Data Augmentation for Autonomous Driving-Based Semantic Segmentation

Gerson C. Kroiz[1], Sander Schulhoff[2], Joshua Anantharaj[3], and Nickolas Drake[3]

[1] University of Maryland, Baltimore County, MD 21250,
[2] University of Maryland, College Park, MD 20742,
[3] University of Tennessee, Knoxville, TN 37996

Abstract. Collecting and segmenting real world data for autonomous driving tasks is financially and temporally expensive. The use of cheap, quickly created synthetic datasets could solve this issue. Unfortunately, the covariate shift between synthetic training data and real world testing data significantly decreases the applicability of synthetic data. To potentially improve the usefulness of synthetic data, we implement several data augmentation techniques including traditional transforms, random Copy+Paste, and style transfer. We apply these techniques to the combined CARLA-Cityscapes data set and train a semantic segmentation model based on the U-Net architecture.

Keywords: Deep learning, Semantic segmentation, Data augmentation, Autonomous driving, U-Net, Style transfer

1 Introduction

Advancements in computer vision and artificial intelligence propel the development of autonomous driving technology (ADT). Semantic segmentation, in particular, is a key driver of improvements in ADT. Semantic segmentation requires deep models with robust datasets. For our application, data can be collected quickly and cheaply using driving simulators to produce synthetic data. Despite recent improvements in the usefulness of synthetic data, current computer vision models trained on synthetic data still perform poorly when applied to real world data. This is due to the covariate shift problem.

Given the extremely varied nature of driving scenery and the time intensive nature of human image segmentation, synthetic images likely must be used to train successful semantic segmentation networks for driving scenes. As such, the covariate shift problem must be dealt with. To attempt this, we built a U-Net model which makes use of modern data augmentation techniques specific to the semantic segmentation domain.

We applied two recent data augmentation techniques: Copy+Paste segmentation overlay and cross-domain stylization. The former involves copying and overlaying specific segmentations onto an image from a separate environmental context. In theory, this increases model robustness in diverse environments. The latter technique involves extracting environmental "style" features and transferring them to the another image. We also applied some common data augmentation transforms.

The remainder of the report is structured as follows: Section 2 provides information on the data set and describes some data processing steps. Section 3 describes the deep learning U-Net architecture used for the semantic segmentation alongside the data augmentations explored. This is followed by Section 4, which provides performance information for the data augmentation methods used with the model. The report discusses the results and overall conclusions in Section 5.

2 Data Processing

2.1 Data Description

The provided training data is split into two sets. The first set consists of synthetic RGB images collected with a wide range of weather and lighting conditions using the CARLA simulator [2]. The different weather and lighting conditions include the following: ClearNoon, HardRainNoon, CloudySunset, CloudyNoon, Default, MidRainSunset, and SoftRainNoon. The second set includes a small subset of data from the Cityscapes training data set, which is comprised of RGB images of various driving scenes in European cities [1]. Overall, this dataset included 5600 samples, or 700 for each weather condition and 700 additional samples for the Cityscape images. Throughout the rest of the paper, the entirety of this data is referred to as the CARLA-Cityscapes data set. The CARLA-Cityscapes data set contains 15 different segmentation classes (Figure 1).

Class	**Original Color**			**New Color**		
	R	**G**	**B**	**R**	**G**	**B**
Building	70	70	70	70	70	70
Fence	190	153	153	190	153	153
Pole	153	153	153	153	153	153
Sidewalk	244	35	232	244	35	232
Vegetation	107	142	35	107	142	35
Wall	102	102	156	102	102	156
Road / road line	128	64	128	128	64	128
	157	234	50			
Traffic light / sign	250	170	30	220	220	0
	220	220	0			
Person / rider	220	20	60	220	20	60
	255	0	0			
Car	0	0	142	0	0	142
Truck	0	0	70	0	0	70
Bus	0	60	100	0	60	100
Train	0	80	100	0	80	100
Motorcycle / Bicycle	0	0	230	119	11	32
	119	11	32			
Other	Anything else			0	0	0

Fig. 1. Class labels for hybrid CARLA-Cityscapes data set

2.2 Multiple RGB values for some classes

There are several segmentation classes to which multiple RGB combinations in the segmentation map correspond. For example, RGB values (128, 64, 128) and (157, 234, 50) both represent the 'Road / road line' class. To simplify our train-time processing, we pre-process the CARLA-Cityscapes data set, replacing all original colors with their corresponding new color (Figure 1).

2.3 Image Scaling

The CARLA-Cityscapes data set contains images of two different sizes: 2048x1024 and 1280x720. Our model requires images of the same size, so during training images of size 2048x1024 are randomly cropped to size 1820x1024 in order to match the aspect ratio of the smaller images. They are then scaled down to 1280x720. This method preserves the most amount of data from the larger images without distorting them.

