

Real-Time Change Detection with Convolutional Density Approximation

Synh Viet-Uyen Ha ^{**†}, Tien-Cuong Nguyen ^{**†}, Hung Ngoc Phan ^{**†}
and Phuong Hoai Ha ^{‡§}

**International University, Vietnam*

†Vietnam National University, Ho Chi Minh City, Vietnam

‡UiT The Arctic University of Norway, Norway

§phuong.hoi.ha@uit.no

Received 15 August 2023

Revised 9 October 2023

Accepted 9 October 2023

Published 2 April 2024

Background Subtraction (BgS) is a widely researched technique to develop online Change Detection algorithms for static video cameras. Many BgS methods have employed the unsupervised, adaptive approach of Gaussian Mixture Model (GMM) to produce decent backgrounds, but they lack proper consideration of scene semantics to produce better foregrounds. On the other hand, with considerable computational expenses, BgS with Deep Neural Networks (DNN) is able to produce accurate background and foreground segments. In our research, we blend both approaches for the best. First, we formulated a network called Convolutional Density Approximation (CDA) for direct density estimation of background models. Then, we propose a self-supervised training strategy for CDA to adaptively capture high-frequency color distributions for the corresponding backgrounds. Finally, we show that background models can indeed assist foreground extraction by an efficient Neural Motion Subtraction (NeMos) network. Our experiments verify competitive results in the balance between effectiveness and efficiency.

Keywords: Neural network; representation learning; motion estimation; background subtraction; change detection.

1. Introduction

Change Detection is a fundamental semantic segmentation task that handles the identification of changing or moving areas in the field of view of a camera. With the

^{**†}In relation to the authors' contract, * also represents "1. International University, Vietnam", † also represents "2. Vietnam National University, Ho Chi Minh City, Vietnam", ‡ also represents "3. UiT The Arctic University of Norway, Norway".

§Corresponding author.

This is an Open Access article published by World Scientific Publishing Company. It is distributed under the terms of the Creative Commons Attribution 4.0 (CC BY) License which permits use, distribution and reproduction in any medium, provided the original work is properly cited.

swift progress in computer vision, practical utilization of it in visual systems has involved advanced tasks such as behavior analysis,^{1,2} instance segmentation³ and traffic analysis.⁴ While there are various online and offline algorithms that have been proposed, online algorithms are arguably much more favorable as they can make predictions on demand for large-scale inputs of essentially all tasks. By making continuous predictions on demand, and even detecting and dealing with out-of-distribution signals, online Change Detection algorithms have been pivotal for proper, timely understanding of scene dynamics and extraction of interesting attributes in many systems.

One popular approach is Change Detection via background modeling and Background Subtraction (BgS). The background modeling step aims to construct ideal backgrounds, which are scene captures containing only stationary objects and features (e.g. streets, houses) uninteresting to the systems analytic purposes. Then, by comparing visual inputs from a video sequence with their backgrounds, the BgS technique can localize all desired targets (i.e. so-called foregrounds like cars, pedestrians, etc.) for analysis. Despite having been challenged by a plethora of real-life scenarios such as shadows, illumination changes, dynamic background, among others,⁵ background modeling and BgS remain prominent topics of research toward being applied to a wide range of applications including video surveillance, smart environments, and content retrieval.

Prominent online approaches⁶ include pixel-based statistical frameworks such as the Gaussian Mixture Models (GMMs).⁷⁻¹⁰ These GMM frameworks are based on the hypothesis that background intensities can be observed most frequently in a video sequence recorded from a still camera, thereby creating explicit mathematical structures for simple inferences. Additionally, this design notably entails general applicability under illumination changes (e.g. from moving clouds), view noises (e.g. rain, snowflakes), and implicit motions (e.g. river water). However, they struggle to perform effectively when the background intensity hypothesis fails (i.e. stopped objects or visual noises in the scene presenting prolonged intensity values in certain regions), resulting in corrupted backgrounds and inaccurate foreground estimations. On the whole, it is thanks to their simplicity and efficiency on CPUs that make GMM-driven approaches very appealing, but their lack of not only explicit parallel-computing design with GPUs, but also limited consideration for scene semantics for better foregrounds have made them much less relevant in modern research, especially concerning big data and deep learning (DL).

On the other hand, ever-advancing processing units specialized for large-scale data are making Deep Neural Networks (DNNs) not only powerful, but also tractable. However, in addition to a reliance on labels for practical utilization in real scenarios, existing architectures have inevitable trade-offs between computational efficiency and high accuracy, among which also include DL architectures for background modeling and Change Detection. With respect to labeled data for Change Detection in particular, authors in Ref. 11 saw a clear lack of labels for training general motion detectors, and there is currently no universal dataset that can ensure

all possible scenes' true properties are appropriately presented. These findings have obviously presented many challenges, but it has also motivated research into designing DNNs that can achieve high accuracy, while utilizing as little labeled data as possible.^{12,13} Nevertheless, because domain generalization¹⁴ is a complex, still unsolved problem in research to overcome data biases, especially with regards to a semantic segmentation task like Change Detection, learned models can always be susceptible to unseen contextual variations that may occur in the real world.^{15,16}

Through highly parallelizable neural architectures, the literature on DNNs has shown that they can approximate any functions up to any arbitrary accuracies. This signifies that we can efficiently utilize their parallelism to not only approximate the mechanism behind the optimization of GMM for background modeling, but we can also facilitate a more efficient data-driven foreground extractor that uses few labels. Unfortunately, little research has focused on striking a balance between effectiveness and efficiency for real-time, scalable, and reliable processing.

Hence, in this paper, we propose a real-time, highly effective network design for BgS that uses few learning labels. Our novel approach essentially reserves most of DNNs' benefits, addresses the sequentialism of GMM-based background modeling, and shows that backgrounds can be used to streamline foreground extraction processes at high accuracies. Essentially, we develop a dual framework of BgS consisting of two modules in this paper: (1) Background Modeling by Convolutional Density Approximation (CDA) for direct density estimation of background distributions; and (2) Foreground Extraction by Neural Motion Subtraction (NeMos) that estimates changed regions based on contextual constraints. Our contributions are summarized as follows:

- First, inspired by the existing computing technologies and Bishop,¹⁷ we present our formulation of a GMM-based background solver via CDA. It is a feed-forward, 2800-parameter parallelizable Convolutional Neural Network (CNN) that simulates a posterior probability function conditioned on the temporal history at each pixel location. The architecture is lightweight, compressed, and efficient by addressing the conventional sequentialism of GMM-based background models, and it performs as an effective codebook for mapping arrays of pixel values to the corresponding GMM functions.
- Second, to technically model the underlying generator of input data, we propose a self-supervised learning strategy based on unsupervised learning and data augmentation. In particular, the strategy includes an unsupervised objective function that guides CDA to approximate the parameters of GMMs via expectation maximization, and teaches it to behave as a permutation invariant network. The proposed background modeling architecture not only achieves high degrees of mathematical interpretability, but also possesses adaptation to contextual dynamics with the neural statistical analysis. Furthermore, with self-supervision, the framework can be pre-trained with an inexhaustible amount of data.
- Third, we propose to use a context-driven, 700-parameter neural foreground extraction component called NeMos, on top of background models, for effectively

and efficiently segmenting the difference mapping between input frames and their corresponding background estimations. This is motivated not only by our construction of GMM-driven background models with CDA to provide for summarized semantic understanding of a context-rich scene, but also by addressing the prohibitive expenses of existing segmentation networks for Change Detection. The network can properly maintain generalization across a scenario's dynamics in real time.

The organization of this paper is as follows. Section 2 encapsulates the synthesis of recent approaches in background initialization and foreground segmentation. The proposed method is described in Sec. 3. Experimental evaluations are discussed in Sec. 4. Finally, our conclusion and motivations toward future works are discussed in Sec. 5.

2. Related Works

The new era of video analysis has witnessed a proliferation of methods that concentrate on Change Detection. In fact, studies in recent decades have been encapsulated in various conceptual and experimental perspectives.^{6,15,18} The literature has specifically remarked on both unsupervised and supervised learning, particularly on the two most prominent concepts used in BgS or foreground detection: statistics-based approaches that are unsupervised and supervised DNNs. This work has been extended from our preprint¹⁹ to further investigate self-supervision and experimental results.

2.1. Statistics

Statistics-driven methods have been widely studied in terms of both research and practical applications due to simplicity, lightweightness, and online adaptation to scene dynamics without label training. Deployed methods in the practice of this category are usually sample-based (e.g. temporal median,²⁰ histograms,²¹ codebooks²²) or via estimations of the multi-modular probability density function (PDF) (e.g. the GMMs⁷) on data inputs.

Sample-based approaches essentially record the history of observed input pixels by sets of intensity values representing a background model. From a new input value, an algorithm compares the corresponding set to that pixel value to determine whether that pixel belongs to the background, and selectly adapts the model. For example, a codebook algorithm like Ref. 23 records all intensities in YCrCb color space at each pixel, which is done over a period of time through quantization of scene multi-modularity. In a similar way, Ref. 24 estimates visual changes by extracting histogram features, and thresholding their means over their highest probable occurrence probability. Another recently proposed approach employs a weight-sample-based strategy²⁵ that rapidly adapts to changing scenarios by a reward-and-penalty function on samples. Recently, Agrawal and Natu²⁶ presented a two-level adaptive thresholding

algorithm to remove shadow pixels and detect foregrounds. The algorithm is based on the YCbCr color space, and uses the intensity ratio method for improved pixel-wise recognition.

In parallel, popular approaches also aim to construct the PDF of data, where pixels' spatio-temporal visual features are captured in the corresponding probabilistic models at either pixel-level or region-level. In the last decades, scientists have proposed a variety of statistical models to resolve the problem of background modeling and subtraction. Stauffer and Grimson⁷ proposed a pioneering work that handled gradual changes in outdoor scenes using pixel-level GMM with a sequential K -means distribution matching algorithm. To enhance the foreground/background discrimination ability regarding scene dynamics, Pulgarin-Giraldo *et al.*²⁷ improved GMM with a contextual sensitivity that used a Least Mean Squares formulation to update the parameter estimation framework. By validating the robustness of background modeling in a high amount of dynamic scene changes, Ha *et al.*²⁸ proposed a GMM with a high variation removal module using entropy estimation. On the other hand, Zhao *et al.*²⁹ showed that BgS is possible with the integration of alternative cues about foreground and background on freely-moving cameras, where foreground cues can be extracted from the GMM compensated with image alignment, and background cues can be obtained from the spatio-temporal features filtered by the homography transformation. Then, in an effort to address the sequential bottleneck among statistical methods in pixel-wise learning, an unsupervised, tensor-driven framework of GMM was proposed by Ha *et al.*¹⁰ with a balanced trade-off between satisfactory foreground mask and exceptional processing speed. However, the approach's number of parameters requires a lot of manual tuning. Overall, statistical models were developed with explicit probabilistic hypotheses to sequentially present the correlation of history observation at each image point or a pixel block, added with a global thresholding approach to extract foregrounds. This global thresholding technique for foreground detection usually leads to a compromise between the segregation of slow-moving objects and rapid adaptation to sudden scene changes within short-term measurement. This trade-off usually damages the image-BgS in multi-contextual scenarios, which is considered a sensitive concern in motion estimation. Hence, regarding foreground segmentation from background modeling, it is critical to improve frame differencing from constructed background scenes with a better approximation mechanism, and utilize parallel technologies.

Drawn from the published methods, statistical studies essentially aim to characterize the history of pixels' intensities with generalistic background models. The construction of these models is conveniently unsupervised and can be effectively adaptive to the dynamics of their input domains. However, indiscriminate adaptability entails compromises between valuable incorporation of domain contexts and over-adaptations of foreground objects into background models. While addressing these effects has shown promising results, they entail extra computational burdens corresponding to improved accuracy, but still without full consideration of scene properties for segmenting accurate foregrounds. Regarding GMM-based approaches

specifically, the GMM mathematical framework has not only demonstrated strong multi-modular approximations of input statistics where effective background extraction procedures may excel, but it has also shown how highly customizable it can be in the extensive literature to address specific problems. Nevertheless, in terms of computations, there has yet to be a common, explicit computing framework for GMM-based approaches in which pixel-wise processing for high dimensionalities and scales can be accomplished with GPUs.

2.2. Deep neural networks

Unlike statistical frameworks, neural networks can explicitly exploit nonlinear data manipulations on parallel distributed computing paradigms with modern technologies by label training. Their goal is to generalize an equivariant function of foreground segmentation across video sequences where there can be visual changes of varying degrees of complexity, which is by either BgS or direct foreground extraction.

Recently, there have been many attempts to apply DNNs to BgS. Inspired by LeNet-5³⁰ used for handwritten digit recognition, one of the earliest efforts to subtract the background from the input image frame was done by Braham *et al.*³¹ This work explores the potential of visual features learned by hidden layers for foreground-background pixel classification. Similarly, Wang *et al.*³² proposed a deep CNN trained on only a small subset of frames as there is a large redundancy in a video taken by surveillance systems. The model requires a hand-labeled segmentation of moving regions as an indicator in observed scenes. Lim *et al.*³³ constructed an encoder-decoder architecture with the encoder inherited from VGG-16.³⁴ The proposed encoder-decoder network takes a video frame, along its corresponding grayscale background and its previous frame as the network's inputs to compute their latent representations, and to deconvolve these latent features into a foreground binary map. Another method is DeepBS³⁵ which was proposed by Babae *et al.* to compute the background model using both SuBSENSE³⁶ and Flux Tensor method.³⁷ The authors extract the foreground mask from a small patch from the current video frame and its corresponding background to feed into the CNN, and the mask is later post-processed to give the result. Nguyen *et al.* proposed a motion feature network³⁸ to exploit motion patterns via encoding motion features from small samples of images. The method's experimental results showed that the network obtained promising results and was well-performed on several unseen data sequences.

