Embedded Motion Detection via Neural Response Mixture Background Modeling

Mohammad Javad Shafiee[†], Parthipan Siva[‡], Paul Fieguth[†], Alexander Wong[†]
[†]VIP Research Group, University of Waterloo, Waterloo, ON, Canada
[‡]Aimetis Corporation, Waterloo, ON, Canada
{mjshafiee, pfieguth, alexander.wong}@uwaterloo.ca
[‡]parthipan.siva@aimetis.com

Abstract

*Recent studies have shown that deep neural networks (DNNs) can outperform state-of-the-art algorithms for a multitude of computer vision tasks. However, the ability to leverage DNNs for near real-time performance on embedded systems have been all but impossible so far without requiring specialized processors or GPUs. In this paper, we present a new motion detection algorithm that leverages the power of DNNs while maintaining low computational complexity needed for near real-time embedded performance without specialized hardware. The proposed **Ne**ural **R**esponse **M**ixture (NeRM) model leverages rich deep features extracted from the neural responses of an efficient, stochastically-formed deep neural network (StochasticNet) for constructing Gaussian mixture models to detect motion in a scene. NeRM was implemented embedded on an Axis surveillance camera, and results demonstrated that the proposed NeRM approach can achieve strong motion detection accuracy while operating at near real-time performance.*

1. Introduction

One of the most basic functionalities required of modern surveillance cameras is the ability to record video when motion is detected within the field of view of the camera. This allows for reduced storage requirements for the videos, as well as the ability to quickly review historical videos focusing only on the times when there is something happening in the scene. This requirement has driven surveillance camera manufacturers (e.g., Axis, Samsung, etc.) to build motion detection algorithms right on the camera. Due to the reduced computational capabilities of these cameras, the embedded motion detection algorithms used tend to be very simple pixel change detection algorithms. For example, the pixel colour can be modelled as a Gaussian mixture model using an on-line approximation [19] and when a pixel value does not conform to the modelled Gaussian it is considered to be "in-motion" (i.e., the pixel has changed value due to a moving object in the scene).

Gaussian mixture models (GMM) are a simple and fast algorithm that can perform motion detection in real-time, right on the camera. However, using colour as the feature to represent each pixel has several drawbacks. Most notably, false motion detection can occur as a result of factors such as: I) illumination changes in the scene (e.g., indoor light flickering, shadows, overhanging clouds passing by, and strong sunlight), and II) subtle motions from waving background objects (e.g., branches and leaves of trees moving because of wind). Both illumination changes and subtle motions from waving background objects will change pixel colour, but should not be considered as true motion in the scene.

A number of strategies have been proposed to reduce the false motion detection [1, 9, 18]; however, such methods remain limited in dealing with subtle motions. Although statistical background subtraction methods [5, 15, 19, 21] have addressed noise and dynamic backgrounds, they are highly depended on a learning rate to update the background model to account for gradual illumination changes, which makes them prone to large errors (false alarms) when subject to sudden illumination and motion changes in the background. Furthermore, the computational complexity of such methods restrict their use on embedded devices.

An alternative strategy for robust motion detection while maintaining the computation efficiency of the GMM is to use GMM with different features such as different colour spaces or texture features [2]. Several texture features have been utilized to model the image. Matsuyama *et al.* [13] obtained the correlation between two blocks in the image based on a normalized vector distance function. Edge and color histograms in a block have been utilized as set of texture features by Mason [12] to model the background. Lo-

cal binary pattern (LBP) [14] is another well-known texture feature to capture the background statistics.

Although the use of different colour spaces or texture features have shown some robustness to noise, the choice of colour space or texture feature is still hand-crafted based on our understanding of the human visual system and statistics which we believe to be illumination invariant, and thus their generic nature limits their ability to comprehensively capture the unique traits of objects (e.g., people, vehicles, animals, etc.) in real-world surveillance environments. Interestingly, recent work in deep neural networks (DNNs) have shown that deep features obtained from convolutional neural networks (CNNs) [10] learned from large natural image datasets can be used to obtain significant improvements over hand-crafted texture features such as histogram of gradients [6]. Girshick *et al.* [6] applied pre-trained CNNs in a hierarchical framework to compute region proposals based on deep features. They reported a 30% improvement on the PASCAL VOC object detection problem. Gupta *et al.* [8] extracted deep features from a CNN learned from RGB-D images for object detection and image segmentation. Razavian *et al.* [16] examined deep features from CNNs as generic descriptors for different recognition tasks and reported consistent superior results compared to the highly-tuned, state-of-the-art systems in all visual classification tasks over various datasets.

