embedded image data which can be used for image recovery (Fig. 2). Meanwhile, the last layer of the CCN can return the class confidence and the bounding box of each object in the test image.

The main difference between the proposed method and other CNN based frameworks is the first compressive convolutional layer. The compressive convolutional layer is basically a 3×3 convolutional layer which is optimized for compressive sensing. As a data-driven approach, the CCN uses deterministic convolution operation for recoverable data embedding. The convolution operation is optimized to fulfil the incoherence condition, so that the sub-sampled feature maps can be used to reconstruct the original image by reconstruction algorithms like the standard orthogonal matching pursuit (Tropp and Gilbert 2007). The rest of this section shows how to learn deterministic convolution operations for recoverable data embedding.

Data Embedding via Convolution Operation

As shown in (Romberg 2009), the convolution operation can be seems as a linear embedding operation. Suppose $\mathbf{W} \in \mathbf{R}^{3\times3}$ is a kernel matrix of the compressive convolutional layer. Suppose $\mathbf{X} \in \mathbf{R}^{\mathrm{P}\times\mathrm{Q}}$ is a channel of the input image data. Given $\mathrm{N} = \mathrm{P}\times\mathrm{Q}$, $\mathbf{X}$ can be vecterized and represented as $\mathbf{x} \in \mathbf{R}^{\mathrm{N}}$. Since the convolution operation $\oplus$ in the CNN only consists of first-order multiplications, it can be reformulated as a linear embedding operation as

$$\mathrm{vec}(\mathbf{W} \oplus \mathbf{X}) \doteq \mathbf{\Phi}\mathbf{x} = \mathbf{y} \tag{3}$$

where $\mathrm{vec}(\cdot)$ is the vectorization operation, $\mathbf{\Phi} \in \mathbf{R}^{\mathrm{N}\times\mathrm{N}}$ is the associated embedding matrix determined by the kernel matrix $\mathbf{W}$, $\mathbf{y} \in \mathbf{R}^{\mathrm{N}}$ is the vectorized output of the convolution operation.

Since the kernel matrices in the CNN model are learned from training samples, the embedding matrix $\mathbf{\Phi}$ is determined by iterative training. Define the row vector $\mathbf{w} \in \mathbf{R}^{2\mathrm{Q}+3}$ based on the kernel matrix $\mathbf{W}$ as $\mathbf{w} = [w_1, w_2, w_3, \mathbf{0}, w_4, w_5, w_6, \mathbf{0}, w_7, w_8, w_9]$, where $w_1, \ldots, w_9$ are the nine elements of the kernel matrix $\mathbf{W}$. The embedding matrix associated with the kernel can be written as

$$\mathbf{\Phi} = \begin{pmatrix} \mathbf{w} & \mathbf{0} & \ldots & \mathbf{0} & \mathbf{0} \\ 0 & \mathbf{w} & \ldots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \ldots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \ldots & \mathbf{w} & 0 \\ \mathbf{0} & \mathbf{0} & \ldots & \mathbf{0} & \mathbf{w} \end{pmatrix}_{\mathrm{N}\times\mathrm{N}} \tag{4}$$

where $\mathbf{0}$ and 0 are the vector and scalar form of zeros. The embedding matrix $\mathbf{\Phi}$ is a sparse matrix with 3×3 nonzero elements in each row.

It is worth noting that Eq. 3 and Eq. 1 have the same linear form. But different from the standard compressive sensing, the convolution operation of the CNN doesn't reduce the dimension of the data. To compress the image to M dimensions ($\mathrm{M} \ll \mathrm{N}$), a *uniform-pooling layer* is built after the compressive convolutional layer for dimension reduction. In practice, a single N×N feature map is selected for sub-sampling, and M×N elements are uniformly extracted in a row-by-row fashion.

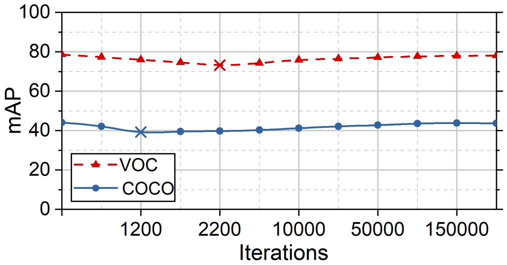

Figure 3: Change of mean average precision (mAP) during the training process over the VOC and COCO data sets.

