ECDNet: Efficient Siamese Convolutional Network for Real-Time Small Object Change Detection from Ground Vehicles

Sander R. Klomp; Eindhoven University of Technology, SPS-VCA group of Electr. Eng.; Eindhoven, The Netherlands
Dennis W.J.M. van de Wouw; ViNotion B.V.; Eindhoven, The Netherlands
Peter H.N. de With; Eindhoven University of Technology, SPS-VCA group of Electr. Eng.; Eindhoven, The Netherlands

Abstract

Change detection from ground vehicles has various applications, such as the detection of roadside Improvised Explosive Devices (IEDs). Although IEDs are hidden, they are often accompanied by visible markers, which can be any kind of object. Because of this, any suspicious change in the environment compared to an earlier moment in time, should be detected. Little work has been published to solve this ill-posed problem using deep learning. This paper shows the feasibility of applying convolutional neural networks (CNNs) to HD video, to accurately predict the presence and location of such markers in real time. The network is trained for the detection of pixel-level changes in HD video, compared to an earlier reference recording. We investigate Siamese CNNs in combination with an encoder-decoder architecture and introduce a modified double-margin contrastive loss function, to achieve pixel-level change detection results. Our dataset consists of seven pairs of challenging real-world recordings with geo-tagged test objects. The proposed network architecture is capable of comparing two images of 1920×1440 pixels in 150 ms on a GTX1080Ti GPU. The proposed network significantly outperforms state-of-the-art networks and algorithms on our dataset in terms of F-1 score, on average by 0.28.

Introduction

Roadside Improvised Explosive Devices (IEDs) have the highest casualty rate per incident among Dutch military in Afghanistan [1]. IEDs are often accompanied by markers, which can be any kind of object, to time the detonation of an IED when a vehicle passes it [2]. In order to reduce casualties by IEDs, identification of IED placement is normally conducted by Mission Payload Operators (MPOs), by using environmental cues. However, experienced MPOs are limited in number [3]. This motivates the desire for real-time automated detection.

Real-time automated systems for the detection of IED cues or markers have been explored in the past, such as by using LiDAR systems [4], ground-based monocular camera imagery [5–7], ground-based stereo imagery [8][9], or imagery from unmanned aerial vehicles [10]. All methods have in common that they search for small environmental changes compared to a previous patrol, under the assumption that the placement of an IED causes small appearance changes in the environment. However, none of these methods are sufficiently capable of detecting small object changes in the dynamic forest environments of our dataset.

This research focuses on change detection between two aligned videos from different moments, with robustness to illumination differences, dynamic surroundings and small alignment errors, using a novel Convolutional Neural Network (CNN) architecture. The videos are acquired from a moving ground vehicle. Image alignment is performed using the registration approach of [8] (to be published, earlier version: [11]). This approach consists of a processing chain using GPS-based image retrieval, depth image generation from stereo cameras, and image alignment based on both features and depth information. After alignment, 2D change detection is performed. Previous work used heuristics and post-processing for decision making, which offered insufficient quality of detection under challenging recording conditions. The purpose of this paper is to replace the decision making by our novel CNN architecture, aiming at improved detection while maintaining real-time operation.

Our contributions are twofold. First, we propose a novel CNN architecture that can perform change detection in real time (around 150 ms per frame) on high-resolution imagery (1920×1440 pixels) and that exploits ideas from related domains, i.e. object detection, patch matching, and semantic segmentation networks. Second, we propose the use of an extended double-margin contrastive loss function for the training of general Siamese networks for change detection. Both proposals are experimentally validated on our own datasets of ground-based images, containing changes in the form of static objects.

Related work

CNNs can be used for change detection in various ways. In the absence of training data, CNNs pretrained on the ImageNet dataset [12] may be used as a feature extractor. The Euclidean distance in feature space is then used to generate a difference image. This is shown to outperform handcrafted features [13]. However, the CNN features were never specifically trained to be well-separable by Euclidean distance, which may limit change detection accuracy. Our preliminary experiments have shown that pretrained feature extractors are not suited for the complex outdoor scenes in our dataset due to poor accuracy.

If training data are available, supervised networks can be used to learn change detection features directly from the training data [14–16]. However, none of them use pretrained weights to initialize their networks, even though it has a proven value for classification and object detection tasks, especially when only a limited amount of annotated data are available [17]. Our dataset is relatively small, hence pretrained weights are essential.

