Multi-branch Neural Networks for Video Anomaly Detection in Adverse Lighting and Weather Conditions

Sam Leroux[*] Bo Li[*] Pieter Simoens
Ghent University - imec
Technologiepark-Zwijnaarde, 9052, Gent, Belgium
first.lastname@ugent.be

Abstract

Automated anomaly detection in surveillance videos has attracted much interest as it provides a scalable alternative to manual monitoring. Most existing approaches achieve good performance on clean benchmark datasets recorded in well-controlled environments. However, detecting anomalies is much more challenging in the real world. Adverse weather conditions like rain or changing brightness levels cause a significant shift in the input data distribution, which in turn can lead to the detector model incorrectly reporting high anomaly scores. Additionally, surveillance cameras are usually deployed in evolving environments such as a city street of which the appearance changes over time because of seasonal changes or roadworks. The anomaly detection model will need to be updated periodically to deal with these issues. In this paper, we introduce a multi-branch model that is equipped with a trainable preprocessing step and multiple identical branches for detecting anomalies during day and night as well as in sunny and rainy conditions. We experimentally validate our approach on a distorted version of the Avenue dataset and provide qualitative results on real-world surveillance camera data. Experimental results show that our method outperforms the existing methods in terms of detection accuracy while being faster and more robust on scenes with varying visibility.

1. Introduction

Automated anomaly detection has attracted significant attention because of its importance for surveillance systems and public security [12, 11, 19, 1, 4, 16]. As anomalies are by definition rare, it is impossible to collect a training data set representative of all the anomalies of interest. Instead, most approaches rely on unsupervised learning to model normal inputs. This is typically done using an autoencoder model that tries to reconstruct input frames [12, 4]. The assumption is that deviating inputs will result in a high reconstruction error. While these techniques usually work well on the common benchmark data sets, deploying them in the real world is challenging for two main reasons: i) In the real world we have to deal with varying visibility and weather conditions. Rain reflections and day-night illumination variation can seriously impact the appearance and perceived motion of an object, resulting in a high reconstruction error and consequently in a high number of false positives. ii) Anomaly detection in a city context is not a static problem. Some visual aspects of the monitored scene, such as the background, may change due to seasonal effects (e.g. leaves on the ground), weather conditions (snow, reflections on wet surfaces, ...) or road construction works. While these background changes do not change the definition of an anomaly, they do result in a high reconstruction error and again in a high number of false positives. Therefore, it is not enough to train the model once. Instead, we should be able to continuously update the anomaly detector to adapt to these changes. Figure 1 illustrates both of these problems. All frames are recorded by the same camera in the course of just two days, yet there is a lot of variation in the illumination, reflections on the surface and background. Without taking these variations into account, the anomaly detector might report large amounts of false alarms and/or miss anomalies. Many computer vision tasks would struggle to deal with these types of variations but it is especially challenging for anomaly detection since the goal of anomaly detection is to detect deviating inputs. In the example of Figure 1, there is a very large difference between the first and last frame, yet the model should predict a low anomaly score for both of them.

In this paper, we focus on two types of changes: day-night transitions and weather shifts (sun-rain). As explained before, these will result in vastly different inputs to the anomaly detection model. Not only because of changes in the brightness and background but also in the behavior of certain objects. Bikes for example have head and tail lights at night which makes them look completely differ-

[*] These authors contributed equally to this work

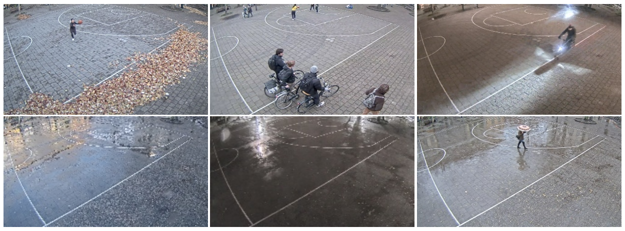


Figure 1. The varying lightning, weather, and background pose challenges for the current anomaly detection approaches.

ent. A possible solution would be to have multiple models, each specialized for one situation. This however would require some controller that activates a different model based on the time of day or the weather condition. In the real world, there are no discrete weather or lighting states, instead the weather lies on a spectrum between sunny and rainy conditions. Similarly, the illumination of the scene can gradually change between dark and bright. In this paper we instead propose a novel multi-branch model that trains individual branches to deal with varying conditions. The model can activate multiple branches and interpolate between them to handle a large spectrum of weather and lighting conditions. We experimentally evaluate our model on an augmented version of the Avenue benchmark data set where we changed the illuminations to simulate day-night transitions and where we added different levels of rain. We demonstrate that our model performs better, is faster and more robust than the existing state-of-the-art approaches. Our augmented Avenue data set can also serve as a challenging benchmark for future anomaly detection in adverse weather research. We also qualitatively validated our approach on real-world data collected in the course of multiple weeks and demonstrate that our proposed model can effectively detect rare events in the real world.

