Play and Learn: Using Video Games to Train Computer Vision Models

Alireza Shafaei
http://cs.ubc.ca/~shafaei

James J. Little
http://cs.ubc.ca/~little

Mark Schmidt
http://cs.ubc.ca/~schmidtm

Department of Computer Science
University of British Columbia
Vancouver, Canada

Abstract

Video games are a compelling source of annotated data as they can readily provide fine-grained groundtruth for diverse tasks. However, it is not clear whether the synthetically generated data has enough resemblance to the real-world images to improve the performance of computer vision models in practice. We present experiments assessing the effectiveness on real-world data of systems trained on synthetic RGB images that are extracted from a video game. We collected over 60,000 synthetic samples from a modern video game with similar conditions to the real-world CamVid and Cityscapes datasets. We provide several experiments to demonstrate that the synthetically generated RGB images can be used to improve the performance of deep neural networks on both image segmentation and depth estimation. These results show that a convolutional network trained on synthetic data achieves a similar test error to a network that is trained on real-world data for dense image classification. Furthermore, the synthetically generated RGB images can provide similar or better results compared to the real-world datasets if a simple domain adaptation technique is applied. Our results suggest that collaboration with game developers for an accessible interface to gather data is potentially a fruitful direction for future work in computer vision.

1 Introduction

Deep neural networks have been setting new records in almost all of the computer vision challenges and continue to grow in the broader field of artificial intelligence. One of the key components of a successful deep network recipe is the availability of a sufficiently large dataset for training and evaluation. The Imagenet Large Scale Visual Recognition Challenge (ILSVRC) [35], with over 1 million training images, has been one of the primary benchmarks for evaluation of several deep network architectures over the past few years. As architectures such as Resnet [15] approach a 3.57% top-5 error on the Imagenet classification task, it is likely that a larger and more complex dataset will become necessary soon. However, compiling datasets has its difficulties – depending on the desired granularity of the groundtruth, the costs and the labor could grow rather quickly.

Meanwhile, to cope with limited available data, there are several approaches to the training of deep networks. The common practices include data augmentation through various image transformations [19] or use of pre-trained networks that are trained for a similar task as a warm-start in the original optimization problem. A more radical approach is to synthesize training samples through computer-generated (CG) imagery.

One of the main concerns in the use of CG data for the training is whether the source domain is close enough to the target domain (*i.e.*, the domain of real-world images) to make an effective application possible. For depth images of the Kinect sensors, for instance, several lines of work [33, 38, 40, 45] demonstrate the efficacy and practical similarities between computer-generated depth images and the output of consumer level depth sensors. There are also methods that rely on indirect representations such as the HOG templates [16] or geometrical descriptors [23, 42], that can be efficiently synthesized and used. For the direct synthesis and application of RGB data, however, there is a limited amount of work that only investigates simple rendering of 3d CAD models [3, 4, 22, 30, 43] or synthetic text [14]. The question of interest is whether existing video games with photorealistic environments can be a useful source of synthetic RGB data to address computer vision tasks.

Although video games generate images from a finite set of textures, there is variation in viewpoint, illumination, weather, and level of detail which could provide valuable augmentation of the data. In addition to full control over the environment, video games can also provide us with groundtruth data such as dense image class annotations, depth information, radiance, irradiance, and reflectance which may not be straightforward, or even possible, to collect from real data. Other measures, such as the precise location of the game character within the environment, could be useful for development of visual SLAM algorithms.

In this work, we focus our attention on the RGB domain of a modern video game and run various experiments to gauge the efficacy of using CG RGB data directly for computer vision problems. We collect over 60,000 outdoor images under conditions similar to the CamVid [5] and the Cityscapes [7] datasets and present experiments on two computer vision problems: (i) dense image annotation, and (ii) depth estimation from RGB. We show that a convolutional network trained on synthetic data achieves a similar test error to a network that is trained on real-world data. Furthermore, after fine-tuning, our results show a network that is pre-trained on synthetic data can outperform a network that is pre-trained on real-world data.

2   Previous Work

Synthetic Data. Synthetic data has a successful history in computer vision. Taylor *et al.* [44] present a system called ObjectVideo Virtual Video (OVVV) based on Half-life [1] for evaluation of tracking in surveillance systems. Marin *et al.* [27] extend OVVV to perform pedestrian detection with HOG [8] features. In the recent literature, a variety of methods [4, 22, 30, 43] use 3d CAD models with simple rendering to tackle vision problems. Peng *et al.* [30], and Sun and Saenko [43] use non-photorealistic 3d CAD models to improve object detection. Lim *et al.* [22], and Aubry *et al.* [4] use CAD models for detection and object alignment in the image. Aubry and Russell [3] use synthetic RGB images rendered from CAD models to analyze the response pattern and the behavior of neurons in the commonly used deep convolutional networks. Rematas *et al.* [31] use 3d models to synthesize novel viewpoints of objects in real world images. Stark *et al.* [42], Lim *et al.* [23], and Liebelt and Schmid [21] learn intermediate geometric descriptors from 3d models to per-

