Honda Research Institute EU
Honda Research Institute Europe GmbH
https://www.honda-ri.de/

Effects of Domain Awareness in Generalizing over Cameras in Road Detection

Sebastian Schrom, Stephan Hasler

2017

Preprint:

This is an accepted article published in New Challenges in Neural Computation (NC2). The final authenticated version is available online at: https://doi.org/[DOI not available]

Effects of Domain Awareness in Generalizing over Cameras in Road Detection

Sebastian Schrom[1,2] and Stephan Hasler[2]

[1] Technical University of Darmstadt
[2] Honda Research Institute Europe GmbH

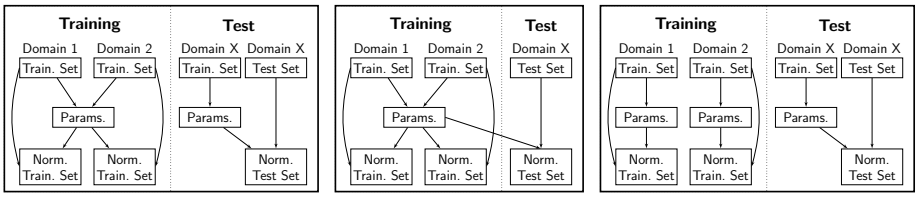
Abstract. In this paper we investigate the effects of domain awareness when using a pre-trained covolutional neural network (CNN). Usually, the only adaptation when using such a CNN for the same task is to normalize the input data by an individual RGB mean from own data. We show that it plays a major role whether the test domain was included during training and if training was aware of domains. For this we investigate generalization over few cameras in a road detection task as a domain transfer scenario. We train CNNs for all combinations of used cameras and test each camera individually. We apply RGB mean subtraction in three different cases of domain awareness during training and test. Our results reveal a harmful effect if the test domain was included during the network training, but not considered as an individual domain.

1 Introduction

In recent years deep neural networks showed impressive results on several large-scale object recognition benchmarks [7, 13]. However, when such pre-trained networks are applied on images of a different benchmark the observed performance can be substantially reduced. This problem is referred to as domain transfer. A standard way to minimize the effects of domain transfer is to adapt some parameters of the pre-trained network to the target domain, e.g. usually the input data is normalized using the RGB mean of the target domain.

Most domain transfer studies focus on the transfer from a single (or multiple) source domain(s) to a single new target domain only [10, 16]. This is the most difficult case which we will call the *new*-case of generalization. Often these results are interpreted in relation to the simplest case, where source and target domain are the same [14, 15]. We call this the *same*-case. Only rarely the case is considered that the target domain was one of multiple source domains (e.g. [6]). From our point of view this case needs to be considered more, since downloadable networks are usually trained on huge image collections from the internet which potentially contain many different domains. So e.g. the test data might be from a camera that was also used to acquire some images of the training data. In our work we will focus on this *part*-case of generalization and show that domain adaptation can be harmful if training was not aware of domains.

Usually the entire training set is treated as a single domain and the parameters used to normalize the input are generated from the entire training set. Later,



(a) *Test*-aware (b) *Never*-aware (c) *Always*-aware

Fig. 1. Three different cases of domain awareness.

during test the parameters are re-estimated for the target domain. This is the standard workflow for using a pre-trained network and we call this case of domain awareness the *test*-aware case (see Fig. 1a). Since we saw a weak performance for the *test*-aware case in the *part*-case of generalization, we concluded this might be due to inconsistent normalization for the same domain during training and test. To evaluate this further we additionally tested two consistent cases of domain awareness. The *never*-aware case and the *always*-aware case. The *never*-aware case (see Fig. 1b) is unaware of domains during both, training and test. Here, the normalization parameters are also calculated over the entire training set but later the same parameter set is used to normalize data of the test domain as well. However, this way of handling domain transfer is not recommended since there is no real adaptation to the test domain and a weak performance for the *new*-case of generalization can be expected. The second consistent awareness case we investigated is the *always*-aware case (see Fig. 1c). It is aware of domains while training and test and computes during training the parameters individually for each domain. However, considering all domains during training is usually not feasible because there might be many which might also overlap.