Because existing change detection networks are less suited for our use case, we prefer building a network from more basic components, by borrowing techniques that have been proven to work well in other computer vision tasks. The research fields from which we draw inspiration for our network are classification, semantic segmentation and patch matching.

Canziani *et al.* [18] provide an overview of speed considerations for recent popular classification networks. This overview supports a base network choice that operates in real time. It shows that especially ResNet variants [19] have high performance for relatively low computational cost.

Change detection requires creating a pixel-level prediction based on image data, which makes it closely related to the field of semantic segmentation. Recent work focuses on segmentation networks that can operate in real time, e.g. ERFNet [20], which achieves surprisingly good performance compared to state-of-the-art expensive semantic segmentation networks.

Patch matching is closely related to change detection, as it compares two patches and determines their similarity score. In general, three main architecture options exist for patch matching: Siamese, Pseudo-Siamese and stacked (or '2-channel') networks [21]. Combining Siamese networks with concepts from semantic segmentation and an efficient base network will allow us to perform pixel-level change detection in real time.

ECDNet: Proposed architecture

This section introduces our novel Efficient Change Detection Network (ECDNet) architecture for real-time change detection between aligned images. The network consists of a Siamese encoder-decoder architecture, as shown in Fig. 1. The diagram portrays all feature maps with their corresponding resolution and layer depth (number of used filters in that layer) of both the encoder and decoder.

This section is structured as follows. First, it describes how to achieve efficient pixel-level features through a smart choice of encoder and decoder. Second, we show how to extend this basic network to be suited for change detection, including the definition of an appropriate loss function. Third, a post-processing operation to suppress invalid detections is proposed, which is uniquely coupled to our network architecture. Finally, all parameters are listed for reproducibility.

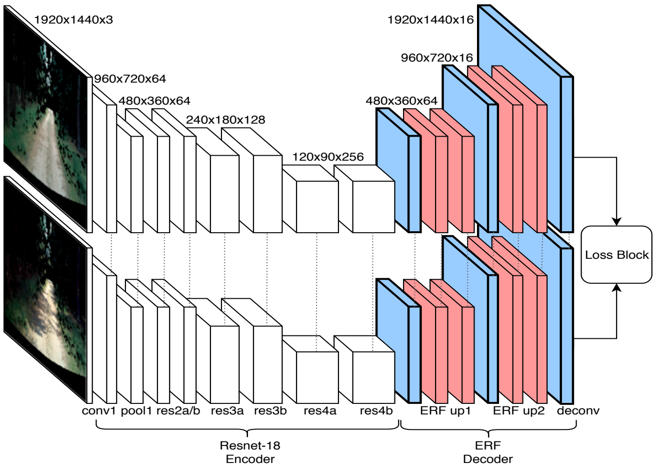

Figure 1. *Proposed architecture for change detection (ECDNet). White blocks refer to the encoder. Colored blocks highlight the decoder, consisting of transposed convolution blocks (blue, bold) followed by residual blocks (red). Top and bottom networks share parameters, indicated by vertical lines. The numbers show activation map sizes after each block.*

Achieving efficient pixel-level features

Finding a good trade-off between computation speed, performance and memory requirement is crucial to allow for real-time processing of high-resolution images. The most impactful component for this trade-off is the encoder, which acts as a feature extractor. Less impactful is the choice of the decoder network, which upsamples the features back to original resolution.

This work employs a ResNet-18 based encoder architecture, based on the accuracy versus speed trade-off analysis from [18]. In the proposed architecture, ResNet-18 is cut-off after the fourth residual block. This enables the detection of small objects, which may otherwise vanish due to excessive downsampling.

A decoder is employed to expand the downsampled encoder features back to pixel-level features.The decoder from ERFNet [20] is adopted for its simplified one-dimensional (1D) residual blocks, which split the residual blocks into multiple 1D convolutions for faster processing at the cost of memory.

Extending the network for change detection

An encoder-decoder Siamese architecture enables pixel-level change detection. In contrast to pretrained network features, the network optimizes to maximal Euclidean distance separability between outputs, provided that it is used with the right loss function.