Incoherence Regularization

As a data-driven approach, Eq. 3 uses the deterministic embedding matrix $\mathbf{\Phi}$ learned from samples. According to the theorem of compressive sensing, $\mathbf{\Phi}$ should satisfy the incoherence condition that the embedding matrix $\mathbf{\Phi}$ and the basis matrix $\mathbf{\Psi}$ should be incoherent. Suppose φ_i is the ith row of $\mathbf{\Phi}$, and ψ_j is the jth column of $\mathbf{\Psi}$. The *mutual coherence* between $\mathbf{\Phi}$ and $\mathbf{\Psi}$ is defined as

$$\mu(\mathbf{\Phi}, \mathbf{\Psi}) = \max | < \varphi_i, \psi_j > | = \max |\varphi_i \psi_j| \qquad (5)$$

Within the compressive sensing framework, lower coherence between $\mathbf{\Phi}$ and $\mathbf{\Psi}$ translates to fewer samples required for recovering the signal. For data-driven approaches, it is intuitive to minimize the mutual coherence during the training process; However, since the mutual coherence in Eq. 5 is not differentiable, we relax it as the following *coherence measurement*:

$$\mathcal{R}_{\mathbf{W}}(\mathbf{\Phi}) = \sum_{ij} < \varphi_i, \psi_j >= \sum_{ij} |\varphi_i \psi_j| \qquad (6)$$

Since the basis matrix $\mathbf{\Psi}$ is usually a predefined constant matrix, e.g. a discreet wavelet transform (DWT) matrix, the coherence measurement defined in Eq. 6 is a weighted L1 norm of the matrix variable $\mathbf{\Phi}$. Given that $\mathbf{\Phi}$ is determined by the kernel matrix $\mathbf{W}$ as in Eq. 4, the function $\mathcal{R}_{\mathbf{W}}$ is differentiable with respect to $\mathbf{W}$; therefore, one could use Eq. 6 as regularization and estimate the embedding matrix $\mathbf{\Phi}$ by minimizing $\mathcal{R}_{\mathbf{W}}$.

In practice, the incoherence regularization term $\mathcal{R}_{\mathbf{W}}$ is used for estimating the kernels of the compressive convolutional layer. The values of $\mathbf{W}$ and $\mathbf{\Phi}$ are calculated via a gradient decent training process. By reducing the coherence between $\mathbf{\Phi}$ and $\mathbf{\Psi}$, the iterative training process could calculate the near-isometric embedding matrix $\mathbf{\Phi}$ for effective compressive sensing.

Model Implementation

To construct the embedding matrix $\mathbf{\Phi}$ for image compression, we selected the *single* best kernel of the compressive convolutional layer that achieved the highest Peak Signal to Noise Ratio (PSNR). It is worth noting that, the output feature maps of the compressive convolutional layer had the same dimension as the input image data. The dimension reduction process and the compression rate were controlled by the uniform pooling layer.

Data Set	Train	Test	Total
BSD100	100	-	100
VOC2007+2012	16551	4952	21053
COCO	117263	5000	122263

Table 1: Data sets used to evaluate the performance of object detection (VOC & COCO) and image compression (VOC & BSD100).

As for the training process of the prototyping CCN model, the pre-trained model of YOLOv2 was used as the initial parameter setting. YOLOv2 was an open-source framework pre-trained for objection detection(Redmon and Farhadi 2017). We further trained the network over the VOC and COCO data sets for 200000 iterations respectively, using stochastic gradient descent with a starting learning rate of 0.0001, incoherence regularization weight of 0.0005 and momentum of 0.9.

Since the YOLOv2 model was designed for object detection, modifying the first convolutional layer might affect the performance of object detection. To remedy the loss of accuracy, we controlled the influence of incoherence regularization via a *two-stage training strategy*. At the first stage, the coherence measurement associated with each kernel of the compressive convolutional layer was monitored during training iterations. Fig. 3 shows the change of mean average precision (mAP) during the training process over the VOC and COCO data sets. For the VOC data set, after 2200 iterations, the coherence measurement began to stabilize, then the values of the top 3 kernels with the lowest coherence measurement were *frozen*. Then at the second stage, we removed the incoherence regularization, and the parameters of the whole deep neural network, including the 29 unfrozen kernels in the compressive convolutional layer, were further updated until reaching convergence. As shown in Fig. 3, By updating the parameters without incoherence regularization, the proposed method suffered almost no loss of detection accuracy over the VOC and COCO data sets.

