Detecting the Unexpected via Image Resynthesis *

Krzysztof Lis Krishna Nakka Pascal Fua Mathieu Salzmann

Computer Vision Laboratory, EPFL

Abstract

Classical semantic segmentation methods, including the recent deep learning ones, assume that all classes observed at test time have been seen during training. In this paper, we tackle the more realistic scenario where unexpected objects of unknown classes can appear at test time. The main trends in this area either leverage the notion of prediction uncertainty to flag the regions with low confidence as unknown, or rely on autoencoders and highlight poorly-decoded regions. Having observed that, in both cases, the detected regions typically do not correspond to unexpected objects, in this paper, we introduce a drastically different strategy: It relies on the intuition that the network will produce spurious labels in regions depicting unexpected objects. Therefore, resynthesizing the image from the resulting semantic map will yield significant appearance differences with respect to the input image. In other words, we translate the problem of detecting unknown classes to one of identifying poorly-resynthesized image regions. We show that this outperforms both uncertainty- and autoencoder-based methods.

1. Introduction

Semantic segmentation has progressed tremendously in recent years and state-of-the-art methods rely on deep learning [4, 5, 47, 45]. Therefore, they typically operate under the assumption that all classes encountered at test time have been seen at training time. In reality, however, guaranteeing that all classes that can ever be found are represented in the database is impossible when dealing with complex outdoors scenes. For instance, in an autonomous driving scenario, one should expect to occasionally find the unexpected, in the form of animals, snow heaps, or lost cargo on the road, as shown in Fig. 1. Note that the corresponding labels are absent from standard segmentation training datasets [7, 46, 15]. Nevertheless, a self-driving vehicle should at least be able to detect that some image regions cannot be labeled properly and warrant further attention.

Figure 1: **Detecting the unexpected.** While uncertainty- and autoencoder-based methods tend to be distracted by the background, our approach focuses much more accurately on the unknown objects.

Recent approaches to addressing this problem follow two trends. The first one involves reasoning about the prediction uncertainty of the deep networks used to perform the segmentation [19, 25, 20, 12]. In the driving scenario, we have observed that the uncertain regions tend not to coincide with unknown objects, and, as illustrated by Fig. 1, these methods therefore fail to detect the unexpected. The second trend consists of leveraging autoencoders to detect anomalies [8, 33, 1], assuming that never-seen-before objects will be decoded poorly. We found, however, that autoencoders tend to learn to simply generate a lower-quality version of the input image. As such, as shown in Fig. 1, they also fail to find the unexpected objects.

In this paper, we therefore introduce a radically different approach to detecting the unexpected. Fig. 2 depicts our pipeline, built on the following intuition: In regions containing unknown classes, the segmentation network will make spurious predictions. Therefore, if one tries to resynthesize the input image from the semantic label map, the resynthe-

*This work was supported in part by the International Chair Drive for All - MINES ParisTech - Peugeot-Citroën - Safran - Valeo, and by the Swiss National Science Foundation.

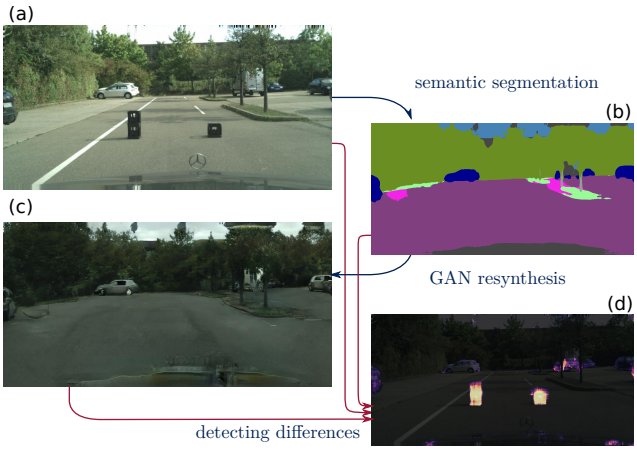


Figure 2: **Our Approach.** **(a)** Input image from the *Lost and Found* [35] dataset containing objects of a class the segmentation algorithm has not been trained for. **(b)** In the resulting semantic segmentation, these objects are lost. **(c)** In the image resynthesized based on the segmentation labels, they are also lost. **(d)** Using a specially trained *discrepancy network* to compare the original image and the resynthesized one highlights the unexpected objects.

sized unknown regions will look significantly different from the original ones. In other words, we reformulate the problem of segmenting unknown classes as one of identifying the differences between the original input image and the one resynthesized from the predicted semantic map. To this end, we leverage a generative network [42] to learn a mapping from semantic maps back to images. We then introduce a discrepancy network that, given as input the original image, the resynthesized one, and the predicted semantic map, produces a binary mask indicating unexpected objects. To train this network *without* ever observing unexpected objects, we simulate such objects by changing the semantic label of known object instances to other, randomly chosen classes. This process, described in Section 3.2, does *not* require seeing the unknown classes during training, which makes our approach applicable to detecting never-seen-before classes at test time.

Our contribution is therefore a radically new approach to identifying regions that have been misclassified by a given semantic segmentation method, based on comparing the original image with a resynthesized one. We demonstrate the ability of our approach to detect unexpected objects using the Lost and Found dataset [35]. This dataset, however, only depicts a limited set of unexpected objects in a fairly constrained scenario. To palliate this lack of data, we create a new dataset depicting unexpected objects, such as animals, rocks, lost tires and construction equipment, on roads. Our method outperforms uncertainty-based baselines, as well as the state-of-the-art autoencoder-based method specifically designed to detect road obstacles [8].

Furthermore, our approach to detecting anomalies by comparing the original image with a resynthesized one is generic and applies to other tasks than unexpected object detection. For example, deep learning segmentation algorithms are vulnerable to adversarial attacks [44, 6], that is, maliciously crafted images that look normal to a human but cause the segmentation algorithm to fail catastrophically. As in the unexpected object detection case, re-synthesizing the image using the erroneous labels results in a synthetic image that looks nothing like the original one. Then, a simple non-differentiable detector, thus less prone to attacks, is sufficient to identify the attack. As shown by our experiments, our approach outperforms the state-of-the-art one of [43] for standard attacks, such as those introduced in [44, 6].

The implementation of our algorithm[1], our new *Road Anomaly* dataset[2], and the labeling tool[3] used to process it are publicly available.

2. Related Work

2.1. Uncertainty in Semantic Segmentation

Reasoning about uncertainty in neural networks can be traced back to the early 90s and Bayesian neural networks [10, 29, 30]. Unfortunately, they are not easy to train and, in practice, *dropout* [40] has often been used to approximate Bayesian inference [11]. An approach relying on explicitly propagating activation uncertainties through the network was recently proposed [12]. However, it has only been studied for a restricted set of distributions, such as the Gaussian one. Another alternative to modeling uncertainty is to replace a single network by an ensemble [25].

For semantic segmentation specifically, the standard approach is to use dropout, as in the *Bayesian SegNet* [19], a framework later extended in [20]. Leveraging such an approach to estimating label uncertainty then becomes an appealing way to detect unknown objects because one would expect these objects to coincide with low confidence regions in the predicted semantic map. This approach was pursued in [16, 18, 17]. These methods build upon the Bayesian SegNet and incorporate an uncertainty threshold to detect potentially mislabeled regions, including unknown objects. However, as shown in our experiments, uncertainty-based methods, such as the Bayesian SegNet [19] and network ensembles [25], yield many false positives in irrelevant regions. By contrast, our resynthesis-based approach learns to focus on the regions depicting unexpected objects.

[1] Implementation: github.com/cvlab-epfl/detecting-the-unexpected
[2] *Road Anomaly* dataset: cvlab.epfl.ch/data/road-anomaly/
[3] Our labeling tool: github.com/cvlab-epfl/LabelGrab

2.2. Anomaly Detection via Resynthesis

Image resynthesis and generation methods, such as autoencoder and GANs, have been used in the past for anomaly detection. The existing methods, however, mostly focus on finding behavioral anomalies in the temporal domain [36, 22]. For example, [36] predicts the optical flow in a video, attempts to reconstruct the images from the flow, and treats significant differences from the original images as evidence for an anomaly. This method, however, was only demonstrated in scenes with a static background. Furthermore, as it relies on flow, it does not apply to single images.

To handle individual images, some algorithms compare the image to the output of a model trained to represent the distribution of the original images. For example, in [1], the image is passed through an adversarial autoencoder and the feature loss between the output and input image is then measured. This can be used to classify whole images but not localize anomalies within the images. Similarly, given a GAN trained to represent an original distribution, the algorithm of [38] searches for the latent vector that yields the image most similar to the input, which is computationally expensive and does not localize anomalies either.

In the context of road scenes, image resynthesis has been employed to detect traffic obstacles. For example, [32] relies on the previous frame to predict the non-anomalous appearance of the road in the current one. In [8, 33], input patches are compared to the output of a shallow autoencoder trained on the road texture, which makes it possible to localize the obstacle. These methods, however, are very specific to roads and lack generality. Furthermore, as shown in our experiments, patch-based approaches such as the one of [8] yield many false positives and our approach outperforms it.

Note that the approaches described above typically rely on autoencoder for image resynthesis. We have observed that autoencoders tend to learn to perform image compression, simply synthesizing a lower-quality version of the input image, independently of its content. By contrast, we resynthesize the image from the semantic label map, and thus incorrect class predictions yield appearance variations between the input and resynthesized image.

2.3. Adversarial Attacks in Semantic Segmentation

As mentioned before, we can also use the comparison of an original image with a resynthesized one for adversarial attack detection. The main focus of the adversarial attack literature has been on image classification [13, 3, 31], leading to several defense strategies [24, 41] and detection methods [14, 26, 28]. Nevertheless, in [44, 6], classification attack schemes were extended to semantic segmentation networks. However, as far as defense schemes are concerned, only [43] has proposed an attack detection method in this scenario. This was achieved by analyzing the spatial consistency of the predictions of overlapping image patches. We will show that our approach outperforms this technique.