form object detection. Butler *et al*. [6] present the synthetic Sintel dataset for evaluation of optical flow methods. Synthetic depth images are also successfully used for human pose estimation [38, 40] and hand pose estimation [43, 45]. Kaneva *et al*. [17] study robustness of image features under viewpoint and illumination changes in a photorealistic virtual world. In contrast to previous work, we take a different approach to synthetic models. Instead of rendering simple 3d CAD models in isolation, we take a step further and collect synthetic data in a simulated photorealistic world within the broad context of street scenes. We are specifically targeting the use of modern video games to generate densely annotated groundtruth to train computer vision models.

Dense Image Classification. Deep Convolutional Networks are extensively used for dense image segmentation [26, 29, 48, 49]. The fully convolutional networks of Long *et al*. [26] is among the first to popularize deep network architectures that densely label input images. Zheng *et al*. [49] build on top of the architecture in Long *et al*. [26] and integrate CRFs with Gaussian pairwise potentials to yield a significant gain in image segmentation. The current state-of-the-art methods, such as the one presented by Liu *et al*. [25], use variants of fully convolutional networks as a building block on top of which different recurrent neural networks or graphical models are proposed. We use the basic fully convolutional architecture of Long *et al*. [26] as it provides the basis of the follow-up developments in this area.

Depth Estimation from RGB. One of the early studies on unconstrained depth estimation from single RGB images is the work of Saxena *et al*. [36] in which the authors present a hierarchical Markov random field to estimate the depth. More recently, Zhuo *et al*. [50] present an energy minimization problem that incorporates semantic information at multiple levels of abstraction to generate a depth estimate. Li *et al*. [20] and Liu *et al*. [24] use deep networks equipped with a conditional random field that estimates the depth. More recently, Eigen and Fergus [10] presented a multi-scale deep convolutional architecture that can predict depth, normals, and dense labels. Instead of regressing against the metric data directly, Zoran *et al*. [51] propose a general method to estimate reflectance, shading, and depth by learning a model to predict ordinal relationships. The input image is first segmented into SLIC superpixels [2] on top of which a multi-scale neighbourhood graph is constructed. Zoran *et al*. [51] use the neighbourhood graph and generate ordinal queries on its edges, and then use the results to construct a globally consistent ranking by solving a quadratic program. We apply the method of Zoran *et al*. [51] in our study and show improvement in the depth estimation task through the use of synthetic data.

Transfer Learning. A closely related area of work to our study is transfer learning (see Pan and Yang [28] for a review). The early studies of transfer learning with Deep Convolutional Networks successfully demonstrated domain adaptation through pre-training on source data and fine-tuning on target data [9, 39, 47]. Further studies such as the work of Ganin *et al*. [12] present more sophisticated approaches to domain adaptation. In this work, we apply the most widely used fine-tuning approach to domain adaptation and leave further studies on feature transferability of the synthetic data to the future work.

3 Datasets

In this work, we focus on outdoor on-the-road datasets as we can easily collect this data from the video game. For real-world images, there are a few available datasets [5, 7, 13, 37] of which we use Cityscapes [7] and CamVid [5]. The selected datasets provide the most appropriate setting regarding the available groundtruth and visual similarity.
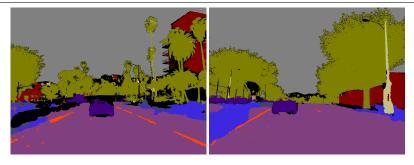
Figure 1: Densely labeled samples from the VG+ dataset. The label space of this dataset is the same as the CamVid [5] dataset.

3.1 VG Dataset

The dataset consists of over 60,000 frames collected from a video game[1]. To gather this data we use a camera on the hood of a car, similar to the configuration of the CamVid [5] or Cityscapes [7] datasets. The weather is kept fixed at sunny and the time of the day is fixed at 11:00 AM. This atmospheric setting was chosen to make the synthetic data similar to the real-world data, although note that a key advantage of VG is that it would be easy to sample data under non-ideal conditions (while it might be impossible to collect reliable real data under many conditions). The image resolution of the game is 1024×768 with the highest possible graphics configuration. The autonomous driver randomly drives around the city while obeying the traffic laws. Every second, a sample data is collected from the game. Each sample contains the RGB image, groundtruth semantic segmentation, depth image, and the surface normals. The groundtruth semantic segmentation that we were able to automatically extract from the game is over the label set {Sky, Pedestrian, Cars, Trees}. The help of the game developers is likely to be required in order to extract more labels.