Regarding direct foreground extraction, these models essentially construct implicit backgrounds within the hidden states, and cluster pixel regions by recognizing semantic classes of interest in the training set. An excellent proposed approach is the scene-specific FgSegNet series of encoder-decoder architectures^{12,13} proposed by Long *et al.* FgSegNet is one of the top-performing approaches in Change Detection that is built on top of VGG-16 of convolutional layers. There is also a published work from Chen *et al.*³⁹ which aims to exploit high-level spatial-temporal features

with a deep pixel-wise attention mechanism and convolutional long short-term memory (ConvLSTM). Chen *et al.* introduced a pixel-wise deep sequence learning architecture with attention mechanism and ConvLSTM to Change Detection. On the other hand, Yang *et al.*⁴⁰ proposed an end-to-end multi-scale spatiotemporal propagation network to detect motions. Instead of using ConvLSTM or 3D convolutions, they developed a feature aggregation block to fuse motion features of various scales. Similarly, to take into account multi-scale features, Houhou *et al.*⁴¹ presented a deep multi-scale network for BgS, which fuse both the RGB color channels and depth maps to perform spatio-semantic BgS at various scales. Recently, Gouizi and Megherbi⁴² extended the U-net architecture with more skip connections on residual micro-autoencoder blocks. The approach is called Nested-Net, which produces high accuracy at the expense of significant computational costs over U-net and many skip connections.

All things considered, neural-network-based methods significantly benefit from learning a transformation from an input batch of consecutive frames to manually labeled foregrounds of visual changes. From training with selected samples, these approaches are able to accurately generalize to varying degrees of contextual dynamics within a scene, essentially by constructing a numerical understanding of foreground extraction within a network parameters. However, recent DNNs-based methods do not ensure real-time performance, which is a crucial requirement for practical systems that need on-the-fly predictions. Despite the fact that DNNs can utilize the parallel-computing mechanisms of modern hardware very well, and can also make use of data for high-accuracy prediction, hardly any work has been done to investigate a proper balance between effectiveness and efficiency for DNNs-based Change Detection models.

Therefore, inspired by how statistics-based models are very popular with application scientists,⁶ we advocate real-time processing and high accuracy to account for the convenience, scalability, and functionality of deploying DNNs in practical scenarios.

3. Methodology

In our work, we propose a framework consisting of two CNNs, as shown in Fig. 1. First, grounded on a generalized GMM model like Ref. 7, the first network models posterior PDFs conditioned on records of temporal information to construct background scenes. The vanilla form of GMM on background modeling is very simple for neural networks to approximate, as highly frequent intensity values are skewed toward background values. Then, the second component is developed to perform deep BgS across thresholds of differences, in which we show that a CNN-based encoder–decoder not only can be used in estimating frame-to-background differences like Ref. 35, but by leveraging generalized backgrounds, it can also make accurate predictions efficiently.

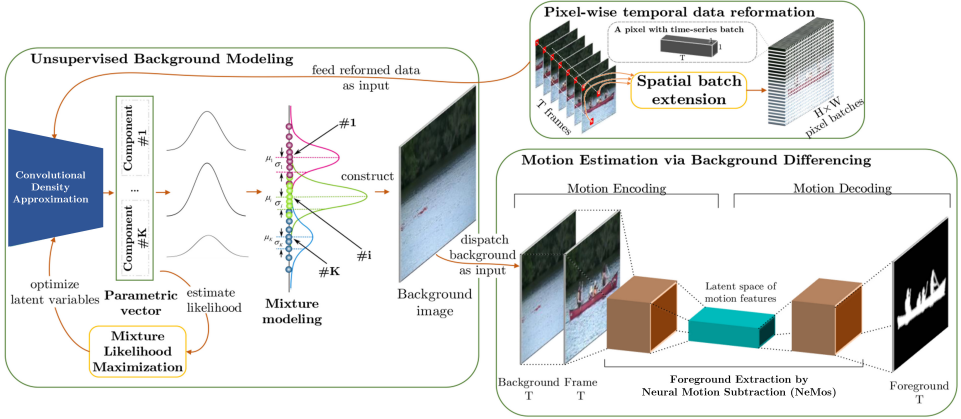


Fig. 1. The overview of the proposed method for background modeling and foreground detection.

3.1. Convolutional density approximation of gaussians

In this section, we first propose to formulate the GMM problem under DNNs' perspectives. Following Zivkovic,⁸ let $\mathcal{X}_c^T = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T | \mathbf{x}_i \in [0, 255]^c\}$ be the set of T observed color signals at a pixel position, where c is the number of dimensions in the color space, the distribution of pixel intensity \mathbf{x}_i can be modeled by a linear combination of K probabilistic components θ_k and their corresponding posterior functions $P(\mathbf{x}_i | \theta_k)$. The marginal probability $P(\mathbf{x}_i)$ of the mixture is defined in the following equation:

$$P(\mathbf{x}) = \sum_{k=1}^K P(\theta_k) \cdot P(\mathbf{x} | \theta_k) = \sum_{k=1}^K \pi_k \cdot \mathcal{N}(\mathbf{x} | \boldsymbol{\mu}_k, \sigma_k), \quad (1)$$

where $P(\theta_k) = \pi_k$ is the non-negative mixing coefficient that sums to unity over all k 's, representing the likelihood of occurrence of the k th Gaussian distribution θ_k .

In practice, real-life recorded scenes have often presented various degrees of changing context dynamics (e.g. body of water, waving trees, changing weather, illumination, etc.). Obviously, while also taking into account acquisition noises, a single Gaussian would not be sufficient to model the pixel's values. This multi-modality ought to be captured by a mixture of adaptive Gaussians. To also avoid performing costly matrix inversion,⁷ each color channel in the color space is assumed to be distributed independently, thus each Gaussian component in the mixture is described with a scalar variance σ_k .

$$\begin{aligned} P(\mathbf{x} | \theta_k) &= \mathcal{N}(\mathbf{x} | \boldsymbol{\mu}_k, \sigma_k) \\ &= \frac{1}{\sqrt{(2\pi)^c \sigma_k^c}} \exp\left(-\frac{\|\mathbf{x} - \boldsymbol{\mu}_k\|^2}{2\sigma_k}\right), \end{aligned} \quad (2)$$

where $\boldsymbol{\mu}_k$ is the estimated mean and σ_k is the estimated universal covariance of color channels in the k th Gaussian component.

From this hypothesis, we propose an architecture called Convolutional Density Approximation (CDA), which employs a set of nonlinear transformations $f_{\theta}(\cdot)$ to formulate a conditional GMM-based density function of \mathbf{x} given a set of randomly selected, vectorized data points χ_T :

$$\mathbf{y}_T = f_{\theta}(\chi_c^T) \sim P(\mathbf{x}|\chi_c^T). \tag{3}$$

In this work, we incorporate the mixture density model with the CNN instead of a multi-layer perceptron as done by Bishop *et al.* in the vanilla research.¹⁷ In the proposed scheme, the network itself learns to act as a feature extractor to formulate statistical inferences on temporal series of intensity values. First, as the background image contains the most frequently presented intensities in the sequence of observed scenes, we take advantage of this in CDA intuition to exploit the most likely intensity value that will rise in the background image via consideration of temporal arrangement. Second, the memory requirement to store so many weights with multi-layer perceptron may rule out certain hardware implementations. In convolutional layers, the scheme of weight sharing in the proposed CNN reduces the number of parameters, making CDA lighter and exploiting the parallel processing of a set of multiple pixel-wise analyses within a batch of video frames.

The architecture of CDA contains seven learned layers, not counting the input — two depthwise convolutional, two convolutional, and three dense layers. Our network is summarized in Fig. 2. The input of our rudimentary architecture of the proposed network is a time series of color intensity at each pixel, which was analyzed with noncomplete connection schemes in four convolution layers regarding temporal perspective. Finally, the feature map of the last convolution layer was connected

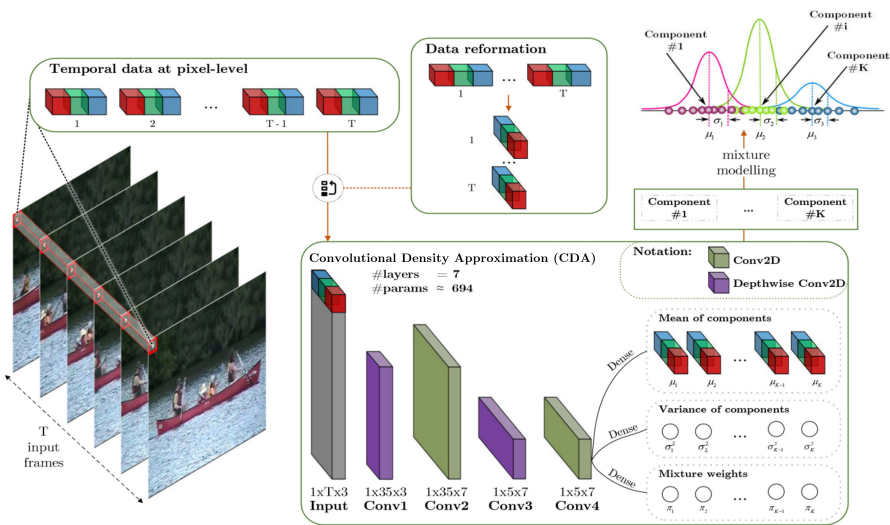


Fig. 2. The proposed architecture of Convolution Density Network of GMM.

with three different configurations of dense layers to form a three-fold output of the network which presents the kernel parameter of the GMM.

The main goal of CDA is to construct an architecture of CNN that presents multivariate mapping in the form of GMM with the mechanism of offline learning. With the simulated probabilistic function, we aim to model the description of the most likely background scenes from actual observed data. In other words, the regularities in the proposed CNN should cover a generalized presentation of the intensity series of a set of consecutive frames at the pixel level. To achieve this proposition, instead of using separate GMM for each pixel-wise statistical learning, we consider using a single GMM to formulate the temporal history of all pixels in the whole image. Accordingly, CDA architecture is extended through a spatial extension of temporal data at image points with an extensive scheme defined in Fig. 2.

The network output \mathbf{y}_T , whose dimension is $(c + 2) \times K$, is partitioned into three portions $\mathbf{y}_\mu(\mathbf{x}_c^T)$, $\mathbf{y}_\sigma(\mathbf{x}_c^T)$, and $\mathbf{y}_\pi(\mathbf{x}_c^T)$ of GMM:

$$\begin{aligned} \mathbf{y}_T &= [\mathbf{y}_\mu(\mathbf{x}_c^T), \mathbf{y}_\sigma(\mathbf{x}_c^T), \mathbf{y}_\pi(\mathbf{x}_c^T)], \\ &= [\mathbf{y}_\mu^1, \dots, \mathbf{y}_\mu^K, \mathbf{y}_\sigma^1, \dots, \mathbf{y}_\sigma^K, \mathbf{y}_\pi^1, \dots, \mathbf{y}_\pi^K]. \end{aligned} \quad (4)$$

With our goal of formulating the GMM, we impose a different restriction on threefold outputs from the network:

- First, as the mixing coefficients π_k indicate the proportion of data accounted for by mixture component k , they must be defined as independent and identically distributed probabilities. To achieve this regulation, in principle, we activate the network output with a softmax activation function:

$$\pi_k(\mathbf{x}_c^T) = \frac{\exp(\mathbf{y}_\pi^k)}{\sum_{l=1}^K \exp(\mathbf{y}_\pi^l)}. \quad (5)$$

- Second, in realistic scenarios, the measured intensity of observed image signals may fluctuate due to a variety of factors, including illumination transformations, dynamic contexts, and bootstrapping. Hence, we restrict the value of the variance of each component to the range $[\bar{\sigma}_{\min}, \bar{\sigma}_{\max}]$ so that each component does not span spread the entire color space, and does not focus on one single color cluster:

$$\sigma_k(\mathbf{x}_c^T) = \frac{\bar{\sigma}_{\min} \times (1 - \hat{\sigma}_k) + \bar{\sigma}_{\max} \times \hat{\sigma}_k}{255}, \quad (6)$$

where $\sigma_k(\mathbf{x}_c^T)$ is normalized toward a range of $[0, 1]$ over the maximum color intensity value, 255; and $\hat{\sigma}_k$ is the normalized variance activated through a hard-sigmoid function, from the output neurons \mathbf{y}_σ that correspond with the variances:

$$\hat{\sigma}_k(\mathbf{x}_c^T) = \max \left[0, \min \left(1, \frac{2 \times \mathbf{y}_\sigma^k + 5}{10} \right) \right]. \quad (7)$$

In this work, we adopt the hard sigmoid function because of the piecewise linear property and correspondence to the bounded form of a linear rectifier function (ReLU) of the technique. Furthermore, this was proposed and proved to be more efficient in both software and specialized hardware implementations by Courbariaux *et al.*⁴³

- Third, the mean of the probabilistic mixture is considered on a normalized RGB color space where the intensity values retain in a range of $[0, 1]$ so that they can be approximated correspondingly with the normalized input. Similar to the normalized variance $\hat{\sigma}_k$, we have

$$\mu_k(\boldsymbol{\chi}_c^T) = \max \left[0, \min \left(1, \frac{2 \times \mathbf{y}_\mu^k + 5}{10} \right) \right]. \quad (8)$$

From the proposed CNN, we extract the periodical background image for each block of pixel-wise time series of data in a period of T . This can be done by taking the weighted average of the estimated means, essentially summarizing the contextual dynamics of the scene into one background image.

$$\text{BG}(\boldsymbol{\chi}_c^T) = \sum_{k=1}^K \pi_k(\boldsymbol{\chi}_c^T) \cdot \mu_k(\boldsymbol{\chi}_c^T). \quad (9)$$

3.2. Learning posterior estimation

In practice, particularly in each real-life scenario, the background model must capture multiple degrees of dynamics, which is more challenging by the fact that scene dynamics may also change gradually under external effects (e.g. lighting deviations). These effects convey the latest information regarding contextual deviations that may constitute new background predictions. Therefore, the modeling of backgrounds must not only take into account the various degrees of dynamics across multiple imaging pixels of the data source, but it must also be able to adaptively update its predictions concerning semantic changes.

