EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

Mathematisch-Naturwissenschaftliche Fakultät

Lernbasierte Computer Vision

Bachelorarbeit

A Comparison of Synthetic-to-Real Domain Adaptation Techniques

Eberhard Karls Universität Tübingen
Mathematisch-Naturwissenschaftliche Fakultät
Wilhelm-Schickard-Institut für Informatik
Lernbasierte Computer Vision
Peter Trost, peter.trost@student.uni-tuebingen.de, 2019

Bearbeitungszeitraum: 24.05.2019-23.09.2019

Betreuer/Gutachter: Prof. Dr. Andreas Geiger, Universität Tübingen

Selbstständigkeitserklärung

Hiermit versichere ich, dass ich die vorliegende Bachelorarbeit selbständig und nur mit den angegebenen Hilfsmitteln angefertigt habe und dass alle Stellen, die dem Wortlaut oder dem Sinne nach anderen Werken entnommen sind, durch Angaben von Quellen als Entlehnung kenntlich gemacht worden sind. Diese Bachelorarbeit wurde in gleicher oder ähnlicher Form in keinem anderen Studiengang als Prüfungsleistung vorgelegt.

Peter Trost (Matrikelnummer 4039682), August 1, 2019

Abstract

Template

Acknowledgments

If you have someone to Acknowledge ;)

Contents

1. Introduction

With autonomous driving being highly popular these days and the need for large amounts of labeled data to train the deep neural networks running the autonomous cars, methods have been developed to generate that data. Real datasets like Cityscapes [COR+16] or the Berkley Deep Drive [YXC+18] are expensive to assemble and even more expensive to label. Especially pixel-level annotations require a lot of working hours and can be inaccurate. In contrast synthetic approaches like the GTAV [RVRK16] or SYNTHIA datasets [RSM+16] enable to create vast amounts of image data automatically and make the annotation process much faster and more accurate and therefore cheaper than their real counterparts. The drawback of using synthetic data to train models that are ran in autonomous cars is that these models don't perform well in real environments. This can result in wrong predictions, which must be avoided in order to make autonomous driving possible. The drop in performance is due to the domain shift, i.e. changes in appearance, texture and others of objects in the synthetic images compared to objects in the real world seen through the car's cameras. An approach to create models to perform better in this scenario is domain adaptation. The idea behind that is to adapt a specific domain to another domain (e.g. synthetic to real, image taken at night to image taken at daytime, images of an object taken from one perspective to another perspective, ...). For autonomous driving this means making the synthetic image look more like an image of a real scene taken with a real camera. There are many techniques for adapting images from a synthetic to a realistic domain. Each having different approaches and using different methods. This work compares three current approaches on synthetic-to-real domain adaptation, namely CycleGAN [ZPIE17], CyCADA [HTP+17] and SG-GAN/Grad-GAN [LLJX18], all three of which are based on Generative Adversarial Networks [GPAM+14]. And tries to show their strengths and weaknesses when using them to translate images from the GTAV dataset [RVRK16] to images looking like the ones in Cityscapes [COR+16]. In order to evaluate the resulting images, a pre-trained deeplabv3 [CPSA17] model (code: [DLR]) is used to predict semantic segmentation maps and use the cityscapes evaluation code [CSR] to compute Intersection over Union (IoU) for these.

2. Foundations

This chapter will describe and explain the foundations necessary for this work. The term **Domain Adaptation** will be defined and described. Furthermore all relevant **Neural Network** architectures will be shown and explained. Specifically **Convolutional Neural Networks** and **Generative Adversarial Networks**.

2.1. Domain Adaptation

As described in [Csu17], Domain Adaptation is the task of transfering a machine learning model that is working well on a source data distribution to a related target data distribution. In this work we will focus on the adaptation from synthetic to real images. When talking about a synthetic image we are implying it was rendered from a virtual scene through a rendering engine like for example blender's "Cycles" [Cycc] or graphics engine like Unity [Uni]. We define real images as taken from a real-world scene through some kind of camera. See Figure 2.1 for examples. In general there are many more domains, for example an image of a painting in the style of a particular artist, images of an object from different viewpoints, and many more.

Figure 2.1.: Example images for the the two domains relevant for this work. A real image from the cityscapes dataset [COR+16] (left) and a synthetic image from the GTA5 dataset [RVRK16] (right)

2.2. Neural Networks

2.2.1. Convolutional Neural Networks

see [Wu17] (Introduction to CNNs) Convolutional Neural Networks (CNNs) are Deep Neural Networks mostly used with images as input, consisting of convolutional layers, that extract features from input data (e.g. edges, curves, circles), pooling layers, commonly using max-pooling (mapping a subregion of the input to its maxium value) or average pooling (mapping the subregion to the average of the values) and a fully connected network often used for classification. **TODO: add more description and example images**

Convolutional Layer. In this layer a so called kernel iterates over the input. The kernel is a matrix of fixed size depending on the method used. A 3x3 kernel is commonly used. These kernels are used to extract features like diagonal lines, horizontal lines, curves and more. For example to extract a diagonal line a kernel could look like this:

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Pooling Layer This layer reduces the size of the image by mapping an area to its maximum or average value in order to focus on the important features and save computing time.

Fully Connected Layer The last type of layers is a standard feed forward network that has one output node for each class the net has to be able to predict. It is used to classify the input of the net given the features exracted by the preceeding layers.

TODO: example architectures VGG16, AlexNet,..?