The remainder of this paper is organized as follows. In Section 2 we give an overview of related anomaly detection methods and deep learning techniques to deal with scenes with varying visibility. In Section 3, we introduce our approach and we then experimentally validate it on a synthetic dataset in Section 4. We finally evaluate our proposed methods on a real world surveillance footage dataset in Section 5 and conclude in Section 6.

2. Related Work

2.1. Anomaly detection in surveillance videos

Most approaches reconstruct input frames or predict future frames during training to model normal behavior. At test time, frames with high reconstruction or prediction errors are flagged as anomalies. The assumption here is that the model can only encode frequently seen events and behaviors through training and thus cannot reconstruct or predict the anomalies accurately. These approaches differ in how they incorporate motion information such as speed and direction of the movement of objects in the model. Motion information is crucial in the context of surveillance since many anomalies happen because of irregular movements [3]. Movement information can be encoded by reconstructing stacked frames [5] or by reconstructing complementary information such as optical flow maps [22]. Recurrent Neural Networks (RNNs) and Long Short Term Memory networks (LSTM) have also been employed to capture motion information [13]. Another possibility is to use 3D convolutional networks that can capture the shape and motion-related information in a video sequence [20, 1].

Most of the above works use pixel-wise metrics between input and output to detect anomalies [5, 13, 4, 16]. These metrics, however, do not necessarily correspond to human vision understanding and are sensitive to small changes in the brightness or color [15, 8]. Some recent works use intermediate feature representations instead, which can capture more high-level information than the raw pixel values.Xu et al. [22] classify these extracted latent codes using an SVM and assume that samples that do not belong to any cluster contain anomalies. Abati et al. [1] further explore the characteristics of the latent code using an autoregressive model. A benefit of using intermediate features instead of pixel reconstructions or predictions is that the model does not need a decoder part, making it less expensive and faster to execute [9].

2.2. Deep learning in adverse weather conditions

Despite the success of deep learning algorithms in many vision-related applications, their performance tends to degenerate under low illumination and adverse weather conditions. The primary reason for this degradation is that these algorithms are trained and evaluated on relatively clean data captured under well-controlled and stable conditions. When deployed in the real world however, the model needs to deal with a much larger variability. There is a lot of interest in unsupervised domain adaptation which allows a model trained on one domain (e.g. clear weather) to adapt to another domain (e.g. rain). Sakaridis et al. [18] enable image segmentation at night using frames that are created via image translation [10]. Hoffman et al.[6] propose a domain adaptation approach by extending the CycleGAN [23] framework for multiple visual recognition and prediction tasks, such as cross-season semantic segmentation. Similarly, Tran et al. [21] design a two-branch framework by leveraging the feature-level and pixel-level information translation between the source domain and target domain to handle different types of domain adaptation. RoyChowdhury et al. [17] propose to adapt an object detector to a new domain (nighttime, foggy, and rainy) via knowledge distillation loss. Other techniques rely on multiple sensors that can improve the performance in different conditions. Bijelic et al. [2] apply a real-time fusion network for lidar and camera measurements in foggy weather to detect objects accurately. These approaches can noticeably enhance the performance in adverse weather and improve the accuracy of the related tasks such as object detection and classification. However, more research is needed in this area to develop a model that can handle the full variety of different weather conditions.

All of these previously mentioned techniques focus on object detection or segmentation tasks. In this work, we perform anomaly detection which is even more sensitive to changing weather conditions as the goal of unsupervised anomaly detection is to detect anything that deviates from a baseline and these changing weather conditions can cause the inputs at test time to differ substantially from the training data.

3. Methods

To tackle the problems associated with the distribution shift caused by varying weather conditions, we propose a multi-branch model. Fig. 2 shows the general framework. Given an input sequence of camera frames, we first estimate their corresponding backgrounds with a shallow background learning module. These estimations are then subtracted from the input frames to highlight the foregrounds. This first step should already remove most of the distractions such as reflections on the background. Contrary to previous work, we do not use fixed background frames but instead can automatically adjust to changing brightness levels as we estimate a background frame for every input frame. We then use multiple branches in our model to encode the remaining foreground. Each branch can specialize in different situations. We interpolate between these branches to get a prediction for the future. The anomaly score is then calculated based on the difference between the actual observation and the predictions. We will now explain each module in detail.

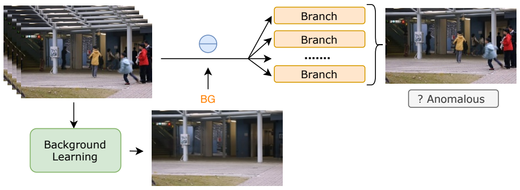


Figure 2. Overview of our approach. Given a video sequence as the input, we first predict the corresponding background frame. This background frame is then subtracted from each of the input frames. We send this preprocessed video sequence to multiple branches that each predict an anomaly score. We calculate a weighted average over the scores of different branches where the weights of each branch are predicted by the model itself, allowing the model to selectively activate branches.