We also consider a label-augmented version of the VG dataset, which we call VG+. To augment the label space with additional classes, we use SegNet [18], one of the top performing methods on CamVid [5] dataset with available code and data, to classify the images of the VG. We then refine the labels with the true labels of the VG dataset and clean-up the data automatically using the depth and the surface normals. See Fig. 1 for samples.

Note that, unlike the groundtruth dense annotations of VG, the groundtruth label space $\mathcal{Y}$ in VG+ is noisy and may partially exhibit the biases of the SegNet [18]. However, the input space $\mathcal{X}$ remains the same. Consequently, the results that are derived from VG+ are weaker but still can provide us with useful insight as the input synthetic RGB is intact. In principle, it should be possible to automatically collect the groundtruth dense annotation of all the objects in the game.

3.2 Real-world Datasets

CamVid [5]. The CamVid dataset contains 701 densely labelled images collected by a driving car in a city (see Fig. 2). The label set is {Sky, Building, Pole, Road Marking, Road, Pavement, Tree, Sign Symbol, Fence, Vehicle, Pedestrian, Bike}. The data is split into 367, 101, and 233 images for train, validation, and test.

---

[1]Due to legal concerns we were asked to not disclose the title of the game.

Figure 2: Two random images and the corresponding annotations from the CamVid [5] dataset.
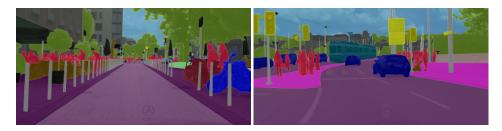

Figure 3: Two random images with the corresponding dense annotations from the Cityscapes [7] dataset.

Cityscapes [7]. The Cityscapes dataset offers 5000 finely annotated images and 20,000 coarsely annotated images of urban street scenes over 33 labels collected in cities in Germany and Switzerland. In addition to the dense and coarse image annotations, the dataset also includes car odometry readings, GPS readings, and the disparity maps that are calculated using the stereo camera. See Fig. 3 for pixel-level annotation examples. The densely annotated images are split into sets of 2975, 500, and 1525 for training, validation, and test. With the exception of 'Road Marking', all the other 11 classes of CamVid are present in Cityscapes. The groundtruth for the test set is kept private for evaluation purposes. Thus, we perform the Cityscapes evaluations on the validation set.

4 Dense Image Classification

For this task we use the fully convolutional network (FCN) of Long *et al.* [26]. More specifically, we use the FCN8 architecture on top of a 16-layer VGG Net [41]. FCN8 is a simple architecture that only uses convolution and deconvolution to perform classification, and provides a competitive baseline for dense image classification. All of the networks will be trained with the same settings: SGD with momentum (0.9), and pre-defined step sizes of 10^{-4}, 10^{-5}, and 10^{-6} for 50, 25, and 5 epochs. We use the MatConvNet [46] library.

To align our datasets with respect to the label space, we define two subsets of the data. The datasets CamVid, Cityscapes, and VG refer to a variation in which the label space is limited to {Pedestrian, Cars, Trees, Sky, Background}. In this setting, the label space of the VG is precise and automatically extracted from the game. The second variation CamVid+, Cityscapes+, and VG+ refers to the setting in which the label space is the full 12 classes of the original CamVid dataset and the labels in the synthetic VG+ is noisy. When we are evaluating on Cityscapes+ we omit the missing 'Road Marking' class.
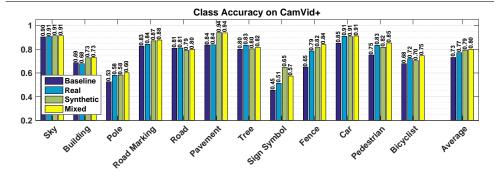
Figure 4: The per-class accuracy on the test set of `CamVid+` dataset. The `baseline` is trained on the target dataset, the `real` is pre-trained on the real alternative dataset, and the `synthetic` is pre-trained on `VG+`. The `Mixed` approach is pre-trained on both synthetic and the real alternative dataset. Pre-training the baseline with the synthetic `VG+` improves the average accuracy by 6%, while pre-training with the real-world `Cityscapes+` improves the average by 4%.

Model	CamVid+			Cityscapes+		
	Pixel Acc.	Class Acc.	Mean IoU	Pixel Acc.	Class Acc.	Mean IoU
Baseline	79%	73%	47%	83%	77%	50%
Real	80%	77%	51%	83%	77%	50%
Synthetic	**82%**	79%	52%	**84%**	**79%**	51%
Mixed	**82%**	**80%**	**53%**	**84%**	**79%**	**52%**

Table 1: Evaluation of different pre-training strategies on `CamVid+` and `Cityscapes+`, comparing pixel accuracy, mean class accuracy, and mean intersection over union (IoU). Pre-training on synthetic data consistently outperforms the equivalent model that is only pre-trained on real-world data. The mixed pre-training strategy gives the most improvement.