2.2.2. Generative Adversarial Networks

Generative Adversarial Networks (GANs) implement a two-player-game:
A Discriminator learns from a given data distribution what is “real”. The Generator generates data. The goal of the generator is to fool the discriminator into believing the generated data is “real” i.e. tries to create samples coming from the same distribution as the “real” data. The discriminator will label anything as “fake” that doesn’t resemble the learned “real” data distribution (2 classes classification, real or fake). This way GANs can learn to generate realistically looking images of faces, translate images of art from one style to another and improve semantic segmentation. The generative model generally uses *maximum likelihood estimation*. In [Goo17] it is described in the following way:

> The basic idea of maximum likelihood is to define a model that provides an estimate of probability distribituion, parametereized by parameters θ. We then refer to the **likelihood** as the probability that the model assigns to the training data: $\prod_{i=1}^{m} p_{\text{model}}(x^{(i)}; \theta)$

The parameter θ that maximizes the likelihood of the data is better found in log space [Goo17]

$$\theta^* = \arg\max_{\theta} \prod_{i=1}^{m} p_{\text{model}}(x^{(i)}; \theta) \tag{2.1}$$

$$= \arg\max_{\theta} \log \prod_{i=1}^{m} p_{\text{model}}(x^{(i)}; \theta) \tag{2.2}$$

$$= \arg\max_{\theta} \sum_{i=1}^{m} \log p_{\text{model}}(x^{(i)}; \theta) \tag{2.3}$$

as the maximum of the function is at the same θ value and we now have a sum which aswell eliminates the possibility of having underflow by multiplying multiple very small probabilities together. Formally GANs are a structured probabilistic model (more info: Chapter 16 of Goodfellow et al. 2016) containing latent variables z and observed variables x. The generator is defined by a function G that takes **z** as input and uses $\theta^{(G)}$ as parameters, the discriminator by a function D that takes **x** as input and uses $\theta^{(D)}$ as parameters. Both players have cost functions that are defined in terms of both players' parameters. The discriminator wishes to minimize $J^{(D)}(\theta^{(D)}, \theta^{(G)})$ and must do so by controlling only $\theta^{(D)}$. This is analogous for the generator: he tries to minimize $J^{(G)}(\theta^{(D)}, \theta^{(G)})$ while controlling only $\theta^{(G)}$. In contrast to an optimization problem that has a solution that is the (local) minimum (a point in parameter space where all neighboring points have greater or equal cost), the GAN objective is a game. The solution to a game is a Nash equilibrium (**TODO: reference john forbes nash jr.**), meaning that each player chooses the best possible option or strategy in respect to what the other player(s) choose. Here the Nash equilibrium is a tuple $(\theta^{(D)}, \theta^{(G)})$ that is a local minimum of $J^{(D)}$ with respect to $\theta^{(D)}$ and a local minimum $J^(G)$ with respect to $\theta^{(G)}$. **The generator** is simply a differentiable function G. When **z** is sampled from some simple prior distribution, G(z) yields a sample of **x** drawn from p_{model}. Typically a deep neural network is used to represent G.
The game plays out in two scenarios. The first is where the discriminator D is given random training examples **x** from the training set. The goal of the discriminator here is for $D(\mathbf{x})$ to be near 1. In the second scenario, inputs **z** to the generator are randomly sampled from the model's prior overt the latent variables. The discriminator then receives input $G(\mathbf{z})$, a fake sample by the generator. The discriminator strives to make $D(G(\mathbf{z}))$ approach 0 while the generator tries to make that approach 1. The games Nash equilibrium corresponds to $G(\mathbf{z})$ being drawn from the same distribution as the training data. Under the assumption that the inputs to the discriminator are half real and half fake, this corresponds to $D(\mathbf{x}) = \frac{1}{2}$ for all **x**.

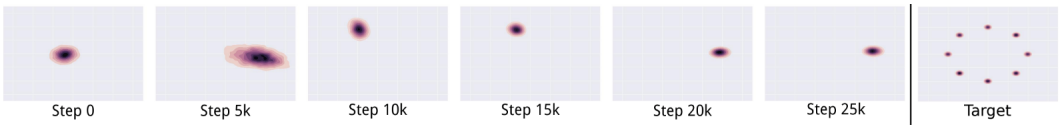

Figure 2.2.: target data distribution of a toy dataset (right) and the data distribution of generated samples. Mind that the generated sample distribution hops from one mode to the next as the discriminator learns to recognize each mode as fake. Image from [MPPS16]

Training On each step, two minibatches are sampled: a minibatch of $\mathbf{x}$ values from the dataset and one of $\mathbf{z}$ values drawn from the model's prior over latent variables. Then two gradient steps are made simultaneously (simultaneous SGD): one updating $\theta^{(D)}$ to reduce $J^{(D)}$ and one updating $\theta^{(G)}$ to reduce $J^{(G)}$.

Cost functions There are many different cost functions that can be used for GANs. See chapter 4 for the ones that the techniques in this work use.

Advantages and Disandvantages

non-convergence GANs implement a two player game and in contrast to optimization problems that find a (local) minimum for example by stochastic gradient decent (SGD) and therefore usually make reliable downhill progress, this game has the following disadvantage: Updating one player and making the best current move downhill on the loss curve for that player might mean going uphill for the counterfeit player. Because of this GANs often oscillate in practice.

mode collapse happens when the generator learns to map different inputs $\mathbf{z}$ to the same output. Partial mode collapse refers to scenarios in which the generator makes multiple images containing the same color or texture themes, or multiple images containing different views of the same dog. See Figure 2.2 for an example.