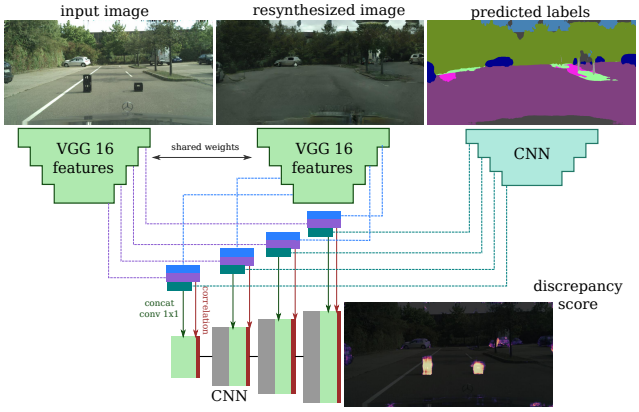


Figure 3: **Discrepancy network.** Given the original image, the predicted semantic labels and the resynthesized image as input, our discrepancy network detects meaningful differences caused by mislabeled objects. The VGG [39] network extracts features from both images, which are correlated at all levels of the pyramid. Image and label features are then fused using 1×1 convolutions. Both the features and their correlations are then fed to a decoder via skip connections to produce the final discrepancy map.

3. Approach

Our goal is to handle unexpected objects at test time in semantic segmentation and to predict the probability that a pixel belongs to a never-seen-before class. This is in contrast to most of the semantic segmentation literature, which focuses on assigning to each pixel a probability to belong to classes it has seen in training, without explicit provision for the unexpected.

Fig. 2 summarizes our approach. We first use a given semantic segmentation algorithm, such as [2] and [47], to generate a semantic map. We then pass this map to a generative network [42] that attempts to resynthesize the input image. If the image contains objects belonging to a class that the segmentation algorithm has not been trained for, the corresponding pixels will be mislabeled in the semantic map and therefore poorly resynthesized. We then identify these *unexpected* objects by detecting significant differences between the original image and the synthetic one. Below, we introduce our approach to detecting these discrepancies and assessing which differences are significant.

3.1. Discrepancy Network

Having synthesized a new image, we compare it to the original one to detect the meaningful differences that denote unexpected objects not captured by the semantic map. While the layout of the known objects is preserved in the synthetic image, precise information about the scene's appearance is lost and simply differencing the images would

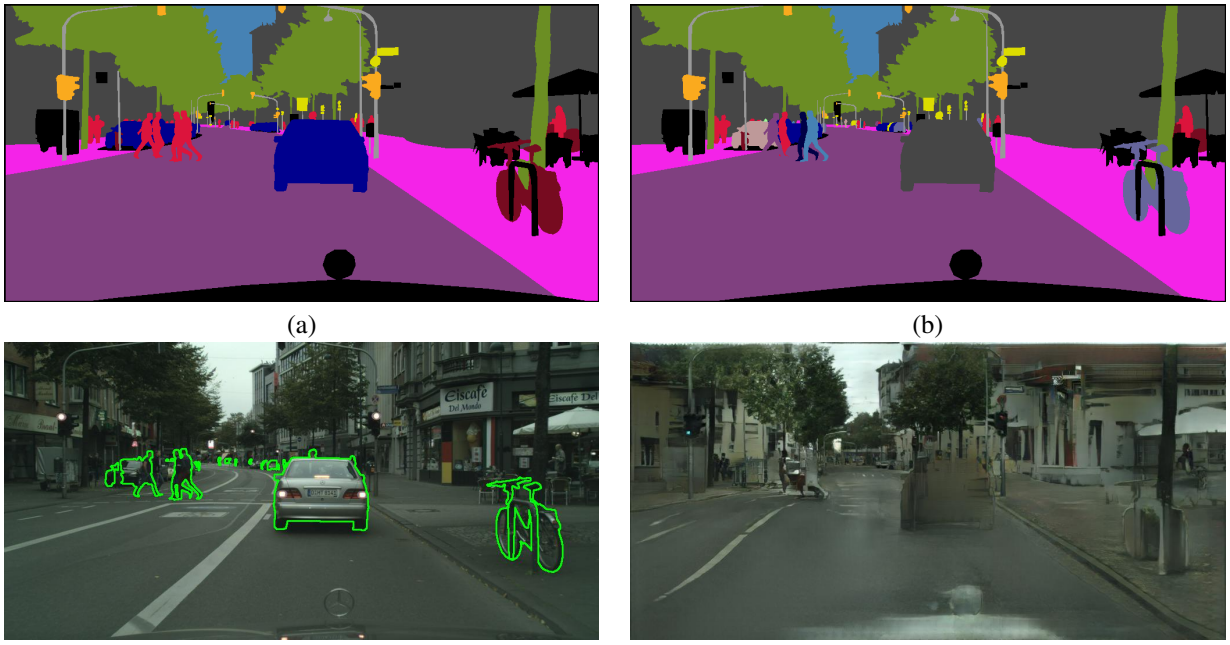


Figure 4: **Creating training examples for the discrepancy detector.** **(a)** Ground-truth semantic map. **(b)** We alter the map by replacing some object instances with randomly chosen labels. **(c)** Original image with the overlaid outlines of the altered objects. **(d)** Image re-synthesized using the altered map. We train the discrepancy detector to find the pixels within the outlines of altered objects shown in **(c)**.

not yield meaningful results. Instead, we train a second network, which we refer to as the *discrepancy network*, to detect the image discrepancies that *are* significant.

Fig. 3 depicts the architecture of our discrepancy network. We drew our inspiration from the co-segmentation network of [27] that uses feature correlations to detect objects co-occurring in two input images. Our network relies on a three-stream architecture that first extracts features from the inputs. We use a pre-trained VGG [39] network for both the original and resynthesized image, and a custom CNN to process the one-hot representation of the predicted labels. At each level of the feature pyramid, the features of all the streams are concatenated and passed through 1×1 convolution filters to reduce the number of channels. In parallel, pointwise correlations between the features of the real image and the resynthesized one are computed and passed, along with the reduced concatenated features, to an upconvolution pyramid that returns the final discrepancy score. The details of this architecture are provided in the supplementary material.

3.2. Training

When training our discrepancy network, we cannot observe the unknown classes. To address this, we therefore train it on synthetic data that mimics what happens in the presence of unexpected objects. In practice, the semantic segmentation network assigns incorrect class labels to the regions belonging to unknown classes. To simulate this, as illustrated in Fig. 4, we therefore replace the label of randomly-chosen object instances with a different random one, sampled uniformly from the set of *Cityscapes* evaluation classes. We then resynthesize the input image from this altered semantic map using the *pix2pixHD* [42] generator trained on the dataset of interest. This creates pairs of real and synthesized images from which we can train our discrepancy network. Note that this strategy does *not* require seeing unexpected objects during training.

3.3. Detecting Adversarial Attacks

As mentioned above, comparing an input image to a resynthesized one also allows us to detect adversarial attacks. To this end, we rely on the following strategy. As for unexpected object detection, we first compute a semantic map from the input image, adversarial or not, and resynthesize the scene from this map using the *pix2pixHD* generator. Here, unlike in the unexpected object case, the semantic map predicted for an adversarial example is completely wrong and the resynthesized image therefore completely distorted. This makes attack detection a simpler problem than unexpected object one. We can thus use a simple non-differentiable heuristic to compare the input image with the resynthesized one. Specifically, we use the L^2 distance be-

tween HOG [9] features computed on the input and resynthesized image. We then train a logisitic regressor on these distances to predict whether the input image is adversarial or not. Note that this simple heuristic is much harder to attack than a more sophisticated, deep learning based one.

4. Experiments

We first evaluate our approach on the task of detecting unexpected objects, such as lost cargo, animals, and rocks, in traffic scenes, which constitute our target application domain and the central evaluation domain for semantic segmentation thanks to the availability of large datasets, such as Cityscapes [7] and BDD100K [46]. For this application, all tested methods output a per-pixel *anomaly score*, and we compare the resulting maps with the ground-truth anomaly annotations using ROC curves and the area under the ROC curve (AUROC) metric. Then, we present our results on the task of adversarial attack detection.

We perform evaluations using the Bayesian SegNet [19] and the PSP Net [47], both trained using the BDD100K dataset [46] (segmentation part) chosen for its large number of diverse frames, allowing the networks to generalize to the anomaly datasets, whose images differ slightly and cannot be used during training. To train the image synthesizer and discrepancy detector, we used the training set of Cityscapes [7], downscaled to a resolution of 1024×512 because of GPU memory constraints.

4.1. Baselines

As a first baseline, we rely on an uncertainty-based semantic segmentation network. Specifically, we use the Bayesian SegNet [19], which samples the distribution of the network's results using random dropouts — the uncertainty measure is computed as the variance of the samples. We will refer to this method as *Uncertainty (Dropout)*.

It requires the semantic segmentation network to contain dropout layers, which is not the case of most state-of-the-art networks, such as PSP [47], which is based on a ResNet backbone. To calculate the uncertainty of the PSP network, we therefore use the ensemble-based method of [25]: We trained the PSP model four times, yielding different weights due to the random initialization. We then use the variance of the outputs of these networks as a proxy for uncertainty. We will refer to this method as *Uncertainty (Ensemble)*.