4.1   Evaluation with Fine-tuning

To measure the effect of using synthetic data, we look at the performance in a domain adaptation setting in which we perform fine-tuning on the target dataset. In this approach we successively train our dense classifier on different datasets and then examine the performance. In the first experiment we focus on the `CamVid+` dataset. We train four different networks and evaluate their performance on `CamVid+` to analyze the influence of our synthetic data. Each model is pre-trained on an alternative dataset(s) before training on the target dataset. For instance, the experiment with name `Synthetic` means that the network has been pre-trained on the synthetic `VG` dataset first, and then fine-tuned on the target dataset (see Fig. 4). The `Real` counterpart is the network that is pre-trained on the alternative real-world dataset first. The `Mixed` approach is when we pre-train on both synthetic and real-world data first.

Figure 4 compares our training strategies with respect to the per-class accuracy. Pre-training on `VG+` improves the average accuracy more than pre-training on `Cityscapes+` does, 79% vs. 77%. The most improvement is for the class 'Sign Symbol' where pre-training with `VG+` improves the accuracy by 20%. The highest improvement is achieved when we pre-train on both real and synthetic datasets. Table 1 shows a summary of our results.
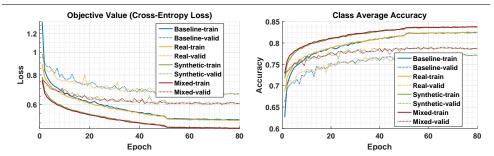
Figure 5: The influence of various pre-training approaches on the `Cityscapes+`. The left image is the objective function throughout training, and the right image is the class average accuracy. The solid lines are the evaluation results on the training set, the dashed lines are the results on the validation set. Pre-training on synthetic data gives a better initialization and final local minima compared to pre-training on a real-world dataset. The figure is best viewed in color.

We can also analyze the benefits of using this synthetic data from an optimization perspective. Figure 5 shows the objective value and the class average accuracy of `Cityscapes+` during the last optimization stage. Pre-training on the `VG+` datasets yields better results on train and validation in comparison to both the baseline, and the pre-trained version on the real-world `CamVid+` dataset. Pre-training on the `VG+` data provides a better initialization and final local minima in the training procedure. In both of the results, the synthetic data provides a clear improvement over the baseline, and helps as much or more than pre-training on the real data. More figures and analysis of these experiments can be found in the supplementary material.

4.2   Cross-dataset Evaluation

We also evaluate the models in a cross-dataset setting in which the network is evaluated on a dataset other than the one it was trained on. We train three dense image classifiers on each dataset and examine the cross-dataset accuracy of these classifiers. The purpose of this experiment is to see how much the domain of our synthetic data differs from a real-world dataset in comparison to another real-world dataset.

Figure 6 shows the cross-dataset accuracy on `CamVid` and `Cityscapes`. The first observation, as anticipated, is that the domain of a real-world dataset is more similar to the domain of another real-world dataset on average. Even though the `VG` network has been only trained on synthetic data, in 'pedestrian', 'car', or 'trees', it competes or even in two cases outperforms the network trained on real data. Although the `VG` network does not outperform the real counterpart on average, the small gap indicates that the network with synthetic data has learned relevant features and is not overfitting to the game specific textures that can be an obstacle to generalization to the real-world domain.

Figure 7 shows the per-class accuracy for each network on `Cityscapes+`. Similar to the previous results, the domain of `CamVid+` is more similar to the `Cityscapes+` than the synthetic `VG+` dataset is. However, `VG+` gives better results for pole, tree, sign symbol, fence, and bicyclist. On average, the network that is trained on `CamVid+` gives a 60% accuracy, and the network obtained from synthetic data gives a similar 56% accuracy. The similarity in performance suggests that for the training of computer vision models the synthetic data
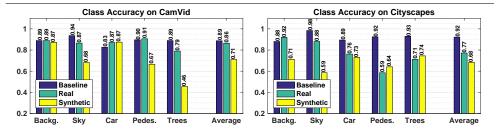
Figure 6: The cross-dataset per-class accuracy on `CamVid` (left) and `Cityscapes` (right). The baseline is trained on the target dataset, the real is trained on the real alternative dataset, and the synthetic is trained on `VG`.
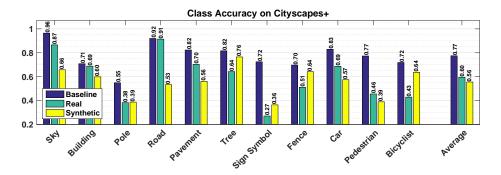

Figure 7: The cross-dataset per-class accuracy on the validation set of the `Cityscapes+` dataset. The baseline is trained on `Cityscapes+`, the real is trained on the `CamVid+`, and the synthetic is trained on `VG+`.

provides a reasonable proxy to the real world images. For more figures on this experiment please see the supplementary material.