3. Related Work

There are many approaches of adapting Images from one domain to a different one. This chapter will show some of those techniques and compare them to the ones focused on in this work.

3.1. section

4. Domain Adaptation Techniques

This chapter will give an overview over the three **Domain Adaptation Techniques** that will be compared in this work, namely CycleGAN [ZPIE17], CyCADA [HTP+17] and SG-GAN [LLJX18]. It will explain how these techniques work, what kind of Network Architecture they use and how training these Nets was approached.

All of the methods compared in this work are based on Generative Adversarial Networks which were initially proposed in [GPAM+14] and are explained in Section 2. CycleGAN [ZPIE17] adds a cycle-consistency loss which consists of an additional generator/discriminator pair with the idea that translating a previously translated image back to the source domain should yield the original image again. Mathematically speaking let G be the function of the Generator translating an image from the source to the target domain. Now add function F for the second generator's translation from target back to source domain. The goal of cycle-consistency is that for any image x from the source domain, $F(G(x)) \approx x$ holds. This analogously also has to hold for any image y of the target domain with $G(F(y)) \approx y$. CyCADA [HTP+17] adds a semantic loss to the CycleGAN approach to enforce semantic consistency (no more translating cats to dogs or drawing trees into the sky). This is achieved by doing semantic segmentation on the source image, then translating that image and doing semantic segmentation on the target image. The loss describes the difference between the two resulting label maps. SG-GAN [LLJX18] expands this by including a gradient-sensitive objective that emphasizes semantic boundaries by regularizing the generator to render distinct color/texture for each semantic region, and a patch-level discriminator that better understands differences in appearance distributions of different semantic regions (e.g. coarse texture of asphalt vs smooth and reflective of a vehicle). Prior to [ZPIE17] some works used paired images of the source and the target domain to train the networks. For example edgedrawings of shoes with a picture of the same shoe (see [IZZE16]). The cycle-consistency loss of CycleGAN makes it possible use datasets without paired images. This makes data acquisition easier and reduces costs which results in more training data and therefore makes better models possible. As CyCADA and SG-GAN are both based on CycleGAN they aswell don't require paired images.

4.1. Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks

see [ZPIE17]

- image-to-image translation: extracting characteristics of an image and translating it to another style while preserving the characteristics (rgb to greyscale, painting to photo,..)
- special in this approach: no paired images necessary (datasets with paired images are far more expensive)
- create mapping $G : X \rightarrow Y$ from source domain X to target domain Y
- The Generator has to trick the discriminator into believing $G(x), x \in X$ is actually a real sample y from the target domain Y (matches distribution $p_{\text{data}}(y)$)
- problem of mode collapse: any inputimage will be translated to the same output image
- add cycle-consistency constraint: create mapping $F : Y \rightarrow X$ and add contraint $F(G(x)) \overset{!}{\approx} x$
- objective for mapping/generator G and discriminator D_Y:
  $\mathcal{L}_{\text{GAN}}(G, D_Y, X, Y) = \mathbb{E}_{y \sim p_{\text{data}}(y)}[\log D_Y(y)] + \mathbb{E}_{x \sim p_{\text{data}}(x)}[1 - \log D_Y(G(x))]$
- analogous for mapping/generator F and discriminator D_X
- generators try to minimize the objective, discriminators try to maximize it
- cycle consistency loss:
  $\mathcal{L}_{\text{cyc}}(G, F) = \mathbb{E}_{x \sim p_{\text{data}}(x)}[\|F(G(x)) - x\|_1] + \mathbb{E}_{y \sim p_{\text{data}}(y)}[\|G(F(y)) - y\|_1]$
- full objective:
  $\mathcal{L}(G, F, D_X, D_Y) = \mathcal{L}_{\text{GAN}}(G, D_Y, X, Y) + \mathcal{L}_{\text{GAN}}(F, D_X, Y, X) + \lambda \mathcal{L}_{\text{cyc}}(G, F)$
- solve: $G^*, F^* = \arg \min_{G,F} \max_{D_X, D_Y} \mathcal{L}(G, F, D_X, D_Y)$

4.1.1. Training Details

- for $\mathcal{L}_{\text{GAN}}$ replaced negative log-likelihood objective by least-squares loss → more stable during training and generates higher quality results
- for GAN loss $\mathcal{L}_{\text{GAN}}(G, D, X, Y)$ they train
  G to minimize $\mathbb{E}_{x \sim p_{\text{data}}(x)}[(D(G(x)) - 1)^2]$
  and D to minimize $\mathbb{E}_{y \sim p_{\text{data}}(y)}[(D(y) - 1)^2] + \mathbb{E}_{x \sim p_{\text{data}}(x)}[D(G(x))^2]$

- reduce model oscillation **TODO: add Shrivasta et al.**s method update discriminator using history f generated images instead of the latest ones generated. Use a buffer of 50 images
- set $\lambda = 10$ in **TODO: Equation 3**, Adam solver, batchsize 1, trained from scratch with learning rate 0.0002 for the first 100 epochs then decay linearly to 0 in the following 100.