Although deep features have demonstrated great applicability and achieved significant performance improvements over state-of-the-art in several computer vision tasks, and hold great potential for achieving improved motion detection performance, the ability to leverage them for near real-time performance on embedded systems have been all but impossible so far without requiring the integration of custom GPUs or specialized processors designed for accelerating DNNs. Not only are the vast majority of surveillance cameras not equipped with GPUs or specialized deep processors, their embedded CPU capabilities are also far inferior to most modern computers and thus further prohibiting the use of existing DNN architectures for real-time embedded motion detection. As such, alternative approaches to leveraging DNNs for improved near real-time, embedded motion detection is highly desired.

The main contribution of this paper is a novel approach to motion detection that leverages the power of DNNs while maintaining low computational complexity necessary for near real-time embedded performance without the need for specialized hardware. In the proposed **Ne**ural **R**esponse **M**ixture (NeRM) model, rich deep features are extracted from the neural responses of a highly efficient deep neural network called a StochasticNet [17], where the synaptic connectivity of such networks is sparsely formed in a stochastic manner. Such StochasticNets have been shown to achieve the same level of modelling accuracy as general DNNs while containing only a small fraction of the synaptic connectivities, thus greatly reducing computational complexity. These deep features, which are obtained from StochasticNets pre-trained on large natural image datasets, are then used to construct Gaussian mixture models in an unsupervised manner to model the background based on past frames in the sequence which is being updated on-line. Given its low computational complexity compared to existing DNN approaches, NeRM was implemented on an embedded system on an Axis surveillance camera to demonstrate that strong motion detection accuracy can be achieved while operating at near real-time performance.

The paper is organized as follows. The theory and design considerations behind NeRM, along with implementation details on the embedded system are discussed in Section 2. Experimental results where we examine the proposed NeRM framework on very difficult video datasets for motion detection are reported and discussed in Section 3. Finally, conclusions are drawn in Section 4

2. Methodology

The methodology of the proposed NeRM framework for motion detection is described in detail as follows. First, the problem of motion detection via background modelling is described. Second, the motivation behind the proposed **Ne**ural **R**esponse **M**ixture (NeRM) model is presented. Third, the probabilistic framework for forming the StochasticNets used in NeRM is explained. Fourth, the implementation details of NeRM on an embedded system of an Axis camera is discussed in detail.

2.1. Motion Detection via Background Modelling

A feasible approach for motion detection on an embedded system is to evaluate each pixel or region by the background model and assigning ones that do not conform with the model as pixels under motion. In other words, motion is detected in the scene by evaluating their likelihood of belonging to the background based on the background model. A common approach to this background modelling for motion detection on an embedded system is the use of a Gaussian mixture model (GMM), which can be described as follows. At time t, pixel $\bar{x}_i^t \in X^t$ (where $X^t = \{\bar{x}_1^t, \cdots, \bar{x}_n^t\}$) of frame t, is classified as background if the probability of being background is larger than 0.5. The Gaussian mixture model is formulated as

$$P(\bar{x}_i^t = bg) = \prod_{m=1}^{M} \omega_{i,m}^t \cdot \mathcal{N}_m(\bar{x}_i^t; \bar{\mu}_{i,m}^t, \bar{\sigma}_{i,m}^{t^2}) \quad (1)$$

where the GMM model contains M normal distributions with mean $\bar{\mu}_m^t$ and standard deviation $\bar{\sigma}_m^t$ at time t, and $\omega_{i,m}^t$ encodes the weight of the normal distribution m in GMM model at time t of pixel i. The probability of being

background ("bg") is evaluated via (1). At each frame, the normal distributions $\mathcal{N}_m(\bar{x}_i^t; \bar{\mu}_m^t, \bar{\sigma}_m^{t^2})$ are updated based on which mixture the pixels was assigned to:

$$\omega_{i,m}^{t+1} = \omega_{i,m}^t + \alpha \cdot (1 - \omega_{i,m}^t) \tag{2}$$

$$\bar{\mu}_{i,m}^{t+1} = \bar{\mu}_{i,m}^t + (\frac{\alpha}{\omega_{i,m}^t}) \cdot d_i^t \tag{3}$$

$$\bar{\sigma}_{i,m}^{t+1^2} = \bar{\sigma}_{i,m}^{t^2} + (\frac{\alpha}{\omega_{i,m}^t}) \cdot (d_{i,m}^t - \bar{\sigma}_{i,m}^{t^2}) \tag{4}$$

$$d_{i,m}^t = \frac{(\bar{x}_i^t - \bar{\mu}_{i,m}^t)^2}{\bar{\sigma}_{i,m}^{t^2}} \tag{5}$$

where $d_{i,m}^t$ represents the distance between the new sample (pixel) $\bar{x}_i^t$ and the m^{th}normal distribution and α encodes the learning rate. It is worth noting that $\bar{x}_i^t$ can be pixel intensity or a set of features extracted from pixel i at frame t. $\bar{x}_i^t$ is commonly modelled with the RGB (Red, Green and Blue) pixel intensities in embedded systems because there are no additional feature computation costs. However, while computationally efficient, pixel intensity is highly sensitive to illumination changes, subtle motion in the background, and camera sensor noises. Texture features have been explored to mitigate some of the issues when dealing with the aforementioned factors, but their generic, hand-crafted nature limits their ability to comprehensively capture the unique traits of objects (e.g., people, vehicles, animals, etc.) in real-world surveillance environments, thus limiting their robustness.