Finally, we also evaluate the road-specific approach of [8], which relies on training a shallow Restricted Boltzmann Machine autoencoder to resynthesize patches of road texture corrupted by Gaussian noise. The regions whose appearance differs from the road are expected not to be reconstructed properly, and thus an anomaly score for each patch can be obtained using the difference between the autoencoder's input and output. As the original implementation was not publicly available, we re-implemented it and make the code available[1] for future comparisons. As in the original article, we use 8×8 patches with stride 6 and a hidden layer of size 20. We extract the empty road patches required by this method for training from the Cityscapes images using the ground-truth labels to determine the road area. We will refer to this approach as *RBM*.

The full version of our discrepancy detector takes as input the original image, the resynthesized one and the predicted semantic labels. To study the importance of using both of these information sources as input, we also report the results of variants of our approach that have access to only one of them. We will refer to these variants as *Ours (Resynthesis only)* and *Ours (Labels only)*.

4.2. Anomaly Detection Results

We evaluate our method's ability to detect unexpected objects using two separate datasets described below. We did not use any portion of these datasets during training, because we tackle the task of finding never-seen-before objects.

4.2.1 Lost and Found

The *Lost And Found* [35] dataset contains images of small items, such as cargo and toys, left on the street, with per-pixel annotations of the obstacle and the free-space in front of the car. We perform our evaluation using the test set, excluding 17 frames for which the annotations are missing. We downscaled the images to 1024×512 to match the size of our training images and selected a region of interest which excludes the ego-vehicle and recording artifacts at the image boundaries. We do not compare our results against the stereo-based ones introduced in [35] because our study focuses on monocular approaches.

The ROC curves of our approach and of the baselines are shown in the left column of Fig. 5. Our method outperforms the baselines in both cases. The Labels-only and Resynthesis-only variants of our approach show lower accuracy but remain competitive. By contrast, the uncertainty-based methods prove to be ill-suited for this task. Qualitative examples are provided in Fig. 6. Note that, while our method still produces false positives, albeit much fewer than the baselines, some of them are valid unexpected objects, such as the garbage bin in the first image. These objects, however, were not annotated as obstacles in the dataset.

Since the RBM method of [8] is specifically trained to reconstruct the road, we further restricted the evaluation to the road area. To this end, we defined the region of interest as the union of the *obstacle* and *freespace* annotations of *Lost And Found*. The resulting ROC curves are shown in the middle column of Fig. 5. The globally-higher scores in this scenario show that distinguishing anomalies from only the road is easier than finding them in the entire scene. While

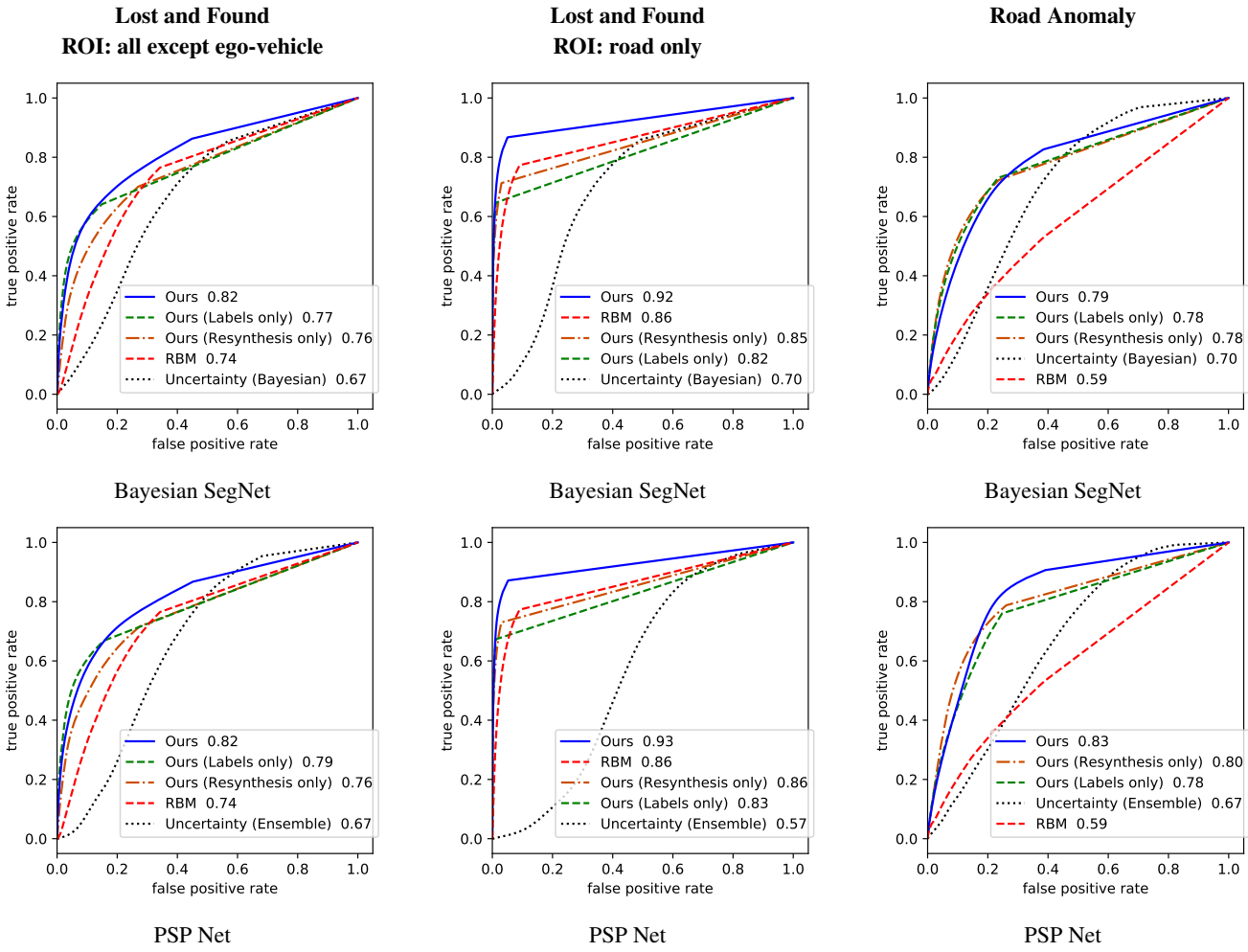


Figure 5: **ROC curves for unexpected object detection.** The first two columns show results for the *Lost and Found* [35] dataset: The curves on the left were computed over the entire images, excluding only the ego-vehicle. Those in the middle were obtained by restricting evaluation to the road, as defined by the ground-truth annotations. The right column depicts the results on our *Road Anomaly* dataset. The top and bottom rows depict the results of the *Bayesian SegNet* and the *PSP Net*, respectively. The methods are ordered according to their AUROC scores, provided on the right of the methods' name.

the RBM approach significantly improves in this scenario, our method still outperforms it.

4.2.2 Our Road Anomaly Dataset

Motivated by the scarcity of available data for unexpected object detection, we collected online images depicting anomalous objects, such as animals, rocks, lost tires, trash cans, and construction equipment, located on or near the road. We then produced per-pixel annotations of these unexpected objects manually, using the *Grab Cut* algorithm [37] to speed up the process. The dataset contains 60 images rescaled to a uniform size of 1280×720. We make this dataset[2] and the labeling tool[3] publicly available.

The results on this dataset are shown in the right column of Fig. 5, with example images in Fig. 7. Our approach outperforms the baselines, demonstrating its ability to generalize to new environments. By contrast, the *RBM* method's performance is strongly affected by the presence of road textures that differ significantly from the Cityscapes ones.

4.3. Adversarial Attack Detection

We now evaluate our approach to detecting attacks using the two types of attack that have been used in the context of semantic segmentation.

Adversarial Attacks: For semantic segmentation, the two state-of-the-art attack strategies are Dense Adversary Generation (DAG) [44] and Houdini [6]. While DAG is an iterative gradient-based method, Houdini combines the standard task loss with an additional stochastic margin factor

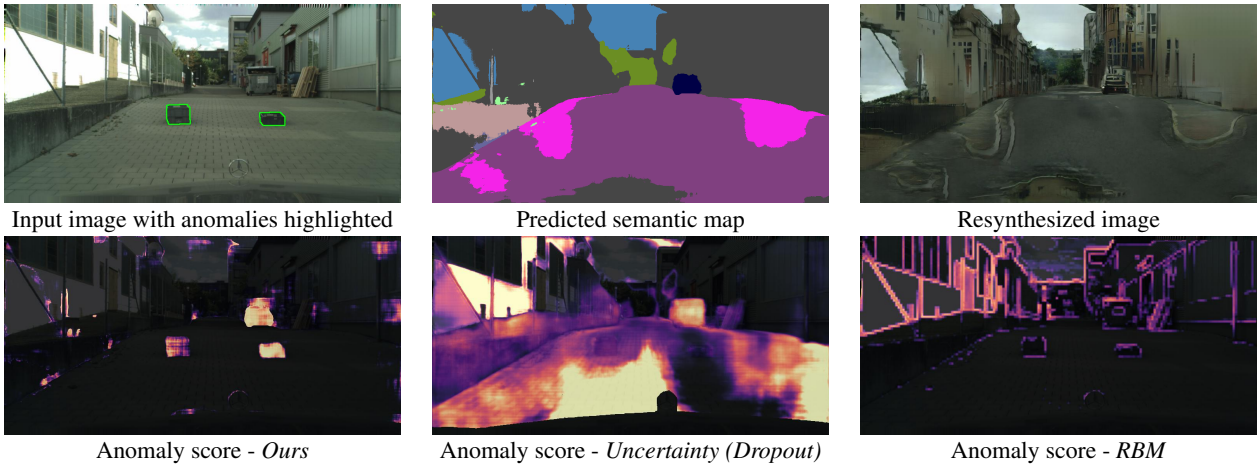


Figure 6: **Lost and Found results.** The top images depict algorithmic steps and the bottom ones our results along with those of the baselines. Our detector finds not only the obstacles on the road but also other unusual objects like the trash container on the right side of the road. By contrast *Uncertainty (Dropout)* reports high uncertainty in irrelevant regions and fails to localize the obstacles. *RBM* finds only the edges of the obstacles. Our approach detects the unexpected objects correctly.

between the score of the actual and predicted semantic maps to yield less perturbed images. Following [43], we generate adversarial examples with two different target semantic maps. In the first case (Shift), we shift the predicted label at each pixel by a constant offset and use the resulting label as target. In the second case (Pure), a single random label is chosen as target for all pixels, thus generating a pure semantic map. We generate adversarial samples on the validation sets of the Cityscapes and BDD100K datasets, yielding 500 and 1000 images, respectively, with every normal sample having an attacked counterpart.

