Enabling Image Recognition on Constrained Devices Using Neural Network Pruning and a CycleGAN

Lidfelt, August; Isaksson, Daniel; Hedlund, Ludwig; Åberg, Simon; Borg, Markus; Larsson, Erik

Published in:
1st international workshop on Internet of Things for Emergency Management

DOI:
10.1145/3423423.3423437

2020

Citation for published version (APA):
Lidfelt, A., Isaksson, D., Hedlund, L., Åberg, S., Borg, M., & Larsson, E. (2020). Enabling Image Recognition on Constrained Devices Using Neural Network Pruning and a CycleGAN. In *1st international workshop on Internet of Things for Emergency Management* Association for Computing Machinery (ACM). https://doi.org/10.1145/3423423.3423437

Total number of authors:
6

Enabling Image Recognition on Constrained Devices Using Neural Network Pruning and a CycleGAN

August Lidfeldt*
Dept. of Computer Science, Lund University
Lund, Sweden
august.lidfeldt@gmail.com

Daniel Isaksson*
Dept. of Computer Science, Lund University
Lund, Sweden
daniel.g.isaksson@gmail.com

Ludwig Hedlund*
Dept. of Computer Science, Lund University
Lund, Sweden
ludwighedlund@outlook.com

Simon Åberg*
Dept. of Computer Science, Lund University
Lund, Sweden
simon.aberg95@gmail.com

Markus Borg
RISE Research Institutes of Sweden AB, Lund
Lund, Sweden
markus.borg@ri.se

Erik Larsson
Dept. of Electrical and Information Technology, Lund University
Lund, Sweden
erik.larsson@eit.lth.se

ABSTRACT

Smart cameras are increasingly used in surveillance solutions in public spaces. Contemporary computer vision applications can be used to recognize events that require intervention by emergency services. Smart cameras can be mounted in locations where citizens feel particularly unsafe, e.g., pathways and underpasses with a history of incidents. One promising approach for smart cameras is edge AI, i.e., deploying AI technology on IoT devices. However, implementing resource-demanding technology such as image recognition using deep neural networks (DNN) on constrained devices is a substantial challenge. In this paper, we explore two approaches to reduce the need for compute in contemporary image recognition in an underpass. First, we showcase successful neural network pruning, i.e., we retain comparable classification accuracy with only 1.1% of the neurons remaining from the state-of-the-art DNN architecture. Second, we demonstrate how a CycleGAN can be used to transform out-of-distribution images to the operational design domain. We posit that both pruning and CycleGANs are promising enablers for efficient edge AI in smart cameras.

CCS CONCEPTS

• **Computing methodologies → Activity recognition and understanding**; **Object recognition**; **Neural networks**.

KEYWORDS

smart camera, image recognition, neural network pruning, generative adversarial network, edge AI

ACM Reference Format:
August Lidfeldt, Daniel Isaksson, Ludwig Hedlund, Simon Åberg, Markus Borg, and Erik Larsson. 2020. Enabling Image Recognition on Constrained Devices Using Neural Network Pruning and a CycleGAN. In *10th International Conference on the Internet of Things Companion (IoT '20 Companion), October 6–9, 2020, Malmö, Sweden.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3423423.3423437

1 INTRODUCTION

According to a study conducted in 2019 by The Swedish National Council for Crime Prevention, 28% of Swedes felt unsafe walking outside in their own neighbourhood at night. This number marks an increase from 21% in 2013 and is indicative of the larger trend of lower perceived safety. The same study also reported an increasing concern about crime in society from 28% in 2013 to 43% in 2019 [18]. The trend leads to the larger question of how to create safe societies.

A measure that often is proposed is strategic placement of cameras in public spaces [5, 7]. Although cameras are increasingly used in surveillance solutions, operators typically are required to detect and classify incidents. AI and machine learning (ML) enable smart cameras [14] that allow automatic recognition of events that require intervention by emergency services. Large scale camera deployment increase the bandwidth requirements, leading to a need to distribute computation to the cameras themselves. Edge AI is based on the idea of decentralized computational platforms, where AI technology such as image recognition is incorporated directly in IoT devices [21].

Image recognition on the constrained edge devices introduces fundamental trade-offs between performance and efficiency. The pursuit of high accuracy image recognition has lead to ever-growing deep neural network (DNN) architectures. State-of-the-art DNN architectures contain trainable parameters in the magnitude of hundreds of millions, which requires considerable computational power and energy. To mitigate the issue, several studies have investigated neural network pruning, i.e., decreasing the size of DNNs by reducing the number of trainable parameters while trying to retain model accuracy [9].

Robustness is another essential quality attribute in image recognition, especially for critical emergency response applications. For a trained DNN, robustness involves handling perturbations or input

*These authors contributed equally to this research.

IoT '20 Companion, October 6–9, 2020, Malmö, Sweden
© 2020 Association for Computing Machinery.
ACM ISBN 978-1-4503-8820-7/20/10…$15.00
https://doi.org/10.1145/3423423.3423437

data that is does not closely resemble the training data – input referred to as being out-of-distribution (OOD) [12]. In the context of smart cameras, perturbations might include dirty or vandalized camera domes. An example that could lead to OOD input would be deployment of the camera in an environment that does not reflect the training data, e.g., due to differences in illumination. Inspired by work in the automotive domain [19], we propose using a CycleGAN to perform style transfers of OOD input.