In our work we extensively evaluate all generalization cases, the *same*-case the *part*-case and the *new*-case with all three cases of domain awareness, the *test*-aware case, the *never*-aware case and the *always*-aware case. For this we have generated a controlled setting to focus on the effects of domain awareness:
1) We have chosen the task of segmenting an image into *road-like-area* and *non-road-like-area*. Usually this is done with architectures that model large receptive fields to take global image relations into account like in [3] and [1]. But we focus on classification of small image patches instead, which allows us the use of a simple CNN architecture. This is sufficient for our goal to show the general effects of domain awareness.
2) We have a controlled dataset where the domains are three cameras that strongly differ in their sensitivity of color channels. Usually other datasets like the frequently used Office dataset proposed in [12] are less controlled as they are partly a result of a web search and might hide several unknown domains inside. Hence, results on these datasets are caused by a mixture of effects, which hinders a qualitative interpretation.
3) We use the the standard input normalization method of RGB mean subtrac-

tion [7, 13], which is usually done before training a CNN to improve convergence speed [9], as a simple unsupervised way of domain adaptation. This means we interpret the "Params." from Fig. 1 as the RGB mean. In general, more sophisticated approaches of domain adaptation range from nonlinear transformations [8] to complex domain confusion losses within CNNs [16]. Nevertheless, there are methods almost as simple as our method as e.g. batch normalization [4], which was originally designed for improving convergence speed by normalizing mean and standard deviation of the batch-wise output of several layers. In [10] they showed that this technique can be successfully used for domain adaptation by re-estimating the normalization parameters for the target data.

The remainder of the paper is structured as follows. In Sec. 2, we describe the used image data and the neural network architecture of our experiments. The results for the different cases of domain awareness will be extensively compared for the different generalization cases in Sec. 3. Finally, we give a conclusion and a short outlook on future work in Sec. 4.

2 Data & Methods

To test the generalization over the three used cameras, we train an individual CNN for each single camera, for each possible pair, and for all cameras together. Each of these seven networks is tested against each individual camera. This means our tests include the three previously presented generalization cases, the *same*-case, the *part*-case and the *new*-case. We repeat this extensive evaluation for the different cases of domain awareness. Before comparing these results in Sec. 3 we will give further details on the used data and methods in the following.

2.1 Data

For our experiments we distinguished between images of three RGB cameras, each looking to the front of the ego car. Our first data set is the publicly available KITTI Road Benchmark [2]. The images from the other two datasets were acquired in-house using a BlackMagic[1] (BMAG) and an ELESYS[2] camera. The images of each camera stem from different recording sessions, containing different routes, illuminations and weather conditions. This increases the chance that the prominent domains in our data are the different cameras and not some hidden condition of the environment. This notion is confirmed in Table 1 when looking at the channel-wise means over pixels of training patches, where we see a strong difference of individual channels over cameras Each image comes with a manual annotation of one or more *road-like-area* polygons. In general, image patches whose center is in the *road-like-area* are used as positive examples and all other patches as negative examples. We limit the extraction of patches with a size of 37×37 pixels to a region that corresponds to a rectangular corridor in the metric Bird's-Eye-View[3][11] space. With this we mainly ensure that the

[1] http://www.blackmagicdesign.com/de/products/blackmagicpocketcinemacamera
[2] Special in-house made, no external reference available
[3] The mounting position and angles are known for each camera

Table 1. Data statistics of the different cameras.

		KITTI	BMAG	ELESYS
Red channel mean		96.5	115.3	83.3
Green channel mean		97.0	136.0	89.6
Blue channel mean		92.2	132.0	55.5
Training	#Images	289	113	166
	#Samples	2,312,000	5,085,000	2,822,000
	Pos./neg.	1/1	1/1	1/1
Test	#Images	290	111	166
	#Samples	42,165,099	85,041,497	49,325,077
	Pos./neg.	∼1/1	∼2/1	∼2/1



Fig. 2. Example image of BMAG camera. The colored area in the perspective view (left) denotes pixels inside the metric Bird's-Eye-View corridor (right).

negative patches focus on regions close to the road, while e.g. removing everything above the horizon, like the sky. In Fig. 2 we show an image of the BMAG camera where the region that corresponds to the Bird's-Eye-View corridor in our work is highlighted.

We split the available images for each camera into a training and a test set of similar size and extract patches with a size corresponding to the input dimension of our neural network. To speed up training we substantially reduced the number of available patches by randomly choosing roughly 5% of them, and we also balanced the number of positive and negative samples. During test we classify all negative and positive patches inside the Bird's-Eye-View corridor. The numbers of used training and test patches are given in Table 1. It can be seen that the number of training samples is imbalanced over different cameras. However, we experimented with more complex balancing strategies but could not find any substantial change in reported results.