Equivalently, to approximate a statistical mapping function for background modeling, the proposed neural network function has to be capable of approximating a conditional PDF, thereby estimating a multi-modular distribution conditioned on its time-wise latest raw imaging inputs. The criteria for the neural statistical function to be instituted can be summarized as follows:

- By taking adaptiveness into account, the neural probabilistic density function can directly interpolate predictions in evolving scenes upon reception of new data.
- As a metric for estimating distributions, input data sequences cannot be weighted in terms of order.

Hence, we have developed a self-supervised approach.

3.2.1. Adaptive objective function

To satisfy the first criterion, we propose to use an unsupervised loss function capable of directing CDA’s parameters toward adaptively capturing the conditional distribution of data inputs.

At every single pixel, the proposed CNN estimates the probabilistic density function on the provided data by parameterizing the GMM. Specifically, given a set $\boldsymbol{\chi}_c^T$ of vectorized data points, π_k , μ_k and σ_k shall be functions parameterized by the set. Thus, Eq. (B.1) can be modified for target \mathbf{x} :

$$P(\mathbf{x}) = \sum_{k=1}^K \pi_k(\boldsymbol{\chi}_c^T) \cdot \mathcal{N}(\mathbf{x} | \mu_k, \sigma_k), \quad (10)$$

where

$$\mathcal{N}(\mathbf{x} | \mu_k, \sigma_k) = \frac{1}{\sqrt{(2\pi)^c \cdot \sigma_k^c(\boldsymbol{\chi}_c^T)}} \exp \left\{ -\frac{\|\mathbf{x} - \mu_k(\boldsymbol{\chi}_c^T)\|^2}{2\sigma_k^c(\boldsymbol{\chi}_c^T)} \right\}. \quad (11)$$

In this loss objective, the data distribution to be approximated is the set of data points relevant to background construction. This is rationalized by the goal of directing the neural network’s variables toward generalizing universal statistical mapping functions. Even with constantly evolving scenes where the batches of data values also vary, this loss measure can constitute fair weighting on the sequence of inputs thanks to explicit design to capture various pixel-wise dynamics over a video scene, and encompass unseen perspectives.

Practical modeling: We establish the mapping function on the RGB color space, which would require optimizing the loss on not just any 3-channeled pixel, but for $b = H \times W$ spatial blocks of image intensity data, over the temporal data axis T

$$\mathcal{L} = \sum_i^b \mathcal{L}^{(i)}(\boldsymbol{\chi}_c^T) = \sum_i^b \sum_j^T \mathcal{L}_j^{(i)} \quad (12)$$

with

$$\mathcal{L}_j^{(i)} = -\ln \left[\sum_{k=1}^K \pi_k^{(i)} \mathcal{N}(\mathbf{x}_j | \mu_k^{(i)}, \sigma_k^{(i)}) \right], \quad (13)$$

where \mathbf{x}_j is the j th element of the i th time-series data $\boldsymbol{\chi}_c^{T,(i)}$ of pixel values; $\pi^{(i)}$, $\mu^{(i)}$, and $\sigma^{(i)}$ are, respectively, the desired mixing coefficients, means, and variances that commonly model the distribution of $\boldsymbol{\chi}_c^{T,(i)}$ in GMM.

We define $\mathcal{L}_j^{(i)}$ as the error function for our learned estimation on an observed data point \mathbf{x}_j , given the locally relevant dataset $\boldsymbol{\chi}_c^{T,(i)}$ for the neural function. $\mathcal{L}_j^{(i)}$ is based on the statistical log-likelihood function and is equal to the negative of its magnitude. Hence, by minimizing this loss measure, we will essentially be

maximizing the expected likelihood value of the GMM-based neural probabilistic density function $P(\mathbf{x})$.

Employing stochastic gradient descent on the negative logarithmic function $\mathcal{L}_j^{(i)}$ involves not only monotonic decreases, which are steep when close to zero, but also upon convergence it also leads to the proposed neural function approaching an optimized mixture of Gaussians PDF. In addition, since this loss function depends entirely on the input and the output of the network (i.e. without external data labels), it is completely unsupervised. Optimization of the function is intended for the network to generalize on new data that is available on the fly without labels.

Learning by back-propagation: Learning can only be achieved if we can obtain suitable equations of the partial derivatives of the error \mathcal{L} with respect to outputs of the network. As we describe in the previous section, \mathbf{y}_μ , \mathbf{y}_σ , and \mathbf{y}_π present the proposed CDA's outputs that formulate to the latent variables of GMM. The partial derivatives $\partial\mathcal{L}_j^{(i)}/\partial\mathbf{y}^{(k)}$ can be evaluated for a particular pattern and then summed up to produce the derivative of the error function \mathcal{L} . To simplify the further analysis of the derivatives, it is convenient to introduce the following notation that presents the posterior probabilities of the component k in the mixture, using Bayes theorem:

$$\Pi_k^{(i)} = \frac{\pi_k^{(i)} \mathcal{N}(\mathbf{x}_j | \boldsymbol{\mu}_k^{(i)}, \sigma_k^{(i)})}{\sum_{l=1}^K \pi_l^{(i)} \mathcal{N}(\mathbf{x}_j | \boldsymbol{\mu}_l^{(i)}, \sigma_l^{(i)})}. \tag{14}$$

First, we need to consider the derivatives of the loss function with respect to the network's outputs \mathbf{y}_π that correspond to the mixing coefficients π_k . Using Eqs. (B.14) and (B.15), we obtain

$$\frac{\partial\mathcal{L}_j^{(i)}}{\partial\pi_k^{(i)}} = -\frac{\Pi_k^{(i)}}{\pi_k^{(i)}}. \tag{15}$$

From this expression, we perceive that the value of $\pi_k^{(i)}$ explicitly depends on $\mathbf{y}_\pi^{(l)}$ for $l = 1, 2, \dots, K$ as $\pi_k^{(i)}$ is the result of the softmax mapping from $\mathbf{y}_\pi^{(l)}$ as indicated in Eq. (B.6). We continue to examine the partial derivative of $\pi_k^{(i)}$ with respect to a particular network output $\mathbf{y}_\pi^{(l)}$, which is

$$\frac{\partial\pi_k^{(i)}}{\partial\mathbf{y}_\pi^{(l)}} = \begin{cases} \pi_k^{(i)}(1 - \pi_l^{(i)}) & \text{if } k = l, \\ -\pi_l^{(i)}\pi_k^{(i)} & \text{otherwise.} \end{cases} \tag{16}$$

By the chain rule, we have

$$\frac{\partial\mathcal{L}_j^{(i)}}{\partial\mathbf{y}_\pi^{(l)}} = \sum_k \frac{\partial\mathcal{L}_j^{(i)}}{\partial\pi_k^{(i)}} \frac{\partial\pi_k^{(i)}}{\partial\mathbf{y}_\pi^{(l)}}. \tag{17}$$

From Eqs. (B.15), (B.18), (B.19), and (B.21), we then obtain

$$\frac{\partial \mathcal{L}_j^{(i)}}{\partial \mathbf{y}_\pi^{(l)}} = \pi_l^{(i)} - \Pi_l^{(i)}. \quad (18)$$

For $\mathbf{y}_\sigma^{(k)}$, we make use of Eqs. (B.3), (B.7), (B.34), (B.14), and (B.15), by differentiation, to obtain

$$\frac{\partial \mathcal{L}_j^{(i)}}{\partial \mathbf{y}_\sigma^{(k)}} = \frac{0.2(\bar{\sigma}_{\max} - \bar{\sigma}_{\min})}{255} \Pi_k \left(\frac{c}{2\sigma_k^{(i)}} - \frac{\|\mathbf{x}_j - \boldsymbol{\mu}_k\|^2}{2(\sigma_k^{(i)})^2} \right) \quad (19)$$

for $-2.5 < \mathbf{y}_\sigma^{(k)} < 2.5$. This is because of the piece-wise property in the definition of the hard-sigmoid activation function.

Finally, for $\mathbf{y}_\mu^{(k)}$, let $\mu_{k,l}^{(i)}$ be the l th element of the mean vector, where l is an integer, which lies in $[0, c)$ and suppose that $\mu_{k,l}^{(i)}$ corresponds to an output o_k^μ of the network. We can get derivative of $\mu_{k,l}^{(i)}$ by taking Eqs. (B.3), (B.9), (B.14), (B.15) into the differentiation process:

$$\frac{\partial \mathcal{L}_j^{(i)}}{\partial \mathbf{y}_\mu^{(k,l)}} = -0.2 \times \Pi_k^{(i)} \cdot \left[\frac{x_{j,l}^{(i)} - \mu_{k,l}^{(i)}}{\sigma_k^{(i)}} \right] \quad (20)$$

for $-2.5 < \mathbf{y}_\mu^{(k)} < 2.5$.

From Eqs. (B.22), (B.34), and (B.31), we validate the primary conceptualization that the loss objective is differentiable and optimizable by our CDA formulation.

3.2.2. Inducing permutation invariance

True to fundamental theories in statistics, the posterior PDF on the history of intensity occurrences ought not to depend on the order of appearances. To satisfy the second requirement, the order of the inputs should not matter upon loading, which is proper for any statistical function that estimates PDFs. In other words, regardless of what sampled pixel values appear first, estimates of the population distribution only depend on their frequency. We propose an augmentation method to revise the loss objective as a self-supervised procedure to induce permutation invariance.

We denote $\rho_n(\boldsymbol{\chi}_c^T)$ as the n permutation of $\boldsymbol{\chi}_c^T$, such that n is an integer and $\rho_1(\boldsymbol{\chi}_c^T) = \boldsymbol{\chi}_c^T$. Thus, we aim to satisfy

$$\begin{aligned} \pi_k(\boldsymbol{\chi}_c^T) &\approx \pi_k(\rho_n(\boldsymbol{\chi}_c^T)), \\ \sigma_k(\boldsymbol{\chi}_c^T) &\approx \sigma_k(\rho_n(\boldsymbol{\chi}_c^T)), \\ \mu_k(\boldsymbol{\chi}_c^T) &\approx \mu_k(\rho_n(\boldsymbol{\chi}_c^T)), \end{aligned} \quad (21)$$

which is applied $\forall n | 1 \leq n \leq T!$.

We implicitly drive the model's parameters to achieve condition (21) by slight modification of the loss function on random samples of integer n :

$$\mathcal{L} = \sum_i^b \mathcal{L}^{(i)}(\rho_n(\mathbf{x}_c^T)), \quad (22)$$

which is to regularize model parameters for generalized inferencing of a diverse range of cases, as the convolutional operations have demonstrated rotational and translational robustness, but not against permutational variance.

3.3. Background modeling for efficient foreground extraction

In this section, we show that the utilization of background models can provide sufficient information for foreground extraction, thereby reducing the required computational expenses involved while maintaining decent accuracies. Hence, we developed a convolutional auto-encoder, called NeMos, to simulate nonlinear frame-background differencing for foreground detection on background models.

Traditionally, thresholding schemes are employed to find the highlighted difference between an imaging input and its corresponding static view in order to segment motion. For example, Stauffer and Grimson⁷ employed variance thresholding on background-input pairs by modeling the static view with GMM. While experimental results suggest certain degrees of applicability due to its simplicity, the approach lacks flexibility as the background model is usually not static and may contain various motion effects such as occlusions, stopped objects and shadow effects.

In practice, a good design of a difference function between the current frame and its background must be capable of facilitating segmentation across a plethora of scenarios and effects. However, regarding countless scenarios in real life, where there are unique image features and object behaviors, there is yet any explicit mathematical model that is general enough to cover them all. Thus, effective subtraction requires high-degreed nonlinearity in order to approximate a model for the underlying mathematical framework. Following the universal approximation theorem,⁴⁴ we design the technologically parallelizable neural function for an approximation of such framework. Specifically, we make use of a CNN to construct a foreground segmentation network. The motive is further complemented by two folds.

- CNNs have long been known for their effectiveness in approximating nonlinear functions with arbitrary accuracy.
- CNNs are capable of balancing between both speed and generalization accuracy, especially when given an effective design and enough representative training data.

We exploit the use of a pair of the current video frame and its corresponding background as the input to the neural function and extract motion estimation. By combining this with a suitable learning objective, we explicitly provide the neural

function with enough information to mold itself into a context-driven nonlinear difference function, thereby restricting model behavior and its search directions. This also allows us to scale down the networks parameter size, width, and depth to focus on learning representations while maintaining generalization for unseen cases. As empirically shown in the experiments, the proposed architecture is lightweight in terms of the number of parameters, and is also extremely resource-efficient.

Compared to approaches that perform semantic segmentation on single images to cluster pixels of certain known classes in the training set (e.g. FgSegNet¹²), NeMos relies on existing input data to perform a learned pixel-wise subtraction procedure on input signals, conditioned on obvious and implicit distinctions. Essentially given,

$$I_t = BG_t + \phi(FG_t), \tag{23}$$

where FG_t is a binary foreground map and ϕ represents the pixel-wise transformation function such that $\phi(FG_t) = I_t - BG_t$. Thus, to solve the equation $FG_t = \phi^{-1}(I_t - BG_t) = \Phi(I_t, BG_t)$, we seek to approximate the equivariant function Φ by a neural network that operates using all necessary information in I_t and BG_t . This means we can not only circumvent expensive analytic operations on spatio-temporal 4D colored tensors, but can also avoid associating pixel regions with certain target classes that may require more data or heavier architectures for full semantic segmentation operations.

3.3.1. Architectural design

The overall flow of the NeMos is shown in Fig. 3. We employ the encoder–decoder design approach for our segmentation function. With this approach, data inputs are compressed into a low-dimensional latent space of learned informative variables in

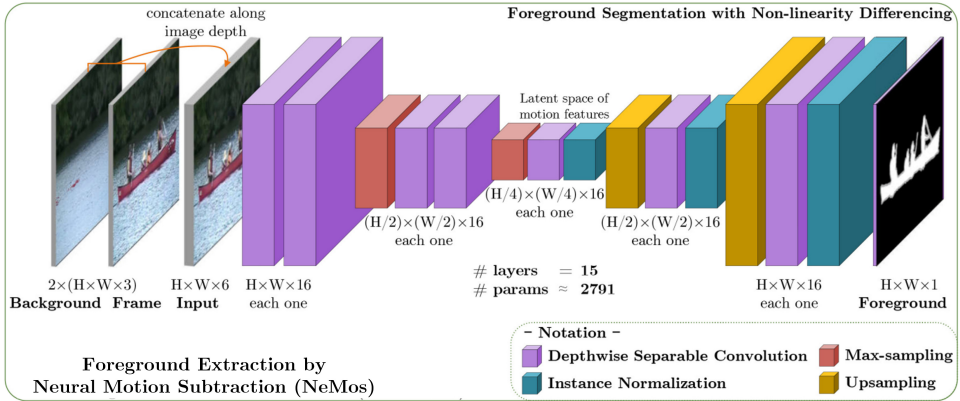


Fig. 3. The proposed architecture of NeMos grounded on convolutional autoencoder for foreground detection.

the encoder, and the encoded feature map is then passed into the decoder, thereby generating foreground masks.

