Deep Learning-Based Improved Object Recognition in Warehouses

Syeda Fouzia[1(✉)], Mark Bell[2], and Reinhard Klette[1]

[1] School of Electrical and Computer Engineering,
Auckland University of Technology, Auckland, New Zealand
syeda.fouzia@aut.ac.nz
[2] Crown Lift Trucks Ltd., Auckland, New Zealand
mark.bell@crown.com

Abstract. Research migrates in recent years from model-based object detection and classification to data-driven approaches. With the efficiency improvement of computational resources, improved acquisition systems, and bulks of data for training, deep learning models have found their way to accurate object category classification. Deep convolution nets have an inherent ability to extract features automatically and are used for accurate category classification. This paper has three parts. First, we extract moving foregrounds by using a mixture-of-Gaussians technique. Next, we aim at improving the quality of object foreground based on a pixel saliency map. Third, the obtained improved foreground is assigned labels using a pre-trained deep learning detector. Altogether, the paper proposes a way for improved video-based object detection and classification for logistics in warehouses.

1 Introduction

Various model-based methods for object detection have been proposed, such as feature-based [20], appearance-based [8], or motion-based [37]. A selection of the best technique for any specified application is relative; it depends on the extent of hardware resources or the scope of the targeted task.

Much of the progress achieved for detecting and classifying objects of interest is made by the development of robust image descriptors such as SIFT [20] or hand-crafted low-level features such as *histogram of oriented gradients* (HOG) [6], bag-of-features representations [4], or deformable part models [8], feature pooling [22], classic classifiers such as Support Vector Machine (SVM) [5] and random forests [2].

Recently, due to efficient computational resources and ease of data availability, data driven approaches have found their way. Deep learning is a form of representation learning. A computer is fed with large amounts of raw data and it finds out the features needed for detection, based on learning [25,26]. Deep convolutional neural nets, proposed by Krizhevsky et al. [17], have achieved tremendous success on bigger benchmark datasets, such as ImageNet. ImageNet

M. Paul et al. (Eds.): PSIVT 2017, LNCS 10749, pp. 350–365, 2018.
https://doi.org/10.1007/978-3-319-75786-5_29

is a dataset of over 15 million labeled high-resolution images belonging to roughly 22,000 categories. It took between five and six days on two GTX 580 3 GB GPUs to train a network with ImageNet dataset. Some modern object recognition models [18,19,32] have millions of parameters and may take some weeks to be fully trained. Hence, traditional deep learning models need a huge amount of training data for training and resources, such as multiple GPUs.

The reported research is motivated by tasks of improved object detection and classification. We also aim at using deep learning for improved recognition accuracy. We target our research towards logistics handling warehouses in our case. As per our observations, these particular indoor scenes come with the following environmental challenges:

1. There are multiple moving objects. We have recordings with pedestrians and forklift trucks moving inside a warehouse.
2. Color contrast between background and foreground is very marginal, most of the time.
3. Multiple occlusions are likely; the environment is semi-cluttered.
4. There are parked forklifts (stationery objects) in some areas.
5. Changes in loads occur frequently for racks in the background; typically these changes are gradual.
6. Illumination changes are also gradual. Warehouse indoor data have usually only a few low-illumination areas.
7. There are entries and exits of vehicles into a scene.
8. People are considered to be part of the background if they are static or moving only slightly.

A typical warehouse is a busy place. With many industrial pick and pack processes going on, we need accurate localization of occluded moving targets, to be used for precise visual surveillance. Also, moving foreground extraction is an essential requirement for object recognition surveillance tasks in computer vision. Thus, we need an indoor adaptive algorithm which can handle lighting changes, repetitive motions from clutter, and long-term scene changes inside a warehouse. See Fig. 1 for few typical warehouse scenes from our recordings.

First, we selected background subtraction, which is well suited for moving targets as in our case. We follow [27] where each pixel value is modelled as a *mixture of Gaussians* (MOG). By this means we can determine whether or not a pixel is part of the background. This supports an effective approach for separating background from foreground.

Second, we need to improve extracted foregrounds as they are not yet accurate. See Figs. 2, 3, 4, 5, 6 and 7. Due to a low background pixel recovery rate and a slow adaptation to scene changes when using the traditional MOG algorithm, foreground quality is not yet fair. We extracted salient pixels using a local contrast method [36], based on a visual saliency map. A pixel-wise saliency map, for each frame, is used to improve the corresponding foreground obtained by the MOG background extraction.