5 Depth Estimation from RGB

The Cityscapes dataset also provides the disparity images which we will be using in this setting. For this problem we use the method of Zoran *et al.* [51], in which the underlying deep network is queried with two points in the image and has to decide which one is closer to the camera. This method is more attractive than the other techniques because it only relies on the ordinal relationships and not on the measurement unit in the target dataset. This is useful for our experiments because the depth images of the video game are tailored to improve the visualization effects in the rendering pipeline and are not directly comparable to the measurement units of the real-world datasets. We use the same deep network architecture as Zoran *et al.* [51]. The images are first decomposed into superpixels. The center of each superpixel is a node in the graph. Adjacent superpixels are connected on this graph in a multiscale fashion. The two end-points of each edge are then used to query a deep network that classifies the relative depth of two input patches as $\{=, >, <\}$. A global ranking of the pixels is then generated based on these queries by solving a quadratic program. Depending on the target ordinal relationships we can estimate the depth, shading, or reflectance. We apply this
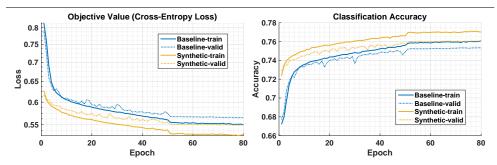
Figure 8: The influence of pre-training on synthetic data for the depth estimation task on the proposed network architecture of Zoran *et al*. [51].

method for depth estimation and train the underlying deep network on the Cityscapes and the VG datasets. While the networks in the previous experiments focused on higher level visual cues and abstractions, the nature of the network in this problem is concerned with the mid-level visual cues, which provides further insight to the quality of the synthetic data.

In Figure 8 we look at the influence of pre-training the network on the VG dataset vs. directly learning from the real dataset Cityscapes. Similar to the previous experiments we observe that pre-training with the synthetic data gives us a consisent improvement in initialization and the final local minima, in both validation and training. The final patch classification accuracy on the validation set is improved from 75% to 76%. For more figures on this experiment see the supplementary material.

6 Conclusion

Concurrently, a number of independent studies that explore similar ideas have been published recently. Gaidon *et al*. [11] present the Virtual KITTI dataset and show experiments on multi-object tracking tasks. Ros *et al*. [34] present the SYNTHIA dataset of urban scenes and also demonstrate improvement in dense image segmentation using synthetic data. Our study on using video games complements the recent work by providing analysis on the photorealistic output of a state-of-the-art game engine. Furthermore, an independent study to be published by Richter *et al*. [32] provide a similar analysis of using video games which we invite the reader to review for a complete picture.

As video games progress towards photorealistic environments, we can also use them for the training of computer vision models at no extra cost. We delivered a proof of concept by exploring the use of synthetic RGB images that we extracted from a video game. Our approach goes beyond the use of simplistic 3d CAD models as we collect a synthetic dataset that encompasses the broad context of street scenes. We presented several experiments to compare our synthetic dataset with the existing real-world ones. Our experiments show that in a cross-dataset setting, the deep neural networks that we trained on synthetic RGB images have a similar generalization power as the networks that we trained on real-world data. Furthermore, with a simple domain adaptation technique such as fine-tuning, pre-training on synthetic data consistently yielded better results than pre-training on real data. Moreover, we showed that pre-training on synthetic data resulted in a better initialization and final local minima in the optimization. On a network that classifies the ordinal relationship of image patches, we also observed how pre-training on synthetic data leads to improvement in optimization and final performance.

Our results suggest that RGB images collected from video games with photorealistic environments are potentially useful for a variety of computer vision tasks. Video games can offer an alternative way to compile large datasets for direct training or augmenting real-world datasets. Although we limited our attention to a fixed atmospheric setting to facilitate comparing to the existing real-world datasets, a key advantage is the ability to capture and use data under a variety of environmental parameters. For example, it would be easy to collect data under winter, night, fog, and other conditions where reliable real data is difficult to obtain.

References

[1] Half-Life 2. http://www.valvesoftware.com/games/hl2.html.

[2] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Susstrunk. SLIC superpixels compared to state-of-the-art superpixel methods. *TPAMI*, 2012.

[3] Mathieu Aubry and Bryan C. Russell. Understanding deep features with computer-generated imagery. In *ICCV*, 2015.