Motivated to leverage the recent advances in DNNs while maintaining low computational complexity needed for near real-time embedded performance without specialized hardware, we propose a novel method for motion detection via background modelling based on rich deep features obtained from the neural responses of a highly efficient, stochastically-formed deep neural network known as StochasticNets [17]. Such deep features are highly discriminative and facilitate for a powerful mechanism to model the background while facilitating for low computational complexity, which is the key for near real-time embedded performance. An overview of the proposed **Ne**ural **R**esponse **M**ixture (NeRM) framework for motion detection is shown in Figure 1. First, the neural responses of a StochasticNet pre-trained on a large natural image dataset (e.g., ImageNet [4]) are extracted from the input video frame. A Gaussian mixture model (GMM) is then constructed based on the extracted neural responses to act as the background model of the scene. Finally, motion is detected in the scene by evaluating their likelihood of belonging to the background based on this constructed neural response GMM model.

2.2. Neural Response Mixture Model

The first step to the NeRM framework is to extract rich deep features with which to build a reliable GMM model of the background that can capture the unique traits of objects (e.g., people, vehicles, animals, trees, etc.) in real-world surveillance environments. To train a DNN for the task of video motion detection requires a large training set under different scenarios such as lighting changes, weather conditions, camera jitter, etc. with full manual annotation indicating the pixels in motion (corresponding to objects like people, vehicles, etc.). Obtaining such a large manually annotated dataset is highly difficult. However, DNNs can be trained on an extensive image dataset, such as ImageNet, with millions of images for object classification, with hundreds of object categories. It is worth noting that this dataset (i.e., ImageNet) is not related to motion detection problem. ImageNet [4] is a large scale ontology of images built upon the backbone of the WordNet structure which is mostly utilized for image classification tasks. This can allow the neural responses of the DNN to provide powerful deep features that can better characterize the unique traits of objects, many of which are present in real-world surveillance environments. Considering that true motion in videos is caused by moving objects of interest such as those found in the aforementioned image datasets, we are motivated to leverage the neural responses of DNNs trained in this manner to provide rich features for GMM modelling of the background.

Specifically, we take the first synaptic layer of a highly efficient StochasticNet trained on the ImageNet dataset as a primitive, low-level, feature representation that can isolate important features required for object classification. Therefore, the neural responses of the first synaptic layer at all pixels in the frame can be used as a feature to distinguish motion caused by objects moving in the scene. It is worth noting that the formation of StochasticNets used in the NeRM framework is a one-time and off-line procedure which is not implemented on an embedded system. The final formed StochasticNet is transferred to the embedded system after as described in Section 2.2.2.

2.2.1 StochasticNet Formation

In order for rich deep features to be obtained for GMM modeling of the background, one must first form a StochasticNet pre-trained on ImageNet. In this work, we leverage knowledge gained from traditional deep convolutional neural networks designed for image classification to form highly-efficient StochasticNets that are adapted for background modelling and optimized for minimal synaptic connectivity for near real-time embedded performance.

The proposed StochasticNet formation process can be described as follows. First, we pre-trained a deep convo-

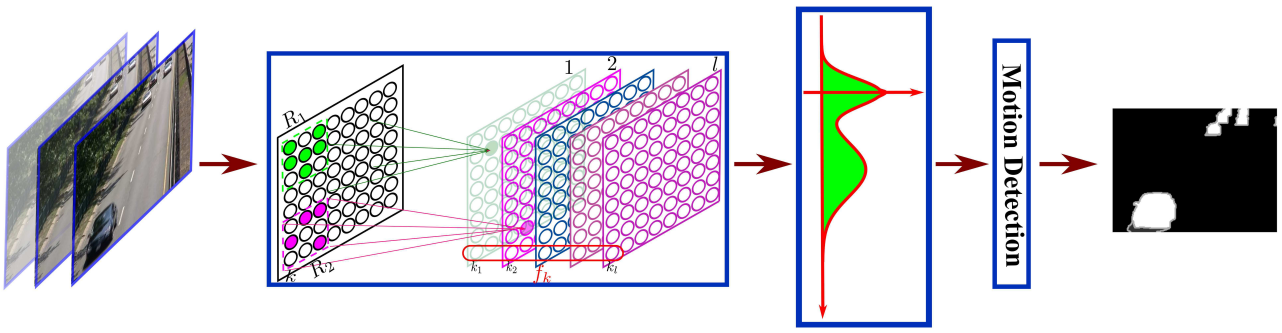

Figure 1. Motion detection based on the NeRM framework. The neural responses from a highly efficient StochasticNet are used as rich deep features. These deep features are then used to construct the Gaussian mixture model to model the background. Finally, motion is detected in the scene by evaluating their likelihood of belonging to the background based on this constructed neural response GMM model.