Experiment Results

We performed a group of experiments to evaluate the CCN for both object detection and image compression. Our experiments showed that, the prototyping CCN-YOLO model achieved relatively high image compression performance, while keeping competitively high accuracy and efficiency for object detection. A demo program of the proposed method is uploaded on the Github website at https://github.com/sosaaaad2/CCN.

Data Sets and Experiment Setting

For better comparison, we use the data sets and configuration of the YOLOv2 research to evaluate the proposed method (Redmon and Farhadi 2017). Specifically, we evaluate the proposed method for object detection and image compression over the BSD100, the VOC (2007+2012) and the COCO data sets (Table 1). Both the VOC and the COCO data sets are widely used for evaluating object detection approaches. The VOC set has 21530 images containing

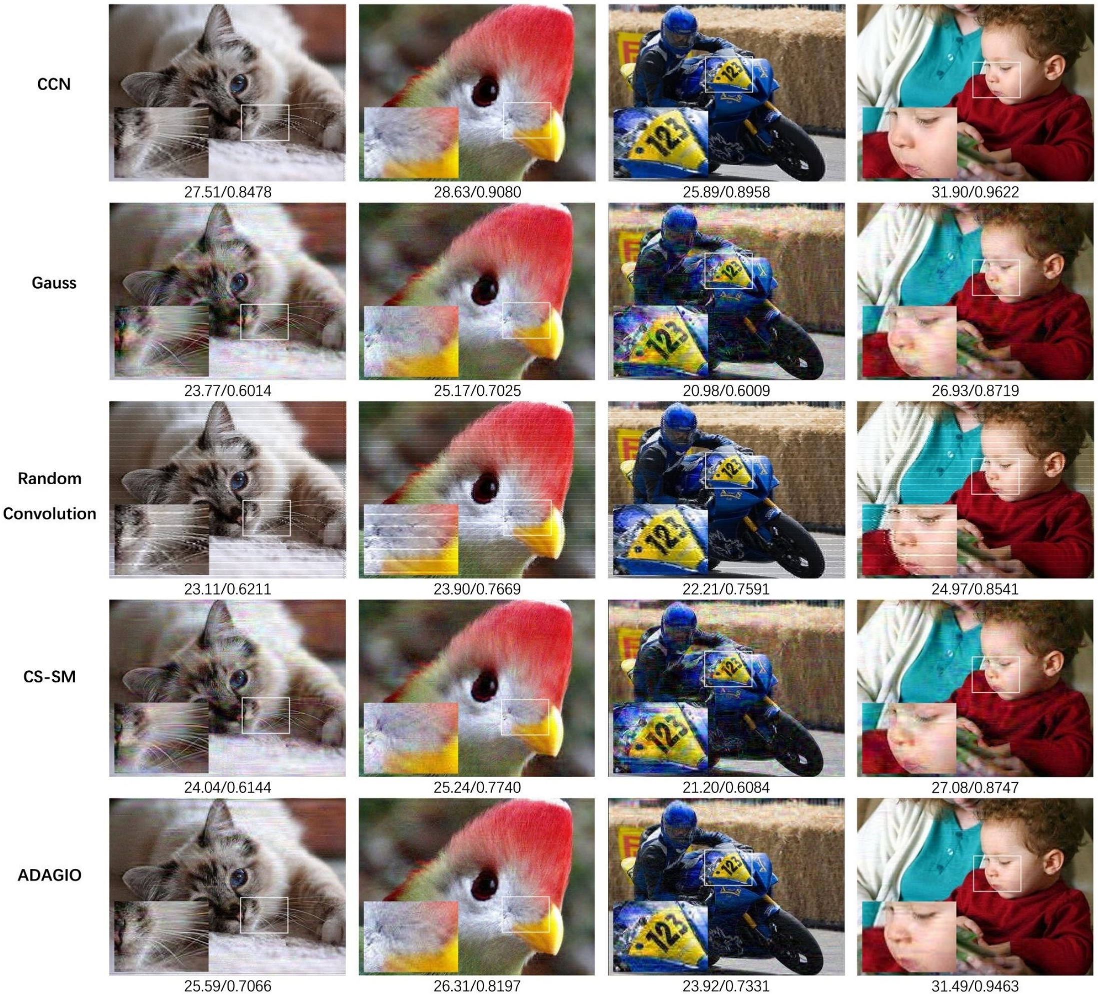

Figure 4: Comparing the proposed CCN approach with different compressive sensing approaches using Gaussian, random convolution, CS-SM and ADAGIO sub-sampling strategies. The image quality measurements of PSNR/SSIM are shown below each subgraph.