3.1. Background prediction

A common preprocessing step in computer vision is to discard the background to bring more attention to the foreground objects [12, 1]. This is especially useful for anomaly detection as anomalies are caused by foreground objects and we should not be distracted by the background. Most approaches use the average frame, calculated over some window as the background. If the window is too short, the extracted background will be noisy due to moving objects. If the window is too long, the average frame might not be representative of the current frame because of changing brightness levels or weather conditions. Instead of manually defining a window or a background, we propose to learn this automatically from data. To deal with a continuous spectrum of backgrounds, we describe the background as a linear combination of several trainable background bases. We add N trainable tensors to the model's parameters, each with the same dimensions as the input frame $(w * h * 3)$. We initialize them with zero values. We also add a small convolutional neural network to the model. This network consists of four convolutional blocks, two fully connected layers, and a softmax layer. This model predicts the N weights that are used to combine the background bases through a weighted sum to obtain the background for the current frame. The predicted background is then subtracted from the frame. The background bases and background selection module are trained end-to-end using gradient descent as we will explain in the following sections. Fig. 3 illustrates this background learning module. This figure shows the two learned background bases (BG-0 and BG-1). These are completely learned from data without supervision. BG-0 represents a typical clear frame while BG-1 shows a darker frame. Given an input frame, we use the small neural network to predict two scalar values (α_0, α_1). We then estimate the current background as the weighted sum of the two background bases, weighted by (α_0, α_1). This allows us to easily adjust to changes in the brightness level. In this example we used two background bases (causing $\alpha_0 = 1 - \alpha_1$) but we can use more background bases if appropriate. In our experiments, we found that 3 bases is usually sufficient.

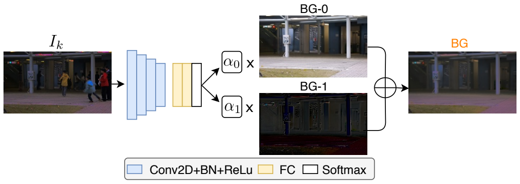


Figure 3. Architecture of the proposed background learning model with two trainable background base frames. *BG-* are the learned background bases. For each input frame, we obtain its background by interpolating between these bases with the ratios that are estimated by a shallow neural network.

3.2. Multiple branches

In the previous section we subtracted an estimated background frame from each input. This allows us to remove distractions in the background and to focus on the objects that might cause anomalies. The remaining foreground might also differ between different conditions. Bikes for example have head and tail lights on during the night, causing reflections on the ground. Instead of using a single model to generalize to all these different situations, we add multiple parallel anomaly detection branches to the model. Each of them can specialize in a different condition. Similar to the background prediction, we do not hard code the role of each branch but learn it unsupervised from data. We again interpolate between the predictions of multiple branches to deal with a wide range of possible inputs.

The idea of using multiple branches is independent of the choice of anomaly detection model that is used inside a branch. In this work, we use a recent state-of-the-art anomaly detector for Decoupled Appearance and Motion Learning (DAML) [9], which consists of an encoder, a motion model, and a decoder. DAML [9] first uses an encoder to extract latent codes (features) from individual input frames. These latent codes are then passed to a motion model that contains several 3D convolutional layers to predict the latent code of a future frame. A decoder is used during training to generate frames based on the combined latent codes. At test time, the decoder is discarded and anomalies are detected by calculating the difference between observed and predicted latent codes. The authors demonstrate that predicting the future in latent space can capture high level information, useful for anomaly detection.

We use the same principle in our multi-branch model. Each branch has a different encoder but we still have a single motion model and a single decoder. Fig. 4 shows the architecture. Given a preprocessed input video sequence, we extract latent codes from each of the encoders, which are then used for the prediction of the latent code of a future frame and for the reconstruction of the input video sequence. We feed the weighted latent codes to the decoder for reconstructing the inputs. Experiments with multiple decoders gave similar results, so we opted for a single decoder to reduce the model size. We extend the background prediction model with a second output layer that predicts the weights of each branch and use these weights to interpolate between the predictions of each branch.

3.3. Training

We train the model in two stages, first, we train the background learning module and background bases from data. We pass each frame through the background selection model to predict the background weight α_i for each of the background bases BG_i. We use these weights to determine the background for the current frame as a weighted sum of the background bases with the corresponding weights (equation 1a). We train the model to minimize the MSE loss between the obtained background and the input frame I (equation 1b). We use gradient descent to jointly train the values of the background bases and the weights of the background selection model. If there would be only one background base, the best the model can do, is to use the average pixel values as the background base. If we allow for multiple backgrounds, the model can learn different prototype backgrounds and the background selection model can interpolate between them to generate the most accurate background estimation for the current frame.