lutional neural network on the Imagenet dataset for the task of image classification. A StochasticNet is then formed by stochastically selecting a very small set of synaptic connections from this pre-trained deep convolutional neural network. Selection is based on an energy function guided by a smaller annotated dataset with ground truth motion detection labels, and adapting them as synaptic connections in the StochasticNet geared for the task of background modelling.

The goal here is to have the best representative deep features to model the background in a video scene based on a Gaussian mixture model, while limiting the number of synaptic connections in the formed StochasticNet to enable near real-time embedded performance. Here, we define an energy function to minimize false detections while minimizing the number of synaptic connections in the formed StochasticNet:

$$E_l = \frac{1}{S}\sum_{i=1}^{T}\sum_{j=1}^{N}\delta(\hat{b}_{ij}^{l} \neq b_{ij}) \tag{6}$$

where S encodes the number of synaptic connections in the StochasticNet, b_{ij} is the ground truth label for pixel j at frame i and $\hat{b}_{ij}^{l}$ encodes the estimated label (i.e., in motion or not in motion classification from the GMM) for pixel j at frame i via the extracted neural response features at iteration l. T represents the total number of frames in the training video which the number of pixels in each frame is represented by N and $\delta(\cdot)$ is Dirac function.

The off-line, stochastic formation of the StochasticNet used for deep features is an iterative procedure such that in each iteration new synaptic connectivities, stochastically selected from the deep convolutional neural network, are included to the network using a stochastic acceptance-rejection criteria based on the energy function gradient (ΔE) between consecutive iterations (see Algorithm 1). Line 9 in Algorithm 1 ($\exp(-\frac{\Delta E}{T}) \geq U(0,1)$) provides more chance to form a synaptic connection if the connection does not decrease the energy function $E(\cdot)$, where T is the controlling parameter to adjust the acceptance probability. Experimental results showed that we can create StochasticNets with up to 95% fewer synaptic connections than a conventional deep convolutional neural network while maintaining good modelling performance. Algorithm 1 demonstrates the stochastic formation procedure to form StochasticNets for NeRM.

For the implementation of NeRM used in experiments, a deep convolutional neural network based on the AlexNet [11] network architecture, trained on ImageNet, is utilized to form a StochasticNet with just 5% of synaptic connectivities compared to the AlexNet network architecture. A set of 200 frames of highway video dataset [7] is used as the small annotated dataset for the formation procedure. A small dataset is selected to make the formation process fast; however, utilizing more datasets with varying conditions can lead to a more efficient and effective network architecture. The StochasticNet formation in this study is implemented via MatConvNet framework [20].

2.2.2 Embedded System Implementation Details

There are many camera manufacturers with open platforms for developing embedded applications; however, Axis cameras and their 3rd party development platform is among the most popular and mature platforms currently available in the surveillance industry. Their latest cameras are available with Axis ARTPEC-5 chip-set (MIPS 1004Kc V2.12 CPU model) running a striped down version of Linux. While our implementation will work on any Axis camera that supports embedded development and has the ARTPEC-5 chip-set,

Algorithm 1 Stochastic Formation

procedure SF
  $\mathcal{X}$:= *Video Stream*; $\mathcal{B}$:= *Ground truth*;
  R_{NeRM} := *Sparse Set of Synaptic Connections* ;
  R_{NeRM} = Null
  while is-not-equal(E^l,E^{l+1}) **do**
    E^l = ComputeEnergy ($R^l_{NeRM}, \mathcal{X}, \mathcal{B}$)
    R^{l+1}_{NeRD} = add(R^l_{NeRM}, i) ▷ add new synaptic connectivity i to the set of synaptic connections R^l_{NeRM}
    E^{l+1} = ComputeEnergy ($R^{l+1}_{NeRM}, \mathcal{X}, \mathcal{B}$) ▷ Compute the energy of R^{l+1}_{NeRM}
    if $\exp(-\frac{\Delta E}{T}) \geq U(0,1)$ **then**
      keep new R^{l+1}_{NeRM}
    else
      $R^{l+1}_{NeRM} = R^l_{NeRM}$

we test our algorithms on the Axis Q7436 Encoder[1].