27450 annotated objects and 6929 segmentations. The COCO data set includes more than 200000 images in 80 classes. The Berkeley BSD100 data set is a small image set with 100 images, which is used to evaluate the computationally-expensive data-driven compressive sensing approaches.

Different measurements are adopted in our experiment to compare the CCN with other approaches for image compression, including the peak signal to noise ratio (PSNR) and the structural similarity index (SSIM). Typical values for the PSNR in lossy images and video compression are between 30 and 50 dB, provided the bit depth is 8 bits, but it is acceptable to be 20 dB to 25 dB for wireless transmission loss (Li and Cai 2007). The SSIM measures the similarity in luminance, contrast, and structure between two images. Both measurements of the PSNR and SSIM are widely used to measure the quality of image compression. For object detection, the mean average precision (mAP), the top-5 class average accuracy and the recall rate over the VOC and COCO test data sets are calculated and compared for evaluation.

Similar to the NuMax and ADAGIO approaches, the CCN is a data-driven near-isometric data embedding approach. Generally, different back-end image reconstruction algorithms can be applied to recover the data compressed by these approaches. It may not be the optimal solution, but for fair comparison, we adopt the standard orthogonal matching pursuit (OMP) algorithm for image reconstruction. The OMP algorithm uses the discrete wavelet transform (DWT) matrix as the basis matrix $\mathbf{\Psi}$ to rebuild the image (Tropp and Gilbert 2007). The block size of the OMP method used in our experiment is the default value of 16×16 pixels.

Performance of Image Compression

We compared the CCN with other approaches over the VOC and BSD100 data sets for image compression. We examined the conventional compressive sensing approach using Gaussian embedding matrices. The random convolution approach (Romberg 2009) and sparse matrix based approach (CS-

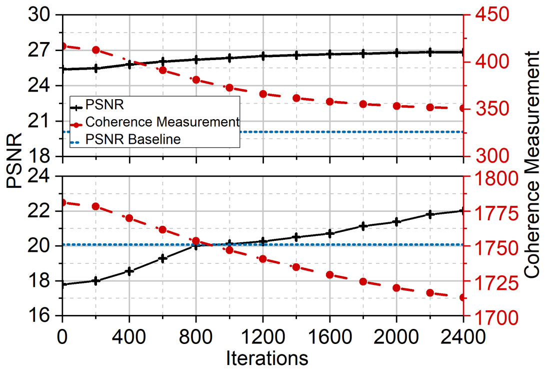

Figure 5: The change of coherence measurement and average PSNR during the training process over the VOC data set. Only the top two kernels with the highest PSNR are shown here to save space.

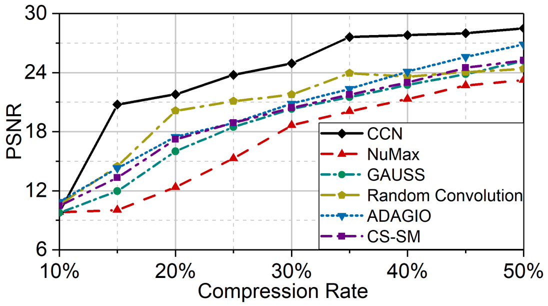

Figure 6: Image reconstruction PSNR for the examined approaches over the BSD100 data set.

SM) (Gilbert and Indyk 2010) were also compared. As a data-driven approach, we compared the CCN with the NuMax and the recently proposed ADAGIO algorithm. A thousand images randomly selected from the VOC data set, along with all the images in the BSD100 image set, were compressed using different approaches. The CCN was set to use a single kernel of the compressive convolutional layer for data embedding. The standard OMP algorithm was used for image reconstruction (Tropp and Gilbert 2007) .