Results: We compare our method with the state-of-the-art spatial consistency (SC) work of [43], which crops random overlapping patches and computes the mean Intersection over Union (mIoU) of the overlapping regions. The results of this comparison are provided in Table 1. Our approach outperforms SC on Cityscapes and performs on par with it on BDD100K despite our use of a Cityscapes-trained generator to resynthesize the images. Note that, in contrast with SC, which requires comparing 50 pairs of patches to detect the attack, our approach only requires a single forward pass through the segmentation and generator networks. In Fig. 8, we show the resynthesized images produced when using adversarial samples. Note that they massively differ from the input one. More examples are provided in the supplementary material.

Dataset	Model	Method	Detection			
			DAG		Houdini	
			Pure	Shift	Pure	Shift
Cityscapes	BSeg	SC	99%	98%	**100%**	**98%**
		Ours	**100%**	**100%**	**100%**	**98%**
	PSP	SC	98%	90%	98%	**100%**
		Ours	**100%**	**99%**	**99%**	**100%**
BDD	BSeg	SC	**100%**	**100%**	98%	**100%**
		Ours	98%	98%	**100%**	90%
	PSP	SC	92%	**100%**	96%	**100%**
		Ours	**100%**	96%	**98%**	95%

Table 1: **Attack detection on Cityscapes and BDD100K.** Our method achieves higher AUROC on Cityscapes than SC and comparable ones on BDD100K, despite the fact that we rely on a generator trained on Cityscapes.

5. Conclusion

In this paper, we have introduced a drastically new approach to detecting the unexpected in images. Our method is built on the intuition that, because unexpected objects have not been seen during training, typical semantic segmentation networks will produce spurious labels in the corresponding regions. Therefore, resynthesizing an image from the semantic map will yield discrepancies with respect to the input image, and we introduced a network that learns to detect the meaningful ones. Our experiments have shown that our approach detects the unexpected objects much more reliably than uncertainty- and autoencoder-based techniques. We have also contributed a new dataset with annotated road anomalies, which we believe will facilitate research in this relatively unexplored field. Our approach still suffers from the presence of some false positives, which, in a real autonomous driving scenario would create a source of distraction. Reducing this false positive rate will therefore be the focus of our future research.

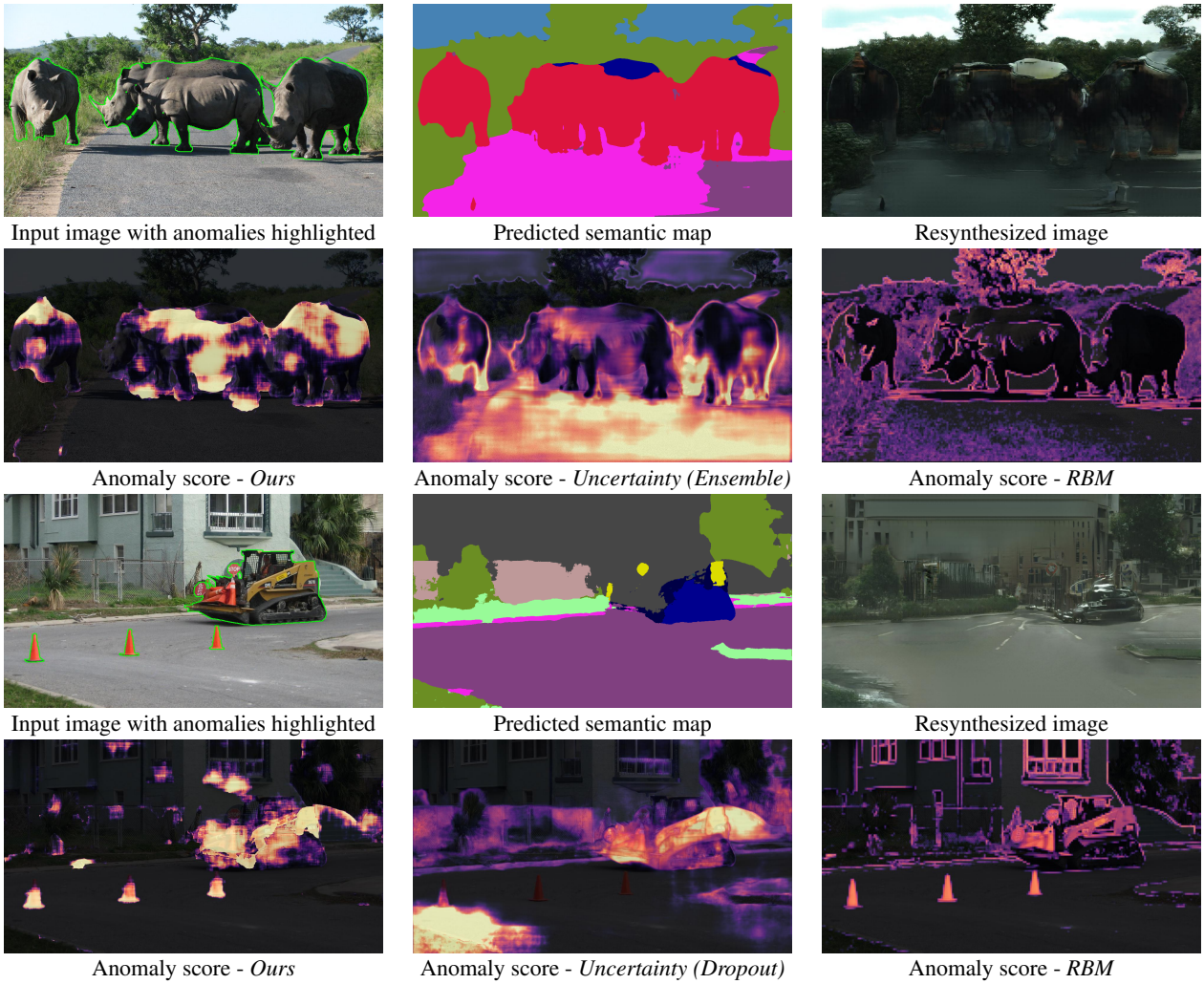


Figure 7: **Road anomaly results.** As in Fig. 6, in each pairs of rows, the consecutive images at the top depict algorithmic steps and the ones at the bottom our results along with those of the baselines.

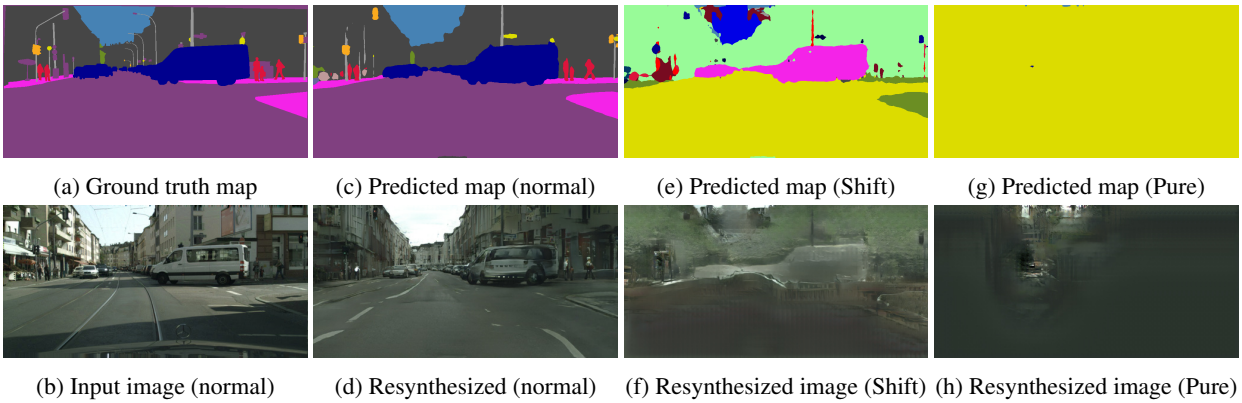


Figure 8: **Visualizing adversarial attacks.** Without attacks, the resynthesized image **(d)**, obtained from **(c)**, looks similar to the input one **(b)**. By contrast, resynthesized images (**(f)** and **(h)**) obtained from the semantic maps (**(e)** and **(g)**) computed from an attacked input differ massively from the original one.