Loss function

The contrastive loss that has been proven effective in patch matching [22] can be re-defined to pixel-level operation to make it useful for change detection, similar to [14]. The pixel contrastive loss as a function of network parameters W and training inputs $\mathbf{X}$ is then defined as a sum over pixel coordinates (i,j) as follows

$$\mathscr{L}(W,\mathbf{X}) = \frac{1}{2}\sum_{k=1}^{P}\sum_{i,j}(1-y_{i,j}^{(k)})(D_{i,j}^{(k)})^2 + y_{i,j}^{(k)}\max(0,m-D_{i,j}^{(k)})^2, \tag{1}$$

where P is the batch size, k the image number in the batch, $y_{i,j}^{(k)}$ is the pixel label of the kth image at position (i,j), with 0 indicating no change and 1 denoting change. Parameter D is a 2D change map, and $m > 0$ is a margin parameter beyond which changed pixels no longer influence the loss [22]. The margin parameter m can be chosen arbitrarily, as the network will accommodate itself to the margin [23]. The change map D is defined as the Euclidean distance between two network output feature maps $\mathbf{G}_W(\mathbf{X}_1)$ and $\mathbf{G}_W(\mathbf{X}_2)$, for the two aligned input images $\mathbf{X}_1$ and $\mathbf{X}_2$, giving

$$D(\mathbf{X}_1,\mathbf{X}_2) = \|\mathbf{G}_W(\mathbf{X}_1) - \mathbf{G}_W(\mathbf{X}_2)\|_2. \tag{2}$$

A binary change mask can then be generated by thresholding this change map with some threshold T.

The above-defined regular contrastive loss suffers from three problems. First, as images cannot be matched perfectly, for example, due to differing occlusion under different viewpoints, aligned images will have unmatched pixels. These pixels should receive a 'don't care' label $y_{i,j}^{(k)} = 2$. The loss function is altered such that pixels with this label do not contribute to the loss, regardless of the value of $D_{i,j}^{(k)}$, which is a simple masking operation on the per-pixel loss with valid mask $M_{i,j}^{(k)} = (y_{i,j}^{(k)} \neq 2)$. The total loss is normalized to the number of valid pixels, to ensure that it does not depend on the number of ‘don’t care’ pixels and the input resolution.

Second, our dataset is severely imbalanced, where less than 1 out of 5,000 pixels is a change pixel, hence class balancing is crucial. Class balancing is implemented as average frequency balancing on a per-image basis. To be partially invariant to object size, the balancing weights are computed per image instead of over the entire training set. This results in balancing weights w_C, where $C == 1$ for changed pixels and $C == 0$ for unchanged pixels, which assign weights to the individual loss terms by the inverse of the frequency of these pixels ($f_C^{(k)}$) as

$$w_C^{(k)} = \frac{K}{f_C^{(k)}} = \frac{0.5}{\varepsilon + \frac{1}{N}\sum_{i,j}(y_{i,j}^{(k)} == C)}, \tag{3}$$

where N is the number of valid pixels ($M_{i,j}^{(k)} == 1$), K is a constant to avoid balancing for a perfectly balanced dataset (0.5 for 2 classes) and ε is a small constant to prevent division by zero.

Third, the margin only affects changed pixels. Unchanged pixels contribute to the loss, even if their distance is already close to zero. This can lead to deteriorated performance, as has been shown for patch matching networks by Lin *et al.* [24]. Because of this behavior, they propose a double-margin contrastive loss, which adds an additional margin to the left side of the loss equation. In contrast to [24], we leave the squaring operations at the same positions as in the regular contrastive loss. This slightly increases computation time, but prevents non-differentiable points and non-convexity in the loss graph.

The final loss function now becomes as follows:

$$\begin{aligned} \mathscr{L}(W,\mathbf{X}) = \frac{1}{2PN}\sum_{k=1}^{P}\sum_{i,j} M_{i,j}^{(k)}\{ \\ (1-y_{i,j}^{(k)})w_0^{(k)}\max(0,(D_{i,j}^{(k)})-m_1)^2 \\ +y_{i,j}^{(k)}w_1^{(k)}\max(0,m_2-D_{i,j}^{(k)})^2\}. \end{aligned} \tag{4}$$