In this paper, we study a motion-activated network camera mounted in an underpass located in Helsingborg, Sweden. We use the camera input to train various DNNs for image recognition, using the well-known VGG16 [22] as the baseline. In this preliminary work, we discuss an application of multi-class classification, i.e., detecting the presence of pedestrians, dog walkers, and bicyclists. Again inspired by automotive engineering, we specify the operational design domain (ODD) [8] of our classification model to cover daytime conditions. Two research questions (RQ) guide us:

RQ1 How does neural network pruning of a state-of-the-art DNN architecture affect the classification accuracy?

RQ2 How can a CycleGAN be used to transfer OOD input to the ODD of an image recognition application?

Our results show that substantial pruning of VGG16 is possible in our case under study. Given the ODD, i.e., homogeneous daytime conditions in the underpass, we obtain a classification accuracy above 90% despite pruning the DNN to contain only 1.1% of the trainable parameters. Second, as a proof-of-concept, we report how style transfers from nighttime to daytime conditions improves the classification accuracy of OOD input images – the CycleGAN might thus extend the ODD of the smart camera by performing classification beyond its underlying training data. We hypothesize that style transfers can be an effective and efficient way to enable smart cameras to operate in environments for which their embedded DNNs were not trained.

The rest of this paper is organized as follows. Section 2 introduces fundamental concepts related to image classification and CycleGANs. In Section 3, we present the overall ML workflow and two experiments that address the RQs. Section 4 presents the results and the discussion follows in Section 5. Section 6 reports the main threats to validity. Finally, Section 7 concludes the paper and outlines directions for future work.

2 BACKGROUND

Thanks to DNNs and massive datasets, image recognition has been reported to outperform humans on specific tasks in the last decade. A key component in this development is convolutional neural networks (CNN) [6, pp. 321-362]. The main purpose of CNNs is to extract key features from images in a computationally efficient way. CNNs are generally made up of several different types of layers with two of the most important being fully connected layers and convolutional layers.

A large computational expense in traditional neural networks stems from the heavy reliance on fully connected layers, where each neuron is connected to all the neurons in the previous layer. CNNs combine fully connected layers with convolutional layers in which each neuron only is connected to a small region of the input volume and thereby greatly reduces the number of trainable parameters and the computational cost. Convolutional layers iterate sequentially over sections of the image and produces an edge feature representation derived from contrasts in light and color. From this the network will learn which specific features at a given spatial position of the input that should trigger an activation.

In this work, we use VGG16 as the baseline DNN architecture – a well-known convolutional DNN that has obtained accurate results in recognized competitions, e.g., the ImageNet Large Scale Visual Recognition Competition[22].

Generative Adversarial Networks (GAN) consist of two competing models, a generator and a discriminator [6, pp. 690-693]. The generator generates fake samples of data, images in our case, while the discriminator's purpose is to distinguish if the samples come from the original dataset or are generated by the generator. These models get updated by an *adversarial loss* where the generator tries to maximize the discriminator probability of incorrectly labeling the generated sample as fake. The discriminator on the other hand tries to minimize the same object function. When these models are successfully trained against each other the generator learns to produce random samples indistinguishable from the original training collection.

Zhu *et al.* proposed CycleGANs, an architecture that combines two GANs [25]. CycleGANs have been trained to transform images between domains while preserving the images' specific features. A CycleGAN is trained on unlabeled datasets from the chosen domains, i.e., unpaired image to image translation. For example, if camera images in rainy conditions are underrepresented in the training set, existing images from the sunny domain can be transformed to the rainy domain as an approach to data augmentation [13]. Moreover, CycleGANs can be used during operation to transform real-time input data that does not resemble the training data to images that are within the ODD [1].

A CycleGAN is, as the original GAN, optimized by an *adversarial loss* but the CycleGANs generators have three additional losses, *identity loss*, *forward-* and *backward cycle consistency loss*. The *identity loss* is calculated by the difference between input and output when the input for the generator is in the same domain as the target domain. The *forward-* and *backward cycle consistency loss* is important to ensure that the output keeps the specific features of the input image and not only generates a sample that resembles the target domain. These losses are calculated when you combine the two different generators in a cycle which put the input and output in the same domain and then calculate the difference between input and output.

3 METHOD

This section describes the training data, the machine learning process, and the experimental setup. A complete replication package is available on GitHub[1].