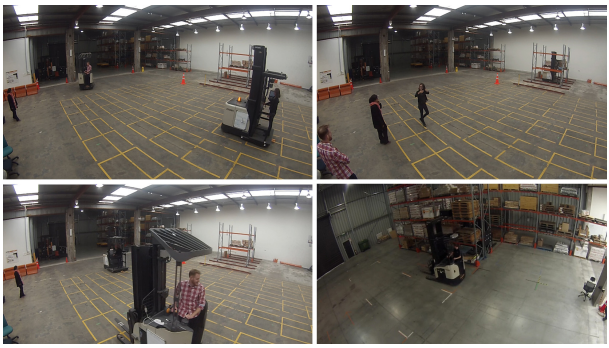

Fig. 1. Warehouse scenes from recorded videos

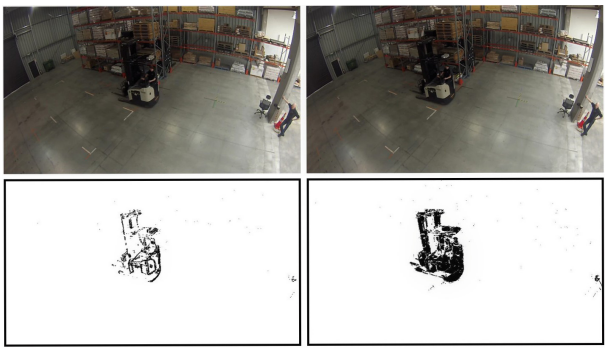

Fig. 2. Forklift crossing low illumination area. Foreground results in bottom left and right images, after applying MOG. The pedestrians at the right are detected poorly due to a more static posture

Fig. 3. A forklift crossing low illumination area. Frame 15 and Frame 20 foreground results in bottom left and right images, after applying MOG. Very slow recovery of background pixels have made the area of forklift foreground bigger than actual

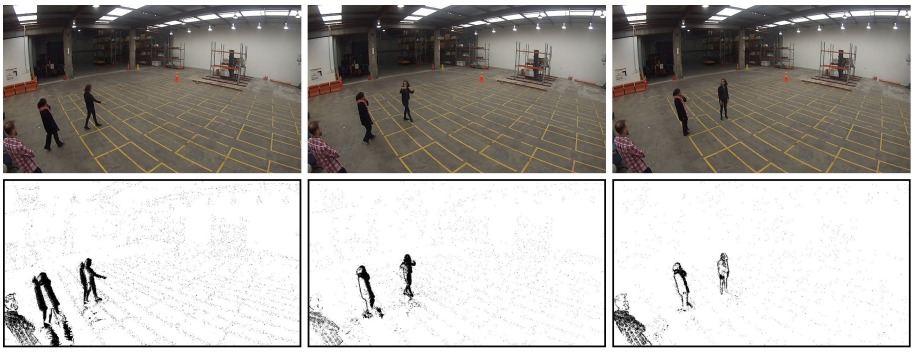

Fig. 4. Scenario of three pedestrians standing inside warehouse. Starting from bottom extreme left, foreground is somewhat distorted. Some foreground pixels from previous frames are still there. For bottom middle, one of the pedestrian, who is more static, is poorly detected in foreground

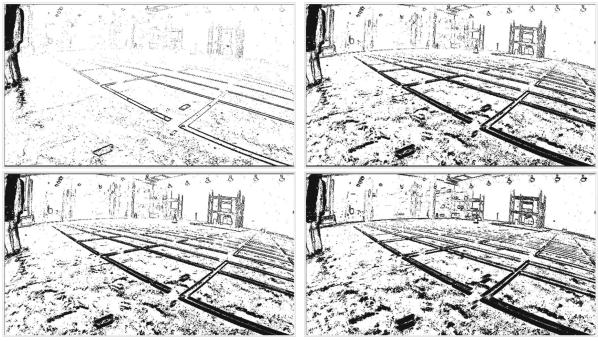

Fig. 5. MOG results in a sudden camera movement scenario. A pedestrian is detected at the left, standing static in a warehouse scene. Due to variance changes in pixels due to camera motion, most of the background pixels appeared as foreground pixels. See upper right, bottom right, and left images

Third, we want to label the detected foreground. We can repurpose features, extracted from a pre-trained deep convolution neural network, for new object category recognition specific to our application [7]. This technique is called *transfer learning*. We transferred the learned features from a pre-trained model (i.e. *Google Inception Model*) for new category classification which are forklifts and pedestrians in a typical warehouse scenario [17,29].

The structure of the paper is as follows. Section 3 reports about the first step of foreground extraction. Section 4 explains the MOG- based foreground quality improvement achieved by computing pixel saliency map. Section 5 illustrates the use of a pre-trained deep-learning architecture model, for foreground category recognition. Section 6 concludes.