4.2. CyCADA: Cycle Consistent Adversarial Domain Adaptation

see [HTP+17]

4.2.1. Introduction

- synthetic datasets cheaper and more accurate in classification than real ones
- per-pixel label accuracy drops from 93%(real) to 54%(synthetic)
- while translating from synth to real semantic information might be lost (e.g translating line-drawing of a cat to a picture of a dog)
- CyCADA uses cycle consistency and semantic losses
- apply model to digit recognition and semantic segmentation of urban scenes across domains.
- improves per-pixel accuracy from 54% to 82% on synth-to-real. (**TODO: compared to what?**)
- shows that domain adaptation benefits greatily from cycle-consistent pixel transformations
- adaptation at both pixel and representation level can offer complementary improvements with joint pixel-space and feature adaptation leading to the highest performing model for digit classification tasks

4.2.2. Related Work

- **TODO: cite everything**
- visual domain adaptation introduced along with a pairwise metric transform solution by Seanko et al. 2010
- further popularized by broad study of visual dataset bias (Torralba & Efros, 2011)
- early deep adaptive works focused on feature space alignment through minimizing distance between first or second order feature space statistics of source and target (Tzeng et al., 2014; Long & Wand, 2015)
- further improved thorugh use of domain adversarial objectives whereby a domain classifier is trained to distinguish between source and target representations while domain representation is learned so as to maximize error of domain classifier

- representation optimized by using standard minimax objective (Ganin & Lempitsky, 2015)
- symmetric confusion objective (Tzeng et al., 2015)
- inverted label objective (Tzeng et al., 2017)
- each related to GAN (Goodfellow et al., 2014) and followup trianing procedures for these networks (Salimans et al., 2016b; Arjovsky et al., 2017)
- these feature-space adaptation methods focus on modifications to the discriminative representation space. Other recent methods hav sought adaptation in the pixel-space using various generative approaches
- one advantage of pixel-space adaptation: result may be more human interpretable, since an image from on domain can now be visualized in a new domain
- CoGANs (Liu & Tuzel, 2016b) jointly learn source and target representation through explicit weight sharing of certain layers, source and target have unique gen adv objective
- Ghifary et al. 2016 use an additional reconstruction objective in target domain to envourage alignment in the unsupervised adaptation setting
- another approach: directly convert target image into a source style image (or vise versa), largely based on GANs (cite Goodfellow..)
- successfully applied GANs to various applications such as image generation (Denton et al., 2015; Radford et al., 2015; Zhao et al., 2016), image editing (Zhu et al., 2016) and feature learning (Salimans et al., 2016a; Donahue et al., 2017). Recent work (Isola et al., 2016; Sangkloy et al., 2016; Karacan et al., 2016) adopt conditional GANs (Mirza & Osindero, 2014) for these image-to-image translation problems (Isola et al., 2016), but require input-output image pairs for training, which is in general not available in domain adaptation problems
- no training pairs: Yoo et al. 2016 learns source to target encoder-decoder along with a generative adversarial objective on reconstruction which is applied for predicting clothing people are wearing
- Domain Transfer Network (Taigman et al. 2017b) trains generator to transform a source image into a target image by enforccing consistency in embedding space
- Shrivastava et al. 2017 instead use L1 reconstruction loss to force generated target images to be similar to original source images. works well for limited domain shifts where domains are similar in pixel-space, but can be too limiting for setting with larger domain shifts

- Bousmalis et al. 2017b use a content similarity loss to ensure the generated target image is similar to original source image; however this requires prior knowledge about which parts of the image stay the same across domains (e.g. foreground)
- cycada method does not require pre-defining what content is shared between domains and instead simply translates images back to their original domains while ensuring that the remain identical to their original version
- BiGAN (Donahue et al., 2017) and ALI (Dumoulin et al., 2016) take an approach of simultaneously learning the transofmrations between pixel and latent space.
- CycleGAN (Zhu et al., 2017) produced compelling image translation results such as generating photorealistic images from impressionism paintings or transfomring horses into zebras at high resolution using cycle-consistency loss
- this loss was simultaneously proposed by Yi et al. 2017 and Kim et al. 2017 to great effect as well
- adaptation across weather conditions in simple road scenes was first studied by Levinkov & and Fritz 2013
- convolutional domain adversarial based approach was proposed for more general drive cam scenes and for adaptation from simulated to real environments (Hoffmann et al., 2016)
- Ros et al. 2016b learns a multi-source model through concatenating all available labeled data and learning a single large model and then transfers to a sparsely labeled target domain through distillation (Hinton et al., 2015)
- Chen et al. 2017 use an adversarial objective to align both global and class-specific statistics, while mining additional temporal data from street view datasets to learn static object prior
- Zhang et al. 2017 instead perform segmentation adaptation by aligning label distributions both globally and across superpixels in an image

4.2.3. Cycle-consistent adversarial domain adaptation

- consider problem of unsupervised adaptation
- provided source data X_S, source labels Y_S, and target data X_T, but no target labels
- goal: learn a model f that can correctly predict label for target data X_T
- begin by learning source model f_S that can perform the task on the source data