[4] Mathieu Aubry, Daniel Maturana, Alexei A. Efros, Bryan C. Russell, and Josef Sivic. Seeing 3d chairs: Exemplar part-based 2d-3d alignment using a large dataset of cad models. In *CVPR*, 2014.

[5] Gabriel J. Brostow, Julien Fauqueur, and Roberto Cipolla. Semantic object classes in video: A high-definition ground truth database. *Pattern Recognition Letters*, 2009.

[6] Daniel J. Butler, Jonas Wulff, Garrett B. Stanley, and Michael J. Black. A naturalistic open source movie for optical flow evaluation. In *ECCV*, 2012.

[7] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The Cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016.

[8] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005.

[9] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. DeCAF: A deep convolutional activation feature for generic visual recognition. In *ICML*, 2014.

[10] David Eigen and Rob Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *ICCV*, 2015.

[11] Adrien Gaidon, Qiao Wang, Yohann Cabon, and Eleonora Vig. Virtual worlds as proxy for multi-object tracking analysis. In *CVPR*, 2016.

[12] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *JMLR*, 2016.

[13] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The KITTI dataset. *The International Journal of Robotics Research*, 2013.

[14] Ankush Gupta, Andrea Vedaldi, and Andrew Zisserman. Synthetic data for text localisation in natural images. *arXiv preprint arXiv:1604.06646*, 2016.

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[16] Mohsen Hejrati and Deva Ramanan. Analysis by synthesis: 3d object recognition by object reconstruction. In *CVPR*, 2014.

[17] Biliana Kaneva, Antonio Torralba, and William T. Freeman. Evaluation of image features using a photorealistic virtual world. In *ICCV*, 2011.

[18] Alex Kendall, Vijay Badrinarayanan, and Roberto Cipolla. Bayesian segnet: Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding. *arXiv preprint arXiv:1511.02680*, 2015.

[19] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

[20] Bo Li, Chunhua Shen, Yuchao Dai, Anton van den Hengel, and Mingyi He. Depth and surface normal estimation from monocular images using regression on deep features and hierarchical CRFs. In *CVPR*, 2015.

[21] Joerg Liebelt and Cordelia Schmid. Multi-view object class detection with a 3d geometric model. In *CVPR*, 2010.

[22] Jasmine J. Lim, Hamed Pirsiavash, and Antonio Torralba. Parsing ikea objects: Fine pose estimation. In *ICCV*, 2013.

[23] Joseph J. Lim, Aditya Khosla, and Antonio Torralba. FPM: Fine pose parts-based model with 3d CAD models. In *ECCV*, 2014.

[24] Fayao Liu, Chunhua Shen, and Guosheng Lin. Deep convolutional neural fields for depth estimation from a single image. In *CVPR*, 2015.

[25] Ziwei Liu, Xiaoxiao Li, Ping Luo, Chen-Change Loy, and Xiaoou Tang. Semantic image segmentation via deep parsing network. In *ICCV*, 2015.

[26] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015.

[27] Javier Marin, David Vázquez, David Gerónimo, and Antonio M. López. Learning appearance in virtual scenarios for pedestrian detection. In *CVPR*, 2010.

[28] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 2010.

[29] George Papandreou, Liang Chen, Kevin Murphy, and Alan L. Yuille. Weakly- and semi-supervised learning of a deep convolutional network for semantic image segmentation. In *ICCV*, 2015.

[30] Xingchao Peng, Baochen Sun, Karim Ali, and Kate Saenko. Learning deep object detectors from 3d models. In *ICCV*, 2015.

[31] Konstantinos Rematas, Tobias Ritschel, Mario Fritz, and Tinne Tuytelaars. Image-based synthesis and re-synthesis of viewpoints guided by 3d models. In *CVPR*, 2014.

[32] Stephan R. Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: Ground truth from computer games. In *ECCV*, 2016.

[33] Grégory Rogez, James S. Supancic III, and Deva Ramanan. First-person pose recognition using egocentric workspaces. In *CVPR*, 2015.

[34] German Ros, Laura Sellart, Joanna Materzynska, David Vazquez, and Antonio M. Lopez. The SYNTHIA dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *CVPR*, 2016.

[35] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *IJCV*, 2015.

[36] Ashutosh Saxena, Sung H. Chung, and Andrew Y. Ng. 3-D depth reconstruction from a single still image. *IJCV*, 2008.

[37] Timo Scharwächter, Markus Enzweiler, Uwe Franke, and Stefan Roth. Efficient multi-cue scene segmentation. In *Pattern Recognition*. 2013.

[38] Alireza Shafaei and James J. Little. Real-time human motion capture with multiple depth cameras. In *CRV*, 2016.

[39] Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In *CVPR Workshops*, 2014.