Fig. 6. A scenario in which three pedestrians and a forklift are detected. One of the three pedestrians is not detected in the bottom right-most result. This pedestrian was having a slightly static posture, so became part of the background pixels

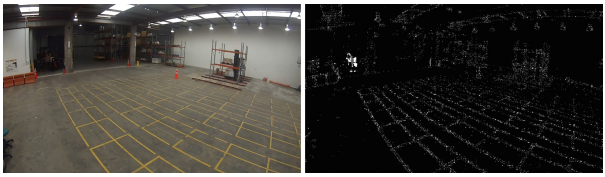

Fig. 7. MOG results for occluded pedestrians. Two occluded pedestrians shown with clustered white pixels

2 Related Work

Deep convolution architectures, employing automatic feature learning and classification, are being researched in recent years. Out of many options, *convolutional neural networks* (CNNs), *region-based CNNs* (R-CNNs), or later versions are frequently explored. An R-CNN is a three-stage pipeline process. Features are extracted for every object proposal in an image and are being cached. A *support vector machine* (SVM) is used as object detector, replacing the *softmax classifier*. In the third stage of training, *bbox regressors* are learned [10]. *Spatial pyramid pooling networks* (SPPnets) were introduced to speed up R-CNNs by sharing the computation burden [14]. This is also a multi-stage process which computes a convolutional feature map for the whole input image and then classifies each object proposal, using a feature vector extracted from the shared feature map.

Fast R-CNNs use single-stage training and a multi-task loss for better detection accuracy and speed. Training can update all the layers of the network at once, and no feature caching is required [11]. These nets still used selective search for region generation; now removed in *faster R-CNNs* came. A cost-free *region proposal network* (RPN) was employed which predicted potential object bounds and an object score at each position in the faster R-CNN. This RPN, integrated with fast RCNN, was trained to share features across layers [24].

Inspired by this work we propose a pipeline for object detection. We use the Gaussian mixture model for possible objectness search in warehouse scenes and improve the foreground quality based on pixel saliency. Once achieved, we use a pre-trained CNN architecture for category label assignment for forklifts and pedestrians.

3 Gaussian Mixture Model for Foreground Extraction

The Gaussian mixture model is a natural choice for our analysis (i.e. for extracting moving targets out of a mostly stationery background). Mixture models are probabilistic models which assume that underlying pixels belong to a particular mixture distribution. To make the model more robust to lighting variations, and to handle multiple surfaces occurring in the view frustum of particular pixels, the mixture models need to be adaptive.

Basics of Gaussian Mixture Models. The values of particular pixels are modelled as a *mixture of adaptive Gaussian distributions*. A pixel process of a pixel (x, y, X) has the history of its previous t values, say from Frame 1 to Frame t. This can be represented by the set $\{X_1, \ldots, X_i, \ldots, X_t\}$, where $1 \leq i \leq t$.

The probability density function of the univariate Gaussian or normal distribution is given by

$$G(X, \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left\{-\frac{1}{2}\left(\frac{X-\mu}{\sigma}\right)^2\right\} \tag{1}$$

for $-\infty < X < \infty$, where μ is the mean and $\sigma^2 > 0$ is the variance.

The probability of observing a specific mixture component at Frame t is given by products of *probability density functions* with their weight. For more than one density function, we have a multivariate case, such as

$$P(X_t) = \sum_{i=1}^{k} w_{i,t} \cdot G(X_t, \mu_{i,t}, \sigma_{i,t}) \tag{2}$$

Here, $w_{i,t}$ is the weight of the i_{th} Gaussian distribution at Frame t.

Aiming at a probabilistic model for separating the *background pixels* from the *foreground* by looking on the distributions, [27] proposed an update of the background model as follows:

(1) *Constructing adaptive mixture of multi-modal Gaussians per pixel.* The number of Gaussian components depends on the environmental complexity one wants to model. In a typical warehouse indoor environment, we observed relatively minor contrast in colors and brightness. Outdoor scenes have different conditions. Following [27], we also keep $k = 3$. Targeting RGB images, we also assume that all three RGB channels have the same σ^2, thus defining a 3×3 covariance matrix $\Sigma_{i,t}$ being the product of a variance with the unit matrix.

(2) *Method for updating the Gaussian parameters.* For every new pixel state for the next frame, we check whether it lies $X_t \leq 2.5$ standard deviations from the mean; we label it *matched* in this case. We update weight, mean, and variance as per the following update equations:

$$w_{i,t} = (1-\alpha) \cdot w_{i,t-1} + \alpha \cdot M_{i,t} \tag{3}$$

$$\mu_{i,t} = (1-\rho) \cdot \mu_{i,t-1} + \rho \cdot X_t \tag{4}$$

$$\sigma^2_{i,t} = (1-\rho) \cdot \sigma^2_{i,t-1} + \rho(X_t - \mu_{i,t})^\top (X_t - \mu_{i,t}) \tag{5}$$

where $\rho = \alpha \cdot P(X_t|\mu_{i,t-1}, \Sigma_{i,t-1})$, $0 < \alpha < 1$ is a selected learning rate, and $M_{i,t}$ equals 1 for a model which is matched, and equals 0 for other models.