Table 2 summarizes the experiment results over the VOC and BSD100 data sets with fixed compression ratio of 0.33. For comparison, we evaluated the original kernels of the YOLOv2 for data embedding. Experiment showed that, the best kernel of the first convolutional layer of YOLOv2 achieved 25.3 dB average PSNR for image compression, which was 2.1 dB higher than the random convolution approach (RandConv). By incorporating the incoherence regularization, the CCN achieved 26.5 dB average PSNR. The CCN also outperformed the data-driven approaches of the NuMax and ADAGIO by 2.6 dB.

Fig. 4 compares the results of four typical VOC images. The measurements of PSNR/SSIM are shown below each subgraph. Generally, the CCN shows higher PSNR and SSIM compared with other approaches. The images compressed by the CCN look better and are less noisy. The square in each graph shows the magnified detail of the recovered images. It seems that, the CCN images has higher sharpness in regions containing complex contexture.

Data Set	**BSD100**		**VOC**	
Method	PSNR	SSIM	PSNR	SSIM
CCN	*26.56*	*0.8192*	*26.54*	*0.8786*
	±2.98	*±0.0550*	*±3.71*	*±0.0745*
ADAGIO	22.42	0.6055	23.89	0.6325
	±1.75	±0.0632	±3.38	±0.0618
RandConv	22.31	0.6243	22.21	0.6608
	±1.91	±0.0642	±2.93	±0.0671
CS-SM	21.39	0.5954	21.46	0.6217
	±2.86	±0.0941	±3.74	±0.0521
GAUSS	21.32	0.5921	22.48	0.6409
	±2.93	±0.0976	±4.18	±0.1188

Table 2: Image recovery quality over the BSD100 and VOC data sets at the fixed compression ratio of 0.33.

Number of Images	10	20	40	100
NuMax	267	838	1537	4460
ADAGIO	0.7432	2.041	7.263	45.782
CCN-YOLO	3.034	6.072	12.325	31.069

Table 3: Training time (second) of the examined data-driven compressive sensing approaches over the BSD100 set.

The CCN approach uses the incoherence regularization to enable the convolutional neural networks to perform data compression. Fig. 5 shows the change of coherence measurement and the average PSNR during the training process over the VOC data set. With the regularization weight set as 0.0005, the coherence measurement continuously drops at the first two thousand iterations, while the PSNR of the compressed images increases by 1.2 dB to over 4.0 dB. Fig. 5 shows the results achieved by the top two convolution kernels of the compressive convolutional layer. The baseline PSNR is achieved by the conventional compressive sensing using Gaussian embedding matrices. This experiment explains the effectiveness of the incoherence regularization for image compression.

Fig. 6 compares the examined compressive sensing approaches across a range of compression rate from 10% to 50% with a step size of 5%. This experiment was performed over the small BSD100 data set, because the NuMax and ADAGIO approaches need to solve semi-definite program and eigenvalue problems, which were difficult to be applied over large-scale data sets. Generally, the CCN achieved the highest average PSNR over the BSD100 data set.

We also compared the training efficiency of the examined data-driven approaches using deterministic embedding matrices. The experiment was performed over a computer with one i7-6800k CPU and 32 GB memory. Since the BSD100 images have 38400 dimensions, it was hard for the NuMax approach to run on a regular computer. We split the images into 16 × 16 blocks for the NuMax algorithm. The training time of the prototyping YOLO-CCN network, the ADAGIO and the NuMax is listed in Table 3. Even over a small data

Method	**Training Set**	**mAP**	**FPS**
Fast R-CNN	VOC 2007+2012	70.0	0.5
Faster R-CNN ResNet	VOC 2007+2012	76.4	5
SSD300	VOC 2007+2012	74.3	22
SSD500	VOC 2007+2012	76.8	19
YOLOv1	VOC 2007+2012	63.4	45
YOLOv2	VOC 2007+2012	78.6	40
CCN-YOLO	VOC 2007+2012	**78.1**	**40**
Fast R-CNN	COCO trainval	35.9	0.5
Faster R-CNN ResNet	COCO trainval	45.3	5
SSD300	COCO trainval	41.2	22
SSD500	COCO trainval	46.5	19
YOLOv2	COCO trainval	44.0	40
CCN-YOLO	COCO trainval	**43.8**	**40**

Table 4: Comparing the proposed method with other CNN based approaches for object detection. Both accuracy measurement of mean average precision (mAP) and efficiency measurement of frames per second (FPS) are listed.

set of 100 images, the NuMax approach still consumed over 4000 seconds to train. The ADAGIO approach relaxed the NuMax SDP problem into an eigenvalue problem, but it was less efficient than the proposed CCN approach when more than 100 images were used for training. As for the large-scale COCO data set, the ADAGIO approach was impractical due to unaffordable memory and time complexity.