References

[1] S. Akcay, A. A. Abarghouei, and T. P. Breckon. Ganomaly: Semi-Supervised Anomaly Detection via Adversarial Training. *arXiv Preprint*, abs/1805.06725, 2018. 1, 3

[2] V. Badrinarayanan, A. Kendall, and R. Cipolla. Segnet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017. 3, 11

[3] N. Carlini and D. Wagner. Towards Evaluating the Robustness of Neural Networks. In *IEEE Symposium on Security and Privacy*, pages 39–57, 2017. 3

[4] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam. Rethinking Atrous Convolution for Semantic Image Segmentation. *arXiv Preprint*, abs/1706.05587, 2017. 1

[5] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam. Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation. In *European Conference on Computer Vision*, 2018. 1

[6] M. M. Cisse, Y. Adi, N. Neverova, and J. Keshet. Houdini: Fooling Deep Structured Visual and Speech Recognition Models with Adversarial Examples. In *Advances in Neural Information Processing Systems*, pages 6977–6987, 2017. 2, 3, 6

[7] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The Cityscapes Dataset for Semantic Urban Scene Understanding. In *Conference on Computer Vision and Pattern Recognition*, 2016. 1, 5, 11

[8] C. Creusot and A. Munawar. Real-Time Small Obstacle Detection on Highways Using Compressive RBM Road Reconstruction. In *Intelligent Vehicles Symposium*, 2015. 1, 2, 3, 5, 14

[9] N. Dalal and B. Triggs. Histograms of Oriented Gradients for Human Detection. In *Conference on Computer Vision and Pattern Recognition*, pages 886–893, 2005. 5

[10] J. S. Denker and Y. LeCun. Transforming Neural-Net Output Levels to Probability Distributions. In *Advances in Neural Information Processing Systems*, 1991. 2

[11] Y. Gal and Z. Ghahramani. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. In *International Conference on Machine Learning*, pages 1050–1059, 2016. 2

[12] J. Gast and S. Roth. Lightweight Probabilistic Deep Networks. In *Conference on Computer Vision and Pattern Recognition*, 2018. 1, 2

[13] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and Harnessing Adversarial Examples. *International Conference on Learning Representations*, 2015. 3

[14] J. Hendrik Metzen, T. Genewein, V. Fischer, and B. Bischoff. On Detecting Adversarial Perturbations. *International Conference on Learning Representations*, 2017. 3

[15] X. Huang, X. Cheng, Q. Geng, B. Cao, D. Zhou, P. Wang, Y. Lin, and R. Yang. The Apolloscape Dataset for Autonomous Driving. In *Conference on Computer Vision and Pattern Recognition Workshops*, 2018. 1

[16] S. Isobe and S. Arai. A Semantic Segmentation Method Using Model Uncertainty. In *IIAE International Conference on Intelligent Systems and Image Processing*, 2017. 2

[17] S. Isobe and S. Arai. Deep Convolutional Encoder-Decoder Network with Model Uncertainty for Semantic Segmentation. In *IEEE International Conference on INnovations in Intelligent SysTems and Applications*, 2017. 2

[18] S. Isobe and S. Arai. Inference with Model Uncertainty on Indoor Scene for Semantic Segmentation. In *IEEE Global Conference on Signal and Information Processing*, 2017. 2

[19] A. Kendall, V. Badrinarayanan, and R. Cipolla. Bayesian Segnet: Model Uncertainty in Deep Convolutional Encoder-Decoder Architectures for Scene Understanding. *arXiv Preprint*, 1511.02680, 2015. 1, 2, 5

[20] A. Kendall and Y. Gal. What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision? In *Advances in Neural Information Processing Systems*, 2017. 1, 2

[21] D. P. Kingma and J. Ba. Adam: A Method for Stochastic Optimisation. In *International Conference on Learning Representations*, 2015. 11

[22] B. R. Kiran, D. M. Thomas, and R. Parakkal. An Overview of Deep Learning Based Methods for Unsupervised and Semi-Supervised Anomaly Detection in Videos. *Journal of Imaging*, 2018. 3

[23] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter. Self-normalizing neural networks. In *Advances in Neural Information Processing Systems*, 2017. 11

[24] A. Kurakin, I. Goodfellow, and S. Bengio. Adversarial Machine Learning at Scale. *International Conference on Learning Representations*, 2017. 3

[25] B. Lakshminarayanan, A. Pritzel, and C. Blundell. Simple and Scalable Predictive Uncertainty Estimation Using Deep Ensembles. In *Advances in Neural Information Processing Systems*, 2017. 1, 2, 5

[26] K. Lee, K. Lee, H. Lee, and J. Shin. A Simple Unified Framework for Detecting Out-Of-Distribution Samples and Adversarial Attacks. In *Advances in Neural Information Processing Systems*, pages 7167–7177, 2018. 3

[27] W. Li, O. H. Jafari, and C. Rother. Deep Object Co-Segmentation. In *Asian Conference on Computer Vision*, 2018. 4, 11

[28] X. Ma, B. Li, Y. Wang, S. M. Erfani, S. Wijewickrema, G. Schoenebeck, D. Song, M. E. Houle, and J. Bailey. Characterizing Adversarial Subspaces Using Local Intrinsic Dimensionality. *International Conference on Learning Representations*, 2018. 3

[29] D. J. MacKay. A Practical Bayesian Framework for Backpropagation Networks. *Neural Computation*, 4(3):448–472, 1992. 2

[30] D. J. Mackay. Bayesian Neural Networks and Density Networks. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 354(1):73–80, 1995. 2

[31] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard. Deepfool: A Simple and Accurate Method to Fool Deep Neural Networks. In *Conference on Computer Vision and Pattern Recognition*, pages 2574–2582, 2016. 3

[32] A. Munawar and C. Creusot. Structural Inpainting of Road Patches for Anomaly Detection. In *IAPR International Conference on Machine Vision Applications*, 2015. 3

[33] A. Munawar, P. Vinayavekhin, and G. De Magistris. Limiting the Reconstruction Capability of Generative Neural Network Using Negative Learning. In *IEEE International Workshop on Machine Learning for Signal Processing*, 2017. 1, 3

[34] A. Paszke, A. Chaurasia, S. Kim, and E. Culurciello. Enet: A Deep Neural Network Architecture for Real-Time Semantic Segmentation. *arXiv Preprint*, abs/1606.02147, 2016. 11

[35] P. Pinggera, S. Ramos, S. Gehrig, U. Franke, C. Rother, and R. Mester. Lost and Found: Detecting Small Road Hazards for Self-Driving Vehicles. In *International Conference on Intelligent Robots and Systems*, 2016. 2, 5, 6

[36] M. Ravanbakhsh, M. Nabi, E. Sangineto, L. Marcenaro, C. Regazzoni, and N. Sebe. Abnormal Event Detection in Videos Using Generative Adversarial Nets. In *International Conference on Image Processing*, 2017. 3

[37] C. Rother, V. Kolmogorov, and A. Blake. "GrabCut" - Interactive Foreground Extraction Using Iterated Graph Cuts. In *ACM SIGGRAPH*, pages 309–314, 2004. 6

[38] T. Schlegl, P. Seeböck, S. M. Waldstein, U. Schmidt-Erfurth, and G. Langs. Unsupervised Anomaly Detection with Generative Adversarial Networks to Guide Marker Discovery. In *International Conference on Information Processing in Medical Imaging*, 2017. 3

[39] K. Simonyan and A. Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *International Conference on Learning Representations*, 2015. 3, 4, 11

[40] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 2014. 2

[41] F. Tramèr, A. Kurakin, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel. Ensemble adversarial training: Attacks and defenses. *arXiv*, 2017. 3

[42] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro. High-Resolution Image Synthesis and Semantic Manipulation with Conditional GANs. *Conference on Computer Vision and Pattern Recognition*, 2018. 2, 3, 4

[43] C. Xiao, R. Deng, B. Li, F. Yu, M. Liu, and D. Song. Characterizing adversarial examples based on spatial consistency information for semantic segmentation. In *European Conference on Computer Vision*, pages 217–234, 2018. 2, 3, 7, 11

[44] C. Xie, J. Wang, Z. Zhang, Y. Zhou, L. Xie, and A. Yuille. Adversarial examples for semantic segmentation and object detection. In *International Conference on Computer Vision*, 2017. 2, 3, 6

[45] C. Yu, J. Wang, C. Peng, C. Gao, G. Yu, and N. Sang. Learning a Discriminative Feature Network for Semantic Segmentation. In *Conference on Computer Vision and Pattern Recognition*, 2018. 1

[46] F. Yu, W. Xian, Y. Chen, F. Liu, M. Liao, V. Madhavan, and T. Darrell. BDD100K: A Diverse Driving Video Database with Scalable Annotation Tooling. *arXiv Preprint*, 2018. 1, 5

[47] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid Scene Parsing Network. In *Conference on Computer Vision and Pattern Recognition*, 2017. 1, 3, 5, 11

Appendices

A. Detecting Unexpected Objects

The legend for the semantic class colors used throughout the article is given in Fig. 9. We present additional examples of the anomaly detection task in Fig. 10.

The synthetic training process alters only foreground objects. A potential failure mode could therefore be for the network to detect *all* foreground objects as anomalies, thus finding not only the true obstacles but also everything else. In Fig. 11, we show that this does not happen and that objects correctly labeled in the semantic segmentation are not detected as discrepancies.

In Fig. 12, we illustrate the fact that, sometimes, objects of known classes differ strongly in appearance from the instances of this class present in the training data, resulting in them being marked as unexpected.

We present a failure case of our method in Fig. 13: Anomalies similar to an existing semantic class are sometimes not detected as discrepancies if the semantic segmentation marks them as this similar class. For example, an animal is assigned to the *person* class and missed by the discrepancy network. In that case, however, the system as a whole is still aware of the obstacle because of its presence in the semantic map.

A.1. Discrepancy Network

Our discrepancy network relies on the implementations of *PSP Net* [47] and *SegNet* [2] kindly provided by Zijun Deng. The detailed architecture of the discrepancy network is shown in Fig. 14. We utilize a pre-trained VGG16 [39] to extract features from images and calculate their pointwise correlation, inspired by the co-segmentation network of [27]. The up-convolution part of the network contains SELU activation functions [23]. The discrepancy network was trained for 50 epochs using the Cityscapes [7] training set with synthetically changed labels as described in Section 3.2 of the main paper. We used the Adam [21] optimizer with a learning rate of 0.0001 and the per-pixel cross-entropy loss. We utilized the class weighting scheme introduced in [34] to offset the unbalanced numbers of pixels belonging to each class.