If the i^{th} Gaussian is marked as *unmatched*, we decrease its initial weight as per below equation:

$$w_{i,t} = (1-\alpha) \cdot w_{i,t-1} \tag{6}$$

If all the k Gaussians in the mixture model, for pixel value X_t, are not matched to the pixel, we mark that specific pixel as a *foreground pixel*. If this is the case, then we find the Gaussian distribution with the lowest weight in the mixture and set its mean equal to X_t. We also adjust the corresponding variance to a higher value, and lower the weight of this distribution.

(3) *Heuristics for determining the background.* For finding the background distributions, we rearrange the distributions in descending order by w/σ. We add up the corresponding weights of the Gaussians in this order, till the final sum is greater than a pre-set threshold T. We set $T = 0.9$ in our case. We observe that there are fewer salient or moving objects, and more background portions in the frames.

Recent Variants for Addressing Challenges. Gaussian mixture models have been an active field of research since two decades. Many variants have been introduced for dealing with various challenges when dealing with "real time moving target detection". *Shadow elimination* has been a major subject. Foregrounds obtained by an MOG technique have shadow pixels as part of the foreground. Much work has been targeted towards shadow elimination, for example [9,31,34,35]. Shadow detection in color space is considered in [9,16]. For the detection of *slowly moving objects*, see [9]. Challenges arise when these objects are incorporated into the background due to less variance.

Adaptation of algorithms to scene changes. This is very important and controlled to some extent by learning rate and parameter selection, see [12,16,37]. *Background recovery rate improvement* is also studied in [34] for solving real-time surveillance issues. To incorporate abandoned objects for surveillance applications, *abandoned/removed object detection* is also throughly researched in [31].

Update to learning equations. This involves controlling the scene changes and slowly or fast moving objects, see [12,21,37]. *Learning rates* and their significance for incorporating scene changes is studied in [33]. The *number of Gaussian*

components or modes depends on scene complexity and pixels modes [37]. For run-time improvements, to adapt MOG to real time, see [34,37]. *Initialization of parameters* is very important for initialization of an MOG model [34]. Parameter analysis and setting as per scenario is dealt with in [35].

Dirichlet-Gaussian distribution. [13] use a Dirichlet process and a Gaussian mixture model to estimate a per-pixel background distribution, which is followed by probabilistic regularization. This work was able to accurately model dynamic backgrounds.

Neighborhood correlation, to update the parameters of MOG [21], is also found effective. Importance of spatial information other than temporal one, for detecting accurate foregrounds [33,34], was able to improve the foreground quality. For using other cues such as intensity and texture, for better foregrounds, see [31]. This approach was not able to deal with resultant holes in foreground masks.

Qualitative Analysis for a Standard MOG Approach. As can be seen in the update equations above, α is the first learning rate. It needs to be adjusted as per the scenario conditions. To incorporate slowly moving objects and large homogeneously colored objects, we kept α small. For scenarios which are changing quickly, it needs to be larger to adapt to the scene. ρ is the second learning rate. Usually it is assigned a much smaller value than α. But, as per our trial experimentation, the use of the second learning rate increases the required computation time. Initial mean and variance are adjusted as per the scenario results. The thresholds are the same (value 0.9) for all the experiments.

We applied the mixture of Gaussian algorithm [27] with the following parameter setting: $\alpha = 0.001$ ranging to 0.79, $\rho = 0.00001$, threshold $T = 0.9$, the number of Gaussian components $k = 3$. We obtained our results by using Matlab 2017a.

For lower alpha values, slowly moving objects are detected with good quality foreground, but for higher values, results are not good for the same object. MOG cannot deal with sudden illumination changes and camera movements. This is as shown in Fig. 5. With passage of time, the variance decreases for more stable pixels. If the variance becomes too small, then even camera noise is marked as foreground pixel that effects the foreground quality. Bigger objects, uniform in color or slowly moving, are sometimes incorporated into background for a few frames. In conclusion, our extensive experiments, here illustrated by a few examples, lead to the conclusion: *We need some improvement in foreground detection, which makes it more robust to the mentioned challenges.*

4 Pixel Saliency for Foreground Improvement

Due to the stated observations above, we improved MOG-based foreground detection using a *saliency map-based foreground extraction scheme.*

We observed that pixel-based *saliency map* values, generated by the method of [36], can be useful for improving the average foreground quality obtained from the MOG method. It was computationally fast. It took 0.5 s or less per frame to compute a saliency map.