Not only do we reduce the network size compared to FgSegNet, we also utilize the use of depthwise separable convolution introduced in MobileNets⁴⁵ so that our method can be suitable for mobile vision applications. Because this type of layer significantly scales down the number of convolutional parameters, we reduced the number of parameters of our network by approximately 81.7% compared to using only standard 2D convolution, rendering a lightweight network of around 2,800 parameters. Interestingly, even with such a small set of parameters, the network still does not lose its ability to generalize predictions at high accuracy. Our architecture also employs normalization layers, but only for the decoder. This design choice is to avoid the loss of information in projecting the contextual differences of background-input pairs into the latent space via the encoder, while formulating normalization to boost the decoders learning.

Encoder. The encoder can be thought of as a folding function that projects the loaded data into an information-rich low-dimensional feature space. In our architecture, the background image estimated by CDA is concatenated with imaging signals such that raw information can be preserved for the neural network to freely learn to manipulate. Moreover, with the background image also in its raw form, context-specific scene dynamics (e.g. moving waves, camera jittering, intermittent objects) are also captured. In addition, by explicitly providing a pair of the current input frame and its background image to segment foregrounds, our designed network essentially constructs a simple difference function that is capable of extending its behaviors to accommodate contextual effects. Thus, we theorize that approximating this neural difference function would not require an enormous number of parameters. In other words, it is possible to reduce the number of layers and the weight size of the foreground extraction network to accomplish the task. Hence, the encoder only consists of a few convolutional layers, with two max-pooling layers for downsampling contextual attributes into a feature-rich latent space.

Decoder. The decoder of our network serves to unfold the encoded feature map into the foreground space using convolutional layers with two upsampling layers to restore the original resolution of its input data. In order to facilitate faster training and better estimation of the final output, we engineered the decoder to include instance normalization, which is more efficient than batch normalization.⁴⁶ Using upsampling to essentially expand the latent tensors, the decoder also employs convolutional layers to induce nonlinearity like the encoder.

The final output of the decoder is a grayscale probability map where each pixel's value represents the chance that it is a component of a foreground object. We use the hard sigmoid activation function because of its property that allows faster gradient propagation, which results in less training time. At inference time, the final segmentation result is a binary image obtained by placing a constant threshold ϵ , which is experimentally determined, on the generated probability map.

3.3.2. Loss objective

We penalize the output of the network using the cross-entropy loss function commonly used for segmentation tasks $[x, y]$, as the goal of the model is to threshold the value of each pixel. The description of the loss function is as follows:

$$\mathcal{L} = - \sum_{i=1}^H \sum_{j=1}^W [\mathbf{Y}_{i,j} \log(\hat{\mathbf{Y}}_{i,j}) + (1 - \mathbf{Y}_{i,j}) \log(1 - \hat{\mathbf{Y}}_{i,j})], \quad (24)$$

where \mathbf{Y} is the corresponding target set of foreground binary masks for $\hat{\mathbf{Y}}$. We minimize $\mathbb{E}(\mathcal{L})$ on batches of predicted foreground probability maps. The network is trained for about 1000 epochs for each sequence in CDnet using Adam optimizer with the learning rate = 0.005. The designed architecture is enabled to learn not only pixel-wise motion estimates of the training set, but it also is taught to recognize inherent dynamics in its data to accurately interpolate region-wise foreground predictions of unseen perspectives.

4. Experiments and Discussion

4.1. Experimental setup

In this section, we verify experimentally the capabilities of the proposed method via comparative evaluations. Our goals are to evaluate the effectiveness and efficiency of CDA and NeMos in background modeling and subtraction. Our proposed scheme is designed to explicitly incorporate probabilistic density properties into the architecture to achieve accurate adaptiveness, while taking advantage of parallel computing technologies often used with DNNs to compete with state-of-the-art works in speed given its light structure. Therefore, we compare the accuracy of the proposed framework not only with unsupervised approaches that are light-weighted and generalizable without pretraining: GMM — Stauffer & Grimson,⁷ GMM — Zivkovic,⁸ SuBSENSE,³⁶ PAWCS,⁴⁷ TensorMoG,¹⁰ BMOG,⁹ FTSG,³⁷ SWCD,⁴⁸ but also with the data-driven, supervised models which trade computational expenses for high accuracy performance: FgSegNet_S,¹² FgSegNet,¹² FgSegNet_v2,¹³ Cascade CNN,³² DeepBS,³⁵ STAM.⁴⁹

First, in terms of BgS results, we employ quantitative analysis on the CDnet-2014⁵⁰ dataset. Our metrics are those that can be appraised from confusion matrices, i.e. Precision, Recall, F -Measure, False-Negative Rate (FNR), False-Positive Rate (FPR), and Percentage of Wrong Classification (PWC). With overall results being drawn from the combination of all confusion matrices across given scenarios, the benchmarks on CDnet-2014 were performed by comparing foreground predictions against provided ground-truths. Through our results, we observe the capabilities of NeMos in leveraging background models of CDA for context-driven BgS.

Then, we proceed with an ablation study to evaluate the contribution of the background generator, CDA, to the overall architecture on the Scene Background Modeling (SBMnet) dataset.⁵¹ The metrics include AGE (Average Gray-level Error), pEPs (Percentage of Error Pixels), pCEPs (Percentage of Clustered Error Pixels), MS-SSIM (MultiScale Structural Similarity Index), PSNR (Peak-Signal-to-Noise-Ratio), and CQM (Color image Quality Measure). The first three measure the intensity-level error difference between the algorithm’s output with the provided ground-truth, where lower estimation values indicate better background estimates. In contrast to how the first three are sensitive to small variations and require intensity-level exactness with referenced ground-truth, the latter three focus on quantifying the visual and structural quality of the background image generated by an algorithm. As exact background images are virtually impossible to obtain due to unavoidable variations of the camera’s capturing process, these structural- and visual-focused metrics provide for more objectiveness in background evaluations against reference ground-truths (higher values indicate better results).

Finally, we will also analyze all methods in terms of processing speed with the image resolution of 320×240 and draw final conclusions.

4.2. Implementation

In our experiments, the number of Gaussians K is empirically and heuristically to balance the CDA’s capability of modeling constantly evolving contexts (e.g. moving body of water) under many effects of potentially corruptive noises. With K too big, many GMM components may be unused or they simply capture various noises within contextual dynamics. As the Gaussian component corresponding to the background intensity revolves around the most frequently occurring color subspaces to draw predictions, the extra components serve only as either placeholders for abrupt changes in backgrounds, be empty, or capture intermittent noises of various degrees. In practice, noise Gaussian components in GMM are pulse-like as they would appear for short durations, and low-weighted because they are not as often matched as background components. Nevertheless, they still present corruptive effects to our model. Our proposed CDA model was set up with the number of Gaussian components $K = 3$ for all experimented sequences, and was trained on the CDnet-2014 dataset with Adam optimizer using a learning rate of $\alpha = 1e^{-4}$.

In addition, the constants $\bar{\sigma}_{\min}$ and $\bar{\sigma}_{\max}$ were chosen such that no Gaussian components span the whole color space while not contracting to a single point that represents noises. If the $[\bar{\sigma}_{\min}, \bar{\sigma}_{\max}]$ interval is too small, all of the Gaussian components will be likely to focus on one single color cluster. Otherwise, if the interval is too large, some of the components might still cover all intensity values, making it hard to find the true background intensity. Based on this assumption and experimental observations, we find that the difference between color clusters usually does not exceed approximately 16 at minimum and 32 at maximum.

Regarding NeMos, the value of ϵ was empirically chosen to be 0.3 to extract the foreground effectively even under high color similarity between objects and background.

The training dataset for NeMos is chosen by hand so that the data maintains a balance between background labels and foreground labels since imbalanced data will increase the model’s likelihood of being overfitted. We chose just 200 labeled ground truths to train the model. This is only up to 20% of the number of labeled frames for some sequences in CDnet, and 8.7% of CDnet’s labeled data overall. During training, the associated background of each chosen frame is directly generated using CDA as NeMos is trained separately from CDA because of the manually chosen input-label pairs.

4.3. Results on CDnet 2014 benchmarks

With 53 video sequences (length varying from 1,000 to 7,000 frames) spread over 11 different scenarios, the CDnet-2014 dataset⁵⁰ is the current biggest, most comprehensive large-scale public dataset for evaluating algorithms in the field of online video Change Detection. Using it, we demonstrate empirically the effectiveness of our proposed approach across a plethora of scenarios and effects. For each thousands-frame sequence of a scenario, we sample only 200 foreground images for training our foreground estimator. This strategy of sampling for supervised learning is the same as that of FgSegNet and Cascade CNN. The experimental results are summarized in Table 1, which highlights the F -measure quantitative results of our approach compared against several existing state-of-the-art approaches. Despite its compact

Table 1. F -measure comparisons over all of 11 categories in the CDnet 2014 dataset.

	Method	BDW	LFR	NVD	PTZ	THM	SHD	IOM	CJT	DBG	BSL	TBL	Average
Unsupervised	GMM — S & G	0.7380	0.5373	0.4097	0.1522	0.6621	0.7156	0.5207	0.5969	0.6330	0.8245	0.4663	0.5707
	GMM — Zivkovic	0.7406	0.5065	0.3960	0.1046	0.6548	0.7232	0.5325	0.5670	0.6328	0.8382	0.4169	0.5566
	SuBSENSE	0.8619 ₍₂₎	0.6445	0.5599 ₍₃₎	0.3476 ₍₃₎	0.8171 ₍₃₎	0.8640 ₍₃₎	0.6569	0.8152 ₍₂₎	0.8177	0.9503 ₍₁₎	0.7792 ₍₂₎	0.7408 ₍₃₎
	PAWCS	0.8152	0.6588 ₍₃₎	0.4152	0.4615 ₍₁₎	0.9921 ₍₁₎	0.8710 ₍₂₎	0.7764 ₍₃₎	0.8137 ₍₃₎	0.8938 ₍₁₎	0.9397 ₍₃₎	0.6450	0.7403
	TensorMoG	0.9298 ₍₁₎	0.6852 ₍₂₎	0.5604 ₍₂₎	0.2626	0.7993	0.9738 ₍₁₎	0.9325 ₍₁₎	0.9325 ₍₁₎	0.6493	0.9488 ₍₂₎	0.8380 ₍₁₎	0.8226 ₍₁₎
	BMOG	0.7836	0.6102	0.4982	0.2350	0.6348	0.8396	0.5291	0.7493	0.7928	0.8301	0.6932	0.6543
	FTSG	0.8228	0.6259	0.5130	0.3241	0.7768	0.8535	0.7891 ₍₂₎	0.7513	0.8792 ₍₂₎	0.9330	0.7127	0.7283
SWCD	0.8233 ₍₃₎	0.7374 ₍₁₎	0.5807 ₍₁₎	0.4545 ₍₂₎	0.8581 ₍₂₎	0.8302	0.7092	0.7411	0.8645 ₍₃₎	0.9214	0.7735 ₍₃₎	0.7583 ₍₂₎	
Supervised	DeepBS	0.8301	0.6002	0.5835	0.3133	0.7583	0.9092	0.6098	0.8990	0.8761	0.9580	0.8455	0.7870
	Cascade CNN	0.9431	0.8370	0.8965	0.9168	0.8958	0.9414	0.8505	0.9758	0.9658	0.9786	0.9108	0.9209
	STAM	0.9703	0.6683	0.7102	0.8648	0.9328	0.9885	0.9483	0.8989	0.9155	0.9663	0.9907 ₍₃₎	0.9651
	FgSegNet	0.9845 ₍₃₎	0.8786 ₍₃₎	0.9655 ₍₃₎	0.9843 ₍₃₎	0.9648 ₍₃₎	0.9973 ₍₂₎	0.9958 ₍₁₎	0.9954 ₍₃₎	0.9951 ₍₃₎	0.9944 ₍₃₎	0.9921 ₍₂₎	0.9770 ₍₃₎
	FgSegNet_S	0.9897 ₍₂₎	0.8972 ₍₂₎	0.9713 ₍₂₎	0.9879 ₍₁₎	0.9921 ₍₁₎	0.9937 ₍₃₎	0.9940 ₍₃₎	0.9957 ₍₂₎	0.9958 ₍₂₎	0.9977 ₍₁₎	0.9681	0.9804 ₍₂₎
	FgSegNet_v2	0.9904 ₍₁₎	0.9336 ₍₁₎	0.9739 ₍₁₎	0.9862 ₍₂₎	0.9727 ₍₂₎	0.9978 ₍₁₎	0.9951 ₍₂₎	0.9971 ₍₁₎	0.9961 ₍₁₎	0.9952 ₍₂₎	0.9938 ₍₁₎	0.9847 ₍₁₎
Ours *	NeMos	0.4739	0.7039	0.5657	0.6338	0.7487	0.6795	0.6543	0.6656	0.2308	0.8769	0.7230	0.5928
	NeMos + CDA	0.8529	0.9433	0.8342	0.8799	0.8910	0.8350	0.9433	0.8427	0.9303	0.9648	0.9236	0.8774

Notes: *Semi-Unsupervised; Experimented scenarios include bad weather (BDW), low frame rate (LFR), night videos (NVD), turbulence (TBL), baseline (BSL), dynamic background (DBG), camera jitter (CJT), intermittent object motion (IOM), shadow (SHD), and thermal (THM). In each column, **Red**₍₁₎ is for the best, **Green**₍₂₎ is for the second best, and **Blue**₍₃₎ is for the third best.

architecture, the proposed approach is shown to be capable of significantly outperforming unsupervised methods, and competing with complex deep-learning-based, supervised approaches in terms of accuracy.