- for K-way classification with cross-entropy loss:

$$\mathcal{L}_{\text{task}}(f_S, X_S, Y_S) = -\mathbb{E}_{(x_s,y_s)\sim(X_S,Y_S)} \sum_{k=1}^{K} \mathbb{1}_{[k=y_s]} \log\Big(\sigma(f_S^{(k)}(x_s))\Big) \tag{4.1}$$

  where σ denotes softmax function

- learned model f_S will perform well on source data but typically domain shift between source and target domain leads to reduced performance when evaluating on target data
- by mapping samples into common space, the model can learn on source data while still generalizing to target data
- mapping from source to target $G_{S\to T}$ trained to produce target samples that fool adversarial discriminator D_T
- discriminator attempts to classify real target data from source target data. loss function:

$$\mathcal{L}_{\text{GAN}}(G_{S\to T}, D_T, X_T, X_S) = \mathbb{E}_{x_t\sim X_T}\Big[\log D_T(x_t)\Big] \tag{4.2}$$

$$+ \mathbb{E}_{x_s\sim X_S}\Big[\log(1 - D_T(G_{S\to T}(x_s)))\Big] \tag{4.3}$$

- this objective ensures that $G_{S\to T}$, given source samples, produces convincing target samples
- this ability to directly map samples between domains allows to learn target model f_T by minimizing $\mathcal{L}_{\text{task}}(f_T, G_{S\to T}(X_S), Y_S)$
- previous approaches that optimized similar objectives have shown effetive results but in practice can often be unstable and prone to failure
- although GAN loss ensures $G_{S\to T}(x_s)$ for some x_s will resemble data drawn from X_T but there is no way to guarantee $G_{S\to T}(x_s)$ preserves structure or content of original sample x_s
- to encourage source content to be preserved during conversion: cycle-consistency constraint (**TODO: cite the 3 works that proposed it**). Mapping $G_{T\to S}$ trained according to GAN loss $\mathcal{L}_{\text{GAN}}(G_{T\to S}, D_S, X_S, X_T)$
- want $G_{T\to S}(G_{S\to T}(x_s)) \approx x_s$ and $G_{S\to T}(G_{T\to S}(x_t)) \approx x_t$
- done by imposing L1 penalty on reconstruction error (reffered to as cycle-consistency loss):

$$\mathcal{L}_{\text{cyc}}(G_{S\to T}, G_{T\to S}, X_S, X_T) = \mathbb{E}_{x_s\sim X_S}\Big[\|G_{T\to S}(G_{S\to T}(x_s)) - x_s\|_1\Big] \tag{4.4}$$

$$+ \mathbb{E}_{x_t\sim X_T}\Big[\|G_{S\to T}(G_{T\to S}(x_t)) - x_t\|_1\Big] \tag{4.5}$$

- also explicitly encourage high semantic consistency before and after image translation
- pretrain source task model f_S, fixing weights and using it as a noisy labeler by which an image to be classified in the same way after translation as it was before translation according to this classifier is encouraged
- define fixed classifier f, predicted label for given input X: $p(f, X) = \arg\max(f(X))$
- semantic consistency before and after image translation:

$$\mathcal{L}_{\text{sem}}(G_{S\to T}, G_{T\to S}, X_S, X_T, f_S) = \mathcal{L}_{\text{task}}(f_S, G_{T\to S}(X_T), p(f_S, X_T)) \tag{4.6}$$
$$+ \mathcal{L}_{\text{task}}(f_S, G_{S\to T}(X_S), p(f_S, X_S)) \tag{4.7}$$

- can be viewed analogously to content loss in style transfer (Gatys et al., 2016) or in pixel adaptation (Taigman et al., 2017a), where shared content to preserve is dictated by the source task model f_S
- could also consider a feature-level method which discriminates between the features or semantics from two image sets as viewed under a task network → additional feature level GAN loss:

$$\mathcal{L}_{\text{GAN}}(f_T, D_{\text{feat}}, f_S(G_{S\to T}(X_S)), X_T) \tag{4.8}$$

- together form complete objective:

$$\mathcal{L}_{\text{CyCADA}}(f_T, X_S, X_T, Y_S, G_{S\to T}, G_{T\to S}, D_S, D_T) \tag{4.9}$$
$$= \mathcal{L}_{\text{task}}(f_T, G_{S\to T}(X_S), Y_S) \tag{4.10}$$
$$+ \mathcal{L}_{\text{GAN}}(G_{S\to T}, D_T, X_T, X_S) + \mathcal{L}_{\text{GAN}}(G_{T\to S}, D_S, X_S, X_T) \tag{4.11}$$
$$+ \mathcal{L}_{\text{GAN}}(f_T, D_{\text{feat}}, f_S(G_{S\to T}(X_S)), X_T) \tag{4.12}$$
$$+ \mathcal{L}_{\text{cyc}}(G_{S\to T}, G_{T\to S}, X_S, X_T) + \mathcal{L}_{\text{sem}}(G_{S\to T}, G_{T\to S}, X_S, X_T, f_S) \tag{4.13}$$

- ultimately corresponds to solving for a target model f_T according to the optimization problem

$$f_T^* = \arg\min_{f_T} \min_{\substack{G_{T\to S}\\ G_{S\to T}}} \max_{D_S, D_T} \mathcal{L}_{\text{CyCADA}}(f_T, X_S, X_T, Y_S, G_{S\to T}, G_{T\to S}, D_S, D_T) \tag{4.14}$$

4.3. Semantic-aware Grad-GAN for Virtual-to-Real Urban Scene Adaption

see [LLJX18]

4.3.1. Introduction

- two main contributions:
  1. gradient-sensitive objective, emphasizes the semantic boundary consistencies between virtual images and adapted images. able to regularize the generator render distinct color/texture for each semantic region in order to keep semantic boundaries, which can alleviate the common blurry issues
  2. previous work often learn a whole image discriminator, makes pixels in image easily collapse into a monotonous pattern. appearance distributions for each semantic region should be regarded differently and purposely (e.g. road: coarse texture of asphalt concrete, vehicle: smooth and reflective)
- semantic-aware discriminator learns distinct discriminate parameters for examining regions with respect to each semantic label
- SG-GAN controllable architecture that personalizes texture rendering for different semantic regions and results in adapted images with finer details