Visual saliency maps are able to mark salient pixels in the images and have good results with occluded objects in warehouse scenes. Different visual cues, such as compactness or uniqueness, are used to detect salient pixels in images [3]. Uniqueness-based methods are further split into *local* and *global* contrast methods. Most uniqueness-based methods use low-level features such as color, direction, or intensity to determine the contrast between image regions and their surrounding pixels.

Compactness-based methods use the variance of spatial features. Salient pixels tend to have a small spatial variance in the image space. The background is distributed on the whole image space and tends to have high spatial variance. Since single visual cue-based salient region detection methods have few limitations in detecting accurate salient pixels, different cues can be combined to make a composite framework [23]. Some methods are based on this approach, but the selection of visual cues depends on context.

Compared with the global contrast method, the local contrast is a relatively better cue to be combined with the compactness cue. Local contrast methods are able to identify the foreground region [15], but they have a limitation that they identify visible object boundaries rather than all the area. This effect can be minimized by propagation of saliency information based on diffusion [15].

To construct the pixel saliency map for the image, we converted it into a superpixel representation for constructing a resultant graph. We used SLIC [1] for an abstract graph representation of an image. Each superpixel, generated by SLIC, corresponds to some node. There are three parameters used in here: The number N of superpixel nodes used in SLIC, σ^2 which controls the fall-off rate of the exponential function, and α which balances the fitting constraints of manifold ranking algorithms. We experimentally set the parameters to $N = 200$, $\sigma^2 = 0.1$, and $\alpha = 0.99$ for experimentation. Next, the two saliency maps are computed based on the compactness visual cue and local contrast [36].

The resulting saliency maps are propagated using a diffusion process and the constructed graph later. Thus, a pixel-wise saliency map is generated from two computed maps. This pixel-wise saliency map for the specific frame is binary thresholded. We apply logical operations between salient binary thresholded pixels and moving pixels from MOG. Finally, some morphological processing is used to generate improved foreground masks.

See Figs. 8 and 9, for the improvement in foregrounds, as per the proposed improvement foreground strategy. It can be seen that foreground is better in quality with less redundant pixels from the background, as part of foreground. See Fig. 8. Fewer foreground holes are present in Fig. 9.

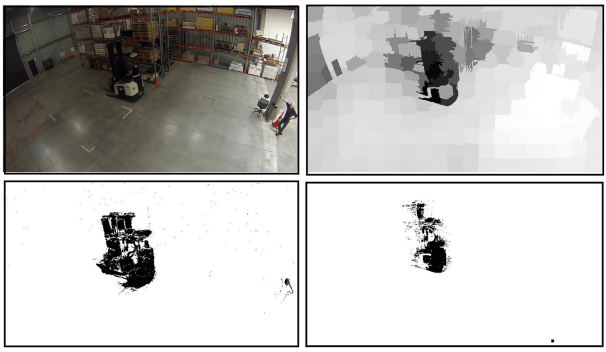

Fig. 8. Occluded forklift, foreground improvement based on pixel saliency, without morphological improvement

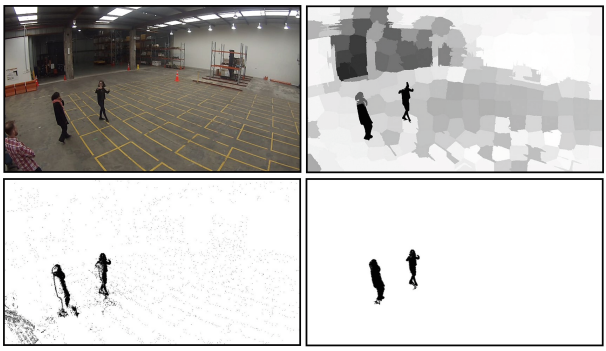

Fig. 9. Improved foreground based on pixel saliency. One of the pedestrians is missed, due to poor visibility both in MOG result and its saliency map

5 Deep Learning for Object Classification

After obtaining our improved foregrounds based on pixel saliency, we aim at classification. We use a pre-trained model of Google's Inception v3 [29] and re-train the top layer, for new categories.

We aim at overcoming the deficiency of the training data and limitations of computation time or resources by adapting a classifier, trained for other categories, to our dataset.

Basics and a Pre-trained Network. Deep *convolution neural nets* (CNNs) are able to learn rich feature representations. They have a reduced number of parameters and connections compared to the same-sized feed-forward networks. This characteristics make it easier to train and test them. They have successfully been used to categorize images and activities or tasks once trained with excess of data samples.

We can use a pre-trained convolutional deep learning model for classification tasks of new categories. We can retrain final or more layers of this model, and adapt it to our new categories and to the limited dataset available to us. This approach is called *transfer learning*. We select a pre-trained model architecture, replace the top layer by a new layer, and adapt the newly added layer to our own data classification task. We selected Google's Inception v3 Architecture model for moving object classification in warehouse scenes into two categories, either forklift or pedestrian.