After the training loss $\mathcal{L}_{BG}$ converges, we fix the weights in the background learning module and focus on training the encoders, the motion models and decoder following [9]. The key here is to extract latent codes from multiple individual frames and use these latent codes to predict the latent code of a future frame. We pass $k = 6$ frames through the background selection model to obtain k backgrounds and subtract these from the frames to obtain the foreground frames f. We also predict the B branch activation values. Each foreground frame is passed through each encoder to obtain B (number of branches) latent codes each. The latent codes are then combined as a weighted sum using the branch activations. The averaged latent code is passed through the decoder to obtain a reconstructed foreground $\hat{f}$. The first part of equation 1c measures the euclidean distance between each input frame and the corresponding reconstruction. Based on the extracted B latent codes, we also predict a future latent code z_{k+n}, $n = 6$ frames in the future. The second part of equation 1c measures the euclidean distance between the predicted future latent code and the actual latent code. For more details, we refer to [9].

$$BG = \sum_{i=1}^{N} \alpha_i * BG_i \tag{1a}$$

$$\mathcal{L}_{BG} = MSE(BG, I) \tag{1b}$$

$$\mathcal{L} = \sum_{i=1}^{k} ||\hat{f}_i - f_i||_2 + \sum_{b=1}^{B} \alpha_b ||\hat{z}_{k+n} - z_{k+n}||_2 \tag{1c}$$

We train our models for 60 epochs with an initial learning rate of $1e-4$, which decays by a factor of 0.5 every 20 epochs. In the first 10 epochs, only the background learning model is trained. Then we keep the background learning model fixed and the rest of the model. We use the Adam Optimizer [7] for all experiments.

3.4. Inference

At inference time, we discard the decoder and calculate the feature-wise difference between the observed and predicted latent codes as the anomaly score following [9]. The

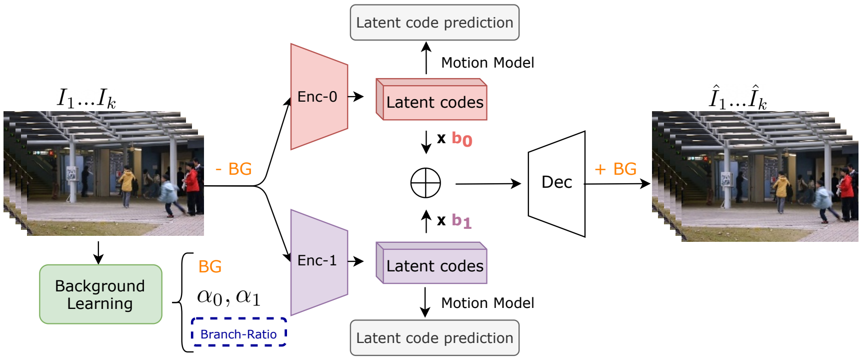


Figure 4. Architecture of the proposed multi-branch model with learnable backgrounds. The stacked latent codes from each encoder are aggregated with the corresponding branch ratio b and sent to the decoder for frame reconstruction. We discard the decoder during inference and only use the difference between the predicted and observed latent codes to flag anomalies.

assumption is that the model can predict the latent code for a normal frame with high accuracy but is unable to do so for anomalous frames. The Mean Square Error (MSE) is used to quantify this feature-wise difference. This is repeated for each branch. We calculate the weighted sum of the anomaly scores of the different branches where the weight corresponds to the branch activation level. A frame is flagged as an anomaly if the total anomaly score is higher than a predefined threshold.

4. Anomaly detection on the augmented Avenue dataset

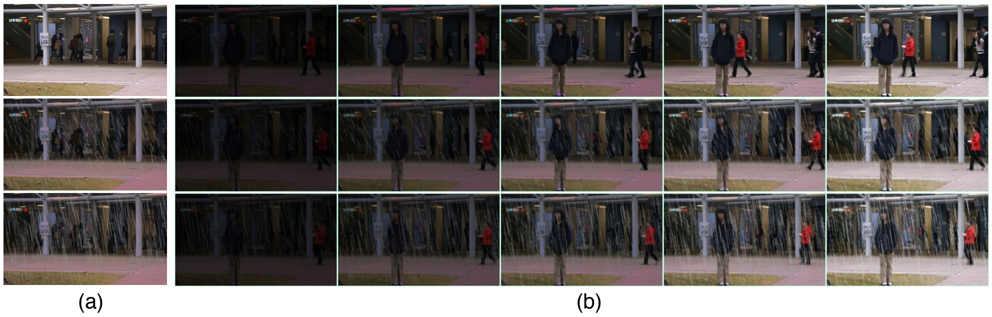
There are many publicly available data sets for anomaly detection in surveillance videos such as the UCSD Pedestrian [14], CUHK Avenue [12], and ShanghaiTech data set [11]. However, all of them contain relatively clean data, recorded at similar times during the day and under clear weather conditions. Therefore, these data sets are not representative of real-world surveillance footage where external factors such as weather and time of the day can severely influence the quality of the collected frames. To evaluate the robustness of our proposed multi-branch model to varying visibility, we generate a synthetic data set that simulates different conditions by augmenting the CUHK Avenue data set.