3.1 End-to-End Machine Learning Process

Figure 1 shows how data were collected and processed. First, we collected 90 minutes of high resolution video clips from the underpass (A) with an average video clip length of 52 seconds. Note that all recordings were action-triggered, i.e., all clips contain activity in

[1] https://github.com/luuddan/EITN35

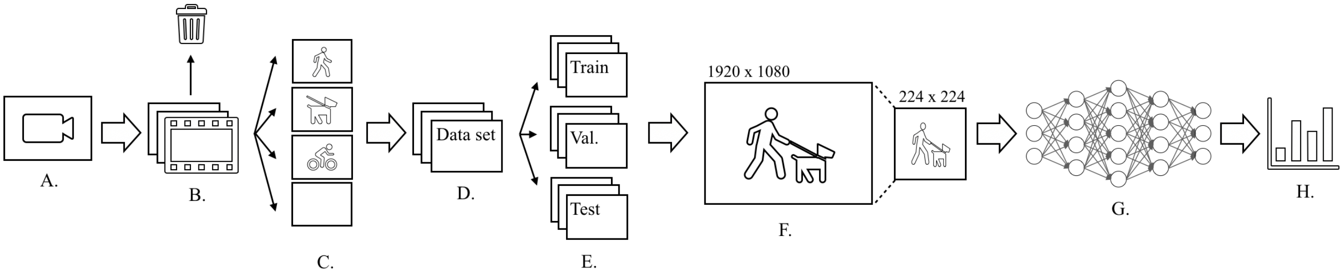


Figure 1: Overview of the end-to-end machine learning process.

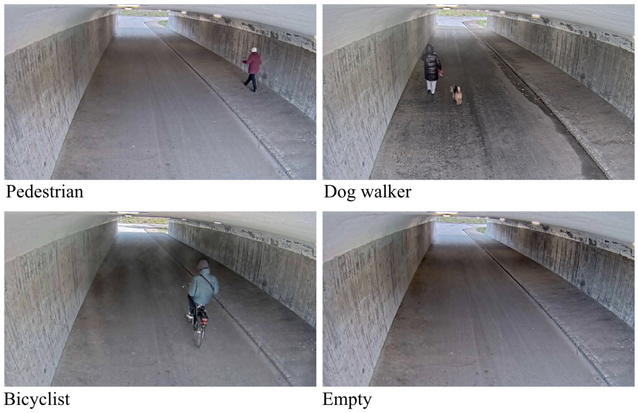


Figure 2: Example images of each class.

the underpass. All raw videos were recorded during five days in the spring of 2020 with a fixed camera angle as presented in Figure 2.

We split the video clips into individual frames (B) at a frame rate of one per second. After step B), we had a dataset containing 5,978 single images. The first four authors manually annotated the images (C) with one of the four labels 1) pedestrian, 2) dog walker, 3) bicyclist, or 4) empty. Images that did not fit into any of these classes or that contained more than one instance of a label were removed in this step. Images that did not match our quality criteria also were excluded, e.g., objects in the far end of the underpass and objects only partially visible. After the labelling step, the final dataset (D) consisted of 180 dog walkers, 904 pedestrians, and 253 bicyclists. To maintain a useful class distribution, we kept only 904 images with the empty label.

In line with standard practice in machine learning, we randomly split the dataset (E) into a training set (64 %), a validation set (16 %), and a test set (20 %). As a final step before using the images as input to the DNN, we downscaled them (F) from a resolution of 1920x1080 to 224x224 pixels.

We trained a DNN for the multi-class classification task (G) using VGG16 as the baseline DNN architecture (G). VGG16 is composed of 16 layers with trainable parameters, whereof the first 13 are convolutional (followed by max-pooling layers) and the last three are fully-connected. The total number of trainable parameters in VGG16 is 134 million. We use early stopping to mitigate overfitting when training all models.

3.2 Neural Network Pruning (RQ1)

Experiment A investigates what effect the number of trainable parameters the DNN architecture has on the classification accuracy. Our baseline architecture (Arch11) is structurally identical to the VGG16 model. We then created 10 additional architectures (Arch1-10) by iteratively reducing the number of parameters by (roughly) 50% in each step. The reduction approach consisted of alternating between removal of convolutional layers, reducing the number of filters in the convolutional layers, and shrinking the dense layers. Table 1 presents an overview of the 11 architectures. Note that after training classification models using these architectures, we instead refer to them as models (M1-M11).

Experiment A uses a full factorial design with two independent variables with discrete values. First, the *DNN architecture* is varied by training classification models using the 11 DNN architectures listed in Table 1. Second, we varied the *amount of training data* by creating subsets containing 25%, 50%, and 75% of the final dataset. The subsets were created through random stratified sampling, i.e., the dataset was split into new training, validation, and test sets while retaining the original class distributions.

For the most promising DNN architecture from experiment A, we performed hyperparameter tuning using grid search, i.e., we evaluated a manually specified subset of the hyperparameter space of the learning algorithm. We used the one-factor-at-a-time method for the tuning, i.e., we did not investigate interaction effects [3].

3.3 CycleGAN Transformation to ODD (RQ2)

We designed Experiment B to act as a proof-of-concept for the approach to use a CycleGAN to transform OOD input to the ODD. In our case, we explore whether a CycleGAN can transform input images from the nighttime domain to the daytime domain.

We used an open source implementation[2] of the CycleGAN architecture proposed by Zhu *et al.* [25]. The architecture consists of two discriminator models and two generator models. The discriminator models consist of 5 convolutional layers. For input, the discriminator models take both real and generated images and outputs a binary value reflecting whether the input was real or fake. We used a mean square error loss as the loss function for the discriminator model.