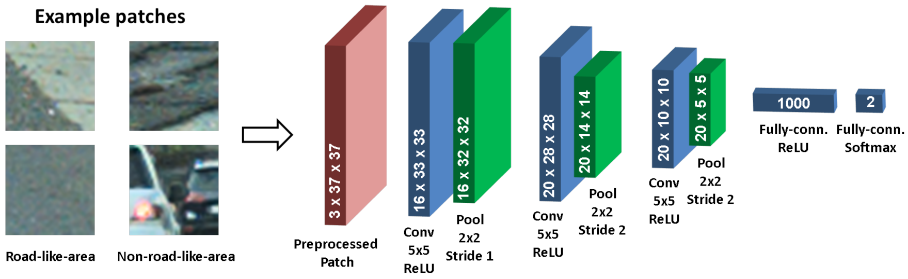


Fig. 3. CNN architecture. The dimension of each layer is shown in white color.

2.2 Network Architecture & Training

For our experiments we use RGB mean subtraction as a simple form of domain adaptation. This method is relatively cheap, as it is a preprocessing step whose few parameters can be estimated from a collection of images without any ground-truth information. More specifically, we used a global channel-wise RGB mean, i.e. we subtracted from each color channel a fixed value for normalization. This corresponds to the notion that the domains that we like to generalize over are defined by different sensitivities of the color channels among the cameras. We use a CNN to predict for a patch of 37×37 pixels whether it belongs to *road-like-area* or *non-road-like-area*. The architecture of the network is shown in Fig. 3 together with some example patches. We implemented the CNN in Caffe [5] and optimized it with Stochastic Gradient Descent using the cross-entropy classification loss. We used a batch size of 50 and trained our architecture for 20 epochs. The initial learning rate was set to 0.001 and constantly halved after ten percent of the maximum iterations. Furthermore, we used a momentum parameter of 0.9, a weight decay factor of 0.0005, and 50% dropout on the output of the first fully-connected layer.

3 Results

During test we interpret the value of the *road-like-area* neuron in the final network layer as a confidence score and measure performance using Average Precision (AP). However, the absolute performance is not in focus, but the change in performance for the different generalization and domain awareness cases. Please note that AP results for the KITTI Road Benchmark [2] are reported for the Bird's-Eye-View space, while we measure performance in image space.

The results of the ***test*-aware** case (see Fig. 1a) are presented in the corresponding column in Table 2, where the upper rows show the results for all experiments and the three lower rows give the averaged results for the three generalization cases. The performance is consistently high whenever the test camera is the only training camera, i.e. in the *same*-case. In the *part*-case, the

Table 2. Results of different cases of domain awareness for different combinations of training and test cameras (AP in %). Blue represents the *same*-case, gray the *part*-case and green the *new*-case. Numbers in gray indicate unchanged results in comparison to previous cases of domain awareness. The averaged results are shown at the bottom.

Used cameras		**Domain awareness**		
Training	**Test**	*Test*	*Never*	*Always*
KITTI	KITTI	93.94	93.94	93.94
KITTI + BMAG	KITTI	89.66	93.76	93.64
KITTI + ELESYS	KITTI	90.12	94.04	93.93
KITTI + BMAG + ELESYS	KITTI	88.03	93.72	93.82
BMAG	KITTI	85.99	58.57	85.99
ELESYS	KITTI	90.10	86.52	90.10
BMAG + ELESYS	KITTI	78.55	59.99	90.04
BMAG	BMAG	96.29	96.29	96.29
BMAG + KITTI	BMAG	95.65	96.27	96.25
BMAG + ELESYS	BMAG	93.12	96.31	96.24
BMAG + KITTI + ELESYS	BMAG	94.80	96.21	96.28
KITTI	BMAG	93.28	73.68	93.28
ELESYS	BMAG	93.70	89.88	93.70
KITTI + ELESYS	BMAG	80.65	80.08	93.46
ELESYS	ELESYS	95.97	95.97	95.97
ELESYS + KITTI	ELESYS	91.33	96.02	95.97
ELESYS + BMAG	ELESYS	86.68	95.77	95.91
ELESYS + KITTI + BMAG	ELESYS	80.25	95.80	95.88
KITTI	ELESYS	86.67	77.12	86.67
BMAG	ELESYS	84.11	74.35	84.11
KITTI + BMAG	ELESYS	80.66	75.46	80.02
Averages	*Same*	95.40	95.40	95.40
	Part	89.96	95.32	95.32
	New	85.97	75.07	88.60

performance drops to 89.96 AP on average, which is surprising, since the network saw similar data already while training and is expected to generalize over such data. A possible explanation for this effect is the inconsistent handling of the same camera during training and test. During test the mean of the considered camera is used, while during training the mean over this camera and one or two other cameras is computed. We think that the CNN learned to specialize too much on the combination of camera data and preprocessing and later is basically surprised to see a familiar camera but with changed characteristics of input channels. For the *new*-case we get the worst performance with 85.97 AP on average, which could be expected, since the cameras differ besides the RGB means also in factors like image noise or resolution.