Our approach is implemented using C++ and compiled with the Axis development SDK. The algorithm has two main parts: StochasticNet (3 layers: convolutional, ReLU) and Gaussian mixture model (GMM) modelling using the deep features. Most deep neural networks employ floating point computations for the convolutional layer. While most desktop computers have a floating point unit (FPU) to handle floating point operations, a majority of surveillance cameras do not have a dedicated FPU. As a result, floating point computations are significantly slower. To overcome this issue, we form StochasticNets on servers using floating points computations and port the networks to the camera using a 32 bit fixed point representation with a dedicated 16 bits for the decimal components. To reduce the computational complexity of the GMM modelling we employ only two modes and the same fixed point representation as for the convolutional layer.

3. Result & Discussion

The proposed NeRM framework is evaluated with the CD.Net datasets [7] and compared with two other methods: I) Gaussian mixture model based on RGB pixel intensity (RGB), and II) Gaussian mixture model based on contrast histograms (CHist) [3]. CHist extracts contrast histogram texture features from the image based on blocks of 8×8. For a fair comparison, all methods uses the same two-mode GMM.

3.1. Dataset

The methods are evaluated comprehensively via CD.Net dataset [7]. CD.Net is one of the largest datasets with a variety of challenging scenarios such as bad weather conditions, night vision, dynamic backgrounds, shadows and thermal cameras. The dataset contains more than 90,000 manually labelled ground truth frames. The performance of the methods are compared with several quantitative measures including recall (Re), specificity (Sp), false positive rate (FPR), false negative rate (FNR), percentage of wrong classifications (PWC), precision (Pr) and F-Measure (FM) as defined in [7].

[1] http://www.axis.com/ca/en/products/axis-q7436

3.2. Results

Tables 1 and 2 show the quantitative results of the compared methods based on all evaluated measures. As seen the proposed method outperforms other approaches in almost all quantitative measures. The F-Measure (FM) is a balance between recall and precision and is used as a summary metric in other works [22]. Based on the overall FM, NeRM has a 5% boost over RGB and CHist features.

Figure 2 and 3 demonstrates the qualitative results of competing algorithms. It must be noted that the background modeling for RGB framework has been done in 352×240 image resolution while the resolution of the frame is reduced since 8×8 non-overlapping texture extraction blocks are used when extracting features in CHist (i.e., due to the high computational complexity of overlapping approach), and NeRM framework convolves the receptive fields with stride 4 which resulted in a blockiness effect in the estimated motion areas for CHist and NeRM methods. As seen, the proposed NeRM framework can handle different conditions and situations to detect motion in videos. Figure 2 shows the comparison of methods in three situations: I) bad weather condition: snow is one of the main issue in the bad weather conditions, first row and last rows demonstrate the snow condition (moving cars or skating persons), II) Thermal: videos are captured with thermal camera and the pixel intensities are different compared to RGB domain in modelling the background (i.e, rows 2), and III) Camera jitter: in this situation camera have slight motion with high frequency while capturing the scene. Results in Figure 2 show that the proposed NeRM framework can detect all motion while producing much less false alarms compared to RGB or CHist approaches.