Post-processing

The network architecture is able to distinguish between objects added or removed from a scene, even though this information was not annotated in the training data. The assumption is that a single Siamese network branch learns a low-magnitude response for uninteresting areas and a high-magnitude response for potential suspicious changes (a sort of ‘objectness’ score). Then, if $|\mathbf{v_1}| > |\mathbf{v_2}|$, where $\mathbf{v_1}$ and $\mathbf{v_2}$ are feature vectors for a single pixel of the output of the first (live) and second (reference) branch respectively, the change will have been caused by strong features in the live frame that were not present in the reference. Network responses for which this does not hold can be suppressed, as we are generally not interested in objects removed from the scene. In case an object has been replaced by a different object, the relative feature map magnitudes are no longer informative. To prevent filtering out an arbitrary half of these cases, all pixels of the live frame that have sufficient response by themselves are kept, regardless of the relative magnitude. This results in a post-processing mask M_{post} defined by

$$M_{post} = (|\mathbf{v_1}| > |\mathbf{v_2}|) \vee (|\mathbf{v_1}| > T), \tag{5}$$

where T is the same threshold as the one used for obtaining the final binary change map.

Network Training

All networks are trained with the dual-margin contrastive loss as defined before. The Adam optimizer [25] with learning rate 10^{-4} is used. We initialize the encoder with ImageNet-pretrained ResNet-18 weights and the decoder with Xavier initialization [26]. All networks are trained for 10 epochs.

Batch size is set to unity to allow a large input image resolution. To cope with limited memory, 512×512-pixel crops are used during training for all network comparisons unless otherwise specified. The crops are chosen such that at least one change pixel occurs in each training crop to speed up training. Additionally, on-the-fly data augmentation in the form of random horizontal mirroring is applied. Batch normalization layers in the encoder are allowed to update in a moving-average sense.

Experimental results

This section describes the experimental validation of both our network and several state-of-the-art alternatives. In this work, all networks are implemented in the Caffe framework [27] and the reported test times are achieved on a GTX1080Ti. The first two subsections provide general information about all experiments. Then, the following experiments are described: (1) investigation of the impact of architectural choices on the speed and accuracy of the network , (2) an impact assessment of our changes to the loss function , (3) a comparison to state-of-the-art methods , and (4) experiments to better understand network behavior.

Dataset acquisition

The training and validation dataset consists of four pairs of videos (1,882 image pairs), with a resolution of 1920×1440 pixels, captured in forest-like environments. A single video pair features a recording prior to the placement of test objects (the reference images) and one afterwards (the live images). The 57 unique test objects consist of, amongst others, colored blocks, bottles, rope, and a large tree trunk, which serve as the changes to be detected. Reference images are aligned to their corresponding live images, as in [8], but without optical flow refinement. This alignment cannot match all pixels, hence the unmatched pixels are masked out. The pixel-level annotated train/validation dataset is split randomly into a training set (80%) and validation set (20%).

A separate dataset consisting of three videos (1,884 image pairs) is employed to test the system and highlight the strengths and weaknesses of the network. This involves: (1) a video pair obtained by driving twice the exact same trajectory, resulting in a pair of videos with practically no alignment errors (‘Test Easy Objects Small Misalignment’); (2) a video pair with a deliberate 5-meter offset in driver path between live and reference videos, causing larger alignment artifacts (‘Test Easy Objects Large Misalignment’); (3) a video pair in a dunes environment instead of a forest environment, which features different objects compared to the training set (‘Test Hard Objects’). The three test videos contain 50 unique objects.

Evaluation metrics

The effectiveness of change detection networks is generally reported via F-1 score or Intersection over Union, both computed at pixel level. For fairest comparison to these networks, we also report pixel F-1 score. Additionally, we report object F-1 scores, based on the number of objects detected. In the context of IED detection, the number of correct objects is a far more relevant metric, since it is important to find all objects without being biased to larger objects. The F-1 score is computed by $F_1 = (2 \cdot \text{Precision} \cdot \text{Recall})/(\text{Precision} + \text{Recall})$, where Recall is the fraction of real pixel/object changes that is correctly detected and Precision is the fraction of the total pixel/object detections that is correct.

Speed versus performance trade-off

There are three main choices that affect the speed and performance of the network, in descending order of importance: the choice of encoder network, the cut-off point of the encoder, and the choice of the decoder network. To achieve an extensive trade-off analysis, a parameter sweep over all combinations of these three architectural choices is performed.