Google's Inception v3 [29] is an architecture that provides a good-performance network with relatively low computational costs. To measure the classification accuracy, there are two main measurements used in the deep learning literature [29]; this is the top-5 error rate and the top-1 error rate. They measures the rate at which the architecture fails to include the correct class in the top-5 and the top-1 output, respectively.

Inception v3 achieved a 5.6% top-5 error rate, and a 21.2% top-1 error rate. Another famous architecture model, i.e., AlexNet [17] achieved a top-5 error rate of 15.3% and Inception (Google Net) [28] achieved 6.67% in the same category. Thus we selected the Inception v3 model architecture, pre-trained on ImageNet. We added a new Softmax and fully connected layer for training, and re-trained it in Tensor Flow 1.0 in the Ubuntu operating system. We re-train the model for classifying our two object categories. The top layer receives as input a 2,048-dimensional vector for each image. Since the Softmax layer contains two labels, this corresponds to learning 4,098 model parameters, corresponding to the learned biases and weights.

Training, Validation and Testing. We prepared our training data set containing 4,000 images, for two object categories of forklifts and pedestrians. We limited the training data to these two categories. Inception network rescales images to 299×299 size. So these are the input-width and input-height flags for the images. Most of the data we processed are from recorded video clips inside of a selected warehouse, showing different scenarios.Testing, validation percentages can be set by adjusting their flags in the script. We will use default values for these i.e. 80

First, a calculation of *transfer-values* is performed, for each of the images, arranged in training, testing, and validation sets. 'Transfer value' is the term we use for the output feature values, at the layer just before the final top layer. We used these feature values to differentiate the objects for new categories [7]. Since each image is reused many times during training and calculation, the transfer-values are being cached (stored on disk), to be reused repeatedly for training, validation, and testing [30]. Once the transfer value computation is complete, the actual training of the top layer of the network begins, for new labels, and for each image.

We have 4,000 training steps. Each step chooses ten images at random from the training set, finds their transfer values from the cache, and feeds them into the final layer to get predictions for the category. Those predictions are then compared against the actual labels, i.e. pedestrians or forklift, to update the top

Table 1. Results after 4,000 training iterations

Accuracy (in percent)	4,000 steps
Training	98
Validation	96
Cross entropy	0.12
Test	99.5

layer's weights through the *back-propagation process*. We obtain the following metrics for each training epoch.

The *training accuracy* shows what percentage of the images, we used in the present training batch, was labeled correctly with the true class.

The *validation accuracy* is the percentage on a randomly-selected set of images from a different data set, which are not present in the training set. If the training accuracy is very good, but the validation accuracy is not, that means that the network is *over fitting* to training data, and we need more training samples for network generalization.

The *cross entropy* is a loss function which evaluates how well the learning process for the model is executing. The training's objective is to make the loss as small as possible. If it is growing lower with each *epoch*, we assume that learning is progressing satisfactory.

As the process continues, we also observe an improvement in accuracy, i.e. in *test accuracy*, the evaluation which is run on a group of images which are kept separate from the training and validation images. This test accuracy is the best estimate of how well the trained model will work for a specified new classification task. This accuracy is based on the percentage of the images in the test set, that is given the correct label after the model is fully trained. We achieved very good testing accuracy of 99.5% after 4,000 training iterations.

Results. See below Table 1 for training, validation and test accuracy after 4,000 training iterations. Cross entropy is also listed.

We tested on images that the model has not been trained for, to check how does it generalize to unseen forklift and pedestrian images. See Figs. 10 and 11 for the category % assignment by the model. Figures illustrate our general finding that the model predicts the class with acceptable test accuracy for unseen images. For dependency upon number of training steps, see Table 2.

Hyper-Parameters. We have kept the hyper-parameters constant throughout training and testing, with learning rate 0.01. We have edited the hyper-parameter and *print misclassified test images*, to print the evaluation metrics for 20 training iterations for simplicity. These metrics are, true positive (TR), true negative (TN), false positive (FP), and false negative (FN). We define them as follows in our warehouse scenario: *TP* is the number of images categorized as forklifts in testing, *TN* is the number of images categorized as pedestrians in testing, *FP* is the number of images categorized as forklifts, although they were pedestrians in real, and *FN* is the number of images categorized as pedestrians, although they were forklifts in real.