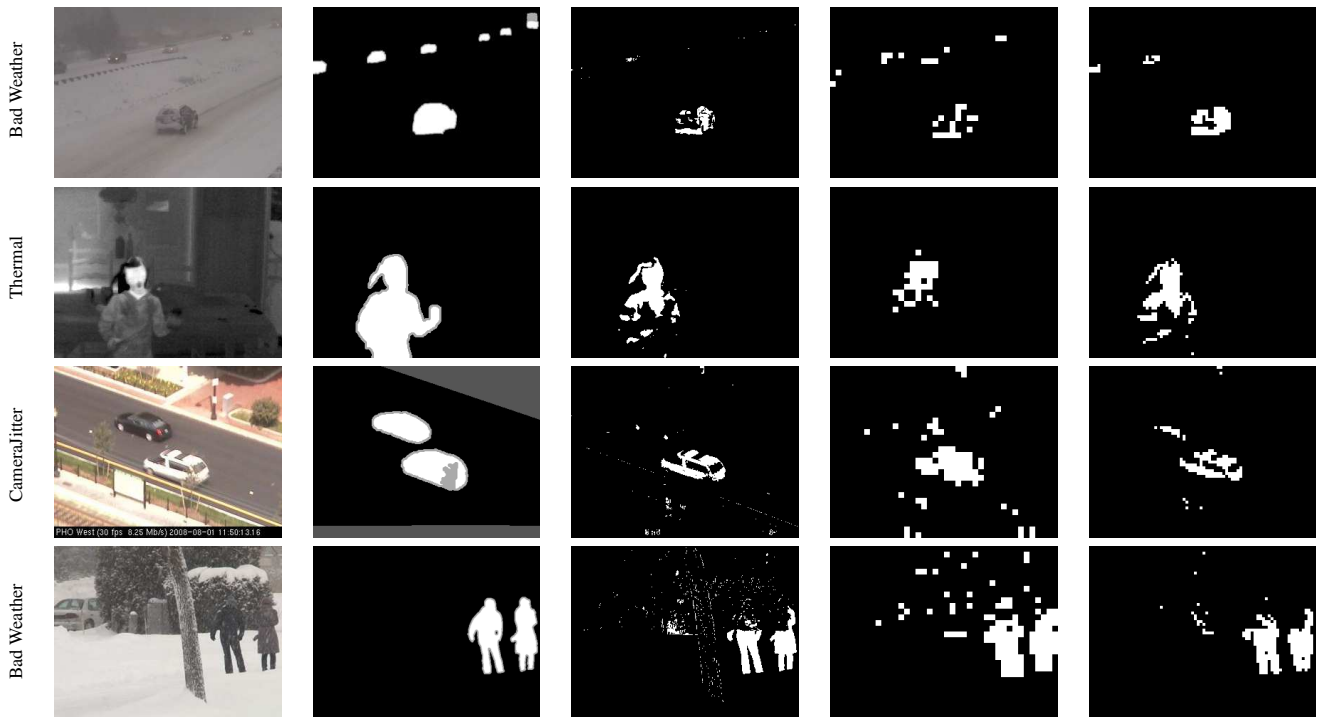

Figure 2. The competing methods are compared with videos captured in bad weather conditions, when the camera has some slight motion and with thermal cameras. Results show that the proposed method outperforms other approaches in these conditions. The blockiness artifact in CHist and NeRM results is due to the shrinking procedure in feature extraction step.

Figure 3 demonstrates the results of competing methods in more complex situations: I) Shadow: in this situation, several regions in the scene are distorted by shadow of other objects, II) Dynamic background: this category examines the method when there are some objects in the background which have motion; III) Low Frame Rate: several surveillance cameras capture the scene with small number of frame per second due to the storage and computation, this category tests performance of competing methods for these situations; IV) Night Videos: detection of motion in both daytimes and low light nighttimes is important. The reported results in Figure 3 support the effectiveness of the proposed NeRM framework compared to the RGB and CHist approaches. Overall results show that NeRM algorithm detects motion with less noise while producing less false alarms. It also demonstrates that the number of missed motion areas by the proposed method is fewer than competing algorithm.

3.3. Running-Time

To validate the efficiency of the proposed framework, NeRM approach was implemented on Axis Q7436 (ARTPEC-5 chip-set) Encoder. The experimental results showed that it took about 470 ms to process a 352×240 video frames. This results in processing speed of about 2 FPS, which is still sufficient frame rate to detect motion in order to determine when to record video. The performance of the proposed approach is compared with the second best method, GMM based on RGB features. RGB GMM takes about 150 ms on the same video frame size. However, the overall Fmeasure result of NeRM is 5% higher than RGB implementation.

4. Conclusion

A new approach was proposed to address the computational complexity of deep convolutional networks to make the use of rich deep features feasible on embedded systems. Here we addressed the motion detection problem on embedded systems based on a neural response mixture (NeRM) model. The proposed NeRM method takes advantage of sparse synaptic connectivities and resolves the computational complexity of running a deep neural network on embedded systems while maintaining its performance and accuracy. Experimental results showed that the extracted neural response features in a Gaussian mixture model can perform better than just using RGB pixel intensity or even hand-crafted texture features. This new approach can open a new avenue to facilitate the use of deep neural networks on embedded systems which has huge applicability in dif-