3 Methods

3.1 Model

For image segmentation, we implemented a U-Net [8]. The U-Net takes an image and pushes it through multiple downscaling layers (the encoder), effectively lowering the image resolution to capture the context in the image. Then, the image goes through upscaling layers (the decoder), which increase the image resolution for precise localization. Dense skip connections are added so that small image details do not get lost. For simplicity, consider each series of layers as a block. There are two different block types: the downBlocks, for the encoder portion of the model, and the upBlocks, for the decoder portion. The downBlocks consist of convolutional layers with ReLU activation functions, followed by max pooling. The upBlocks consist of transposed convolutional layers and an upscaling portion, which includes a concatenation of the corresponding downBlock (skip layers), as denoted in the figure by the grey arrows (Figure 2).

Section 2.1 describes the class configuration of this data set. To account for the 15 different classes that exist within the data set, the model's final convolutional layer produces a tensor with dimensions (15, 1280, 720). We use this tensor to calculate loss via a multi-class pixel-level cross-entropy function.

3.2 Data Augmentation Techniques

We implemented three primary data augmentation techniques: common data augmentations, cross-domain style transfer, and random segmentation channel overlay.

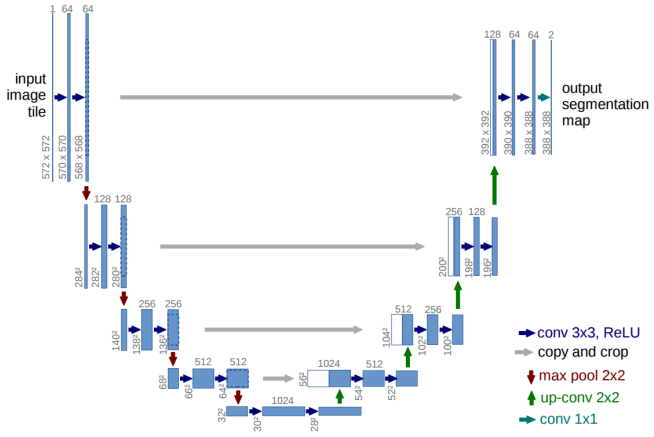

Fig. 2. U-Net architecture (example for 32x32 pixels in the lowest resolution). Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. The x-y-size is provided at the lower left edge of the box. White boxes represent copied feature maps. The arrows denote the different operations [8].

Unaugmented Image | Image w/ Jitter | Flipped Image w/ Jitter

Fig. 3. Types of traditional image augmentation techniques

Traditional Data Augmentation Techniques For a base level of data augmentation, we implemented random horizontal images flips and pixel color jitters (brightness, contrast, saturation, and hue) [7]. This provided increased variety of image colorings. This augmentation is done according to the following procedure:

Cross-Domain Stylization Following recent work on domain adaptation [3], we implement a style transfer network [4] on our data set to increase our quantity of training data and deal with the covariate shift problem related to our synthetic and real data.

We have two domains of data: real and synthetic images. We apply synthetic styles to the real images and real styles to the synthetic images according to the procedure defined in Figure 4. See an example of cross-domain stylization in Figure 5.

Random Segmentation Copy+Paste We also use a Copy+Paste technique [5] to add variety to our data samples. When an image is sampled, we assign a

Procedure for generating Cross-Domain styled images

Partition the data set into real images D^r and synthetic images D^s
For each image D_i^r in D^r:
  Obtain random image D_j^s from D^s
  Create new image via style transfer using D_i^r as the content image and D_j^s as the style image
For each image D_i^s in D^s:
  Obtain random image D_j^r from D^r
  Create new image via style transfer using D_i^s as the content image and D_j^r as the style image

Fig. 4.

Content Image Style Image Output

Fig. 5. Example of cross-domain styled images with synthetic content image and real style image.

50% probability to a Copy+Paste overlay augmentation occurring. When this happens, we select a random image from the data set, select a random segmentation layer to copy (e.g. cars or road), and copy this data from the random image to our newly augmented image. See the procedure in (Figure 6) and an example in (Figure 7).

4 Results

In this section, we discuss the results of training and testing the U-Net on the CARLA-Cityscapes data set with different combinations of the data augmentation techniques described in Section 3.2. We train the U-Net with no data augmentation (default case study), the U-Net with only traditional augmentation (traditional case study), and the combination of all three data augmentation techniques (all-transforms case study) described in Section 3.2

4.1 Training Discrepancies

To save time during training, we performed the style transform on the entire CARLA-Cityscapes data set and saved all of the new images. This doubled the size of our all-transforms data set to about 11K samples. The all-transforms

Procedure for generating Copy+Paste image overlays

Select sample S with segmentation map S_{seg} and image S_{img} from the data set.
Select sample R with segmentation map R_{seg} and image R_{img}, at random, from the data set.
Select index I at random, where I $\in \{x|x \geq 0 \wedge x < N\}$ and N is the number of classes in the data set.
Copy all segmentation values corresponding to class I from R_{seg} to S_{seg}, overwriting values as necessary.
Copy all image values corresponding to class I from R_{img} to S_{img}, overwriting values as necessary.

Fig. 6.

First Image Second Image New Image

Fig. 7. An example of an instance of Random Segmentation Copy+Paste where the car in the second image is copied into the first.

case study was trained on this modified data set which contains twice as many samples as the un-styled data set for the default and traditional case studies. For the cases that included either the traditional or overlay data augmentation, we applied the data augmentations during training.