The generator models are using an encoder-decoder approach. Input images are downsampled to extract the features and then

[2] https://machinelearningmastery.com/cyclegan-tutorial-with-keras/

Table 1: Characteristics of the 11 investigated DNN architectures. Arch11 is the VGG16 baseline.

Arch1	Arch2	Arch3	Arch4	Arch5	Arch6	Arch7	Arch8	Arch9	Arch10	Arch11	
0.1 M	0.2 M	0.4 M	0.8 M	1.5 M	3 M	6 M	13 M	35 M	67 M	134 M	#Parameters
2	3	3	4	4	4	4	8	10	11	13	#Convolutional layers
8	16	32	128	128	256	256	512	1,024	2x2,048	2x4,096	#Dense layers

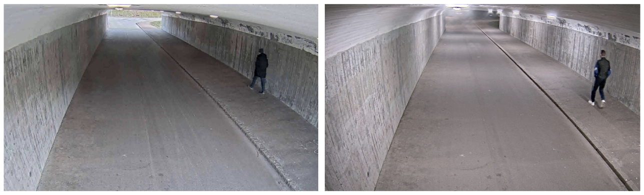


Figure 3: Illumination in the daytime domain (left) and the nighttime domain (right)

Table 2: Distribution of images in the three test sets.

		Domain		
Test set	**Label**	**Day**	**Night**	**Night2Day**
PedSet	Pedestrian	180	106	106
BikeSet	Bicyclist	50	60	60
EmpSet	Empty	180	106	106

upsampled into a new image in the new domain based on the extracted features.

The encoder consists of three convolutional layers followed by a section of six ResNet blocks which are used in deep neural networks for convergence while avoiding exploding or vanishing gradients[11]. Next, the decoder follows, consisting of two transpose convolutional layers, i.e. convolutional layers upscaling the resolution as opposed to normal convolutional layers which downscale it. Lastly a final normal convolutional layer follows.

We trained the CycleGAN using a dataset containing 1,128 images from the underpass with objects close to the camera, i.e., a sample from step B in Figure 1. The dataset contained an equal share of images from the daytime domain and the nighttime domain. Figure 3 illustrates how the illumination differs in the two domains.

Experiment B constituted an evaluation of M5 from Experiment A, trained on 100% of the training data (step D in Figure 1). Table 2 describes the three new test sets we created for this evaluation, containing images with pedestrians (PedSet), bicyclists (BikeSet), and empty underpass (EmpSet), respectively. Each test set contained a combination of randomly sampled images from the CycleGAN dataset from the daytime and nighttime domains. Furthermore, we used the trained CycleGAN to transform the nighttime images to the daytime domain (cf. Night2Day in Table 2). Two examples of transformed images are presented in Figure 4.

4 RESULTS

This section presents experimental results concerning the two RQs.

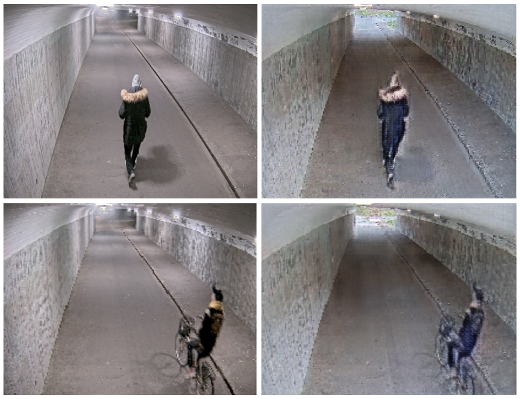


Figure 4: Images transformed from nighttime (left) to daytime (right).

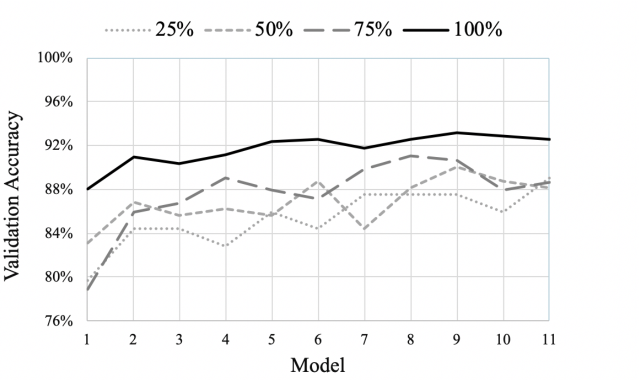


Figure 5: Validation accuracy per model. The four lines show the size of the datasets.

4.1 Neural Network Pruning (RQ1)

Figure 5 shows the classification accuracy on the validation set from Experiment A. The X-axis shows classification models trained according to the architectures listed in Table 1. The models shall be considered on an ordinal scale, with increasing numbers of trainable parameters toward the right.

The four lines represent validation accuracy for different dataset sizes. The solid black line, corresponding to 100% of the dataset, displays as expected the best results. As all lines show increasing trends, the results suggest that more complex DNN architectures result in more accurate object recognition in the underpass.