Performance of Object Detection

Generally, the incorporation of image compression in the prototyping CCN-YOLO model did not degrade the performance of object detection. The CCN-YOLO model trained over the VOC data set achieved 78.1% mean average precision (mAP), which was almost the same as the original YOLOv2 model. The CCN-YOLO also achieved similar recall rate of 86.6%, and competitively high average accuracy of 92.8% over the top 5 classes of the VOC data set. Giving that the recall rate and the average accuracy over the top 5 classes for the original YOLOv2 model were 86.8% and 93.3% respectively, the CCN suffered marginal loss of performance compared to the YOLOv2 model.

Table 4 summarizes the results of inference accuracy and speed of the CCN-YOLO model, which is compared with different variants of the R-CNN, the SSD and the YOLO models. The CCN-YOLO model shows competitively high mAP over the VOC and COCO data sets. Moreover, the incorporation of image compression function doesn't affect inference efficiency; therefore, the CCN-YOLO model could perform object detection and image compression at the speed of 40 frames per second on a computer accelerated by one NVIDIA 1080Ti GPU.

Block Size	16*16	32*32	64*64	128*128
CCN-YOLO	2.82	2.82	2.82	2.82
Original YOLOv2	2.81	2.81	2.81	2.81
Gauss+YOLOv2	8.81	9.25	11.65	14.15
ADAGIO+YOLOv2	8.81	9.24	11.67	14.13
CS-SM+YOLOv2	8.80	9.21	11.47	13.69

Table 5: Comparing the inference time (second) of proposed unified CCN approach and the conventional approaches with separate object detection and sub-sampling components.

The CCN is a compressive-sensing-enabled convolutional neural network, which is a unified approach for object detection and image compression. Table 5 compares the combined latency of image compression and object detection of the compared designs illustrated in Fig. 1. The proposed method is compared with the conventional strategies that a YOLOv2 model is performing object detection side-by-side with a compressive sensing module using different types of embedding matrices. It is worth noting that, the compared compressive sensing approaches require $O(\frac{M\times N}{B})$ times of multiplications at the front end for data embedding. Though the complexity of these strategies depends on the block size B, the CCN-YOLO model is 3.1 to 5.0 fold more efficient than the compared approaches, because no computational overhead is required for compression sensing.

Conclusion and Discussion

To address the challenge of implementing CNN based computer vision applications on wireless embedded devices, this paper presents an efficient Compressive Convolutional Network (CCN) for unified object detection and image compression. A incoherence regularization approach is proposed to enable the convolution operation to perform near-isometric data embedding for compressive sensing. This paper focuses on the CCN-YOLO framework; however, the incoherence regularization approach can be easily applied to other CNN based frameworks.

The benefits of the proposed method are three fold. First, as a regularization based approach, the CCN suffers almost no loss of detection accuracy. Secondly, since no computational overhead is caused by incorporating image compression function, the unified approach is over three-fold more efficient than the conventional approaches using separate compressive sensing and object detection modules. Thirdly, the CCN is more efficient than the existing approaches for data-driven near-isometric embedding. It can be applied to large-scale data sets and achieves over 3.1 dB higher image compression PSNR than the conventional approaches.

There are several limitations for the current CCN-YOLO system that can be improved in the future. First, for fair comparison, the standard OMP image reconstruction approach is adopted in our implementation, which may not be the optimal solution for the CCN or other compared approaches. It may be the reason why all the compared approaches had lower PSNR than 35 dB. Moreover, from a system point of view, the OMP method is computationally expensive, which requires extra computational overhead for image reconstruction on the back-end cloud or servers. In the future, we plan to extend the CCN and build a back-end neural network for efficient and accurate image reconstruction.