There are 15 328 training and 15 324 testing frames in the CUHK Avenue dataset with a total of 47 abnormal events, including people throwing objects, loitering, and running. We train a model with two branches and two background bases. To allow for a fair comparison, we reduce the size of our encoders and motion models to have a similar total number of parameters as the DAML model it was based on. This means that for our model with B branches, the encoder and motion model is roughly B times smaller than the original model. We compare our methods with the existing state-of-the-art approaches: FFP-MC [11], DAML [9], MemAE [4], and MNAD [16]. As is common, we report the frame-level Area Under Curve (AUC) score together with the 95% confidence interval over 4 runs.

To simulate the day-night conditions, we adjust the brightness of the input frames. A value of 1.0 corresponds to the original brightness while the lowest value of 0.2 corresponds to a frame with low visibility. We also add raindrops on the frames using the publicly available Automold toolkit[1]. We consider heavy and torrential rain, which differ in the amount and the length of the raindrops. The slant and the location of the raindrop are randomly placed on every frame. During training, we use brightness levels 0.7 and 1.0 and No rain/ Heavy Rain augmentations. We then test the robustness for all combinations of brightness levels and rain intensity. The results are shown in table 1. Our model with two background bases and two branches outperforms the existing methods on all combinations of brightness level and rain intensity, including the ones it was not trained for. We also compare against a model with just one branch (but stil two background bases). This collapses to the DAML model [9] except that it has a trainable background extraction step. We can clearly see that this alone already drastically improves the anomaly detection performance.

In addition to the anomaly detection accuracy, the inference speed is also an important factor for real-world deployment since the anomaly detection must be fast enough to cope with the frame rate of the surveillance camera. In this regard, we benchmark our methods with the approaches that have publicly available code on an Intel(R)Core(TM) i7-8700 CPU@3.2GHz with a GeForce GTX1080 Ti GPU. We report the number of frames that we can process per second (FPS) in Fig 6 on the x-axis. The y-axis shows the performance of the model. For each model, the dot represents the averaged AUC score over all the weather conditions. The confidence interval shows the standard deviation of the model's AUC score for different weather conditions.

[1] https://github.com/UjjwalSaxena/Automold--Road-Augmentation-Library


(a)

(b)

Figure 5. The example training and testing frames from our sunny-rainy Avenue dataset. The frames from top to bottom are original frames, frames with heavy rain and torrential rain. The training data (a) only includes original frames, frames with heavy and torrential rain at illumination 0.7. The test data (b) contains the frames with and without rain at each illumination level. The illuminations are 0.2, 0.4, 0.6, 0.8, 1.0 from left to right.

Table 1. Anomaly detection accuracy across multiple illuminations and raining conditions

Amount of Rain	No rain					Heavy rain					Torrential rain				
Illumination	0.2	0.4	0.6	0.8	1.0	0.2	0.4	0.6	0.8	1.0	0.2	0.4	0.6	0.8	1.0
FFP-MC [11]	72.4±4.9	79.3±3.4	82.5±1.1	84.2±0.4	85.0±0.6	57.7±6.2	60.6±2.6	66.5±1.2	68.0±2.4	69.7±1.0	53.9±6.5	62.0±3.7	62.4±0.4	62.3±1.0	61.1±1.6
DAML [9]	62.2±3.5	81.5±3.4	83.1±1.1	80.3±2.1	80.7±3.0	62.2±3.4	81.4±3.4	82.9±1.2	80.3±2.1	80.8±3.0	58.7±2.8	78.3±3.0	79.5±1.4	75.6±2.5	76.4±3.5
MemAE [4]	72.6±3.6	78.2±1.1	80.6±0.3	80.4±0.3	82.1±0.7	72.0±2.2	76.7±0.5	79.2±1.3	80.0±0.6	81.9±0.2	71.3±1.6	76.0±0.2	76.2±1.4	76.6±0.9	78.9±1.2
MNAD [16]	72.3±2.9	73.7±1.6	78.6±1.9	82.5±0.4	83.9±0.5	58.7±6.5	59.2±5.7	63.6±3.0	69.2±3.6	71.0±3.1	52.8±5.1	56.7±3.9	62.2±3.8	65.4±5.4	64.9±3.3
Ours (1 branch)	71.3±1.4	85.0±1.5	88.9±1.0	88.0±2.4	**88.3±0.6**	68.7±1.0	78.2±1.9	88.0±2.9	86.8±0.7	86.4±2.3	67.8±0.6	75.9±1.9	86.4±2.2	85.5±1.4	83.3±1.5
Ours (2 branches)	**79.1±1.9**	**86.0±0.8**	**89.2±1.3**	**88.1±2.2**	87.8±2.8	**75.7±0.3**	**84.9±0.4**	**89.5±0.6**	**87.9±1.9**	**87.6±2.3**	**72.8±1.3**	**84.5±2.3**	**88.1±0.4**	**86.1±3.0**	**86.0±2.1**

A larger interval indicates that the accuracy of the model is severely impacted by the weather and lighting conditions.