Supervised Discrepancy Network. To get an upper bound on its accuracy, we test the discrepancy network in a supervised setting. To this end, we use the ground-truth anomaly labels of the *Lost and Found* training set, with semantics predicted by PSP Net. The AUROC scores, measured on the test set, are shown in Table 2.

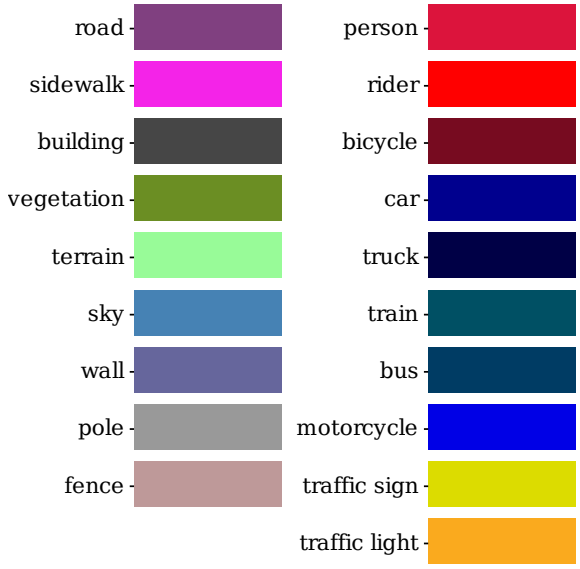


Figure 9: **Semantic map legend.** The colors used in semantic maps throughout this article correspond to the object classes listed above.

	Full	Labels only	Resynthesis only
Supervised	0.94	0.93	0.96
Unsupervised	0.82	0.79	0.76

Table 2: **Performance of the discrepancy network in a supervised setting.** AUROC scores measured on the *Lost and Found* dataset.

B. Detecting Adversarial Samples

We show additional results on adversarial example detection on the Cityscapes and BDD datasets using the Houdini and DAG attack schemes in Figs. 15 and 16. To obtain these results, we set the maximal number of iterations to 200 in all settings and L_∞ perturbation of 0.05 across each iteration of the attack. We randomly choose 80% of the original validation samples to train the logistic detectors and the rest of the samples are used for evaluation. While evaluating the state-of-the-art Scale Consistency method [43], we found by cross-validation that a patch size of 256×256 resulted in the best performance for an input image of size 1024×512.

C. Image Attribution

We used Wikimedia Commons images kindly provided under the Creative Commons Attribution license by the following authors: Thomas R Machnitzki [link], Megan Beckett [link], Infrogmation [link], Kyah [link], PIXNIO [link], Matt Buck [link], Luca Canepa [link], Jonas Buchholz [link] and Kelvin JM [link].

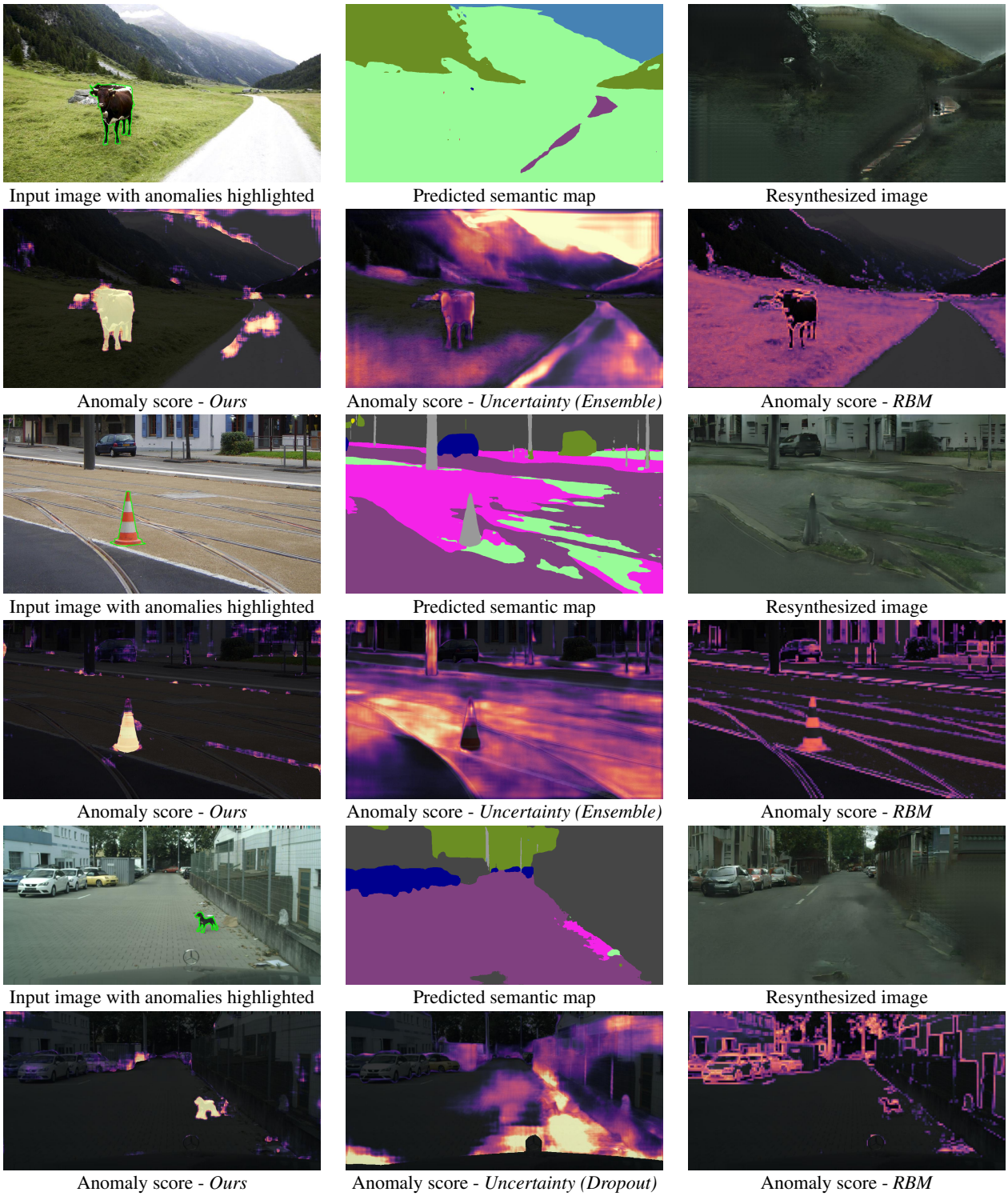


Figure 10: Additional examples of the anomaly detection task

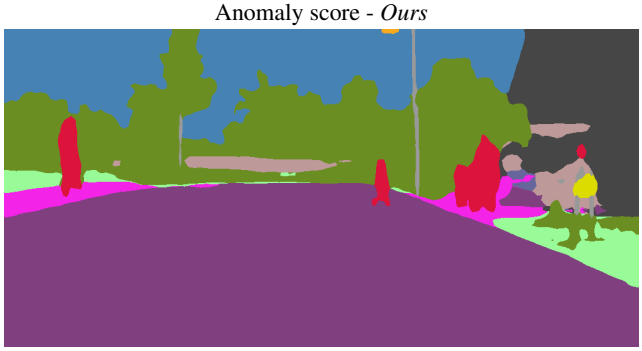
Input image

Predicted semantic map - Baysesian Seg Net

Resynthesized image (labels from Baysesian Seg Net)

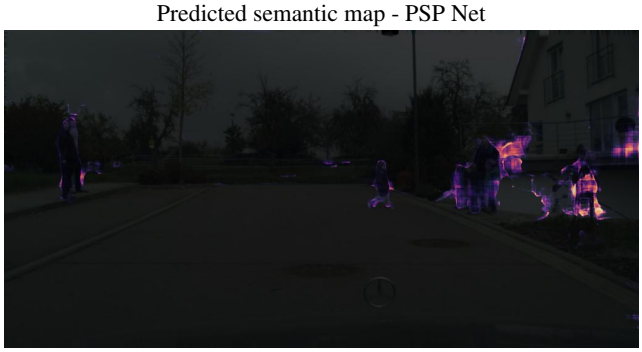
Anomaly score - *Ours*

Input image



Predicted semantic map - PSP Net

Resynthesized image (labels from PSP Net)



Anomaly score - *Ours*

Figure 11: The synthetic training process alters only foreground objects, but that does not mean our discrepancy network learns to blindly mark all such objects. In the top row, we show an example where the *Bayesian SegNet* failed to correctly label some of the people present, and this discrepancy is detected by our network. However, our detector reports no discrepancy when the *PSP Net* correctly labels the people in the image (third row).