For the encoder, both deeper and shallower networks than the theoretically good ResNet-18 are tested: ResNet-10, ResNet-18, ResNet-50, ResNet-101, ResNet-152. Next, each point immediately after a residual block of the ResNet architecture is considered as a cut-off point. Only two decoder options are investigated, the entire ERFNet decoder and a simple bilinear upsampler.

The results of the parameter sweep are shown in Fig. 2. Several interesting interpretations can be made from Fig. 2. First, deeper networks do not necessarily perform better on our dataset, likely because of both the small size of the objects and the small number of frames in the dataset. The deepest networks (those whose timings do not fit on the axes in Fig. 2) also require smaller training crops to fit in memory, which cause them to see less context during training, so that they perform worse. Second, at first sight it may appear that blockwise processing is the cause of reduced F-1 scores. However, this is unlikely, since applying it to the ResNet-10 or ResNet-18 networks reduces F-1 scores by only 0.1%. Blockwise processing primarily negatively affects execution time. Third, the inclusion of the ERFNet decoder blocks improves performance in all cases over a regular bilinear upsampler.

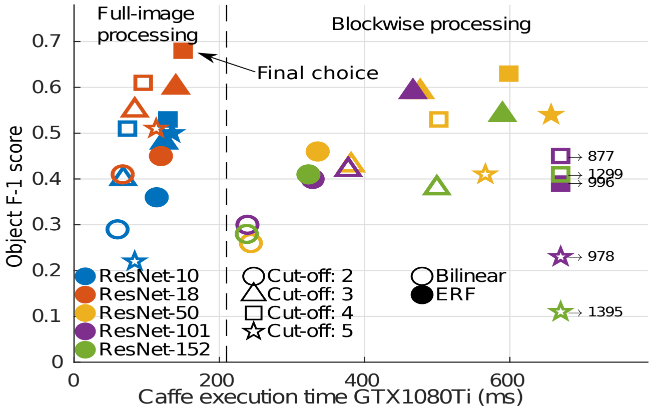

Figure 2. Performance impact of architectural choices of the encoder (marker color), encoder cut-off point (marker shape), and decoder (marker fill). The final network is denoted with an arrow.

Impact of the chosen loss function

The effectiveness of our alterations to the contrastive loss is evaluated in an ablation study. The results are portrayed in Table 1, which shows that class balancing is crucial for good performance. At first glance it appears disabling 'don't care' masking slightly improves performance, albeit by less than one standard deviation. However, disabling 'don't care' masking slows training convergence by almost a factor two, hence enabling it is still a valid improvement. The double-margin addition achieves no obvious gain. Thus, in a pixel-level change detection setting, the singularity problem as mentioned in [24] is apparently much less impactful, probably due to the large number of different background pixels in the dataset. Hence, by Occam's razor, maintaining a single margin is the best solution with fewest parameters.

Next, the proposed contrastive loss is compared with the commonly used softmax cross-entropy loss (with class balancing and masking). Table 1 shows the results. As we expected, the contrastive loss is more powerful for the current architecture.

Table 1. Ablation study of our contrastive loss function.

Loss Function Comparisons	**Object F-1 Validation**	**Object F-1 Test**
Baseline: Our contrastive loss	0.67	0.60
- no class balancing	0.22	0.17
- no 'don't care' masking	0.69	0.60
- no double margin	0.67	0.60
Softmax cross-entropy loss	0.49	0.29

Comparison to the state-of-the-art

The proposed network is compared to the state-of-the-art in both conventional methods and CNNs in Fig. 3: (1) a baseline difference image-based change detection method of Van de Wouw *et al.*, (2) CDNet by Sakurada *et al.* [16], which is a stacked architecture, and (3) a Siamese network by Zhan *et al.* [14], who present the only other Siamese change detection architecture. The implementations have been performed as follows.

Van de Wouw's Adaptive YUV: Van de Wouw's method consists of adaptive thresholding on a difference image in YUV color space [8] with alignment refinement through optical flow.

Zhan's Siamese Network: The network from [14] is implemented and altered to improve its performance on our dataset. We use our improved contrastive loss with class balancing and 'don't care' handling and tune their parameters to our dataset. Their network also results in noise pixels in the change map, which we remove using an additional morphological filtering step.