4.3.2. Related Work

Real-world vs. virtual-world data acquiring

- -

Domain adaptation

- either by adapting scene images or adapting hidden feature representations guided by the targets
- Image-based adaption (aka image-to-image translation) can be summarized into two directions:
  1. generated through feature matching (e.g. **TODO: cite Gatys et al. 10**: combine content of one with style of other images through matching Gram matrix on deep feature maps, at expense of loss of some content information)

2. GAN (e.g. **TODO: Isola et al. 21** conditional GANs with mapping function for paired data, for unpaired data e.g. regularization term **TODO: cite following** 37, cycle structure 24 45 47, weight sharing 29 30)

- some make use of both feature matching and adversarial training (5, 44)
- challenge: existing approaches often modify semantic information, e.g. sky adapted to tree structure or road lamp rendered from nothing
- hidden feature based adaption aims at adapting learned models to target domain (**TODO: multiple citations**), by sharing weight (12) or incorporating adversarial discriminative setting (39), those mitigate performance degradation caused by domain shifting
- feature based adaption methods require different objectives or architecture for different vision tasks, thus not as widely applicable as image-based adaption

4.3.3. Semantic-aware Grad-GAN

Semantic-aware cycle objective

- based on cycle-structured GAN objective
- unpaired images from virtual-world domain V and real-world domain R as $\{v\}_{i=1}^{N} \in V$ and $\{r\}_{i=1}^{M} \in R$
- SG-GAN learns symmetric mappings $G_{V \to R}$ and $G_{R \Rightarrow V}$ alogn with corresponding semantic-aware discriminators SD_R, SD_V
- SD_R distinguishes between real-world images $\{r\}$ and fake real-world images $\{G_{V \to R}(v)\}$ and vice versa for SD_V

Adversarial loss

- two sets of adversial losses applied to $(G_{V \Rightarrow R}, SD_R)$ and $(G_{R \Rightarrow V}, SD_V)$ pairs
- adversarial loss:

$$\mathcal{L}_{\text{adv}}(G_{V \to R}, SD_R, V, R) = \mathbb{E}_{r \sim p_{\text{data}}(r)}[\log SD_R(r)] \tag{4.15}$$

$$+ \mathbb{E}_{v \sim p_{\text{data}}(v)}[\log(1 - SD_R(G_{V \to R}(v)))] \tag{4.16}$$

- objective

$$G^*_{V \to R} = \arg \min_{G_{V \to R}} \max_{SD_R} \mathcal{L}_{\text{adv}}(G_{V \to R}, SD_R, V, R) \tag{4.17}$$

- analogous for other generator discriminator: $\mathcal{L}_{\text{adv}}(G_{R \to V}, SD_V, R, V)$

Cycle consistency loss

$$\mathcal{L}_{\text{cyc}}(G_{V\to R}, G_{R\to V}, V, R) = \mathbb{E}_{r\sim p_{\text{data}}(r)}[\|G_{V\to R}(G_{R\to V}(r)) - r\|_1] \tag{4.18}$$
$$+ \mathbb{E}_{v\sim p_{\text{data}}(v)}[\|G_{R\to V}(G_{V\to R}(v)) - v\|_1] \tag{4.19}$$

- can be seen as introduction of a regularization on positions of image elements
- moving positions of image components is not encouraged
- model only with cycle-consistency can still map sky to tree (not semantic aware)

Soft gradient-sensitive objective

- motivation of gradient-sensitive loss: no matter how texture of semantic classes change, there should be some distinguishable visual difference at boundaries of semantic classes
- visual differences for adjacent pixels can be captured through convolving gradient filters upon the image
- typical choice of gradient filter is Sobel filter (**TODO: cite 38**) as $\mathbf{C} = \{C_x, C_y\}$:
  $$C_x = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}, C_y = \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix}$$
- since focus is visual difference on semantic boundaries, a 0-1 mask is necessary that only has non-zero values on semantic boundaries
- such mask can be retrieved by convolving a gradient filter upon semantic labeling since it only has different adjacent values on semantic boundaries
- semantic labeling obtained by human annotation, segmentation models, computer graphics tools (**TODO: cite**)
- multiply convolved semantic labeling and convolved image element-wise, to only pay attention to visual differences on semantic boundaries
- for input image v and corresponding semantic labeling s_v since we desire v and $G_{V\to R}(v)$ share the same semantic information, greadient-sensitive loss for image v can be defined as follows with C_i and C_s gradient filters for image and semantic labeling, $*$ convolution, $\odot$ element-wise multiplication, $|\cdot|$ absolute value, $\|\cdot\|_1$ L1-norm, sgn sign function:

$$l_{\text{grad}}(v, s_v, (G_{V\to R})) = \|(|(|C_i * v| - |C_i * G_{V\to R}(v)|)|) \tag{4.20}$$
$$\odot (\alpha \times |sgn(C_s * s_v)| + \beta)\|_1 \tag{4.21}$$
$$s.t. \quad \alpha + \beta = 1 \quad \alpha, \beta \geq 0 \tag{4.22}$$