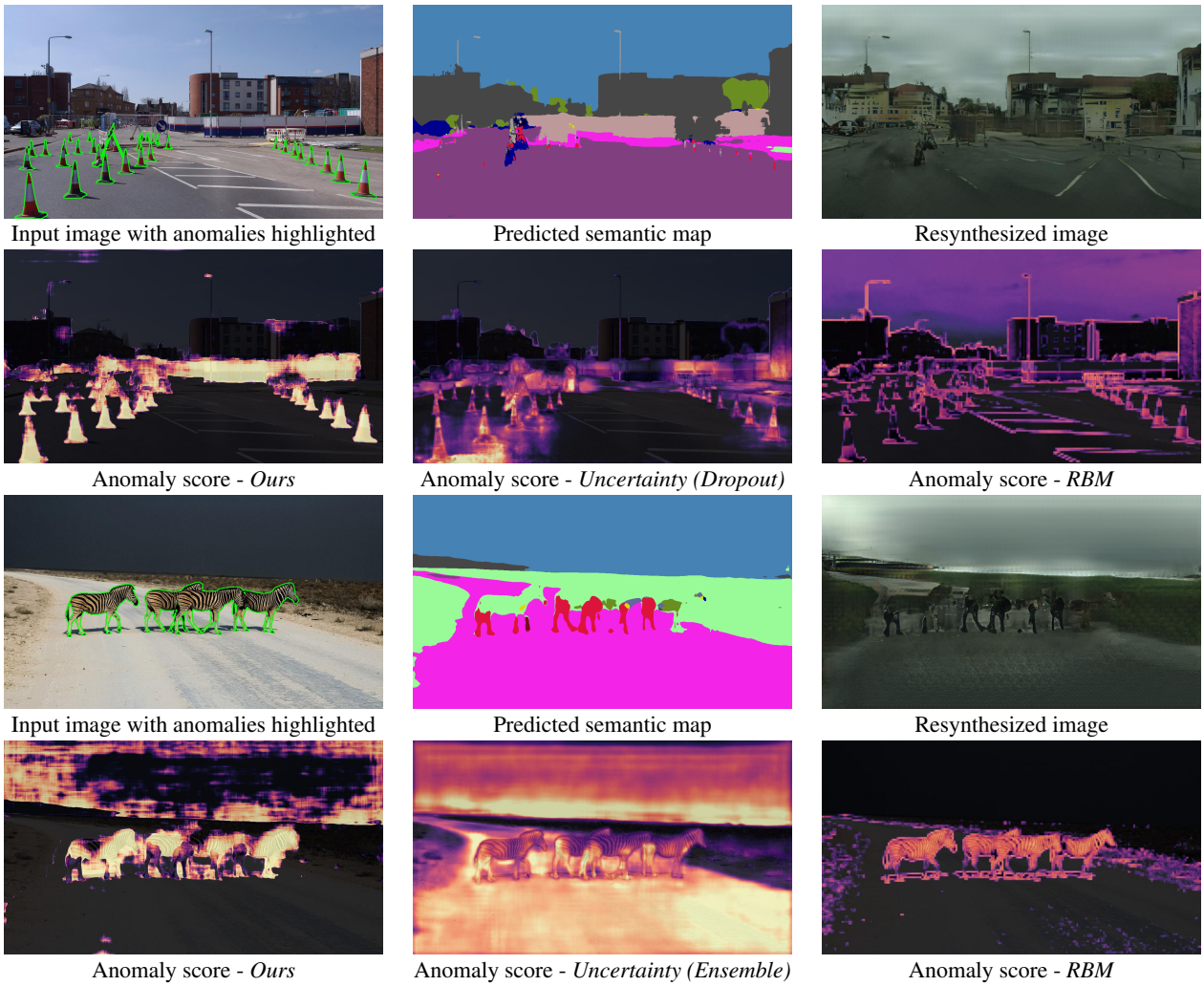


Figure 12: **Unusual versions of known objects**. Objects of known classes are marked as anomalies because their appearance differs from the examples of this class present in the training data, for example the fence in the first row (*fence* class) and the dark sky in the third row. Note that the *RBM* patch-based method [8] is especially sensitive to edges and so it detects the zebras very well.

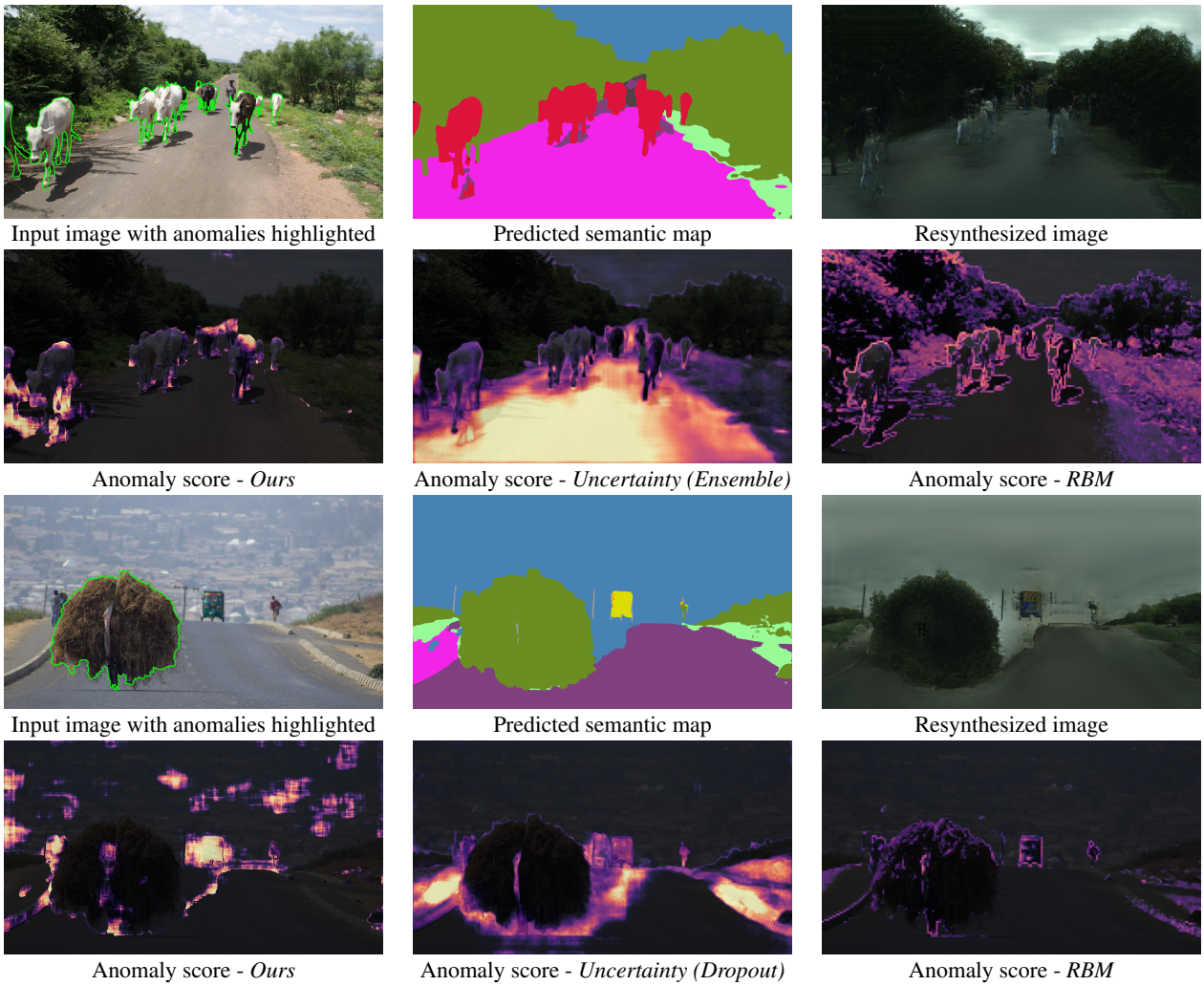


Figure 13: **Failure cases.** Our approach sometimes fails when the anomaly bears resemblance to an existing class: For example, animals classified as people in the first row or transported hay classified as vegetation in the third row. The system as a whole is nonetheless still aware of the obstacle because of its presence in the semantic map.