Sakurada's CDNet: CDNet [16] is implemented without the additional optical flow input, which according to their paper should only change performance by a few percent. The following additional design assumptions are made for parameters not explicitly mentioned in [16]: concatenate-skip connections, ReLU in decoder, convolution padding to maintain feature map size, default dropout parameters, and the Adam optimizer [25] with default parameters. Furthermore, to achieve good results on our dataset, the following additions are necessary: morphological filtering, replacement of the L1 loss by the softmax loss, removal of the deepest three layers, training for four times as many epochs.

The resulting F-1 scores and execution speeds in Frames Per Second (FPS) are shown in Fig. 3. The error bars indicate standard deviations for 5-fold cross-validation. The proposed approach outperforms all earlier work in terms of F-1 score, while having a smaller computational cost than other CNN approaches.

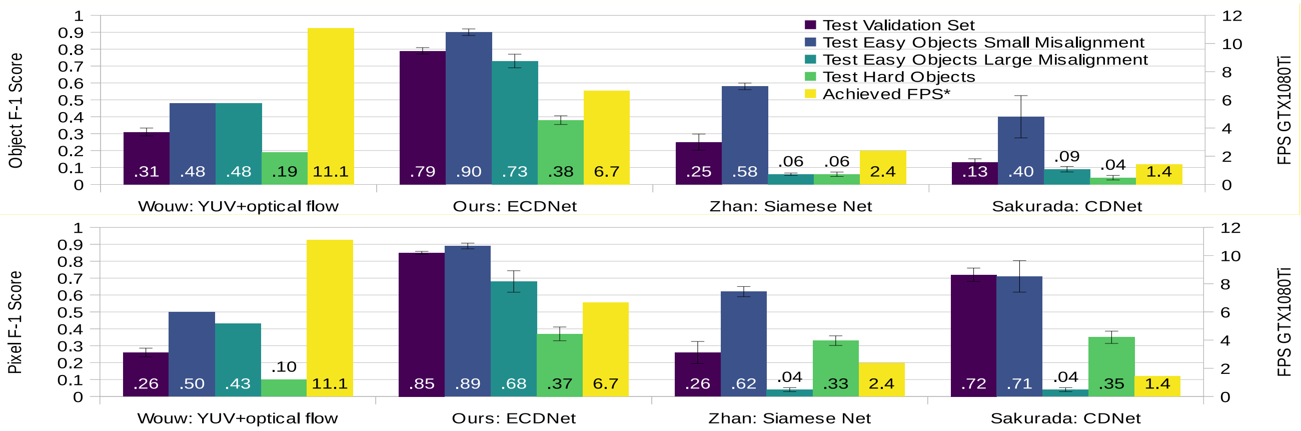

Figure 3. *F-1 scores and execution speed of our network versus state-of-the-art methods when applied to our validation dataset, sorted on speed. *Execution time for Zhan and Sakurada is computed using overlapping 1024x1024 blocks due to memory constraints. With sufficient memory the FPS values would increase to 3.7 and 2.2 respectively.*

Both Zhan's and Sakurada's networks perform poorly under large misalignment errors due to the vast numbers of false positives. In contrast, our network performance is not strongly affected by alignment errors, since a single network branch approximately falls back to being an 'objectness' detector if the misalignment is large. For example, neither grass nor roads result in 'objectness', hence poor alignment that causes these two to overlap does not result in false positives in our network.

Our network's pixel F-1 score on the 'Hard Objects' video is similar to Zhan's and Sakurada's networks, because ours has difficulty finding all pixels of a large change. This means our network likely contains less scale invariance than Sakurada's and Zhan's. The pixel F-1 scores of their networks on the 'Hard Objects' video are high due to a few well-detected large objects. Generally Sakurada's network is better at finding an accurate object outline, while our network is better at finding objects as blobs, not necessarily with an accurate outline.

Are we just detecting objects?

It is possible to use a single branch of the network and ignore the output of the other, resulting in an object detector. Comparing the use of the network as an object detector versus change detection mode shows that change detection achieves higher object F-1 scores than pure object detection by as little as 5% for the 'Large Misalignment' video, up to as much as 21% for the 'Hard Objects' test video. Manual inspection of the frames reveals that false positives are primarily caused by non-objects looking like objects in either the live or reference frames, but not in both. These types of false positives indeed cannot be filtered by comparing 'objectness'. Furthermore, certain shapes of sharp hard shadow edges are sometimes detected as false positives, though in general the network is robust to lighting differences and shadows. These results appear to confirm our suspicion that the network has primarily learned to be an 'objectness' detector that simply compares the 'objectness' score between two images to determine changes. This also explains the robustness of the system to most dynamic backgrounds. A significant disadvantage to change detection based on 'objectness' is that an object that is replaced by a different, yet same-sized object will not be detected as a change.