[40] Jamie Shotton, Ross Girshick, Andrew Fitzgibbon, Toby Sharp, Mat Cook, Mark Finocchio, Richard Moore, Pushmeet Kohli, Antonio Criminisi, and Alex Kipman. Efficient human pose estimation from single depth images. *TPAMI*, 2013.

[41] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[42] Michael Stark, Michael Goesele, and Bernt Schiele. Back to the future: learning shape models from 3d CAD data. In *BMVC*, 2010.

[43] Baochen Sun and Kate Saenko. From virtual to reality: Fast adaptation of virtual object detectors to real domains. In *BMVC*, 2014.

[44] Geoffrey R. Taylor, Andrew J. Chosak, and Paul C. Brewer. OVVV: Using virtual worlds to design and evaluate surveillance systems. In *CVPR*, 2007.

[45] Jonathan Tompson, Murphy Stein, Yann Lecun, and Ken Perlin. Real-time continuous pose recovery of human hands using convolutional networks. *TOG*, 2014.

[46] Andrea Vedaldi and Karel Lenc. MatConvNet – Convolutional neural networks for MATLAB. In *Proceeding of the ACM Int. Conf. on Multimedia*, 2015.

[47] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *NIPS*. 2014.

[48] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. In *ICLR*, 2016.

[49] Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip Torr. Conditional random fields as recurrent neural networks. In *ICCV*, 2015.

[50] Wei Zhuo, Mathieu Salzmann, Xuming He, and Miaomiao Liu. Indoor scene structure analysis for single image depth estimation. In *CVPR*, 2015.

[51] Daniel Zoran, Phillip Isola, Dilip Krishnan, and William T. Freeman. Learning ordinal relationships for mid-level vision. In *ICCV*, 2015.
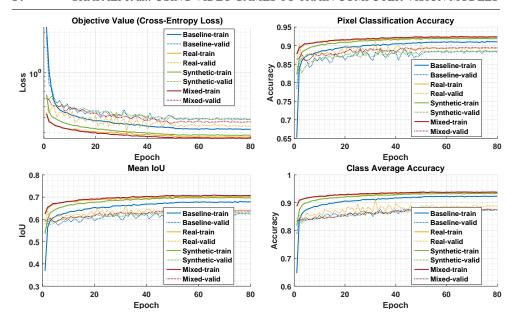
Figure 9: The influence of various pre-training approaches on the `CamVid` dataset. The solid lines are the evaluation results on the training set, the dashed lines are the results on the *validation* set.

Supplementary Material

Evaluation with Fine-tuning

Figures 9-12 compare the behavior of our convolutional networks in the training phase. We present the evolution of *objective value*, *pixel classification accuracy*, *class average accuracy*, and *mean intersection-over-union* for `CamVid`, `Cityscapes`, `CamVid+`, and `Cityscapes+` datasets. Pre-training on synthetic data consistently improves the initialization and the final solution, and in most cases also outperforms pre-training on real-world data.

Figure 13 compares the per-class accuracy of each training strategy on the test set of `CamVid` and the validation set of `Cityscapes`. Using synthetic data yields a consistent improvement over the baseline. On `CamVid`, pre-training on real data leads to a better model than pre-training on synthetic data, but the mixed approach has the best accuracy. On `Cityscapes`, however, pre-training on synthetic data has a higher average accuracy than pre-training on real-world data. Figure 14 shows the per-class accuracy on the `Cityscapes+` dataset. Similar to the previous experiments using synthetic data results in more improvement than using real-world data. Combining synthetic and real data gives the highest performance boost in these experiments.

Cross-dataset Evaluation

In the cross-dataset setting, we train one network on each dataset and evaluate the accuracy of each network on the other datasets. The purpose of this experiment is to measure and
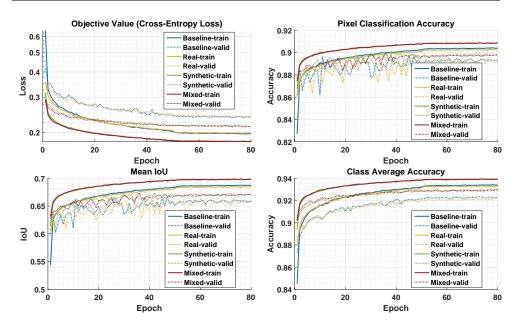
Figure 10: The influence of various pre-training approaches on the `Cityscapes` dataset. The solid lines are the evaluation results on the training set, the dashed lines are the results on the *validation* set.
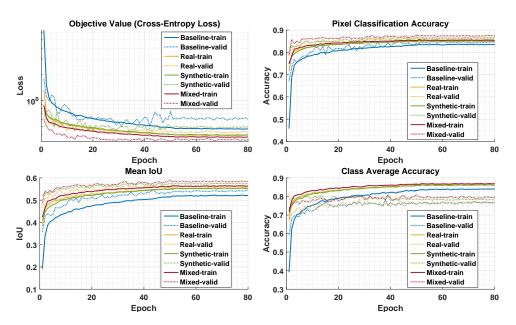