Our multi-branch model with one branch surpasses the other methods over a large margin in terms of the inference speed. Compared to DAML [9], we found that the preprocessing done by the background subtraction makes it possible to use a smaller encoder, decoder and motion model, resulting in a faster model. The model with two branches is significantly slower, yet it achieves a higher AUC score and is still faster than most related approaches.

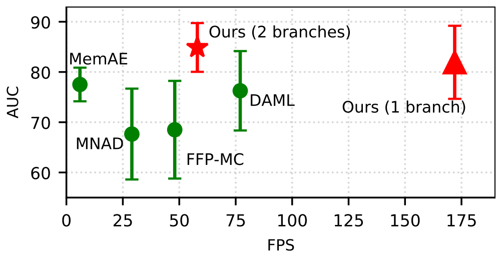


Figure 6. FPS and the averaged AUC score together with its standard deviation on the augmented Avenue data set for different methods. The closer a model is to the top right corner, the more accurate and faster the model is. Using one branch, our multi-branch model is 2 to 29 times faster than the existing approaches. By increasing the number of branches to 2, we achieve a better and more robust accuracy than the existing approaches.

5. Real world anomaly detection

In the previous section, we used an artificially generated data set to simulate the day-night and sunny-rainy conditions. This synthetic data set helps us gain insights into detecting anomalies in varying illuminations and weather conditions. However, it is not able to replicate all the properties of the real-world surveillance footage. We therefore qualitatively evaluate our proposed methods on real-world surveillance data that is collected using the cameras deployed in the Antwerp Smart Zone[2]. The Antwerp Smart Zone is located around a typical Antwerp (Belgium) neighborhood with narrow residential streets, many people, busy shops, and bars. The Smart Zone is a part of the city where cameras and sensors are deployed to research technologies that could improve life in the city. The camera data contains various traffic types such as pedestrians, cyclists, and cars under varying illumination and weather conditions. We recorded 18 video sequences from a static surveillance camera looking down a public square in November, December and January. Each video sequence contains 43200 frames with 1800 frames per hour, with a resolution of 1080x1920. This data set is challenging because of the variety of weather conditions (sunny and raining), changing shadows, reflections of external lights at night, and multiple activities such as walking, cycling, and playing basketball as shown in

[2] https://antwerpsmartzone.be/en/

Fig. 1. Unfortunately, we are not able to publicly share this data due to privacy constraints.

We equip our multi-branch model with three background bases and two branches. We train the initial model with the frames from a first video sequence (21600) and use the remaining video sequences as the test data. The train and test data were recorded at different days.

5.1. Interpolation of background bases

Fig. 7 shows how the model interpolates between the learned background bases in the course of several days. It learns three distinct background bases where one focuses on daylight features, and the other two specialize in frames recorded at night. One of these two bases is used when the street lights are on, the other one when they are off. Note that the model has learned this behavior in a completely unsupervised way. By interpolating between the bases, the model can handle the continuous day-night cycle. We clearly see that background 0 (red line) is most active during the day while background 1 (green line) is only activated at night. Background 2 (blue line) seems to work together with background 1 in a more complicated way.

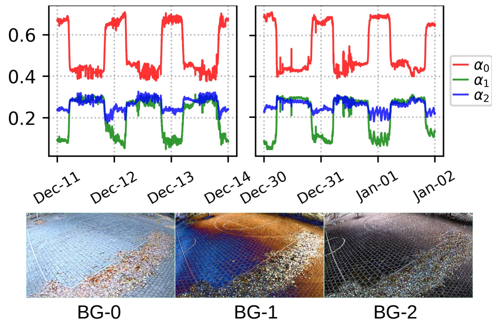


Figure 7. The interpolation between three learned background bases in different time windows using our model that is with a single branch. The model can interpolate between the learned complementary background bases to handle the periodic day-night cycle

5.2. Detected anomalies

We next show the detected anomalies (top 0.01% highest anomaly score) in Fig. 8 (a) . The model accurately detects trucks and cars as anomalies. These are valid anomalies as usually only pedestrians are allowed on the playground. The model is also able to accurately flag the (illegal) fireworks on New Year's Eve as an anomaly. Most of the false positives are frames recorded at night where light reflects on a wet surface. Also note how the leaves on the ground are cleaned up compared to the training frames (Figure 8 compared to figure 7). As the learned backgrounds still contain these leaves, this will cause higher prediction errors and will result in a higher number of false positives.