In comparison with unsupervised models built on the GMM background modeling framework like GMM — Stauffer & Grimson, GMM — Zivkovic, BMOG, and TensorMoG, the proposed approach is better augmented by the context-driven motion estimation plugin, without being constrained by simple thresholding schemes. Thus, it is able to provide remarkably superior F -measure results across the scenarios, especially on those where there are high degrees of noises or background dynamics like LFR, NVD, IOM, CJT, DBG and TBL. However, it is a little worse than TensorMoG on BDW, SHD, IOM, and CJT, which may be attributed to TensorMoG's carefully tuned hyperparameters on segmenting foreground, thereby suggesting that the proposed method is still limited possibly by its architectural size and training data. Comparison with other unsupervised methods is also conducted, using mathematically rigorous approaches such as SuBSENSE, PAWCS, FTSG, and SWCD that are designed to tackle scenarios commonly seen in real life (i.e. BSL, DBG, SHD, and BDW). Nevertheless, the F -measure results of the proposed approach around 0.90 suggest that it is still able to outperform these complex unsupervised approaches, possibly ascribing to its use of hand-labeled data for explicitly enabling context capturing.

In comparison with supervised approaches, the proposed approach is apparently very competitive against the more computationally expensive state of the arts. For instance, our approach considerably surpasses the generalistic methods of STAM and DeepBS on LFR and NVD, but it loses against both of these methods on SHD and CMJ, and especially is outperformed by STAM on many scenarios. While STAM and DeepBS are constructed using only 5% of CDnet-2014, they demonstrate good generalization capability across multiple scenarios by capturing the holistic features of their training dataset. However, despite being trained on all scenarios, their behaviors showcase higher degrees of instability (e.g. with LFR, NVD) than our proposed approach on scenarios that deviate from common features of the dataset. Finally, as our proposed method is compared against similarly scene-specific approaches like FgSegNets, Cascade CNN, the results were within expectations for almost all scenarios that ours would not be significantly outperformed, as the compared models could accommodate various features of each sequence in their big architectures. However, surprisingly, our method surpasses even these computationally expensive to be at the top of the LFR scenarios. This suggests that, with a background for facilitating motion segmentation from an input, our trained model can better tackle scenarios where objects are constantly changing and moving than even existing state-of-the-arts.

Interestingly, NeMos+CDA on the PTZ sequence returns substantially correct results, even better than its performances on BDW, NVD, SHD, or CJT. It can be hypothesized that NeMos would rather work with averaged-out backgrounds to perform raw semantic extraction, than with noisy motions (i.e. snow droplets,

shadows, jitters, lighting shifts) for context-driven BgS. Nevertheless, the sub-dataset PTZ is more limited in terms of observable objects and images in the region and scenario of interest compared to others, as can also be observed in the poor performance of DeepBS which attempted to generalize learning on imbalanced learned data.

Overall, with small training sets, NeMos+CDA achieved decent results in Precision, Recall, FPR, FNR, PWC, and a score of 0.8774 in average F -measure, which is much higher than any compared unsupervised approaches and can practically compete with other, more computationally expensive, supervised approaches despite its light-weighted structure. Table 2 presents evaluation metrics of a confusion matrix.

4.4. Result on SBMnet benchmarks

We perform an empirical ablation study of how the background generator, CDA, contributes to the overall architecture with the SBMnet dataset⁵¹ for evaluating background estimation results. The SBMnet dataset has 80 real-life video sequences and their corresponding ground-truth backgrounds for references over eight scenarios (illumination changes, cluttering, camera jitter, intermittent motion, etc.). It is an often-used dataset to quantitatively evaluate background modeling algorithms. Some of the algorithms that do not model the background, e.g. FgSegNet, SWCD, etc., are left out by default. For brevity, Table 3 provides the overall quantitative rankings (across all dataset sequences) of the proposed method along with state-of-the-art

Table 2. Result of quantitative evaluation on CDnet 2014 dataset.

	Method	Average Recall	Average FPR	Average FNR	Average PWC	Average Precision
Unsupervised	GMM — S & G	0.6846	0.0250	0.3154	3.7667	0.6025
	GMM — Zivkovic	0.6604	0.0275	0.3396	3.9953	0.5973
	SuBSENSE	0.8124	0.0096	0.1876 ₍₁₎	1.6780	0.7509
	PAWCS	0.7718 ₍₃₎	0.0051 ₍₁₎	0.2282	1.1992 ₍₁₎	0.7857 ₍₂₎
	TensorMoG	0.7772 ₍₂₎	0.0107	0.2228 ₍₃₎	2.3315	0.8215 ₍₁₎
	BMOG	0.7265	0.0187	0.2735	2.9757	0.6981
	FTSG	0.7657	0.0078 ₍₃₎	0.2343	1.3763 ₍₃₎	0.7696 ₍₃₎
	SWCD	0.7839 ₍₁₎	0.0070 ₍₂₎	0.2161 ₍₂₎	1.3414 ₍₂₎	0.7527
Supervised	FgSegNet_S	0.9896 ₍₁₎	0.0003 ₍₂₎	0.0104 ₍₁₎	0.0461 ₍₂₎	0.9751
	FgSegNet	0.9836 ₍₃₎	0.0002 ₍₁₎	0.0164 ₍₃₎	0.0559 ₍₃₎	0.9758
	FgSegNet_v2	0.9891 ₍₂₎	0.0002 ₍₁₎	0.0109 ₍₂₎	0.0402 ₍₁₎	0.9823 ₍₂₎
	Cascade CNN	0.9506	0.0032	0.0494	0.4052	0.8997
	DeepBS	0.7545	0.0095	0.2455	1.9920	0.8332
	STAM	0.9458	0.0005 ₍₃₎	0.0542	0.2293	0.9851 ₍₁₎
Ours*	NeMos	0.8718	0.0303	0.1282	3.2975	0.4491
	NeMos + CDA	0.9225	0.0051	0.0775	0.7097	0.8366

Notes: *Semi-Unsupervised; In each column, **Red**₍₁₎ is for the best, **Green**₍₂₎ is for the second best, and **Blue**₍₃₎ is for the third best.

Table 3. Comparison on the SBMnet dataset.

Method	AGE ↓	pEPs ↓	pCEPs ↓	MS-SSIM ↑	PSNR ↑	CQM ↑
GMM — S & G	15.7189 ₍₃₎	0.1275 ₍₁₎	0.1018 ₍₃₎	0.8639 ₍₂₎	26.1304 ₍₁₎	26.9061 ₍₁₎
GMM — Zivkovic	12.9308 ₍₁₎	0.1342 ₍₂₎	0.0995 ₍₂₎	0.8554	23.5434 ₍₃₎	24.3670 ₍₃₎
SuBSENSE	19.5009	0.2367	0.1353	0.8562 ₍₃₎	18.0343	19.2097
PAWCS	25.0482	0.3244	0.1752	0.7800	17.1190	18.2663
TensorMoG	13.8699 ₍₂₎	0.1446 ₍₃₎	0.0745 ₍₁₎	0.8769 ₍₁₎	22.1011	23.2473
CDA (Ours)	19.6865	0.2033	0.1486	0.8405	23.5831 ₍₂₎	24.5391 ₍₂₎

Notes: In each column, Red₍₁₎ is for the best, Green₍₂₎ is for the second best, and Blue₍₃₎ is for the third best.

background estimation algorithms which are originally based on Gaussian Mixture Estimation.

In general, Table 3 demonstrates that the traditional GMM-based methods, GMM — Stauffer & Grimson, GMM — Zivkovic, and TensorMoG, are the top-performing methods in the background modeling domain. The proposed CDA module is outperformed by these traditional GMM-based algorithms in terms of the pixel-based metrics, i.e. AGE, pEPs, and pCEPs, which measure the intensity difference between the generated background and the ground-truth. However, the gains of CDA on visual quality measurements, i.e. MS-SSIM, PSNR, and CQM, signify that the background generated by CDA is competitive against the top GMM-based methods on the background estimation domain in terms of textural and semantic information compared to the ground-truth.

The shortcomings of the proposed background extraction methods in its exact background grayscale estimation show up very clearly in the three metrics AGE, pEPs, and pCEPs, where lower results are better. The background component of the proposed method consistently falls out of the top-3 best methods. There are two main possible reasons why such shortcomings exist. First, the first three grayscale-based metrics are highly sensitive to small variations in the estimated background as these metrics measure the estimation result based entirely on its absolute difference with the provided ground-truth. In real life, however, obtaining a completely accurate background image is inherently impossible since the camera cannot consistently capture the same signal for every pixel in the image, i.e. avoiding variations in capturing pixel signals is inherently impossible. Thus, while these metrics surely provide some degree of confidence in the computed background image quality, they cannot serve as the absolute determination of the image quality. Second, our design of CDA focuses on speed efficiency with a small temporal window in contrast to traditional GMM-based methods that can capture long-term pixel signals to estimate the background intensities with high accuracy. This tradeoff between the efficiency and effectiveness of the algorithm results in a clear disadvantage for CDA in estimating the grayscale signal as close as possible to the ground-truth compared to the other five methods. However, the absolute difference of the CDA module with the top performing method on each metric is still within an acceptable margin: 6.7557 in

AGE (compared to GMM — Zivkovic), 0.0758 in pEPs (compared to GMM — Stauffer & Grimson), and 0.0741 in pCEPs (compared to TensorMoG).

In contrast, on the other three metrics (MS-SSIM, PSNR, and CQM) which measure the visual and structural distortion of estimated backgrounds against ground-truths, the proposed CDA yields very good results. The focus of these metrics was to quantify the errors in artificially generated image’s visual quality as perceived by humans as closely as possible. In these metrics, higher quantitative values correspond to better background estimations. The effectiveness of CDA’s background images’ quality is showcased with (1) the difference with the top-1 method in MS-SSIM, TensorMoG with parameter tuning, is only a marginal value of 0.0364, and (2) CDA consistently shows up as the second best method in PSNR and CQM. Thus, backgrounds approximated by CDA are images of decent quality, with good textural and semantic information compared to ground-truths.

As an unsupervised, generalistic approach, although our proposed CDA module is less competitive against traditional GMM-based methods on grayscale error estimations of the background, the good results on visual quality metrics imply that the semantic information of the generated background and the input image are very similar. This semantic similarity between the background and the input frame possibly has suppressed a large number of background distractors from the input frame for the imbalance foreground segmentation learning task (very high difference in the number of pixels classified as background compared to the number of foreground pixels). However, it should be noted that CDA is still independent of NeMos, which means that any background generation algorithm can theoretically replace CDA in reducing background distractors.

Nevertheless, there are two main reasons for the preference of CDA over other algorithms. First, because CDA is advantageous in maintaining adaptation in cases where environmental changes happen often (e.g., illumination changes) like traditional GMM approaches, its learning to generalize for a small local temporal history is comparable to the slow, gradual adaptation of GMM-based family of algorithms. Thus, with CDA providing suitable backgrounds (via feed-forwarded GMM approximations of windowed data) in a timely manner for NeMos, the latter module is supported with suppression of distractions, which contributes greatly to the reason why NeMos can be such light-weighted but still maintains effective context-driven segmentation. Secondly, most importantly, CDA is the more modern paradigm of background modeling with GMM, in which CDA is highly parallelizable on modern hardware and avoids the speed-throttling nature of sequential paradigm of methods such as GMM — Stauffer & Grimson, GMM — Zivkovic, SuBSENSE, and PAWCS in pixel-wise background generation.

4.5. Computational speed comparison

The proposed framework was implemented on a CUDA-capable machine with an NVIDIA GTX 1070 Ti GPU or similar, along with the methods that require CUDA

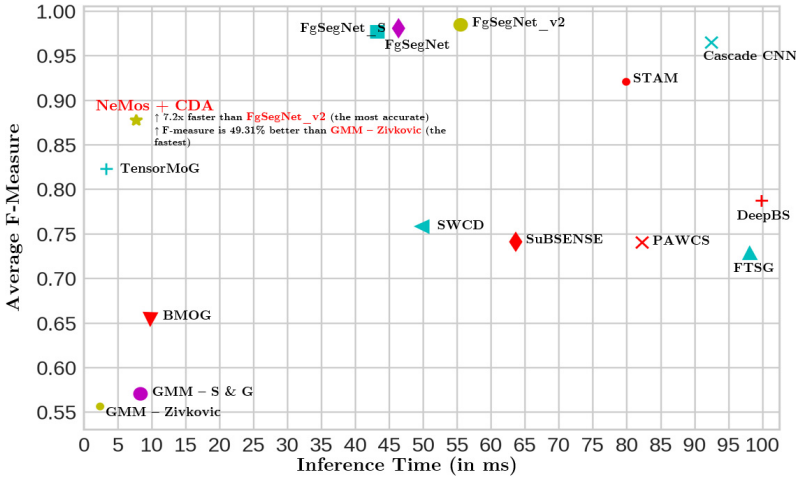


Fig. 4. Computational speed and average F -measure comparison with state-of-the-art methods.

runtime, i.e. TensorMoG, DeepBS, STAM, FgSegNet, and Cascade CNN. For unsupervised approaches, we conducted our speed tests on the configuration of an Intel Core i7 with 16 GB RAM. Our results are recorded quantitatively with execution performance in frame-per-seconds (fps), and time (milliseconds) versus accuracy in Fig. 4. At the overall speed of 129.4510 fps (from about 3,500 parameters), with CDA (about 2,800 parameters) module processing at 402.1087 fps, NeMos+CDA is much faster than other supervised deep learning approaches, of which the fastest — FgSegNet_S — runs at 23.1275 fps. By concatenating estimations of background scenes with raw signals for foreground extraction, our approach makes efficient use of hardware resources due to its completely lightweight architecture and the latent-space-limitation approach. In contrast, other DNNs are burdened with a large number of trainable parameters to achieve accurate input-target mapping. Furthermore, the proposed scheme dominates the mathematically rigorous unsupervised methods frameworks in terms of speed and accuracy such as SuBSENSE, SWCD, and PAWCS, as their paradigms of sequential processing are penalized by significant penalties in execution. Significantly, the average speeds of the top three methods are dramatically disparate. With the objective of parallelizing the traditional imperative outline of rough statistical learning on GMM, TensorMoG reformulates a tensor-based framework that surpasses our dual architecture at 302.5261 fps. On the other hand, GMM — Zivkovic’s design focuses on optimizing its mixture components, thereby significantly trading off its accuracy to attain the highest performance.