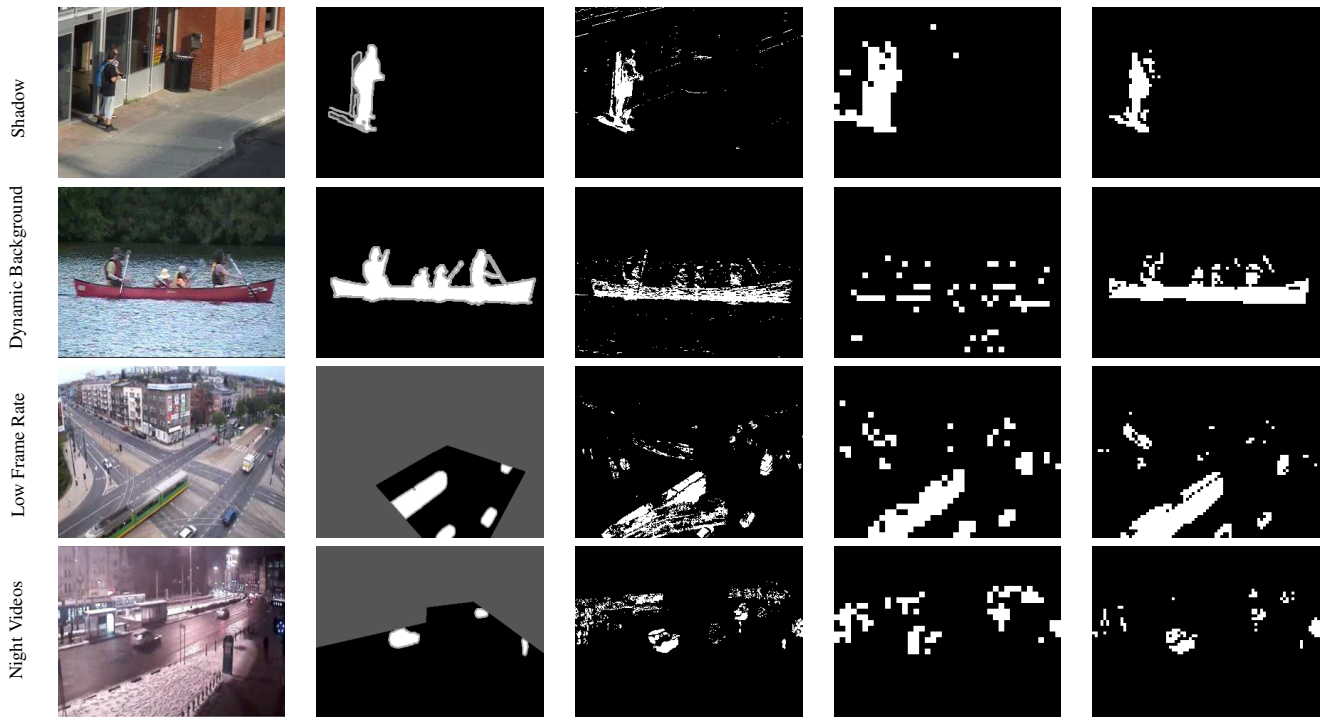

Figure 3. Qualitative results for complex situations. In this Figure, the competing methods are compared with video categories which are considered as difficult conditions to motion detection. The comparison demonstrates that the proposed NeRM approach performs better than other algorithms.

Table 1. Quantitative comparison via several performance measures; results shows that the proposed method outperforms other methods overall in detecting movement. The results are reported based on the best threshold which maximizes the FMeasure per category.

	Specificity (Sp)			Recall (Re)			Precision (Pr)			FMeasure (FM)		
	RGB	CHist	NeRM	RGB	CHist	NeRM	RGB	CHist	NeRM	RGB	CHist	NeRM
badWeathert	**0.99**	0.98	**0.99**	0.59	0.70	**0.72**	0.68	0.54	**0.82**	0.60	0.59	**0.77**
cameraJitter	0.96	0.89	**0.97**	0.40	**0.54**	0.46	0.40	0.21	**0.54**	0.39	0.29	**0.48**
dynamicBackground	**0.98**	0.97	**0.98**	0.40	0.37	**0.60**	0.37	0.20	**0.49**	0.35	0.21	**0.49**
intermittentObjectMotion	0.97	0.95	**0.98**	**0.59**	0.52	0.43	**0.58**	0.42	0.55	**0.53**	0.42	0.44
lowFramerate	0.92	0.94	**0.97**	0.53	**0.66**	0.54	0.26	0.44	**0.50**	0.30	**0.40**	0.39
nightVideos	0.95	**0.97**	0.95	0.51	0.60	**0.61**	0.31	**0.39**	0.29	0.34	**0.45**	0.35
shadow	0.97	0.96	**0.98**	0.70	**0.81**	0.75	0.56	0.55	**0.68**	0.60	0.64	**0.70**
thermal	**0.99**	0.89	0.97	0.67	**0.87**	0.77	**0.81**	0.45	0.60	**0.71**	0.57	0.65
turbulence	0.98	0.98	**0.99**	0.63	0.58	**0.71**	0.57	0.36	**0.63**	0.50	0.37	**0.58**
Overall:	0.96	0.94	**0.97**	0.55	**0.62**	**0.62**	0.50	0.39	**0.56**	0.48	0.43	**0.53**

ferent industrial problems.