On the other hand, training models with more data influences the accuracy more than having more complex DNN architectures. The

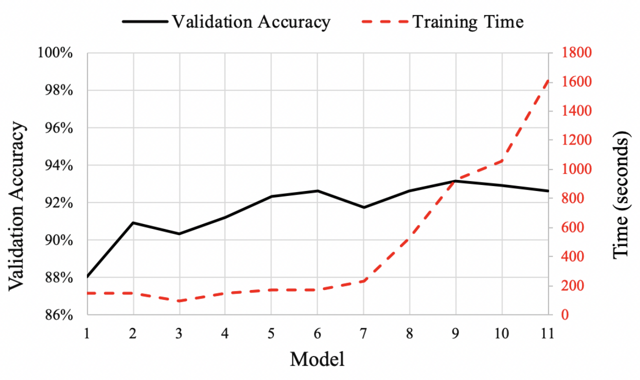


Figure 6: Validation accuracy per model with respective training time

Table 3: Hyperparameters evaluated during tuning and the final setting in bold font.

Hyperparameter	Values
Learning rate	5E-5, 1E-4, 5E-4, **0.001**, 0.005, 0.01
Dropout rate	0, 0.1, ... 0.45, **0.5**, 0.55, ... 0.65
L2 Regularization rate	**0**, 1E-4, 5E-4, 0.001, 0.005, 0.01, 0.1

second smallest model (M2) using 100% of the dataset outperforms all of the more complex models using 25%-75% of the data (except M8 that obtained almost the same validation accuracy). This observation is in line with previous work on object recognition [16, 22], and demonstrates the potential of pruning DNN architectures while retaining acceptable accuracy.

Figure 6 depicts the training time for the different models. The results correspond to training using the complete dataset until early stopping (avg. #epochs=115, SD=16). All measurements are reported in seconds, as shown on the Y-axis to the right. The results show that the training times drastically increase with more complex DNN architectures, i.e., from M8 containing 65M trainable parameters.

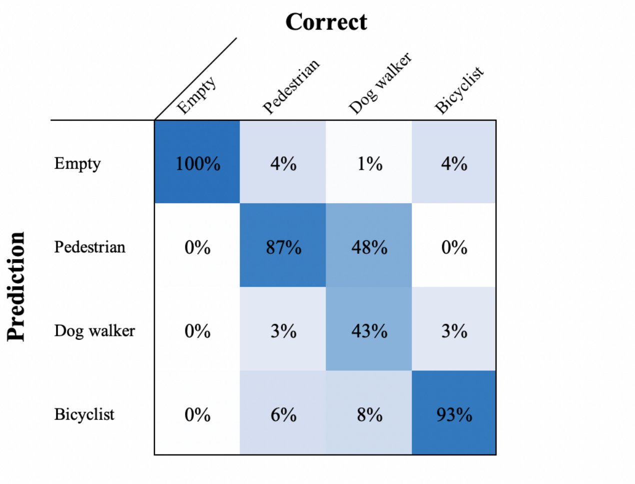
Based on the results from Experiment A, we selected M5 for further development and evaluation. Our rationale was that M5 was the smallest model that performed within 1% of the baseline VGG16 architecture (M11) – M5 has 1.5M trainable parameters, corresponding to 1.1% of M11. Table 3 lists the hyperparameters that we tuned for M5, the values we evaluated, and the final settings we used. Our final results after hyperparameter tuning on the training set and the validation set were 98.8% and 93.5%, respectively.

Table 4 shows the classification accuracy on the test set. M5 obtained an overall accuracy of 91.0%. Looking at the individual classes, we note the least accurate results for the dog walker class (57.1%) whereas the empty underpass was no problem for the classifier.

Figure 7 presents a confusion matrix, enabling further examination of erroneous predictions. The confusion matrix shows that for input of the class dog walker, the classifier could not properly distinguish between pedestrians (48%) and the correct class (43%). We report two possible explanations. First, the proportion of dog walkers in the training set was low (8%). Second, the differences between some images of pedestrians and dog walkers are indeed minor, e.g., small dog breeds on a leash.

Table 4: M5 Classification Accuracy on the test set.

Class	Accuracy
Total	91.0%
Empty	100.0%
Pedestrian	92.0%
Dog walker	57.1%
Bicyclist	80.0%



	Empty	Pedestrian	Dog walker	Bicyclist
Empty	100%	4%	1%	4%
Pedestrian	0%	87%	48%	0%
Dog walker	0%	3%	43%	3%
Bicyclist	0%	6%	8%	93%

Figure 7: Confusion matrix for M5

4.2 CycleGAN Transformation to ODD (RQ2)

Figure 8 reports the classification accuracy of M5, after hyperparameter tuning, on PedSet, BikeSet, and EmpSet as well as the overall result. For EmpSet, the different illuminations in the daytime and nighttime domains yield completely opposite results. All predictions in daytime are correct, but none in nighttime. However, after the Night2Day transformation back to the ODD, M5 again obtains a perfect result.