Discussion

Despite its good performance on our dataset, the trained network also contains several limitations. First, the network is closer to an object detector than a pure change detector, as the decision for change is mostly based on 'objectness'. This causes the network to fail in recognizing the change type 'object replaced by a different object', as both objects will have a high 'objectness' score. The network has never learned this change type, because it is not present in the training set.

Second, almost all objects in the dataset are non-natural objects that would normally not belong in the scene. This is not necessarily the case for general IED markers in practice. It is unlikely that the trained network will perform well on more natural changes, as the current ground-truth annotations mark all natural changes as 'no change', which means the network is rewarded for suppressing them actively.

Finally, while some of our improvements to the contrastive loss are powerful, the addition of the second margin is ineffective for change detection. Since the single-margin loss is both simpler and achieves equal results (for pixel and object F-1 scores), we recommend employing a single margin.

Conclusion

We are the first to propose a neural network architecture for object-like change detection from high-resolution ground-based vehicle imagery in real time, including the following contributions. First, we have designed an efficient Siamese encoder-decoder network based on ResNet-18 and ERFNet that significantly outperforms state-of-the-art change detection networks for IED marker detection. Second, we have developed extensions to the contrastive loss function, by making it applicable to pixel-level problems, adding class balancing, and allowing 'don't care'–label processing. These extensions are suitable for improving performance of any Siamese change detection network. Our architecture can be useful for general mobile change detection platforms with imperfect alignment.

Our experimental results have shown that for small-object change detection, our network outperforms the state-of-the-art considerably on our datasets, both in terms of pixel F-1 score and object F-1 score. Furthermore, the network is only slightly slower than a non-CNN baseline and more than a factor two faster than other CNN approaches.

Future work can focus on expanding the dataset, such as through artificial data, and on investigating the generalization strength over more diverse object types.

Overall, we conclude that the application of deep learning for roadside IED detection can significantly improve robustness of existing counter-IED change detection systems.

Acknowledgment

The authors would like to thank ViNotion for providing datasets and the hardware used for training the networks.

References

[1] R. Hoencamp, E. P. Huizinga, T. T. C. F. Van Dongen, F. J. Idenburg, A. Ramasamy, L. P. H. Leenen, and J. F. Hamming, "Impact of explosive devices in modern armed conflicts: in-depth analysis of Dutch battle casualties in southern Afghanistan," *World J. Surg.*, vol. 38, no. 10, pp. 2551–2557, 2015.

[2] D. W. J. M. van de Wouw, G. Dubbelman, and P. H. N. D. With, "On improving IED object detection by exploiting scene geometry using stereo processing," in *Video Surveill. Transp. Imaging Appl.*, vol. 9407, 2015.

[3] N. J. Mcneese, N. J. Cooke, R. Branaghan, A. Knobloch, and A. Taylor, "Identification of the emplacement of improvised explosive devices by experienced mission payload operators," *Appl. Ergon.*, vol. 60, pp. 43–51, 2017.

[4] M. Wathen, N. Link, P. Iles, J. Jinkerson, P. Mrstik, K. Kusevic, and D. Kovats, "Real-time 3D change detection of IEDs," in *Proc. SPIE*, vol. 8379, 2012.

[5] T. Müller, "Change detection on UGV patrols with respect to a reference tour using VIS imagery," in *SPIE Defense+ Secur.*, vol. 9460, 2015.

[6] M. Tektonidis and D. Monnin, "Color consistency and local contrast enhancement for a mobile image-based change detection system," *J. Imaging*, vol. 3, no. 3, 2017.

[7] D. W. J. M. van de Wouw, K. van Rens, H. van Lint, E. G. T. Jaspers, and P. H. N. de With, "Real-time change detection for countering improvised explosive devices," in *Video Surveill. Transp. Imaging Appl.*, vol. 9026, 2014.