References

[1] O. Barnich and M. Van Droogenbroeck. Vibe: A universal background subtraction algorithm for video sequences. *IEEE Transactions on Image Processing*, 20(6), 2011. 1

[2] T. Bouwmans, F. El Baf, and B. Vachon. Background modeling using mixture of gaussians for foreground detection-a survey. *Recent Patents on Computer Science*, 1(3), 2008. 1

[3] Y.-T. Chen, C.-S. Chen, C.-R. Huang, and Y.-P. Hung. Efficient hierarchical method for background subtraction. *Pattern Recognition*, 40(10), 2007. 5

[4] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2009. 3

[5] A. Elgammal, D. Harwood, and L. Davis. Non-parametric model for background subtraction. In *European Conference on Computer Vision (ECCV)*. Springer, 2000. 1

[6] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *conference on computer vision and pattern recognition (CVPR)*. IEEE, 2014. 2

[7] N. Goyette, P.-M. Jodoin, F. Porikli, J. Konrad, and P. Ishwar. Changedetection. net: A new change detection bench-

Table 2. Quantitative comparison of the competing methods based on three error measures: I) FNR (False Negative Rate), II) PWC (Percentage of Wrong Classifications) and III) FPR (False Positive Rate).

	FNR			PWC			FPR		
	RGB	CHist	NeRM	RGB	CHist	NeRM	RGB	CHist	NeRM
badWeathert	0.49	0.29	**0.27**	1.06	1.79	0.69	**0.00**	0.01	**0.00**
cameraJitter	**0.39**	0.45	0.53	9.50	12.29	**4.40**	0.08	0.10	**0.02**
dynamicBackground	**0.32**	0.62	0.39	6.12	3.39	**1.56**	0.05	0.02	**0.01**
intermittentObjectMotion	0.50	**0.47**	0.56	**5.23**	7.48	5.79	**0.01**	0.04	**0.01**
lowFramerate	0.56	**0.33**	0.45	6.50	5.93	**3.61**	0.04	0.05	**0.02**
nightVideos	0.59	0.39	**0.38**	3.96	**2.91**	5.33	**0.02**	**0.02**	0.04
shadow	0.32	**0.18**	0.24	3.40	4.08	**2.62**	0.02	0.03	**0.01**
thermal	0.56	**0.12**	0.22	**4.17**	10.85	4.73	**0.02**	0.11	**0.02**
turbulence	**0.19**	0.41	0.28	3.65	1.39	**1.06**	0.03	0.08	**0.00**
Overall:	0.43	**0.36**	**0.36**	4.84	5.56	**3.31**	0.03	0.05	**0.01**

mark dataset. In *Computer Vision and Pattern Recognition Workshops (CVPRW)*. IEEE, 2012. 4, 5

[8] S. Gupta, R. Girshick, P. Arbeláez, and J. Malik. Learning rich features from rgb-d images for object detection and segmentation. In *European Conference on Computer Vision (ECCV)*. Springer, 2014. 2

[9] R. Jenifa, C. Akila, and V. Kavitha. Rapid background subtraction from video sequences. In *International Conference on Computing, Electronics and Electrical Technologies (ICCEET)*. IEEE, 2012. 1

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems (NIPS)*, 2012. 2

[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems (NIPS)*, 2012. 4

[12] M. Mason and Z. Duric. Using histograms to detect and track objects in color video. In *Applied Imagery Pattern Recognition Workshop, AIPR 2001 30th*. IEEE, 2001. 1

[13] T. Matsuyama, T. Ohya, and H. Habe. Background subtraction for non-stationary scenes,. 1999. *Department of Electronics and Communications, Graduate School of Engineering, Kyoto University: Sakyo, Kyoto, Japan*. 1

[14] T. Ojala, M. Pietikäinen, and T. Mäenpää. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 24(7), 2002. 2

[15] N. M. Oliver, B. Rosario, and A. P. Pentland. A bayesian computer vision system for modeling human interactions. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 22(8), 2000. 1

[16] A. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In *Conference on Computer Vision and Pattern Recognition Workshops (CVPR)*. IEEE, 2014. 2

[17] M. J. Shafiee, P. Siva, and A. Wong. Stochasticnet: Forming deep neural networks via stochastic connectivity. *IEEE Access*, (99), 2016. 2, 3

[18] P. Siva, M. J. Shafiee, F. Li, and A. Wong. Pirm: Fast background subtraction under sudden, local illumination changes via probabilistic illumination range modelling. In *International Conference on Image Processing (ICIP)*. IEEE, 2015. 1

[19] C. Stauffer and W. E. L. Grimson. Adaptive background mixture models for real-time tracking. In *Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2. IEEE, 1999. 1

[20] A. Vedaldi and K. Lenc. Matconvnet – convolutional neural networks for matlab. 4

[21] C. R. Wren, A. Azarbayejani, T. Darrell, and A. P. Pentland. Pfinder: Real-time tracking of the human body. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 19(7), 1997. 1

[22] Z. Zivkovic. Improved adaptive gaussian mixture model for background subtraction. In *International Conference on Pattern Recognition (ICPR)*, volume 2. IEEE, 2004. 5