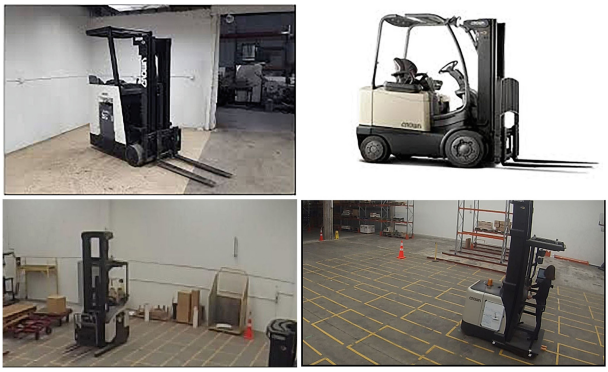

Fig. 10. Model test accuracy for forklift test images. Upper left: 99.92, upper right: 86.8, bottom left: 46.1, and bottom right: 71.76

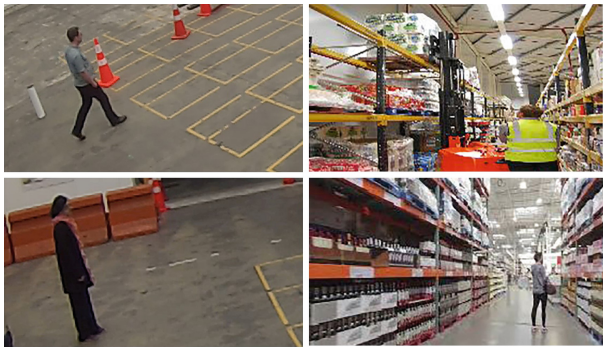

Fig. 11. Model test accuracy for pedestrian test images. Upper left: 86.7, upper right: 89.23, bottom left: 86.26, and bottom right: 99.5

Table 2. Evaluation metrics for 10 and 20 training steps

Evaluation metrics	10 steps	20 steps
True positive	197	200
False positive	1	2
True negative	217	217
False negative	1	8
Precision	99.5	99
Recall	94.7	96.2
f1	97	97.6
Accuracy	97.2	97.7

6 Conclusions

For detecting and classifying objects in warehouse scenes, we proposed a refinement of a standard MOG method by subsequent use of saliency maps, and the application of a CNN for the final step of object classification for the detected foreground segments.

Using the Inception pre-trained model, with only top-layer re-training, we achieved 99.5 classification accuracy for the specified task. The model is not able to generalize well for the test images of forklift models for which we have a relatively small number of training images only in our dataset. This can be seen in the bottom-left of Fig. 10 where a forklift is misclassified. Hence limitation in the current study is the availability of more training data. The current study is based on 4,000 labelled images.

We found that pixel saliency values have the potential to improve MOG-based foreground extraction. This also affects the learning rate selection for background or foreground pixels updates. We expect that this can help to further improve MOG-based real-time video surveillance in general.

References

1. Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., Sasstrunk, S.: SLIC superpixels compared to state-of-the-art superpixel methods. IEEE Trans. Pattern Anal. Mach. Intell. **34**(11), 2274–2282 (2012)
2. Breiman, L., Cutler, A.: Random Forests (2004)
3. Cheng, M.M., Mitra, N.J., Huang, X., Torr, P.H., Hu, S.M.: Global contrast based salient region detection. IEEE Trans. Pattern Anal. Mach. Intell. **37**(3), 569–582 (2015)
4. Csurka, G., et al.: Visual categorization with bags of keypoints. Workshop Stat. Learn. Comput. Vis. **1**, 1–22 (2004)
5. Chang, C.C., Lin, C.: LIBSVM: a library for support vector machines. ACM Trans. Intell. Syst. Technol. **2**(3), 27 (2011)
6. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. Proc. Comput. Vis. Pattern Recognit. **1**, 886–893 (2005)
7. Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., Darrell, T.: DeCAF: a deep convolutional activation feature for generic visual recognition. In: Proceedings of the International Conference on Machine Learning, pp. 647–655 (2014)
8. Felzenszwalb, P.F., et al.: Object detection with discriminatively trained part-based models. IEEE Trans. Pattern Anal. Mach. Intell. **32**(9), 1627–1645 (2010)
9. Friedman, N., Russell, S.: Image segmentation in video sequences: a probabilistic approach. In: Proceedings of the Conference on Uncertainty in Artificial Intelligence, pp. 175–181 (1997)
10. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: IEEE Conference on Computer Vision Pattern Recognition, pp. 580–587 (2014)
11. Girshick, R., Fast R-CNN. In: IEEE International Conference on Computer Vision, pp. 1440–1448 (2015)