- in practice we may hold belief that v and $G_{V\to R}(v)$ share smiliar texture within semantic classes
- texture can be extracted from image gradient, therefore a soft gradient-sensitive loss for v can be defined as following to represent such belief, in which β controls how much belief we have on texture similarities

$$l_{s-\text{grad}}(v, s_v, G_{V\to R}, \alpha, \beta) = \|(|(|C_i * v| - |C_i * G_{V\to R}(v)|)|) \tag{4.23}$$
$$\odot(\alpha \times |sgn(C_s * s_v)| + \beta)\|_1 \tag{4.24}$$
$$s.t. \quad \alpha + \beta = 1 \quad \alpha, \beta \geq 0 \tag{4.25}$$

- S_V semantic labeling for V, S_R for R. final objective for soft gradient-sensitive loss for single image:

$$\mathcal{L}_{\text{grad}}(G_{V\to R}, G_{R\to V}, V, R, S_V, S_R, \alpha, \beta) = \mathbb{E}_{r\sim p_{\text{data}}(r)}[l_{s-\text{grad}}(r, s_r, G_{R\to V}, \alpha, \beta)] \tag{4.26}$$
$$+ \mathbb{E}_{v\sim p_{\text{data}}(v)}[l_{s-\text{grad}}(v, s_v, G_{V\to R}, \alpha, \beta)] \tag{4.27}$$

- full objective with λ_c, λ_g controlling relative importance of cycle consistency and soft gradient-sensitive loss compared to adversarial loss:

$$\mathcal{L}(G_{V\to R}, G_{R\to V}, SD_V, SD_R) = \mathcal{L}_{\text{adv}}(G_{V\to R}, SD_R, V, R) \tag{4.28}$$
$$+ \mathcal{L}_{\text{adv}}(G_{R\to V}, SD_V, R, V) \tag{4.29}$$
$$+ \lambda_c \mathcal{L}_{\text{cyc}}(G_{V\to R}, G_{R\to V}, V, R) \tag{4.30}$$
$$+ \lambda_g \mathcal{L}_{\text{grad}}(G_{V\to R}, G_{R\to V}, V, R, S_V, S_R, \alpha, \beta) \tag{4.31}$$

- optimization target:

$$G^*_{V\to R}, G^*_{R\to V} = \arg\min_{\substack{G_{V\to R} \\ G_{R\to V}}} \max_{\substack{SD_R \\ SD_V}} \mathcal{L}(G_{V\to R}, G_{R\to V}, SD_V, SD_R) \tag{4.32}$$

4.3.4. Semantic-aware discriminator

- introduction of soft gradient-sensitive loss contributes to smoother textures and clearer semantic boundaries
- scene adaption also needs to retain higher-level semantic consistencies (e.g. tone goes dark cause real world less ilumination, but may only want roads to be darker without changing the sky or even making it lighter)
- inappropriate holistic scene adaption because of traditional discriminator judging realism image-wise, regardless of texture difference in semantic-aware manner

- semantic-aware discriminators SD_V, SD_R
- idea: create separate pipeline for each different semantic class in the discriminator
- can be achieved by transiting number of filters in last layer of standard discriminator to number of semantic classes, then applying semantic masks upon filter to let each of them focus on different semantic classes
- last (k-th) layer's feature map of standard discriminator is typically a tensor $\mathbf{T}_k$ with shape $(w_k, h_k, 1)$, where w_k stands for width and h_k stands for height
- T_k then compared with an all-one or all-zero tensor to calculate adversarial objective
- in contrast semantic-aware discriminator will change $\mathbf{T}_k$ as a tensor with shape (w_k, h_k, s) where s is number of semantic classes
- then convert image's semantic labeling to one-hot style and resize to (w_k, h_k) which will result in mask M with shape (w_k, h_k, s) and $\mathbf{M}_{ij} \in \{01\}$
- by multiplying $\mathbf{T}_k$ and $\mathbf{M}$ element-wise, each filter within $\mathbf{T}_k$ will only focus on one particular semantic class
- Finally, by summing up $\mathbf{T}_k$ along the last dimension, tensor with shape $(w_k, h_k, 1)$ will be acquired and adversarial objective can be calculated the same way as standard discriminator

5. Experiments

In this chapter the three Domain Adaptation Techniques *Cycle Consistent Adversarial Domain Adaptation*, *Cycle-consistent Generative Adversarial Network* and **TODO: add 3rd technique** will be compared by analysing similarities and differences. Furthermore pretrained models of each architecture were tested and the results will be compared on their Intersection over Union scores **TODO: add benchmark(s)**.

5.1. Datasets

5.1.1. Synthetic dataset: Playing for Data: Ground Truth from Computer Games

see [RVRK16]

The GTA5 (Grand Theft Auto V) dataset is proposed in [RVRK16]. It contains 24966 images taken from a street view in the game Grand Theft Auto V by Rockstar Games [GTA]. The images are provided with 1914×1052 pixels and are containing moving cars, objects, pedestrians, bikes, have changing lighting and weather conditions aswell as day and night scenes. For all of these images the authors provide label images that are compatible with those of the Cityscapes dataset [COR+16]. Detouring, i.e. injecting a wrapper between the game and the graphics hardware to log function calls and reconstruct the 3D scene is used. This enables a faster labeling process as objects in a scene can be assigned an object ID through which the assigned label is propagated to other images containing this same object. Due to being more realistic than other existing synthetic street view datasets (e.g. SYNTHIA [RSM+16]) the GTAV dataset is very popular for training machine learning models related to autonomous driving and is therefore used in this work.