Notwithstanding, our proposed framework gives the most balanced trade-off (top-left-most) in addressing the speed-and-accuracy dilemma. Our model outperforms other approaches of top accuracy ranking when processing at exceptionally high speed, while obtaining good accuracy scores, at over 90% on more than half of CDnet’s categories and at least 84%.

5. Conclusion

This paper has developed a novel, two-stage BgS framework with a GMM-based CNN for background modeling, and a convolutional auto-encoder NeMos to simulate input-BgS for foreground detection, thus being considered as a search space limitation approach to compress a model of DNNs, while keeping up good accuracy. Our first and second contributions in this paper include a pixel-wise, light-weighted, feed-forward CNN representing a multi-modular conditional PDF of the temporal history of data, and a corresponding self-supervised training strategy for the CNN to learn from virtually inexhaustible datasets for approximating the mixture of Gaussian density function. In such a way, the proposed CDA not only gains the better capability of adaptation in contextual dynamics with humanly interpretable statistical learning for extension, but it is also designed in the tensor form to exploit modern parallelizing hardware. Secondly, we showed that incorporating such statistical features into NeMos’s motion-region extraction phase promises more efficient use of powerful hardware, with prominent speed performance and high accuracy, along with a decent generalization ability using a small-scale set of training labels, in a deep nonlinear scheme of only a few thousand parameters.

Since CDN constructs GMMs out of each pixel’s fixed-sized temporal window, neighborhood information is not captured while redundant temporal information may have been incorporated. Inspired by DGCNN,⁵² which learns irregular neighborhood patterns through the GMM and optimizes the neural network kernels accordingly over graph data, we are investigating irregular patterns of convolution on spatio-temporal data to efficiently and adaptively distinguish background and foreground features. In particular, sampling window widths out of the constructed GMMs can potentially overcome the issues of fixed-width temporal convolution, so an extension to spatio-temporal irregular convolution can address the efficiency issues commonly observed in 3D convolution.

Acknowledgments

This research is funded by the Vietnam National University HoChiMinh City (VNU-HCM) under grant number DS2022-28-04. We thank the Ho Chi Minh City International University — Vietnam National University (HCMIU-VNU) for facilitating this work. Our sincere appreciation also goes to our colleagues for their support, which significantly improved this paper.

Appendix A. Formulation of Equation (2)

Definition 3.2.1, Eqs. (3.2.1) and (3.2.2) in the textbook by Tong⁵³ can be revisited in Fig. A.1.

We adopt the original mathematical formulation of the Gaussian distribution to the setting of background modeling on multi-channeled videos. To avoid performing costly matrix inversion, each color channel in the color space is assumed to be

Definition 3.2.1. An n -dimensional random variable \mathbf{X} with mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$ is said to have a nonsingular multivariate normal distribution, in symbols $\mathbf{X} \sim \mathcal{N}_n(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, $\boldsymbol{\Sigma} > 0$, if (i) $\boldsymbol{\Sigma}$ is positive definite, and (ii) the density function of \mathbf{X} is of the form

$$f(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{n/2} |\boldsymbol{\Sigma}|^{1/2}} e^{-Q_n(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})/2}, \quad \mathbf{x} \in \mathfrak{R}^n, \quad (3.2.1)$$

where

$$Q_n(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = (\mathbf{x} - \boldsymbol{\mu})' \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}). \quad (3.2.2)$$

Fig. A.1. Snippet from p. 26 of Ref. 53.

distributed independently, so each Gaussian component in the mixture is simply described with a positive scalar variance value (i.e. σ). Hence, we get a positive definite covariance matrix (i.e. $\boldsymbol{\Sigma}$) for each Gaussian distribution, represented as a diagonal matrix with values equal to the same positive scalar across the diagonal. As a result, under general input data of c dimensions (e.g. if the video is encoded in RGB, then $c = 3$), the determinant of $\boldsymbol{\Sigma}$ is the multiplication of the same value σ across c diagonal values (i.e. $|\boldsymbol{\Sigma}| = \sigma^c$), and the inversion of $\boldsymbol{\Sigma}$ can simply be $\boldsymbol{\Sigma}^{-1} = \frac{1}{\sigma}$ so that $\boldsymbol{\Sigma}^{-1} \boldsymbol{\Sigma}$ is an identity matrix.

$$\boldsymbol{\Sigma} = \begin{bmatrix} \sigma & & \\ & \sigma & \\ & & \sigma \end{bmatrix}. \quad (A.1)$$

Equations (3.2.1) and (3.2.2) are combined and adopted as

$$f(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{c/2} |\boldsymbol{\Sigma}|^{1/2}} \exp^{-(\mathbf{x} - \boldsymbol{\mu})' \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu})/2}, \quad \mathbf{x} \in [\mathfrak{R}^+]^c \quad (A.2)$$

then equivalently for our case, at the k th Gaussian component,

$$\begin{aligned} f(\mathbf{x}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) &= \frac{1}{\sqrt{(2\pi)^c |\boldsymbol{\Sigma}_k|}} \exp\left(-\frac{(\mathbf{x} - \boldsymbol{\mu}_k)' \boldsymbol{\Sigma}_k^{-1} (\mathbf{x} - \boldsymbol{\mu}_k)}{2}\right), \quad \mathbf{x} \in [\mathfrak{R}^+]^c \\ &= \frac{1}{\sqrt{(2\pi)^c \sigma_k^c}} \exp\left(-\frac{\|\mathbf{x} - \boldsymbol{\mu}_k\|^2}{2\sigma_k}\right). \end{aligned} \quad (A.3)$$

Appendix B. Formulation Proof of CDA-GM

B.1. Formulation of CDA-GM

Let $\boldsymbol{\chi}_c^T = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T | \mathbf{x}_i \in [0, 255]^c\}$ be the time series of the T most recently observed color signals of a pixel where the dimension of the vector \mathbf{x}_i in the color space is c , the distribution of pixel intensity \mathbf{x}_i can be modeled by a linear combination of K probabilistic components $\boldsymbol{\theta}_k$ and their corresponding conditional PDFs $P(\mathbf{x}_i | \boldsymbol{\theta}_k)$. The marginal probability $P(\mathbf{x}_i)$ of the mixture is

$$P(\mathbf{x}) = \sum_{k=1}^K P(\boldsymbol{\theta}_k) P(\mathbf{x} | \boldsymbol{\theta}_k) = \sum_{k=1}^K \pi_k \cdot \mathcal{N}(\mathbf{x} | \boldsymbol{\mu}_k, \sigma_k), \quad (B.1)$$

where π_k is a non-negative mixing coefficient for θ_k :

$$\sum_{k=1}^K \pi_k = 1. \quad (\text{B.2})$$

We use the re-formulated multivariate Gaussian distribution as

$$\mathcal{N}(\mathbf{x}|\mu_k, \sigma_k) = \frac{1}{\sqrt{(2\pi)^c \sigma_k^c}} \exp\left(-\frac{\|\mathbf{x} - \boldsymbol{\mu}_k\|^2}{2\sigma_k}\right), \quad (\text{B.3})$$

where $\boldsymbol{\mu}_k$ is the estimated mean and σ_k is the estimated universal covariance of examined color channels in the k th Gaussian component θ_k .

Our proposed CNN formulates a conditional formalism of GMM density function of \mathbf{x} given a set of randomly selected, vectorized data points $\boldsymbol{\chi}_T$:

$$\mathbf{y}_T = f_\theta(\boldsymbol{\chi}_c^T) \sim P(\mathbf{x}|\boldsymbol{\chi}_c^T), \quad (\text{B.4})$$

where $f_\theta(\cdot)$ is a set of nonlinear transformations.

The network output \mathbf{y}_T , whose dimension is $(c+2) \times K$, is partitioned into three portions $\mathbf{y}_\mu(\boldsymbol{\chi}_c^T)$, $\mathbf{y}_\sigma(\boldsymbol{\chi}_c^T)$, and $\mathbf{y}_\pi(\boldsymbol{\chi}_c^T)$ of GMMs:

$$\begin{aligned} \mathbf{y}_T &= [\mathbf{y}_\mu(\boldsymbol{\chi}_c^T), \mathbf{y}_\sigma(\boldsymbol{\chi}_c^T), \mathbf{y}_\pi(\boldsymbol{\chi}_c^T)] \\ &= [\mathbf{y}_\mu^1, \dots, \mathbf{y}_\mu^K, \mathbf{y}_\sigma^1, \dots, \mathbf{y}_\sigma^K, \mathbf{y}_\pi^1, \dots, \mathbf{y}_\pi^K]. \end{aligned} \quad (\text{B.5})$$

With our goal of formulating the GMM, we restate the three restrictions on network outputs:

- First, as π_k indicates the proportion of data accounted for by mixture component k , they are defined as independent, weighted scores:

$$\pi_k(\boldsymbol{\chi}_c^T) = \frac{\exp(\mathbf{y}_\pi^k)}{\sum_{l=1}^K \exp(\mathbf{y}_\pi^l)}. \quad (\text{B.6})$$

- Second, we restrict the value of the variance of each component to the range $[\bar{\sigma}_{\min}, \bar{\sigma}_{\max}]$ so that the components do not span the entire color space.

$$\sigma_k(\boldsymbol{\chi}_c^T) = \frac{\bar{\sigma}_{\min} \times (1 - \hat{\sigma}_k) + \bar{\sigma}_{\max} \times \hat{\sigma}_k}{255}, \quad (\text{B.7})$$

where $\hat{\sigma}_k$ is the normalized variance that was activated through a hard-sigmoid function from the output neurons \mathbf{y}_σ :

$$\hat{\sigma}_k(\boldsymbol{\chi}_c^T) = \begin{cases} 0 & \text{if } \mathbf{y}_\sigma^k < -2.5, \\ 0.2 \times \mathbf{y}_\sigma^k + 0.5 & \text{if } -2.5 \leq \mathbf{y}_\sigma^k \leq 2.5, \\ 1 & \text{otherwise.} \end{cases} \quad (\text{B.8})$$

- Third, the mixture mean is standardized from the corresponding network outputs with a hard-sigmoid function:

$$\mu_k(\boldsymbol{\chi}_c^T) = \begin{cases} 0 & \text{if } \mathbf{y}_\mu^k < -2.5, \\ 0.2 \times \mathbf{y}_\mu^k + 0.5 & \text{if } -2.5 \leq \mathbf{y}_\mu^k \leq 2.5, \\ 1 & \text{otherwise.} \end{cases} \quad (\text{B.9})$$

We choose the hard-sigmoid function for the means and the variances, as explained.

From the proposed CNN, we extract the periodical background image for each block of pixel-wise time series of data in a period of T , by taking the weighted average of the estimated means,

$$\text{BG}(\boldsymbol{\chi}_c^T) = \sum_{k=1}^K \pi_k(\boldsymbol{\chi}_c^T) \cdot \mu_k(\boldsymbol{\chi}_c^T). \quad (\text{B.10})$$

Accordingly, the corresponding frame-wise foreground mask of each input frame is extracted from the Gaussian mixtures at each pixel location. Specifically, we applied a threshold ξ on the squared Mahalanobis distance between the input frame and the background distribution.

$$\text{FG}(\mathbf{x}_c^T) = \left[\frac{(\mathbf{x}_c^T - \text{BG}(\boldsymbol{\chi}_c^T))^2}{\tilde{\sigma}_t^2} > \xi \right], \quad (\text{B.11})$$

where

$$\tilde{\sigma}_t^2 = \max[\sigma_k^2(\boldsymbol{\chi}_c^T) \cdot \hat{\text{BG}}_{k,T}(\boldsymbol{\chi}_c^T)], \quad \text{for } k \in [1, K]. \quad (\text{B.12})$$

B.2. Unsupervised training via backpropagation

Specifically, given the set $\boldsymbol{\chi}_c^T$ randomly selected, vectorized data points, it is possible to retrieve the continuous conditional distribution of the data target \mathbf{x} .

In our proposed loss function, the data distributions to be approximated are the sets of data points that are relevant to background construction themselves.

$$\mathcal{L} = \sum_i^b \sum_j^T \mathcal{L}_j^{(i)}, \quad (\text{B.13})$$

where

$$\mathcal{L}_j^{(i)} = -\ln \left(\sum_{k=1}^K \pi_k^{(i)} \mathcal{N}(\mathbf{x}_j | \boldsymbol{\mu}_k^{(i)}, \sigma_k^{(i)}) \right), \quad (\text{B.14})$$

where \mathbf{x}_j is the j th element of the i th time-series data $\boldsymbol{\chi}_c^{T,(i)}$ of pixel values; $\pi^{(i)}$, $\boldsymbol{\mu}^{(i)}$, and $\sigma^{(i)}$ are, respectively, the desired mixing coefficients, means, and variances that

commonly model the distribution of $\boldsymbol{\chi}_c^{T,(i)}$ in GMM. We define $\mathcal{L}_j^{(i)}$ as the error function for our learned estimation on an observed data point \mathbf{x}_j , given the locally relevant dataset $\boldsymbol{\chi}_c^{T,(i)}$ for the neural function. $\mathcal{L}_j^{(i)}$ is based on the statistical log-likelihood function and is equal to the negative of its magnitude. Hence, by minimizing this loss measure, we will essentially be maximizing the expectation value of the GMM-based neural probabilistic density function $P(\mathbf{x})$, from the history of pixel intensities at a pixel position.