In the ***never*-aware** case (see Fig. 1b) the mean that was used while training is also used for normalizing the test images. This strategy does not influence the *same*-case. As expected, since there is no real domain adaptation, it strongly affects the *new*-case where the average performance is only 75.07 AP. A visual analysis revealed that for the *new*-case the predicted confidence for *road-like-area* is either quite high or low over the test set. This is a potential indicator that the different characteristics of the input channels of the new camera drive the trained network outside its working range. Interestingly, which is one major result of this work, we see a significant enhancement of performance for the *part*-case. This shows that consistent treatment of domains while training and test is for this case even better than domain adaptation.

In the ***always*-aware** case the data of each camera is normalized independently during training and test (see Fig. 1c). With this we also avoid the inconsistency in the *part*-case, as now each camera uses its own mean during training and test. Correspondingly, we see in Table 2 that the performance for this case stays at the high level of the *never*-aware case. Furthermore, in the *new*-case on KITTI we also see a strong improvement when trained on BMAG+ELESYS, and also for BMAG when trained on KITTI+ELESYS. A reason for this can be that the awareness of domains during training helps the classifier to focus resources on a more general and consistent representation of *road-like-area*.

4 Conclusion

In this paper we investigated the effects of domain awareness in a controlled setting that was based on a simple road segmentation task with a CNN and RGB mean normalization as a simple domain adaptation method. We could reveal that the standard workflow of re-estimating the normalization parameters for the test domain can have a harmful effect if the test domain was already involved among other domains during training. Here, the performance is significantly reduced if a domain is not treated consistently during training and test. However, due to insufficient knowledge about domains when using a pre-trained network it is often not clear whether the test domain was involved and how it was treated if it was one of several. To summarize, the major outcome of our work is that for practice it is important to find out whether the training data of a pre-trained network contained already the target domain, and if so, if the optimization process was aware of different domains and then acting according to that. We think that our rather controlled domain setting together with the simple adaptation strategy strongly helped to discover some of the reported effects, but in general, we expect that our results also hold for less controlled settings. To test this will be the primary direction for future work.

References

1. Badrinarayanan, V., Kendall, A., Cipolla, R.: Segnet: A deep convolutional encoder-decoder architecture for image segmentation. arXiv preprint arXiv:1511.00561 (2015)

2. Fritsch, J., Kühnl, T., Geiger, A.: A new performance measure and evaluation benchmark for road detection algorithms. In: ITSC. pp. 1693–1700 (2013)
3. Fritsch, J., Kühnl, T., Kummert, F.: Monocular road terrain detection by combining visual and spatial information. IEEE Transactions on Intelligent Transportation Systems 15(4), 1586–1596 (2014)
4. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: ICML. pp. 448–456 (2015)
5. Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., Darrell, T.: Caffe: Convolutional architecture for fast feature embedding. In: ACM-MM. pp. 675–678 (2014)
6. Khosla, A., Zhou, T., Malisiewicz, T., Efros, A.A., Torralba, A.: Undoing the damage of dataset bias. In: ECCV. pp. 158–171 (2012)
7. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NIPS. pp. 1097–1105 (2012)
8. Kulis, B., Saenko, K., Darrell, T.: What you saw is not what you get: Domain adaptation using asymmetric kernel transforms. In: CVPR. pp. 1785–1792 (2011)
9. LeCun, Y.A., Bottou, L., Orr, G.B., Müller, K.R.: Efficient backprop. In: Neural networks: Tricks of the trade, pp. 9–48. Springer (2012)
10. Li, Y., Wang, N., Shi, J., Liu, J., Hou, X.: Revisiting batch normalization for practical domain adaptation. arXiv preprint arXiv:1603.04779 (2017)
11. Mallot, H.A., Bülthoff, H.H., Little, J., Bohrer, S.: Inverse perspective mapping simplifies optical flow computation and obstacle detection. Biological cybernetics 64(3), 177–185 (1991)
12. Saenko, K., Kulis, B., Fritz, M., Darrell, T.: Adapting visual category models to new domains. In: ECCV. pp. 213–226. Springer (2010)
13. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
14. Tommasi, T., Patricia, N., Caputo, B., Tuytelaars, T.: A deeper look at dataset bias. In: GCPR. pp. 504–516. Springer (2015)
15. Torralba, A., Efros, A.A.: Unbiased look at dataset bias. In: CVPR. pp. 1521–1528 (2011)
16. Tzeng, E., Hoffman, J., Zhang, N., Saenko, K., Darrell, T.: Deep domain confusion: Maximizing for domain invariance. arXiv preprint arXiv:1412.3474 (2014)