[8] D. W. J. M. van de Wouw, F. B. ter Haar, G. Dubbelman, and P. H. N. de With, "Development analysis of a real-time demonstrator for automated detection of IED from ground vehicles," *Manuscript submitted for publication.*, 2019.

[9] H. Haberdar and S. K. Shah, "Disparity map refinement for video based scene change detection using a mobile stereo camera platform," in *20th Int. Conf. Pattern Recognit.*, 2010, pp. 3890–3893.

[10] G. Saur and W. Krüger, "Change detection in UAV video mosaics combining a feature based approach and extended image differencing," *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.*, vol. 41, pp. 557–562, 2016.

[11] D. van de Wouw, G. Dubbelman, and P. de With, "Hierarchical 2.5D scene alignment for change detection with large viewpoint differences," *IEEE Robot. Autom. Lett.*, vol. 1, no. 1, pp. 361–368, 2016.

[12] Jia Deng, Wei Dong, R. Socher, Li-Jia Li, Kai Li, and Li Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *2009 IEEE Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 248–255.

[13] B. Hou, Y. Wang, and Q. Liu, "Change detection based on deep features and low rank," *IEEE Geosci. Remote Sens. Lett.*, vol. 14, no. 12, pp. 2418–2422, 2017.

[14] Y. Zhan, S. Member, K. Fu, M. Yan, X. Sun, H. Wang, and X. Qiu, "Change detection based on deep siamese convolutional network for optical aerial images," *IEEE Geosci. Remote Sens. Lett.*, vol. 14, no. 10, pp. 1845–1849, 2017.

[15] P. F. Alcantarilla, S. Stent, G. Ros, R. Arroyo, and R. Gherardi, "Street-view change detection with deconvolutional networks," in *Int. Conf. Robot. Sci. Syst.*, 2016.

[16] K. Sakurada, W. Wang, N. Kawaguchi, and R. Nakamura, "Dense optical flow based change detection network robust to difference of camera viewpoints," in *arXiv:1712.02941*, 2017.

[17] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *Neural Inf. Process. Syst. Conf.*, 2014.

[18] A. Canziani, A. Paszke, and E. Culurciello, "An analysis of deep neural network models for practical applications," *arXiv:1605.07678v4*, 2017.

[19] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," Tech. Rep., 2015.

[20] E. Romera, J. M. Alvarez, L. M. Bergasa, and R. Arroyo, "ERFNet: Efficient residual factorized ConvNet for real-time semantic segmentation," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 1, pp. 263–272, 2018.

[21] S. Zagoruyko and N. Komodakis, "Deep compare: A study on using convolutional neural networks to compare image patches," *Comput. Vis. Image Underst.*, vol. 164, pp. 38–55, 2017.

[22] R. Hadsell, S. Chopra, and Y. Lecun, "Dimensionality reduction by learning an invariant mapping," in *CVPR'06*, vol. 2, 2006.

[23] S. Bell and K. Bala, "Learning visual similarity for product design with convolutional neural networks," *ACM Trans. Graph.*, vol. 34, no. 4, pp. 98:1–98:10, 2015.

[24] J. Lin, O. Morere, V. Chandrasekhar, A. Veillard, and H. Goh, "DeepHash: Getting regularization, depth and fine-tuning right," *arXiv:1501.04711*, 2015.

[25] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *ICLR 2015*.

[26] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. Thirteen. Int. Conf. Artif. Intell. Stat.*, vol. 9, 2010, pp. 249–256.

[27] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," *arXiv:1408.5093*, 2014.

Author Biography

Sander Klomp received both his BSc and MSc degrees from the Eindhoven University of Technology (2016,2018) with the designation Cum Laude. He is now pursuing a PhD degree at TU/e in collaboration with ViNotion, with a focus on efficient deep learning algorithms.

Dennis van de Wouw received his MSc degree from the Eindhoven University of Technology in 2011. As the change detection system architect at ViNotion, he is now finalizing his PhD thesis on depth-based extensions for change detection, in close collaboration with TU/e.

Peter H.N. de With is Full Professor of the Video Coding and Architectures group in the Department of Electrical Engineering at Eindhoven University of Technology. He is an IEEE Fellow, has (co-)authored over 400 papers on video coding, analysis, architectures, and 3D processing and has received multiple papers awards. He is a program committee member of the IEEE CES and ICIP and holds some 30 patents.