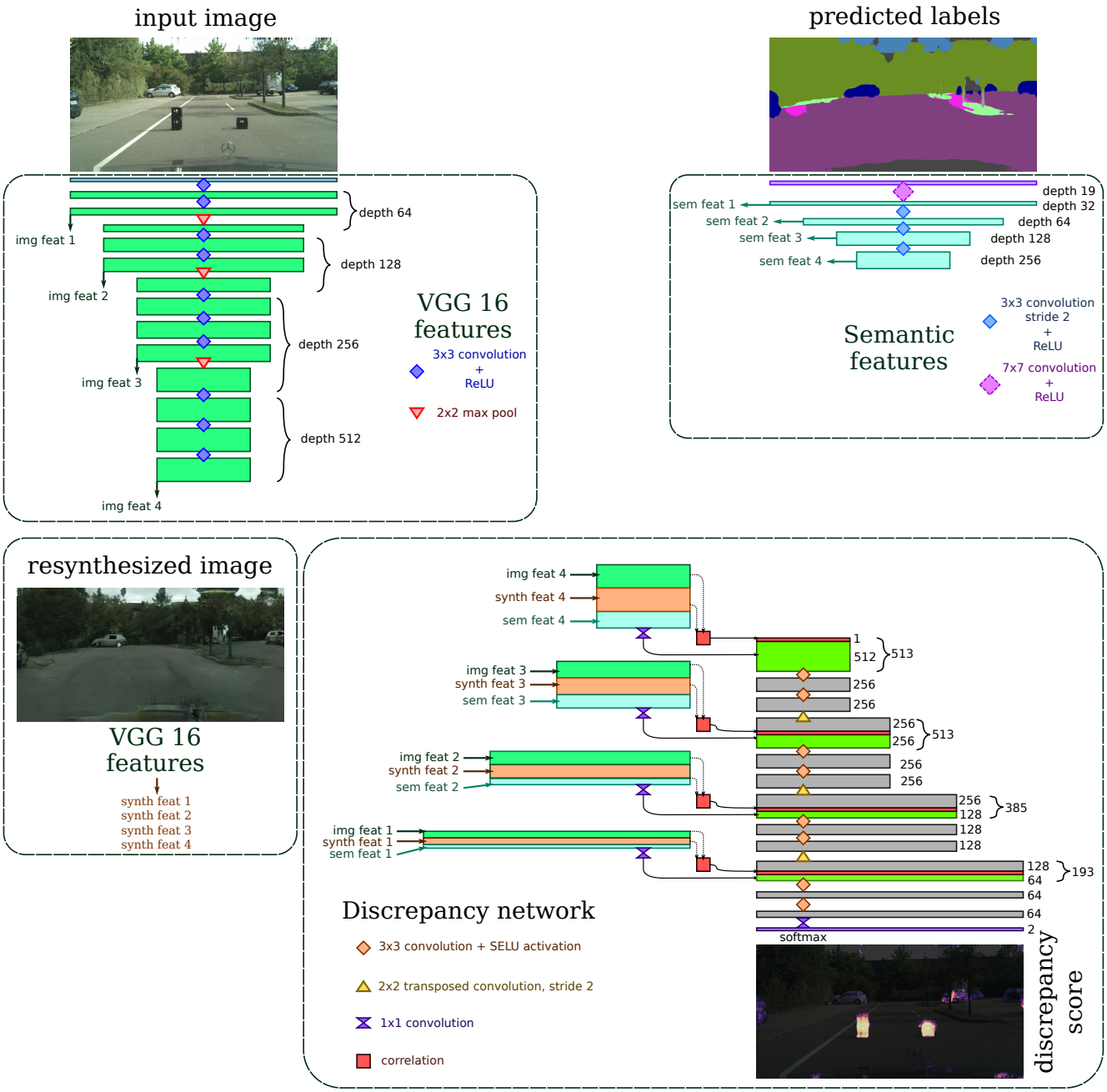
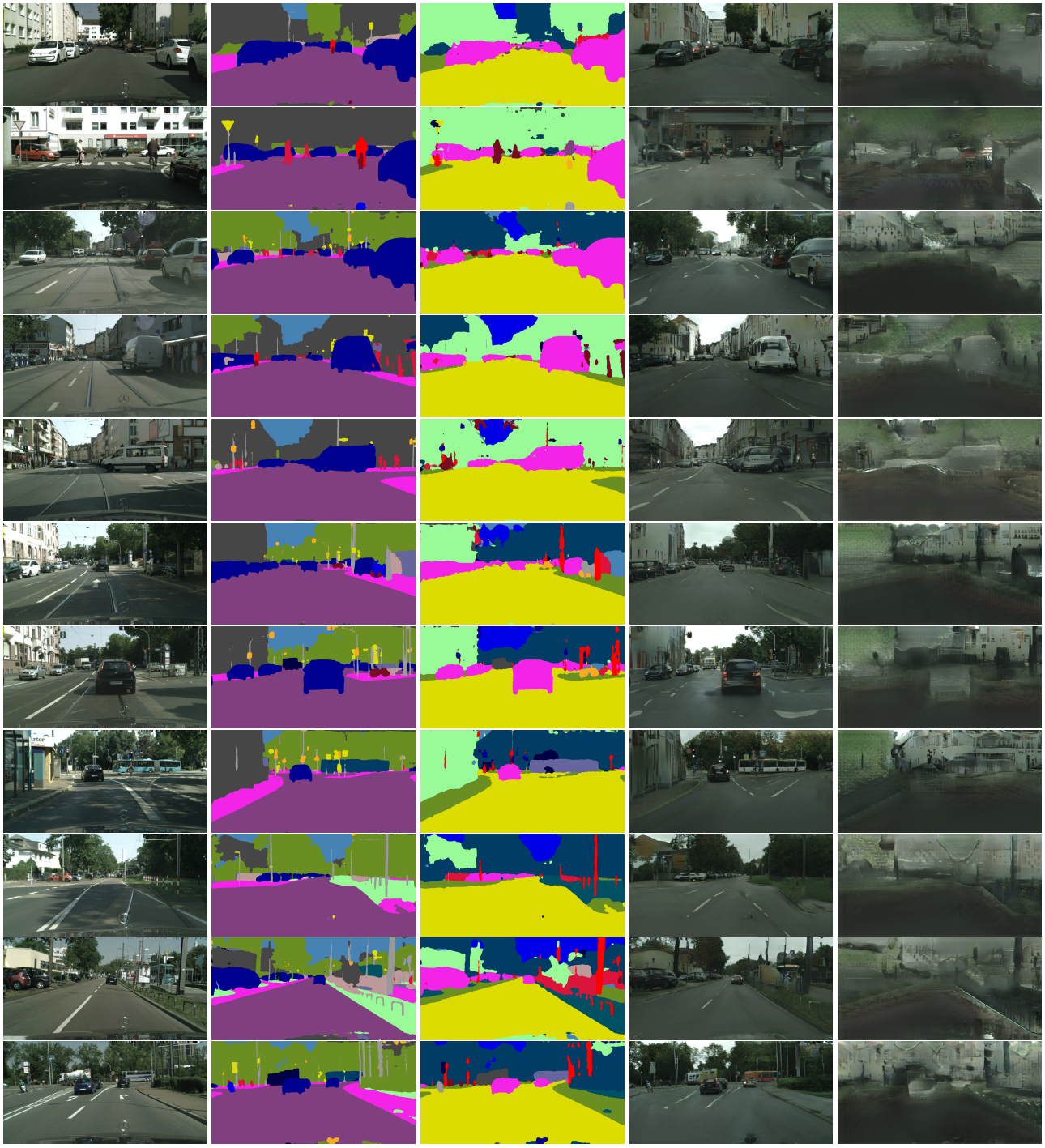
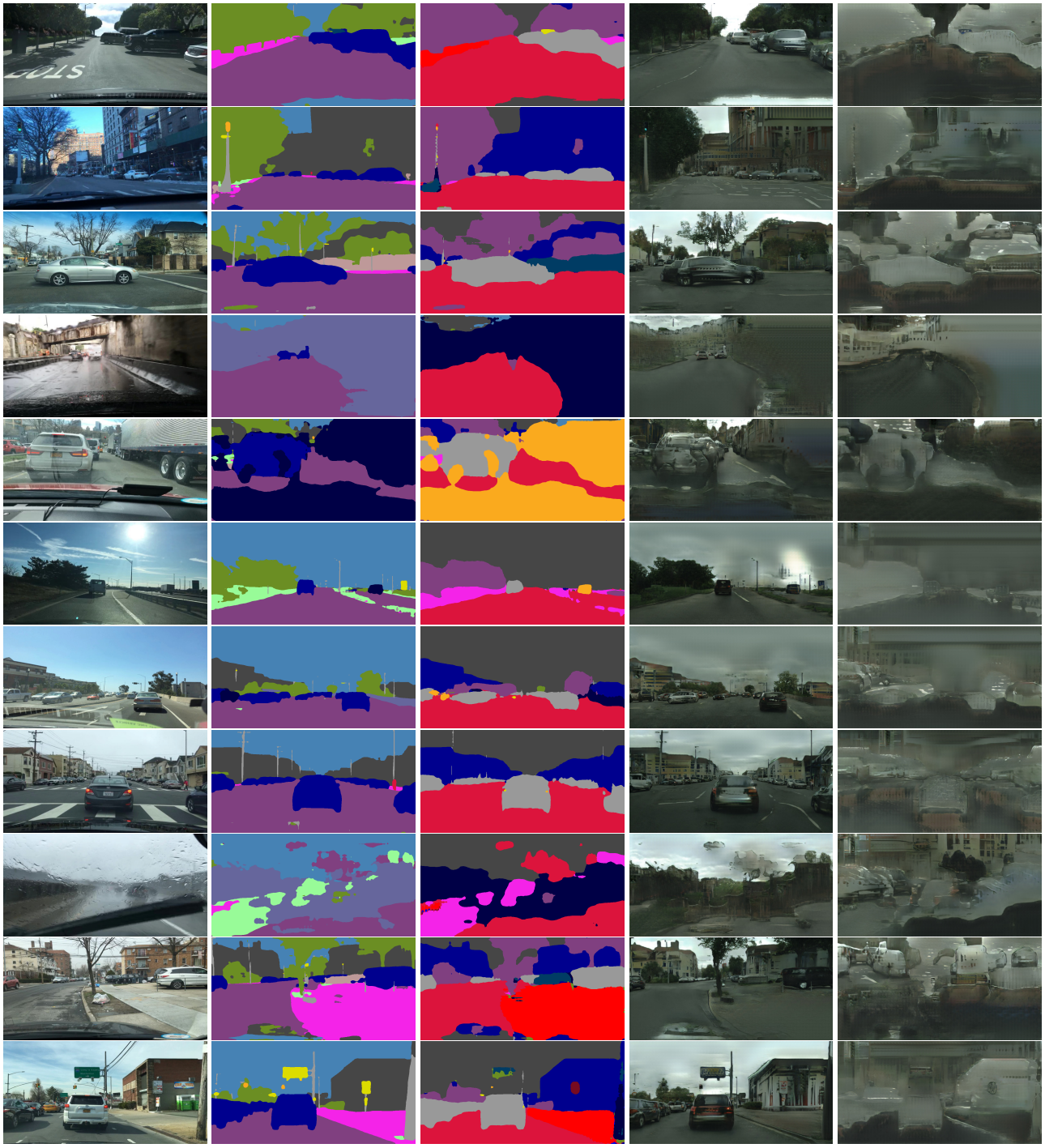


Figure 14: **Architecture of our discrepancy network.**



(a) Input image (normal) (b) Predicted map (normal) (c) Predicted map (Shift) (d) Resynthesized image (normal) (e) Resynthesized image (Shift)

Figure 15: **Detecting Houdini adversarial attacks on Cityscapes.** Without attack, the re-synthesized image **(d)** obtained from **(b)** looks similar to it. By contrast, the resynthesized image **(e)** obtained from the semantic maps **(c)** computed from a Houdini-compromised input differs massively from the original one.



(a) Input image (normal) (b) Predicted map (normal) (c) Predicted map (Shift) (d) Resynthesized image (normal) (e) Resynthesized image (Shift)

Figure 16: **Detecting DAG adversarial attacks on the BDD dataset.** Without attack, the re-synthesized image **(d)** obtained from **(b)** looks similar to it. By contrast, the resynthesized image **(e)** obtained from the semantic maps **(c)** computed from a DAG-compromised input differs massively from the original one.