The key thing here is that whether the neural network can learn to optimize the loss function with the standard stochastic gradient descent algorithm with *back-propagation*. To simplify the further analysis of the derivatives using Bayes theorem, it is convenient to introduce the following notation:

$$\Pi_k = \frac{\pi_k \cdot \mathcal{N}(\mathbf{x}_j | \boldsymbol{\mu}_k, \sigma_k)}{\sum_{l=1}^K \pi_l \cdot \mathcal{N}(\mathbf{x}_j | \boldsymbol{\mu}_l, \sigma_l)}. \quad (\text{B.15})$$

First, we need to consider the derivatives of the loss function with respect to network outputs \mathbf{y}_π that correspond to the mixing coefficients π_k . Using Eq. (B.14) and (B.15), we obtain

$$\frac{\partial \mathcal{L}_j}{\partial \mathbf{y}_\pi^k} = \frac{\partial \mathcal{L}_j}{\partial \pi_k} \cdot \frac{\partial \pi_k}{\partial \mathbf{y}_\pi^k}. \quad (\text{B.16})$$

Thus,

$$\frac{\partial \mathcal{L}_j}{\partial \pi_k} = - \frac{\mathcal{N}(\mathbf{x}_j | \boldsymbol{\mu}_k, \sigma_k)}{\sum_{l=1}^K \pi_l \cdot \mathcal{N}(\mathbf{x}_j | \boldsymbol{\mu}_l, \sigma_l)} = - \frac{\Pi_k}{\pi_k}. \quad (\text{B.17})$$

From this expression, we perceive that the value of $\pi_k^{(i)}$ explicitly depends on $\mathbf{y}_\pi^{(l)}$ for $l = 1, 2, \dots, K$ as $\pi_k^{(i)}$ is the result of the softmax mapping from $\mathbf{y}_\pi^{(l)}$ as indicated in Eq. (B.6). We continue to examine the partial derivative of $\pi_k^{(i)}$ with respect to a particular network output $\mathbf{y}_\pi^{(l)}$, which is

$$\frac{\partial \pi_k}{\partial \mathbf{y}_\pi^l} = \begin{cases} \frac{\exp(\mathbf{y}_\pi^l) \cdot \sum_{p=1}^K \exp(\mathbf{y}_\pi^p) - [\exp(\mathbf{y}_\pi^k)]^2}{\left[\sum_{p=1}^K \exp(\mathbf{y}_\pi^p) \right]^2} & \text{if } k = l, \\ - \frac{\exp(\mathbf{y}_\pi^l) \cdot \exp(\mathbf{y}_\pi^k)}{\left[\sum_{p=1}^K \exp(\mathbf{y}_\pi^p) \right]^2} & \text{otherwise,} \end{cases} \quad (\text{B.18})$$

which can be simplified to

$$\frac{\partial \pi_k}{\partial \mathbf{y}_\pi^l} = \begin{cases} \pi_k \cdot (1 - \pi_k) & \text{if } k = l, \\ -\pi_l \cdot \pi_k & \text{otherwise.} \end{cases} \quad (\text{B.19})$$

By chain rule, we have

$$\frac{\partial \mathcal{L}_j}{\partial \mathbf{y}_\pi^l} = \sum_k \frac{\partial \mathcal{L}_j}{\partial \pi_k} \cdot \frac{\partial \pi_k}{\partial \mathbf{y}_\pi^l} \quad (\text{B.20})$$

thus,

$$\begin{aligned} \frac{\partial \mathcal{L}_j}{\partial \mathbf{y}_\pi^l} &= \sum_k -\frac{\Pi_k}{\pi_k} \cdot \frac{\partial \pi_k}{\partial \mathbf{y}_\pi^l} \\ &= \left(\sum_k \Pi_k \cdot \pi_l \right) - \Pi_l. \end{aligned} \tag{B.21}$$

From Eqs. (B.15), (B.18), (B.19), and (B.21), we then obtain

$$\frac{\partial \mathcal{L}_j}{\partial \mathbf{y}_\pi^l} = \pi_l - \Pi_l. \tag{B.22}$$

For $\mathbf{y}_\sigma^{(k)}$, we make use of Eqs. (B.3), (B.7), (B.33), (B.14), and (B.15), by differentiation, to obtain

$$\begin{aligned} \frac{\partial \mathcal{L}_j}{\partial \sigma_k} &= \frac{\partial}{\partial \sigma_k} \cdot \left[-\ln \left(\sum_{l=1}^K \pi_l \cdot \frac{1}{\sqrt{(2\pi)^c \sigma_l^c}} \exp\left(-\frac{\|\mathbf{x}_j - \boldsymbol{\mu}_l\|^2}{2\sigma_l}\right) \right) \right] \\ &= -\frac{\frac{\partial}{\partial \sigma_k} \cdot \left[\sum_{l=1}^K \pi_l \cdot \frac{1}{\sqrt{(2\pi)^c \sigma_l^c}} \exp\left(-\frac{\|\mathbf{x}_j - \boldsymbol{\mu}_l\|^2}{2\sigma_l}\right) \right]}{\sum_{l=1}^K \pi_l \cdot \frac{1}{\sqrt{(2\pi)^c \sigma_l^c}} \exp\left(-\frac{\|\mathbf{x}_j - \boldsymbol{\mu}_l\|^2}{2\sigma_l}\right)} \\ &= -\frac{\pi_k \cdot \frac{1}{\sqrt{(2\pi)^c}} \cdot \frac{\partial}{\partial \sigma_k} \left[\frac{1}{\sqrt{\sigma_k^c}} \cdot \exp\left(-\frac{\|\mathbf{x}_j - \boldsymbol{\mu}_k\|^2}{2\sigma_k}\right) \right]}{\sum_{l=1}^K \pi_l \cdot \frac{1}{\sqrt{(2\pi)^c \sigma_l^c}} \exp\left(-\frac{\|\mathbf{x}_j - \boldsymbol{\mu}_l\|^2}{2\sigma_l}\right)}. \end{aligned} \tag{B.23}$$

Let $A = \pi_k \cdot \frac{1}{\sqrt{(2\pi)^c}}$, and $B = \sum_{l=1}^K \pi_l \cdot \frac{1}{\sqrt{(2\pi)^c \sigma_l^c}} \exp\left(-\frac{\|\mathbf{x}_j - \boldsymbol{\mu}_l\|^2}{2\sigma_l}\right)$.

Thus,

$$\frac{\partial \mathcal{L}_j}{\partial \sigma_k} = -\frac{A \cdot \frac{\partial}{\partial \sigma_k} \left[\frac{1}{\sqrt{\sigma_k^c}} \cdot \exp\left(-\frac{\|\mathbf{x}_j - \boldsymbol{\mu}_k\|^2}{2\sigma_k}\right) \right]}{B}. \tag{B.24}$$

We also let

$$\begin{aligned} C &= \frac{\partial}{\partial \sigma_k} \left[\frac{1}{\sqrt{\sigma_k^c}} \cdot \exp\left(-\frac{\|\mathbf{x}_j - \boldsymbol{\mu}_k\|^2}{2\sigma_k}\right) \right] \\ &= \frac{\partial}{\partial \sigma_k} \left[\frac{\exp\left(-\frac{\|\mathbf{x}_j - \boldsymbol{\mu}_k\|^2}{2\sigma_k}\right)}{\sqrt{\sigma_k^c}} \right]. \end{aligned} \tag{B.25}$$

Thus,

$$\begin{aligned}
 C &= \frac{\left[\exp\left(-\frac{\|\mathbf{x}_j - \boldsymbol{\mu}_k\|^2}{2\sigma_k}\right) \cdot \left(-\frac{\|\mathbf{x}_j - \boldsymbol{\mu}_k\|^2}{2}\right) \cdot \left(-\frac{1}{\sigma_k^2}\right) \cdot \sqrt{\sigma_k^c}\right]}{-\left[\frac{c}{2} \cdot \sigma_k^{\frac{c}{2}-1} \cdot \exp\left(-\frac{\|\mathbf{x}_j - \boldsymbol{\mu}_k\|^2}{2\sigma_k}\right)\right]} \\
 &= \frac{1}{\sigma_k^c} \cdot \exp\left(-\frac{\|\mathbf{x}_j - \boldsymbol{\mu}_k\|^2}{2\sigma_k}\right) \cdot \left[\frac{\|\mathbf{x}_j - \boldsymbol{\mu}_k\|^2}{2} \cdot \sigma_k^{\frac{c}{2}-2} - \frac{c}{2} \cdot \sigma_k^{\frac{c}{2}-1}\right] \\
 &= \frac{1}{\sqrt{\sigma_k^c}} \cdot \exp\left(-\frac{\|\mathbf{x}_j - \boldsymbol{\mu}_k\|^2}{2\sigma_k}\right) \cdot \left[\frac{\|\mathbf{x}_j - \boldsymbol{\mu}_k\|^2}{2\sigma_k^2} - \frac{c}{2\sigma_k}\right]. \tag{B.26}
 \end{aligned}$$

So plugging in A , B and C , we get

$$\begin{aligned}
 \frac{\partial \mathcal{L}_j}{\partial \sigma_k} &= -\frac{\pi_k \cdot \frac{1}{\sqrt{(2\pi)^c \sigma_k^c}} \cdot \exp\left(-\frac{\|\mathbf{x}_j - \boldsymbol{\mu}_k\|^2}{2\sigma_k}\right)}{\sum_{l=1}^K \pi_l \cdot \frac{1}{\sqrt{(2\pi)^c \sigma_l^c}} \exp\left(-\frac{\|\mathbf{x}_j - \boldsymbol{\mu}_l\|^2}{2\sigma_l}\right)} \cdot \left[\frac{\|\mathbf{x}_j - \boldsymbol{\mu}_k\|^2}{2\sigma_k^2} - \frac{c}{2\sigma_k}\right] \\
 &= \Pi_k \cdot \left[\frac{c}{2\sigma_k} - \frac{\|\mathbf{x}_j - \boldsymbol{\mu}_k\|^2}{2\sigma_k^2}\right]. \tag{B.27}
 \end{aligned}$$

We also have

$$\frac{\partial \sigma_k}{\partial \hat{\sigma}_k} = \frac{\bar{\sigma}_{\max} - \bar{\sigma}_{\min}}{255}. \tag{B.28}$$

And,

$$\frac{\partial \hat{\sigma}_k}{\partial \mathbf{y}_\sigma^k} = \begin{cases} 0.2 & \text{if } -2.5 \leq \mathbf{y}_\sigma^k \leq 2.5, \\ 0 & \text{otherwise.} \end{cases} \tag{B.29}$$

Thus,

$$\begin{aligned}
 \frac{\partial \mathcal{L}_j}{\partial \mathbf{y}_\sigma^k} &= \frac{\partial \mathcal{L}_j}{\partial \sigma_k} \cdot \frac{\partial \sigma_k}{\partial \hat{\sigma}_k} \cdot \frac{\partial \hat{\sigma}_k}{\partial \mathbf{y}_\sigma^k} \\
 &= \frac{0.2(\bar{\sigma}_{\max} - \bar{\sigma}_{\min})}{255} \Pi_k \cdot \left[\frac{c}{2\sigma_k} - \frac{\|\mathbf{x}_j - \boldsymbol{\mu}_k\|^2}{2\sigma_k^2}\right] \tag{B.30}
 \end{aligned}$$

for $-2.5 < \mathbf{y}_\sigma^{(k)} < 2.5$. This is because of the piece-wise property in the definition of the hard-sigmoid activation function.

Finally, for $\mathbf{y}_\mu^{(k)}$, let $\mu_{k,l}^{(i)}$ be the l th element of the mean vector where l is an integer lies in $[0, c)$ and suppose that $\mu_{k,l}^{(i)}$ corresponds to an output o_k^μ of the network. We can get derivative of $\mu_{k,l}^{(i)}$ by taking Eqs. (B.3), (B.9), (B.14), (B.15) into the

differentiation process:

$$\begin{aligned}
 \frac{\partial \mathcal{L}_j}{\partial \boldsymbol{\mu}_k} &= \frac{\partial}{\partial \boldsymbol{\mu}_k} \cdot \left[-\ln \left(\sum_{l=1}^K \pi_l \cdot \frac{1}{\sqrt{(2\pi)^c \sigma_l^c}} \exp\left(-\frac{\|\mathbf{x}_j - \boldsymbol{\mu}_l\|^2}{2\sigma_l}\right) \right) \right] \\
 &= -\frac{\frac{\partial}{\partial \boldsymbol{\mu}_k} \cdot \left[\sum_{l=1}^K \pi_l \cdot \frac{1}{\sqrt{(2\pi)^c \sigma_l^c}} \exp\left(-\frac{\|\mathbf{x}_j - \boldsymbol{\mu}_l\|^2}{2\sigma_l}\right) \right]}{\sum_{l=1}^K \pi_l \cdot \frac{1}{\sqrt{(2\pi)^c \sigma_l^c}} \exp\left(-\frac{\|\mathbf{x}_j - \boldsymbol{\mu}_l\|^2}{2\sigma_l}\right)} \\
 &= -\frac{\pi_k \cdot \frac{1}{\sqrt{(2\pi)^c \sigma_k^c}} \cdot \frac{\partial}{\partial \boldsymbol{\mu}_k} \left[\exp\left(-\frac{\|\mathbf{x}_j - \boldsymbol{\mu}_k\|^2}{2\sigma_k}\right) \right]}{\sum_{l=1}^K \pi_l \cdot \frac{1}{\sqrt{(2\pi)^c \sigma_l^c}} \exp\left(-\frac{\|\mathbf{x}_j - \boldsymbol{\mu}_l\|^2}{2\sigma_l}\right)} \\
 &= -\frac{\pi_k \cdot \frac{1}{\sqrt{(2\pi)^c \sigma_k^c}} \cdot \exp\left(-\frac{\|\mathbf{x}_j - \boldsymbol{\mu}_k\|^2}{2\sigma_k}\right)}{\sum_{l=1}^K \pi_l \cdot \frac{1}{\sqrt{(2\pi)^c \sigma_l^c}} \exp\left(-\frac{\|\mathbf{x}_j - \boldsymbol{\mu}_l\|^2}{2\sigma_l}\right)} \cdot \frac{\partial}{\partial \boldsymbol{\mu}_k} \left[-\frac{\|\mathbf{x}_j - \boldsymbol{\mu}_k\|^2}{2\sigma_k} \right]. \tag{B.31}
 \end{aligned}$$

Then, we get

$$\begin{aligned}
 \frac{\partial \mathcal{L}_j}{\partial \boldsymbol{\mu}_k} &= \frac{\Pi_k}{2\sigma_k} \cdot \frac{\partial}{\partial \boldsymbol{\mu}_k} \left[[\mathbf{x}_j - \boldsymbol{\mu}_k]^T [\mathbf{x}_j - \boldsymbol{\mu}_k] \right] \\
 &= -\frac{\Pi_k}{\sigma_k} \cdot [\mathbf{x}_j - \boldsymbol{\mu}_k]. \tag{B.32}
 \end{aligned}$$

For data at each color channel l , we have


$$\frac{\partial \mu_{k,l}}{\partial y_{\mu}^{k,l}} = \begin{cases} 0.2 & \text{if } -2.5 \leq y_{\mu}^{k,l} \leq 2.5, \\ 0 & \text{otherwise.} \end{cases} \tag{B.33}$$


Thus, for $-2.5 < y_{\mu}^{k,l} < 2.5$,


$$\frac{\partial \mathcal{L}_j}{\partial y_{\mu}^{k,l}} = \frac{\partial \mathcal{L}_j}{\partial \mu_{k,l}} \cdot \frac{\partial \mu_{k,l}}{\partial y_{\mu}^{k,l}} = -0.2 \times \Pi_k \cdot \left[\frac{x_{j,l} - \mu_{k,l}}{\sigma_k} \right]. \tag{B.34}$$


From Eqs. (B.22), (B.34), and (B.31), when CDA-GM is performed data-driven learning individually on each video sequence using Adam optimizer with a learning rate of α , the process tries to regulate the values of latent parameters in the mixture model via minimizing the negative of log likelihood function.