5.3. Continuous learning of the anomaly detection model

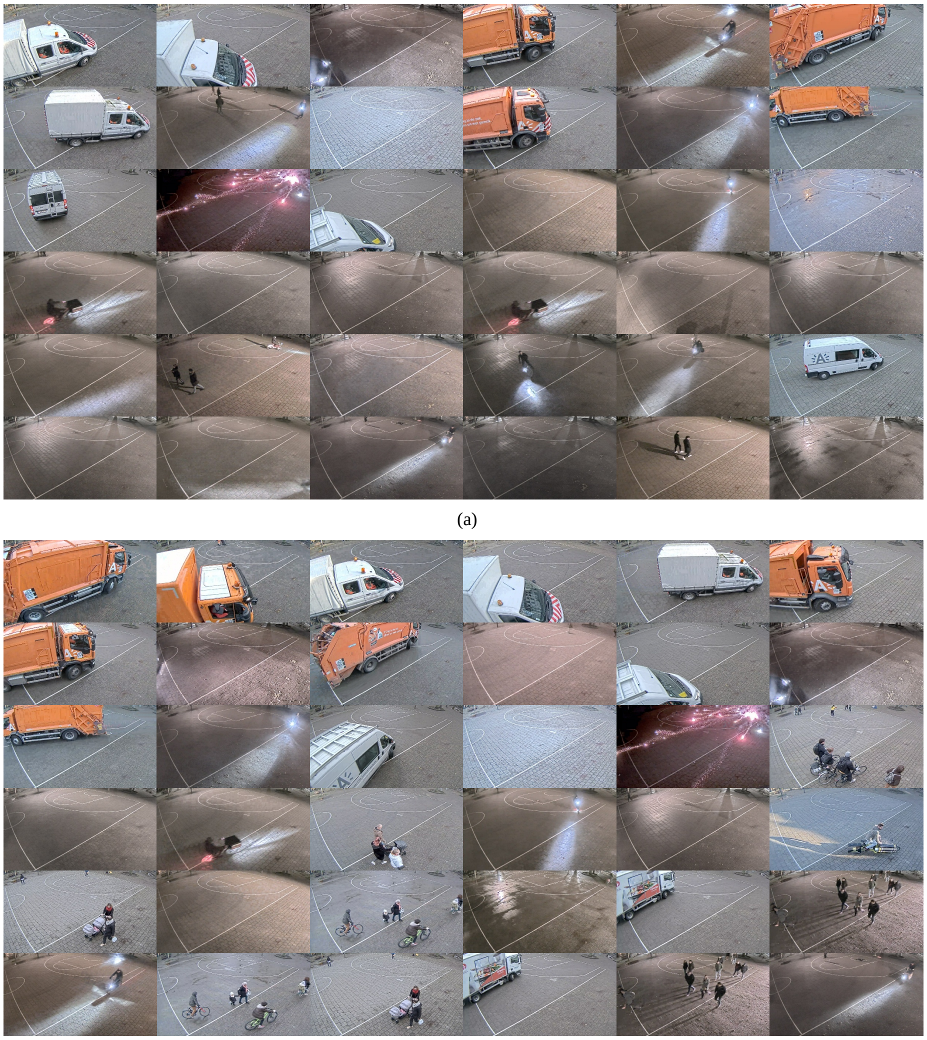
In real-world surveillance scenarios, changes are happening all the time. For example, the background looks different because the leaves are cleaned up in figure 1, the wet surface causes reflections that were not seen during training in figure 8 (a), trees and plants look different across seasons, and semi-permanent objects appear and disappear regularly. Therefore, we argue that the deployed anomaly detector needs to be updated regularly to keep pace with these changes.

Our proposed architecture allow us to quickly update a single or several components to adapt to these changes. We experiment with updating the background learning module over time and freeze the rest of the model. The intuition is that the background might change over time but the behavior of the foreground objects should remain more or less the same. By only updating this background learning module, we need less computational resources, which can benefit the deployment on a resource-constrained edge device. It is also important to note that we train and update the model without any human intervention in a completely unsupervised manner. This allows us to keep the model running continuously without having to send any frames to a cloud backend which is a very desirable property for a model that deals with privacy-sensitive data such as surveillance camera frames.

We show the detected anomalies using the updated models in 8 (b). The updated model correctly identifies most nighttime frames as normal resulting in less false positives. These results suggest that just updating the background learning module already allows us to deal with changes in the background such as those caused by reflections on a wet surface.

6. Conclusion

Anomaly detection in real-world environments is a very challenging open problem. In this paper, we introduced a novel approach that is better at dealing with varying illuminations and weather conditions compared to the typically used anomaly detection techniques. Our multi-branch model is equipped with a background learning module that can interpolate between multiple bases to predict a suitable background for every input frame, adapted to the actual brightness level of that frame. By making the preprocessing step a trainable part of our network, we can quickly deal with the small changes over time. We then employ multiple branches to boost the exploration of the foreground features further. We quantitatively evaluated our models on an artificially distorted version of the Avenue dataset where we changed the brightness levels and added rain and show that our models outperform existing approaches. We also vali-



(a)

(b)

Figure 8. Detected anomalies by the initial (a) and the updated (b) multi-branch (two branches) model (top 0.01%). The anomaly score of the frames decreases from left to right, top to bottom. If the model flags more than one frame as anomalies within two minutes, we only show one. Most of the nighttime frames that are falsely detected by the initial model are filtered out by the fine-tuned one

date our models on real-world data collected in the course of multiple weeks on a camera deployed in a city. We show that our models can capture meaningful anomalies and that they can deal with changing environments by updating a part of the model in an unsupervised way.