Figure 11: The influence of various pre-training approaches on the `CamVid+` dataset. The solid lines are the evaluation results on the training set, the dashed lines are the results on the *validation* set.
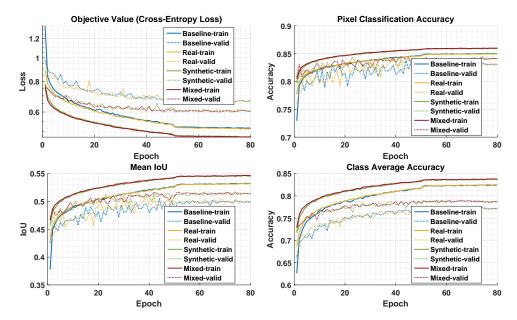
Figure 12: The influence of various pre-training approaches on the `Cityscapes+` dataset. The solid lines are the evaluation results on the training set, the dashed lines are the results on the *validation* set.
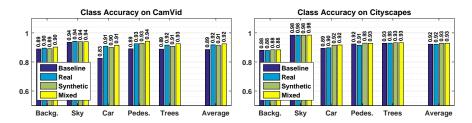

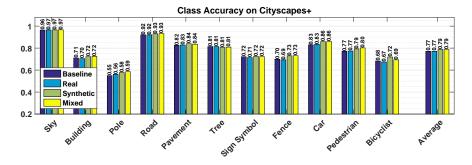Figure 13: The per-class accuracy on `CamVid` (left) and `Cityscapes` (right).

Figure 14: The per-class accuracy on `Cityscapes+`.

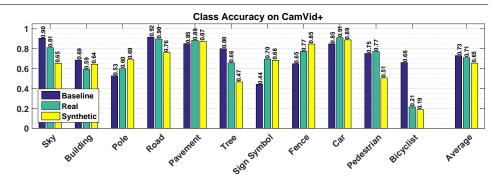**Class Accuracy on CamVid+**

Figure 15: Cross-dataset evaluation. The per-class accuracy on the test set of the `CamVid+` dataset.

compare the generalization power of the networks that are trained on synthetic or real data only. Figure 15 shows the per-class accuracy for evaluation on the `Camvid+` dataset. The `Baseline` network is directly trained on the target dataset, while the `Real` network is trained on the alternative real dataset, and the `Synthetic` network is trained on synthetic data only. Without domain adaptation, both of the `Real` and `Synthetic` networks have a lower accuracy than the `Baseline`. The network that is trained on real data has a better accuracy than the network that is trained on synthetic data only. Even though the `Synthetic` network is only trained on synthetic data, it outperforms the real network on 'Building', 'Pole', and 'Fence'. While the `Synthetic` network does not exceed the accuracy of the `Real` network on average, the small gap indicates that the network with synthetic data is relying on relevant features and is not merely overfitting to the game specific textures.

6.1 Depth Estimation from RGB

Zoran *et al.* [51] present a depth estimation method that only relies on the ordinal relationships between a set of image patch pairs. The image is first decomposed into SLIC [2] superpixels. A deep convolutional network classifies the ordinal relationship between the adjacent superpixels by generating a local relationship label $\{<, =, >\}$ with the corresponding probabilities. A quadratic program is then constructed to generate a total ordering (ranking) over the superpixels which will represent the depth. Note that this method does not rely on the depth measurement unit. Hence, it is not directly comparable to the prior work that directly regress to the depth value. We use this method because the depth information that is collected in the video game is not directly comparable to the real-world depth metrics. The ordinal relationships, however, can be consistently inferred from the extracted depth information. In the main paper, we demonstrated how using the synthetic RGB images can improve the patch classifier of Zoran *et al.* [51]. Figure 16 shows the groundtruth depth and the predicted depth image of a sample input from the `Cityscapes` dataset.
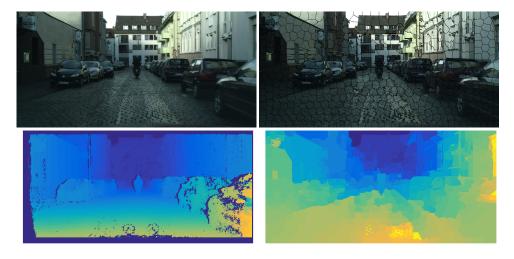
Figure 16: (top left) A sample image from the `Cityscapes` [7] dataset, (top right) decomposition of the RGB image to SLIC superpixels [2], (bottom left) the groundtruth disparity map, (bottom right) the globalized depth output of the method presented by Zoran *et al*. [51].