ORCID

Synh Viet-Uyen Ha  <https://orcid.org/0000-0002-5056-8337>

Tien-Cuong Nguyen  <https://orcid.org/0000-0001-9084-8977>

Hung Ngoc Phan  <https://orcid.org/0000-0003-1909-1793>

Phuong Hoai Ha  <https://orcid.org/0000-0001-8366-5590>

References

1. S. Zhang, H. Zhou, D. Xu, M. E. Celebi and T. Bouwmans, Introduction to the special issue on multimodal machine learning for human behavior analysis, *ACM Trans. Multimed. Comput. Commun. Appl.* **16** (2020) 1–2.
2. L. Li and K. Wang, Research on automatic recognition method of basketball shooting action based on background subtraction method, *Int. J. Biom.* **14**(3/4) (2022) 318–335.
3. S. Ammar, T. Bouwmans, N. Zaghden and M. Neji, Deep detector classifier (DeepDC) for moving objects segmentation and classification in video surveillance, *IET Image Process.* **14** (2020) 1490–1501.
4. M. M. Ata, M. El-Dariby, M. M. A. El-nabi and S. A. Napoleon, Using modified background subtraction for detecting vehicles in videos, *Int. J. Adv. Intell. Paradig.* **22**(1/2) (2022) 21–36.
5. S. R. Sanches, C. Oliveira, A. C. Sementille and V. Freire, Challenging situations for background subtraction algorithms, *Appl. Intell.* **49** (2019) 1771–1784.
6. B. Garcia-Garcia, T. Bouwmans and A. J. Rosales Silva, Background subtraction in real applications: Challenges, current models and future directions, *Comput. Sci. Rev.* **35** (2020) 100204.
7. C. Stauffer and W. E. L. Grimson, Adaptive background mixture models for real-time tracking, in *Proc. 1999 IEEE Computer Society Conf. Computer Vision and Pattern Recognition (Cat. No. PR00149)*, Fort Collins, CO, USA, Vol. **2** (IEEE, 1999), pp. 246–252.
8. Z. Zivkovic, Improved adaptive Gaussian mixture model for background subtraction, in *Proc. 17th Int. Conf. Pattern Recognition, 2004, ICPR 2004*, Cambridge, UK, Vol. **2** (IEEE, 2004), pp. 28–31.
9. I. Martins, P. Carvalho, L. Corte-Real and J. L. Alba-Castro, BMOG: Boosted Gaussian mixture model with controlled complexity for background subtraction, *Pattern Anal. Appl.* **21** (2018) 641–654.
10. S. V.-U. Ha, N. M. Chung, H. N. Phan and C. T. Nguyen, TensorMoG: A tensor-driven Gaussian mixture model with dynamic scene adaptation for background modelling, *Sensors* **20** (2020) 6973.
11. R. Kalsotra and S. Arora, A comprehensive survey of video datasets for background subtraction, *IEEE Access* **7** (2019) 59143–59171.
12. L. A. Lim and H. Yalim Keles, Foreground segmentation using convolutional neural networks for multiscale feature encoding, *Pattern Recognit. Lett.* **112** (2018) 256–262.
13. L. A. Lim and H. Y. Keles, Learning multi-scale features for foreground segmentation, *Pattern Anal. Appl.* **23** (2020) 1369–1380.
14. J. Wang, C. Lan, C. Liu, Y. Ouyang, T. Qin, W. Lu, Y. Chen, W. Zeng and P. Yu, Generalizing to unseen domains: A survey on domain generalization, *IEEE Trans. Knowl. Data Eng.* **35** (2022) 8052–8072.
15. T. Bouwmans, S. Javed, M. Sultana and S. K. Jung, Deep neural network concepts for background subtraction: A systematic review and comparative evaluation, *Neural Netw.* **117** (2019) 8–66.
16. G. Hinton, O. Vinyals and J. Dean, Distilling the knowledge in a neural network, *CoRR*. **abs/1503.02531** (2015), <http://arxiv.org/abs/1503.02531>.
17. C. M. Bishop, Mixture density networks (Aston University, 1994).
18. T. Bouwmans, Traditional approaches in background modeling for static cameras, in *Background Modeling and Foreground Detection for Video Surveillance* (CRC Press, Taylor and Francis Group, 2014), pp. 1–632.

19. S. V. Ha, C. T. Nguyen, H. N. Phan, N. M. Chung and P. H. Ha, CDN-MEDAL: Two-stage density and difference approximation framework for motion analysis, preprint (2021), arXiv:2106.03776.
20. P. Shi, E. Jones and Q. Zhu, Median model for background subtraction in intelligent transportation system, *Proc. SPIE — Int. Soc. Opt. Eng.* **5298** (2004) 168–176.
21. J. Zheng, Y. Wang, N. L. Nihan and M. E. Hallenbeck, Extracting roadway background image: Mode-based approach, *Transp. Res Rec.* **1944**(1) (2006) 82–88.
22. K. Kim, T. Chalidabhongse, D. Harwood and L. Davis, Background modeling and subtraction by codebook construction, in *Proc. — Int. Conf. Image Processing, ICIP 2004*, Singapore, 2004, Vol. **5**, pp. 3061–3064.
23. Y. Zhao and W. Liu, Fast robust foreground–background segmentation based on variable rate codebook method in Bayesian framework for detecting objects of interest, in *Proc. — 2014 7th Int. Congr. Image and Signal Processing, CISP 2014*, Dalian, China, pp. 55–59.
24. S. Maity, A. Chakrabarti and D. Bhattacharjee, Block-based quantized histogram (BBQH) for efficient background modeling and foreground extraction in video, in *2017 Int. Conf. Data Management, Analytics and Innovation (ICDMAI)*, Pune, India (IEEE, 2017), pp. 224–229.
25. S. Jiang and X. Lu, WeSamBE: A weight-sample-based method for background subtraction, *IEEE Trans. Circuits Syst. Video Technol.* **28** (2018) 2105–2115.
26. S. Agrawal and P. Natu, ABGS segmenter: Pixel wise adaptive background subtraction and intensity ratio based shadow removal approach for moving object detection, *J. Supercomput.* **79**(7) (2023) 7937–7969.
27. J. D. Pulgarin-Giraldo, A. Alvarez-Meza, D. Insuasti-Ceballos, T. Bouwmans and G. Castellanos-Dominguez, Progress in pattern recognition, image analysis, computer vision, and applications, in *GMM background modeling using divergence-based weight updating* (Springer International Publishing, 2017), pp. 282–290.
28. S. Viet-Uyen Ha, D. Nguyen-Ngoc Tran, T. P. Nguyen and S. Vu-Truong Dao, High variation removal for background subtraction in traffic surveillance systems, *IET Comput. Vis.* **12** (2018) 1163–1170.
29. C. Zhao, A. Sain, Y. Qu, Y. Ge and H. Hu, Background subtraction based on integration of alternative cues in freely moving camera, *IEEE Trans. Circuits Syst. Video Technol.* **29** (2019) 1933–1945.
30. Y. Lecun, L. Bottou, Y. Bengio and P. Haffner, Gradient-based learning applied to document recognition, *Proc. IEEE* **86**(11) (1998) 2278–2324.
31. M. Braham and M. Van Droogenbroeck, Deep background subtraction with scene-specific convolutional neural networks, in *2016 Int. Conf. Systems, Signals and Image Processing (IWSSIP)*, Bratislava, Slovakia (IEEE, 2016), pp. 1–4.
32. Y. Wang, Z. Luo and P.-M. Jodoin, Interactive deep learning method for segmenting moving objects, *Pattern Recognit. Lett.* **96** (2017) 66–75.
33. K. Lim, W.-D. Jang and C.-S. Kim, Background subtraction using encoder–decoder structured convolutional neural network, in *2017 14th IEEE Int. Conf. Advanced Video and Signal Based Surveillance (AVSS)*, Lecce, Italy (IEEE, 2017), pp. 1–6.
34. K. Simonyan and A. Zisserman, Very deep convolutional networks for large-scale image recognition, in *3rd Int. Conf. Learning Representations, ICLR 2015 — Conf. Track Proc.* (San Diego, CA, USA, 2015), pp. 1–14.
35. M. Babaei, D. T. Dinh and G. Rigoll, A deep convolutional neural network for video sequence background subtraction, *Pattern Recognit.* **76** (2018) 635–649.
36. P.-L. St-Charles, G.-A. Bilodeau and R. Bergevin, SuBSENSE: A universal change detection method with local adaptive sensitivity, *IEEE Trans. Image Process.* **24** (2015) 359–373.

37. R. Wang, F. Bunyak, G. Seetharaman and K. Palaniappan, Static and moving object detection using flux tensor with split Gaussian models, in *2014 IEEE Conf. Computer Vision and Pattern Recognition Workshops*, Columbus, OH, USA (IEEE, 2014), pp. 420–424.
38. T. P. Nguyen, C. C. Pham, S. V.-U. Ha and J. W. Jeon, Change detection by training a triplet network for motion feature extraction, *IEEE Trans. Circuits Syst. Video Technol.* **29** (2019) 433–446.
39. Y. Chen, J. Wang, B. Zhu, M. Tang and H. Lu, Pixelwise deep sequence learning for moving object detection, *IEEE Trans. Circuits Syst. Video Technol.* **29** (2019) 2567–2579.
40. Y. Yang, J. Ruan, Y. Zhang, X. Cheng, Z. Zhang and G.-J. Xie, STPnet: A spatial-temporal propagation network for background subtraction, *IEEE Trans. Circuits Syst. Video Technol.* **32** (2022) 2145–2157.
41. I. Houhou, A. Zitouni, Y. Ruichek, S. E. Bekhouche, M. Kas and A. Taleb-Ahmed, RGBD deep multi-scale network for background subtraction, *Int. J. Multimed. Inf. Retr.* **11**(3) (2022) 395–407.
42. F. Gouizi and A. C. Megherbi, Nested-Net: A deep nested network for background subtraction, *Int. J. Multimed. Inf. Retr.* **12**(1) (2023) 5.
43. M. Courbariaux, Y. Bengio and J.-P. David, BinaryConnect: Training deep neural networks with binary weights during propagations, in *Proc. 28th Int. Conf. Neural Information Processing Systems — Vol. 2, NIPS’15* (MIT Press, Cambridge, MA, USA, 2015), pp. 3123–3131.
44. K. Hornik, M. Stinchcombe and H. White, Multilayer feedforward networks are universal approximators, *Neural Netw.* **2** (1989) 359–366.
45. A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto and H. Adam, MobileNets: Efficient convolutional neural networks for mobile vision applications, *CoRR* **abs/1704.04861** (2017), <http://arxiv.org/abs/1704.04861>.
46. D. Ulyanov, A. Vedaldi and V. Lempitsky, Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis, in *2017 IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, Honolulu, Hawaii, USA (IEEE, 2017), pp. 4105–4113.
47. P.-L. St-Charles, G.-A. Bilodeau and R. Bergevin, A self-adjusting approach to change detection based on background word consensus, in *2015 IEEE Winter Conf. Applications of Computer Vision*, Waikoloa, HI, USA (IEEE, 2015), pp. 990–997.
48. S. Işık, K. Özkan, S. Günal and Ö. N. Gerek, SWCD: A sliding window and self-regulated learning-based background updating method for change detection in videos, *J. Electron. Imag.* **27**(2) (2018) 1–11.
49. D. Liang, J. Pan, H. Sun and H. Zhou, Spatio-temporal attention model for foreground detection in cross-scene surveillance videos, *Sensors* **19** (2019) 5142.
50. Y. Wang, P.-M. Jodoin, F. Porikli, J. Konrad, Y. Benezeth and P. Ishwar, CDnet 2014: An expanded change detection benchmark dataset, in *2014 IEEE Conf. Computer Vision and Pattern Recognition Workshops*, Columbus, Ohio, USA (IEEE, 2014), pp. 393–400.
51. P.-M. Jodoin, L. Maddalena, A. Petrosino and Y. Wang, Extensive benchmark and survey of modeling methods for scene background initialization, *IEEE Trans. Image Process.* **26**(11) (2017) 5244–5256.
52. B. Wu, Y. Liu, B. Lang and L. Huang, DGCNN: Disordered graph convolutional neural network based on the Gaussian mixture model, *Neurocomputing* **321** (2018) 346–356.
53. Y. L. Tong, *The Multivariate Normal Distribution*, 1st edn. (Springer, 1990).