References

Amato, G.; Carrara, F.; Falchi, F.; Gennaro, C.; and Vairo, C. 2016. Car parking occupancy detection using smart camera networks and deep learning. In *Computers and Communication (ISCC), 2016 IEEE Symposium on*, 1212–1217. IEEE.

Baldassarre, L.; Li, Y.-H.; Scarlett, J.; Gözcü, B.; Bogunovic, I.; and Cevher, V. 2016. Learning-based compressive subsampling. *IEEE Journal of Selected Topics in Signal Processing* 10(4):809–822.

Błasiok, J., and Tsourakakis, C. E. 2016. Adagio: Fast data-aware near-isometric linear embeddings. *arXiv preprint arXiv:1609.05388*.

Candes, E. J., and Tao, T. 2006. Near-optimal signal recovery from random projections: Universal encoding strategies? *IEEE transactions on information theory* 52(12):5406–5425.

Candès, E. J., and Wakin, M. B. 2008. An introduction to compressive sampling. *IEEE signal processing magazine* 25(2):21–30.

Candès, E. J.; Romberg, J.; and Tao, T. 2006. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on information theory* 52(2):489–509.

Donoho, D. L. 2006. Compressed sensing. *IEEE Transactions on information theory* 52(4):1289–1306.

Gilbert, A., and Indyk, P. 2010. Sparse recovery using sparse matrices. *Proceedings of the IEEE* 98(6):937–947.

Girshick, R.; Donahue, J.; Darrell, T.; and Malik, J. 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 580–587.

Girshick, R. 2015. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, 1440–1448.

Hegde, C.; Sankaranarayanan, A. C.; Yin, W.; and Baraniuk, R. G. 2015. Numax: A convex approach for learning near-isometric linear embeddings. *IEEE Transactions on Signal Processing* 63(22):6109–6121.

Iliadis, M.; Spinoulas, L.; and Katsaggelos, A. K. 2018. Deep fully-connected networks for video compressive sensing. *Digital Signal Processing* 72:9–18.

Li, X., and Cai, J. 2007. Robust transmission of jpeg2000 encoded images over packet loss channels. In *Multimedia and Expo, 2007 IEEE International Conference on*, 947–950. IEEE.

Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.-Y.; and Berg, A. C. 2016. Ssd: Single shot multibox detector. In *European conference on computer vision*, 21–37. Springer.

Loquercio, A.; Maqueda, A. I.; del Blanco, C. R.; and Scaramuzza, D. 2018. Dronet: Learning to fly by driving. *IEEE Robotics and Automation Letters* 3(2):1088–1095.

Lu, X.; Dong, W.; Wang, P.; Shi, G.; and Xie, X. 2018. Convcsnet: A convolutional compressive sensing framework based on deep learning. *arXiv preprint arXiv:1801.10342*.

Mousavi, A., and Baraniuk, R. G. 2017. Learning to invert: Signal recovery via deep convolutional networks. In *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*, 2272–2276. IEEE.

Mousavi, A.; Dasarathy, G.; and Baraniuk, R. G. 2017. Deepcodec: Adaptive sensing and recovery via deep convolutional neural networks. *arXiv preprint arXiv:1707.03386*.

Redmon, J., and Farhadi, A. 2017. Yolo9000: better, faster, stronger. *arXiv preprint*.

Ren, S.; He, K.; Girshick, R.; and Sun, J. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, 91–99.

Romberg, J. 2009. Compressive sensing by random convolution. *SIAM Journal on Imaging Sciences* 2(4):1098–1128.

SanMiguel, J. C., and Cavallaro, A. 2016. Energy consumption models for smart-camera networks. *IEEE Transactions on Circuits and Systems for Video Technology*.

Sun, Y.; Chen, Y.; Wang, X.; and Tang, X. 2014. Deep learning face representation by joint identification-verification. In *Advances in neural information processing systems*, 1988–1996.

Sze, V.; Chen, Y.-H.; Emer, J.; Suleiman, A.; and Zhang, Z. 2017. Hardware for machine learning: Challenges and opportunities. In *Custom Integrated Circuits Conference (CICC), 2017 IEEE*, 1–8. IEEE.

Tropp, J. A., and Gilbert, A. C. 2007. Signal recovery from random measurements via orthogonal matching pursuit. *IEEE Transactions on information theory* 53(12):4655–4666.