The classification results for PedSet and BikeSet are contrasting. For PedSet, M5 obtains 90% and 80% classification accuracy for the daytime and nighttime domains, respectively. The accuracy for the images transformed to the ODD using Night2Day was only 65%. The results for BikeSet, on the other hand, were orthogonal. M5 obtains 60% accuracy for the daytime domain and 35% for nighttime, but the Night2Day transformation enables a substantial improvement – 75% of the bicyclists are correctly classified. We consider this a promising proof-of-concept for ODD extension using a CycleGAN, i.e., input images that do not resemble the training data can be be transformed to the ODD.

Figure 9 presents confusion matrices for the nighttime domain and the images transformed from night to day, respectively. For the nighttime domain, we notice that an empty underpass in most cases (97%) resulted in the pedestrian label, i.e., the M5 classifier

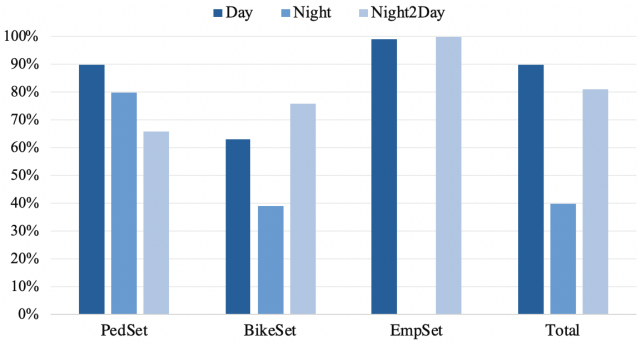
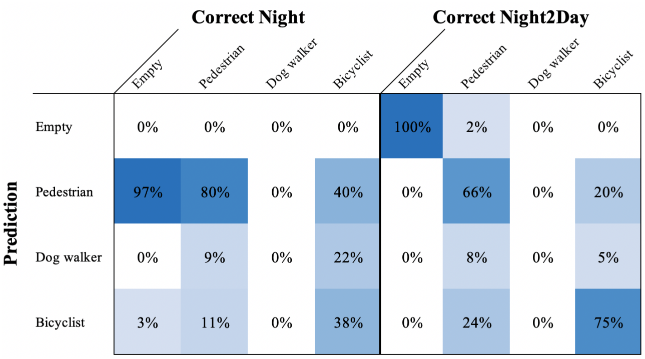


Figure 8: Classification accuracy for M5 on EmpSet, PedSet, BikeSet, and results for the combined test sets (Total).



		Correct Night				Correct Night2Day			
		Empty	Pedestrian	Dog walker	Bicyclist	Empty	Pedestrian	Dog walker	Bicyclist
Prediction	Empty	0%	0%	0%	0%	100%	2%	0%	0%
	Pedestrian	97%	80%	0%	40%	0%	66%	0%	20%
	Dog walker	0%	9%	0%	22%	0%	8%	0%	5%
	Bicyclist	3%	11%	0%	38%	0%	24%	0%	75%

Figure 9: Confusion matrices for M5 on images in the nighttime domain (left) and images transformed from night2day (right).

identified features suggestion people in the underpass background. Furthermore, we highlight that M5 predicted input of bicyclists in the nighttime domain as pedestrians, dog walkers, and bicyclists rather arbitrarily. Figure 9 shows how the Night2Day transformation mitigated this, as 75% of the bicyclists were correctly classified. Moreover, after the Night2Day transformation, M5 identified no pedestrians in input images containing an empty underpass.

5 DISCUSSION

Enabling dependable image recognition on edge devices is an important topic to realize IoT solutions for emergency management. Our study demonstrates two approaches that make applications of edge AI practically feasible on constrained devices in a controlled environment such as an underpass.

Neural network pruning can be used to substantially reduce the size of a DNN model for image recognition. Reduced DNNs lead to reduced needs for computation as well as limited energy consumption [10]. In line with previous work [2, 15], we found that substantially smaller DNNs can perform comparably in terms of classification accuracy. We hypothesize that the pruning worked particularly well in our case due the relatively low complexity in the recognition task, a low number of classes, and the static image background. VGG16, on the other hand, was developed to classify 1,000 labels in arbitrary input. For our specific application, deploying VGG16 on an edge device would have constituted considerable over-engineering.

Changing the DNN size between Arch11's maximum of 134 million trainable parameters down to 0.1 million for Arch1 did have an effect on our validation accuracy, but not as large of a change as one might expect. Decrease the number of trainable parameters by three magnitudes only resulted in a validation accuracy drop of 4.5%. While our results show the potential of pruning, finding the appropriate balance between DNN size and accuracy is truly application specific – conflicting quality requirements must be managed.

Our results also highlight the trade-off between classification accuracy and training time (cf. Figure 5). Moreover, while the training must not necessarily be performed on edge devices, there is growing interest in federated learning [24], a privacy-preserving technology highly relevant to surveillance applications [23].

We demonstrate a novel application of CycleGANs in the context of edge AI. Instead of collecting additional training data to extend the ODD of the smart camera, we used a CycleGAN to transform OOD input to the ODD. As CycleGANs learn style transformations from unpaired training data, this might enable a cost-efficient approach to ODD extension. In our case under study, we train a CyclaGAN to perform style transfers between the daytime and the nighttime domains.