We randomly selected 80% of the data within each case study for their training data sets. Note that the randomly selected samples within the case study for all transformations is different than the samples selected for the other studies. For all of the U-Nets trained with different combinations of data augmentation techniques, we used an initial learning rate of 0.001 with the Adam optimizer [6]. The default and traditional case studies trained for 100 epochs and the all-transforms case study trained for 165 epochs.

4.2 Testing Results

To best facilitate comparisons, the testing data sets for each case consist of a random 20% selection from the CARLA-Cityscapes data set. Despite the discrepancies between the different case studies, the results for each case study are comparable since the testing data sets include the same samples. For each case study, Table 1 displays the Intersection Over Union (IOU) percentages for each class within the data set, the mean IOU, and the weighted mean IOU. For each

of the classes, the model trained with no data augmentation techniques performed best. Furthermore, despite training for 45 additional epochs, the model trained with all data augmentation techniques barely outperformed the model trained only with the traditional data augmentation. With the current selection of data augmentation methods, the U-Net is able to best predict the more common classes, such as Road, Vegetation, and Other. All models fail to predict any pixels of trucks, buses, trains, or bicycles, possibly due to data imbalance.

	None	Traditional	All
Building	65.5%	59.4%	60.5%
Fence	49.9%	34.9%	43.2%
Pole	22.7%	17.3%	19.3%
Sidewalk	66.3%	50.2%	53.2%
Vegetation	80.0%	70.8%	75.2%
Wall	65.0%	38.1%	50.6%
Road	90.0%	82.4%	85.2%
Traffic light	16.9%	3.7%	11.9%
Person	26.7%	15.3%	2.7%
Car	51.3%	41.0%	47.0%
Truck	0%	0%	0%
Bus	0%	0%	0%
Train	0%	0%	0%
Bicycle	0%	0%	0%
Other	82.6%	78.7%	81.1%
mIOU	51.4%	44.7%	48.2%
wMIOU	78.9%	71.4%	74.1%

Table 1. Comparison between no augmentation, only traditional, and all data augmentation methods. The table shows the Intersection Over Union (IOU) value for each class, the mean IOU, and the weighted mean IOU.

These results certainly appear to be poor, as the data augmentations worsen the IOU metrics. However, there are several reasons for the poor result with data augmentations. For each case study, there is little distinction between the samples in the training and testing data sets since they are randomly selected. As such, it would make sense that the case studies that trained with fewer similarities between the training testing data performed worse. We see this within the results, where the two case studies that use data augmentations, and hence train on data that shares less similarities to the testing data, did not perform as well as the case study with no data augmentations. If these case studies were tested on samples outside of the CARLA-Cityscapes data set, the results may significantly differ such that the model trained with all data augmentations will perform best. The case study with all data augmentation methods trained on the largest variety of data, and thus has more potential to successfully generalize outside of the CARL-Cityscapes dataset.

5 Conclusions

In summary, we designed a U-Net deep learning architecture in order to perform semantic segmentation on driving scenery. We implemented several data augmentation techniques as described in Section 3.2, including traditional augmentations, overlay augmentations, and style augmentations. Our results in Section 4 highlight the performance of the U-Net when trained with different combinations of the data augmentations. We Predict how our model with data augmentation techniques will perform on different data sets. Despite our results indicating that the model without data augmentation performs best, for reasoning described earlier, we suggest the U-Net trained with all data augmentation methods may generalize better on data outside of the CARLA-Cityscapes domain.

Acknowledgments

We would like to thank General Motors and the organizers for sponsoring and creating the data challenge. All of our results were computed via small GPU computer clusters provided by Oak Ridge National Laboratory. For more information about our work, please check the github repository: https://github.com/trigaten/smc-data-challenge.

References

[1] Marius Cordts et al. *The Cityscapes Dataset for Semantic Urban Scene Understanding*. 2016. arXiv: 1604.01685 [cs.CV].

[2] Alexey Dosovitskiy et al. *CARLA: An Open Urban Driving Simulator*. 2017. arXiv: 1711.03938 [cs.LG].

[3] Aysegul Dundar et al. *Domain Stylization: A Strong, Simple Baseline for Synthetic to Real Image Domain Adaptation*. 2018. arXiv: 1807.09384 [cs.CV].

[4] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. *A Neural Algorithm of Artistic Style*. 2015. arXiv: 1508.06576 [cs.CV].

[5] Golnaz Ghiasi et al. *Simple Copy-Paste is a Strong Data Augmentation Method for Instance Segmentation*. 2021. arXiv: 2012.07177 [cs.CV].

[6] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2017. arXiv: 1412.6980 [cs.LG].

[7] Adam Paszke et al. "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *Advances in Neural Information Processing Systems 32*. Ed. by H. Wallach et al. Curran Associates, Inc., 2019, pp. 8024–8035. URL: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[8] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. *U-Net: Convolutional Networks for Biomedical Image Segmentation*. 2015. arXiv: 1505.04597 [cs.CV].