5.1.2. Real dataset: The Cityscapes Dataset for Semantic Urban Scene Understanding

see [COR+16]

The Cityscapes dataset is a large scale dataset containing dashcam view images from 50 european cities. It includes 30 classes relevant for autonomous driving. The images include scenes in spring, summer and fall seasons and under different weather conditions. There are 5000 images provided together with fine annotations

and 20000 together with coarse annotations. Due to the large amount of labeled data from a dashvam view and the inclusion of scenes with different weather and lighting conditions this dataset is often used to train deep neural networks that are related to autonomous driving. Due to the popularity and the GTAV dataset containing compatible labeling maps, this work uses Cityscapes as the real dataset for the experiments.

5.2. comparison Benchmark(s)

5.2.1. Intersection over Union (IoU)

The Intersection over Union is a metric often used to compare semantic segmentation methods (**TODO: add references to works that use it**). It follows following formula:

$$\frac{\text{predicted Pixels} \cap \text{ground truth Pixels}}{\text{predicted Pixels} \cup \text{ground truth Pixels}}$$

where predicted Pixels are the pixels predicted for a specific class by the semantic segmentation model and ground truth Pixels are the Pixels containing the ground truth for that image. Usually there are multiple different classes to predict and therefore it is common to calculate the mean IoU (mIoU) over all images that have predictions. To compare how well classes themselves are predicted by a model, one can also calculate the class IoU (cIoU).

5.2.2. Methodology

For the comparison each technique was used to translate a sample of 500 images from the GTA dataset to the Cityscapes domain. For CycleGAN and SG-GAN each, the authors provided pre-trained models. For CyCADA pretranslated images are provided in the project repository [CyCa]. Due to CyCADA only providing 22k translated images instead of the full 25k images provided in the GTA dataset, from the randomly chosen 500 images only 450 are available here. This has to be regarded in the following comparison. To translate images with SG-GAN and CycleGAN the provided code [SG],[Cycb] and for the semantic segmentation task an implementation [DLR] of deeplabv3 [CPSA17] was used. To compute the IoU values the deeplabv3 implementation uses the benchmark code provided by Cityscapes [CSR]. All computations were run on Ubuntu 18.04 using CPU only in VirtualBox on a Windows Machine due problems with dependencies while running the code on the tcml cluster of the university of tuebingen [tcm], a cluster for machine learning applications.

6. Conclusion

To conclude...

A. Blub

Bibliography

[COR+16] 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 *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[CPSA17] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *CoRR*, abs/1706.05587, 2017.

[CSR] Cityscapes repository. `https://github.com/mcordts/cityscapesScripts`. Accessed: 2019-26-07.

[Csu17] Gabriela Csurka. Domain adaptation for visual applications: A comprehensive survey. *CoRR*, abs/1702.05374, 2017.

[CyCa] CyCADA repository. `https://github.com/jhoffman/cycada_release`. Accessed: 2019-01-08.

[Cycb] CycleGAN lua repository. `https://github.com/junyanz/CycleGAN`. Accessed: 2019-01-08.

[Cycc] Cycles rendering engine. `https://www.cycles-renderer.org/`. Accessed: 2019-10-07.

[DLR] Deeplabv3 repository. `https://github.com/fregu856/deeplabv3`. Accessed: 2019-26-07.

[Goo17] Ian J. Goodfellow. NIPS 2016 tutorial: Generative adversarial networks. *CoRR*, abs/1701.00160, 2017.

[GPAM+14] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014.

[GTA] Grand Theft Auto V website. `https://www.rockstargames.com/V/`. Accessed: 2019-26-07.

[HTP+17] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei A. Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. *CoRR*, abs/1711.03213, 2017.

[IZZE16] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. *CoRR*, abs/1611.07004, 2016.

[LLJX18] Peilun Li, Xiaodan Liang, Daoyuan Jia, and Eric P. Xing. Semantic-aware grad-gan for virtual-to-real urban scene adaption. *CoRR*, abs/1801.01726, 2018.

[MPPS16] Luke Metz, Ben Poole, David Pfau, and Jascha Sohl-Dickstein. Unrolled generative adversarial networks. *CoRR*, abs/1611.02163, 2016.

[RSM+16] German Ros, Laura Sellart, Joanna Materzynska, David Vazquez, and Antonio Lopez. The SYNTHIA Dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *CVPR*, 2016.

[RVRK16] Stephan R. Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: Ground truth from computer games. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *European Conference on Computer Vision (ECCV)*, volume 9906 of *LNCS*, pages 102–118. Springer International Publishing, 2016.

[SG] SG-GAN repository. `https://github.com/Peilun-Li/SG-GAN`. Accessed: 2019-01-08.

[tcm] tcml cluster universität tübingen. `https://uni-tuebingen.de/fakultaeten/mathematisch-naturwissenschaftliche-fakultaet/fachbereiche/informatik/lehrstuehle/kognitive-systeme/projects/tcml-cluster/`. Accessed: 2019-01-08.

[Uni] Unity graphics engine. `https://unity.com/`. Accessed: 2019-10-07.

[Wu17] Jianxin Wu. Introduction to convolutional neural networks. *National Key Lab for Novel Software Technology. Nanjing University. China*, pages 5–23, 2017.

[YXC+18] Fisher Yu, Wenqi Xian, Yingying Chen, Fangchen Liu, Mike Liao, Vashisht Madhavan, and Trevor Darrell. BDD100K: A diverse driving video database with scalable annotation tooling. *CoRR*, abs/1805.04687, 2018.

[ZPIE17] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *CoRR*, abs/1703.10593, 2017.