In the underpass, we specified the ODD of the image recognition to perform classification in daytime conditions. In Experiment A, we trained a DNN model accordingly and report satisfactory results (cf. Table 4). In Experiment B, we illustrate the limited robustness of the DNN model as it underperforms on OOD input, i.e., nighttime images. Subsequently, we used the CycleGAN to transform nighttime images to the daytime domain and reclassify the input.

While the results are inconsistent across the different classes (cf. Figure 8), we argue that the overall results indicate that the approach is promising: CycleGANs can make OOD input fit the ODD. Thanks to a learned style transfer, a small DNN model operating in tandem with a CycleGAN might be able to make predictions for input that goes beyond its training data, i.e., increasing the robustness of image recognition on constrained devices.

6 THREATS TO VALIDITY

All empirical research is subject to threats to validity. While the results reported in this study are preliminary, we report the main threats as our findings guide our future work on smart cameras for emergency management.

External validity reflects the generalizability of our results. Our initial work targets standard classes in image recognition, thus future work is needed to investigate how our findings extrapolate to classes customized for emergency management, e.g., person on the ground, a brawl, or bicycle accidents. Moreover, we did not study camera input with more than one class present at the same time. Future work should explore more complex activities in the underpass.

Furthermore, underpasses provide homogeneous environments for image recognition and our results cannot be extrapolated to less controlled public spaces such as pathways in parks. However, underpasses are prioritized locations for camera surveillance as they are known to be emergency hotspots. Finally, all video was recorded in the spring, thus we need to extend the dataset to cover seasonal variations.

Internal validity concerns casual relationships and potentially confounding factors. We report that neural network pruning and CycleGANs are promising approaches to enable efficient image recognition on edge devices. However, our conclusion is based on training DNNs using small datasets. It is possible that pruning would be less useful if the dataset was magnitudes larger. Moreover, perhaps a larger training set would also make the DNN robust enough to make CycleGAN transformations to the ODD superfluous.

7 CONCLUSION AND FUTURE WORK

Edge AI paves the way for numerous applications of IoT for emergency management, e.g., image recognition in smart cameras. However, state-of-the-art DNN architectures are far from deployable on constrained edge devices. In this paper, we study DNN architectures for image classification for camera input in an underpass.

Our contributions are twofold. First, we report successful neural network pruning, i.e., we retain comparable classification accuracy using only 1.1% of the size of the VGG16 architecture. Such a small DNN architecture can be deployed on constrained devices. Second, we propose that CycleGANs can be used to allow classification of OOD camera input by performing style transfers to the ODD. We present a proof-of-concept involving transforming nighttime input to the daytime domain, supporting the robustness of the application. The small DNN classifier can remain trained for only daytime conditions although the ODD of the image recognition solution can be extended to encompass additional environmental conditions.

The preliminary work reported in this paper identified several interesting directions for future work. The obvious first step is to extend the dataset used for both the classification model and the CycleGAN. As the data labelling is labor-intensive, we plan to rely on our previous experience in active learning to focus annotation effort for maximum return on investment [4]. With more data, we can train the classifier to predict additional classes, including input related to emergency response.

Second, we plan to replace the image recognition with object recognition. While image recognition serves as a good first application, complementing the predictions with bounding boxes would be the natural next step. Accurate object detection and recognition can be made in real-time using DNN architectures such as YOLO [20]. However, the prerequisite data labeling requires more manual effort.

Third, we intend to perform more systematic neural network pruning. In this paper, we explored the concept using an *ad hoc* approach. However, state-of-the-art pruning involves sophisticated measurements of which neurons carry the most importance to the classification task [17].

Fourth, we will explore using CycleGANs as an approach to tackle vandalism. During the study, the camera dome in the underpass was targeted by an antagonistic spray paint attack. The image quality was compromised, but not totally useless. Figure 10 shows our initial efforts to learn a style transfer between the daytime domain and the sprayed domain. The results indicate that the approach deserves future study, and we propose that the concept could be used to temporarily recover from vandalism, especially for cameras mounted in difficult-to-reach locations.

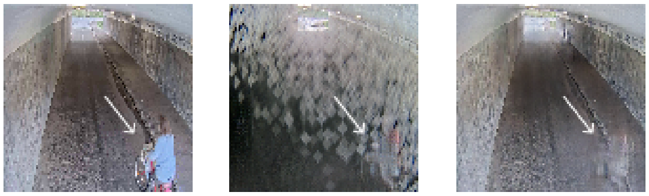


Figure 10: Original daytime image (left), transformed to the spray domain (middle), and then reconstructed in the daytime domain (right). The arrows show the bicyclist.

ACKNOWLEDGMENTS

This work was funded by Plattformen at Campus Helsingborg, Lund University, Sweden.

REFERENCES

[1] Asha Anoosheh, Torsten Sattler, Radu Timofte, Marc Pollefeys, and Luc Van Gool. 2019. Night-to-Day Image Translation for Retrieval-based Localization. In *Proc. of the 2019 International Conference on Robotics and Automation*. 5958–5964.

[2] Davis Blalock, Jose Javier Gonzalez Ortiz, Jonathan Frankle, and John Guttag. 2020. What is the State of Neural Network pruning? *arXiv preprint arXiv:2003.03033* (2020).