Computer vision in adverse weather conditions is still an open research topic. Many of the breakthroughs in the computer vision field during the past years have used relatively clean datasets. The next step is to make it possible to use these techniques in the wild, where we have to deal with all kinds of distortions. Anomaly detection in adverse weather conditions, e.g., foggy and snowy weather, is even more challenging and requires more research.

Acknowledgements

This research received funding from the Flemish Government under the “Onderzoeksprogramma Artificiële Intelligentie (AI) Vlaanderen” programme, and from imec under the CityFlows AAA programme.

References

[1] Davide Abati, Angelo Porrello, Simone Calderara, and Rita Cucchiara. Latent Space Autoregression for Novelty Detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision and Pattern Recognition*, 2019.

[2] Mario Bijelic, Fahim Mannan, Tobias Gruber, Werner Ritter, Klaus Dietmayer, and Felix Heide. Seeing through fog without seeing fog: Deep sensor fusion in the absence of labeled training data. *CoRR*, abs/1902.08913, 2019.

[3] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM Comput. Surv.*, 41, July 2009.

[4] Dong Gong, Lingqiao Liu, Vuong Le, Budhaditya Saha, Moussa Reda Mansour, Svetha Venkatesh, and Anton van den Hengel. Memorizing normality to detect anomaly: Memory-augmented deep autoencoder for unsupervised anomaly detection. In *IEEE International Conference on Computer Vision (ICCV)*, 2019.

[5] Mahmudul Hasan, Jonghyun Choi, Jan Neumann, Amit K. Roy-Chowdhury, and Larry S. Davis. Learning temporal regularity in video sequences. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 733–742, 2016.

[6] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei A. Efros, and Trevor Darrell. Cycada: Cycle consistent adversarial domain adaptation. In *International Conference on Machine Learning (ICML)*, 2018.

[7] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[8] Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, Hugo Larochelle, and Ole Winther. Autoencoding beyond pixels using a learned similarity metric. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, pages 1558–1566, 2016.

[9] Bo Li, Sam Leroux, and Pieter Simoens. Decoupled appearance and motion learning for efficient anomaly detection in surveillance video. *Computer Vision and Image Understanding*, 210:103249, 2021.

[10] Ming-Yu Liu, Thomas Breuel, and Jan Kautz. Unsupervised image-to-image translation networks. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, page 700708. Curran Associates Inc., 2017.

[11] W. Liu, D. Lian W. Luo, and S. Gao. Future frame prediction for anomaly detection – a new baseline. In *2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[12] Cewu Lu, Jianping Shi, and Jiaya Jia. Abnormal event detection at 150 fps in matlab. *2013 IEEE International Conference on Computer Vision*, pages 2720–2727, 2013.

[13] W. Luo, W. Liu, and S. Gao. Remembering history with convolutional lstm for anomaly detection. In *2017 IEEE International Conference on Multimedia and Expo (ICME)*, pages 439–444, 2017.

[14] Vijay Mahadevan, Wei-Xin LI, Viral Bhalodia, and Nuno Vasconcelos. Anomaly detection in crowded scenes. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1975–1981, 2010.

[15] Michaël Mathieu, Camille Couprie, and Yann LeCun. Deep multi-scale video prediction beyond mean square error. *CoRR*, abs/1511.05440, 2015.

[16] Hyunjong Park, Jongyoun Noh, and Bumsub Ham. Learning memory-guided normality for anomaly detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14372–14381, 2020.

[17] Aruni RoyChowdhury, Prithvijit Chakrabarty, Ashish Singh, SouYoung Jin, Huaizu Jiang, Liangliang Cao, and Erik Learned-Miller. Automatic adaptation of object detectors to new domains using self-training. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[18] Christos Sakaridis, Dengxin Dai, and Luc Van Gool. Semantic nighttime image segmentation with synthetic stylized data, gradual adaptation and uncertainty-aware evaluation. *CoRR*, abs/1901.05946, 2019.

[19] Waqas Sultani, Chen Chen, and Mubarak Shah. Real-world anomaly detection in surveillance videos. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[20] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 4489–4497, 2015.

[21] Luan Tran, Kihyuk Sohn, Xiang Yu, Xiaoming Liu, and Manmohan Chandraker. Gotta adapt em all: Joint pixel and feature-level domain adaptation for recognition in the wild. In *In Proceeding of IEEE Computer Vision and Pattern Recognition*, Long Beach, CA, June 2019.

[22] Dan Xu, Yan Yan, Elisa Ricci, and Nicu Sebe. Detecting anomalous events in videos by learning deep representations of appearance and motion. *Computer Vision and Image Understanding*, 156:117 – 127, 2017.

[23] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, 2017.