12. Guo, D.J., Zhe-Ming, L., Hao, L.: Multi-channel adaptive mixture background model for real-time tracking. J. Inf. Hiding Multimed. Sig. Process. **7**, 216–221 (2016)
13. Haines, T.S.F., Xiang, T.: Background subtraction with Dirichlet processes. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012 Part IV. LNCS, vol. 7575, pp. 99–113. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-33765-9_8
14. He, K., Zhang, X., Ren, S., Sun, J.: Spatial pyramid pooling in deep convolutional networks for visual recognition. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014 Part III. LNCS, vol. 8691, pp. 346–361. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10578-9_23
15. Hou, X., Zhang, L.: Saliency detection: a spectral residual approach. In: Proceedings of the IEEE Conference on Computer Vision Pattern Recognition, pp. 1–8 (2007)
16. KaewTraKulPong, P., Bowden, R.: An improved adaptive background mixture model for real time tracking with shadow detection. In: Remagnino, P., Jones, G.A., Paragios, N., Regazzoni, C.S. (eds.) Video-Based Surveillance Systems, pp. 135–144. Springer, Boston (2002). https://doi.org/10.1007/978-1-4615-0913-4_11
17. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: Proceedings of the Advances Neural Information Processing Systems, pp. 1097–1105 (2012)
18. LeCun, Y., Yoshua, B., Geoffrey, H.: Deep learning. Nature **521**, 436–444 (2015)
19. Lee, H., Grosse, R., Ranganath, R., Ng, A.Y.: Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In: Proceedings of the International Conference on Machine Learning, pp. 609–616 (2009)
20. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vis. **60**(2), 91–110 (2004)
21. Panda, D.K., Meher, S.: A Gaussian mixture model with Gaussian weight learning rate and foreground detection using neighbourhood correlation. In: Proceedings of the IEEE Asia Pacific Conference on Postgraduate Research Microelectronics, pp. 158–163 (2013)
22. Perronnin, F., Sánchez, J., Mensink, T.: Improving the fisher Kernel for large-scale image classification. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010 Part IV. LNCS, vol. 6314, pp. 143–156. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15561-1_11
23. Perazzi, F., Krahenbuhl, P., Pritch, Y., Hornung, A.: Saliency filters, contrast based filtering for salient region detection. In: Proceedings of the IEEE Conference on Computing Vision Pattern Recognition, pp. 733–740 (2012)
24. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. In: Proceedings of the Advances Neural Inform processing systems, pp. 91–99 (2015)
25. Sermanet, P., Kavukcuoglu, K., Chintala, S.C., Le-Cun, Y.: Pedestrian detection with unsupervised multi-stage feature learning. In: Proceedings of the IEEE Conference on Computer Vision Pattern Recognition, pp. 3626–3633 (2013)
26. Sohn, K., Zhou, G., Lee, C., Lee, H.: Learning and selecting features jointly with point-wise gated Boltzmann machines. In: Proceedings of the International Conference on Machine Learning, pp. 217–225 (2013)
27. Stauffer, C., Grimson, W.E.L.: Adaptive background mixture models for real-time tracking. Proc. IEEE Comput. Vis. Pattern Recognit. **2**, 246–252 (1999)

28. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: Proceedings of the IEEE Conference Computer Vision Pattern Recognition, pp. 1–9 (2015)
29. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE Conference on Computer Vision Pattern Recognition, pp. 2818–2826 (2016)
30. TensorFlow: how to retrain inception's final layer for new categories. www.tensorflow.org/tutorials/. Accessed 26 July 2017
31. Tian, Y.L., Lu, M., Hampapur, A.: Robust and efficient foreground analysis for real-time video surveillance. Proc. IEEE Conf. Comput. Vis. Pattern Recognit. **1**, 1182–1187 (2005)
32. Wu, Y., Liu, Y., Li, J., Liu, H., and Hu, X.: Traffic sign detection based on convolutional neural networks. In: Proceedings of the International Joint Conference on Neural Networks, pp. 1–7 (2013)
33. Xia, Y., Hu, R., Wang, Z., Lu, T.: Moving foreground detection based on spatiotemporal saliency. Int. J. Comput. Sci. Issues **10**(3), 79–84 (2013)
34. Xia, H., Song, S., He, L.: A modified Gaussian mixture background model via spatiotemporal distribution with shadow detection. Sig. Image Video Process. **2**(10), 343–350 (2016)
35. Zang, Q., Klette, R.: Evaluation of an adaptive composite gaussian model in video surveillance. In: Petkov, N., Westenberg, M.A. (eds.) CAIP 2003. LNCS, vol. 2756, pp. 165–172. Springer, Heidelberg (2003). https://doi.org/10.1007/978-3-540-45179-2_21
36. Zhou, L., Yang, Z., Yuan, Q., Zhou, Z., Hu, D.: Salient region detection via integrating diffusion-based compactness and local contrast. Proc. IEEE Trans. Image Process. **11**, 3308–3320 (2015)
37. Zivkovic, Z.: Improved adaptive Gaussian mixture model for background subtraction. Proc. IEEE Int. Conf. Pattern Recognit. **2**, 28–31 (2004)