[3] Markus Borg. 2016. TuneR: A Framework for Tuning Software Engineering Tools with Hands-on Instructions in R. *Journal of Software: Evolution and Process* 28, 6 (2016), 427–459.

[4] Markus Borg, Iben Lennerstad, Rasmus Ros, and Elizabeth Bjarnason. 2017. On Using Active Learning and Self-training When Mining Performance Discussions on Stack Overflow. In *Proc. of the 21st International Conference on Evaluation and Assessment in Software Engineering*. 308–313.

[5] Yun Won Choi and Jang Woon Baek. 2020. Edge Camera System Using Deep Learning Method with Model Compression on Embedded Applications. *Proc. of the 2020 IEEE International Conference on Consumer Electronics* (2020), 1–4.

[6] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT press.

[7] Anhong Guo, Anuraag Jain, Shomiron Ghose, Gierad Laput, Chris Harrison, and Jeffrey P. Bigham. 2018. Crowd-AI Camera Sensing in the Real World. *Proc. of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2 (2018), 1 – 20.

[8] Magnus Gyllenhammar, Rolf Johansson, Fredrik Warg, DeJiu Chen, Hans-Martin Heyn, Martin Sanfridson, Jan Söderberg, Anders Thorsén, and Stig Ursing. 2020. Towards an Operational Design Domain That Supports the Safety Argumentation of an Automated Driving System. In *Proc. of the 10th European Congress on Embedded Real Time Systems*.

[9] Song Han, Huizi Mao, and William J Dally. 2015. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding. *arXiv preprint arXiv:1510.00149* (2015).

[10] Karen Hao. 2019. Training a Single AI Model Can Emit as Much Carbon as Five Cars in Their Lifetimes. *MIT Technology Review* (June 6 2019).

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep Residual Learning for Image Recognition. *arXiv preprint arXiv:1512.03385* (2015). arXiv:1512.03385

[12] Jens Henriksson, Christian Berger, Markus Borg, Lars Tornberg, Sankar Raman Sathyamoorthy, and Cristofer Englund. 2019. Performance Analysis of Out-of-Distribution Detection on Various Trained Neural Networks. In *Proc. of the 45th Euromicro Conference on Software Engineering and Advanced Applications*. 113–120.

[13] Sheng-Wei Huang, Che-Tsung Lin, Shu-Ping Chen, Yen-Yi Wu, Po-Hao Hsu, and Shang-Hong Lai. 2018. Auggan: Cross Domain Adaptation with GAN-based Data Augmentation. In *Proc. of the European Conference on Computer Vision*. 718–731.

[14] Honghai Liu, Shengyong Chen, and Naoyuki Kubota. 2013. Intelligent Video Systems and Analytics: A Survey. *IEEE Transactions on Industrial Informatics* 9, 3 (2013), 1222–1233.

[15] Charles Lucero and Guangzhi Qu. 2017. Effects Analysis of Archetecture Changes to Convolutional Neural Networks. *Proc. of the 13th International Conference on Semantics, Knowledge and Grids* (2017), 98 – 105.

[16] Chao Luo, Xiaojie Li, Lutao Wang, Jia He, Denggao Li, and Jiliu Zhou. 2018. How Does the Data Set Affect CNN-based Image Classification Performance?. In *Proc. of the 5th International Conference on Systems and Informatics*. 361–366.

[17] Pavlo Molchanov, Arun Mallya, Stephen Tyree, Iuri Frosio, and Jan Kautz. 2019. Importance Estimation for Neural Network Pruning. *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2019), 11256–11264.

[18] Maria Molin and Sofie Lifvin. 2019. *Swedish Crime Survey 2019*. Technical Report 2019:11. The Swedish National Council for Crime Prevention.

[19] Horia Porav, Tom Bruls, and Paul Newman. 2019. I Can See Clearly Now: Image Restoration via De-raining. In *Prov. of the International Conference on Robotics and Automation*. 7087–7093.

[20] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. You Only Look Once: Unified, Real-time Object Detection. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*. 779–788.

[21] Weisong Shi, Jie Cao, Quan Zhang, Youhuizi Li, and Lanyu Xu. 2016. Edge Computing: Vision and Challenges. *IEEE Internet of Things Journal* 3, 5 (2016), 637–646.

[22] Karen Simonyan and Andrew Zisserman. 2014. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv preprint arXiv:1409.1556* (2014).

[23] Stacey Truex, Nathalie Baracaldo, Ali Anwar, Thomas Steinke, Heiko Ludwig, Rui Zhang, and Yi Zhou. 2019. A Hybrid Approach to Privacy-preserving Federated Learning. In *Proc. of the 12th ACM Workshop on Artificial Intelligence and Security*. 1–11.

[24] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. 2019. Federated Machine Learning: Concept and Applications. *ACM Transactions on Intelligent Systems and Technology (TIST)* 10, 2 (2019), 1–19.

[25] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. 2017. Unpaired Image-To-Image Translation Using Cycle-Consistent Adversarial Networks. In *Proc. of the 2017 IEEE International Conference on Computer Vision*.