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), July 17, 2019

Abstract

Template

Acknowledgments

If you have someone to Acknowledge ;)

Contents

1. Introduction

What is this all about?

Cite like this: [GPAM$^+$14]

1.1. Problem Statement

There are many techniques for adapting images from a synthetic to a realistic domain. Each having different approaches and using different methods. This workcompares three current approaches on synthetic-to-real domain adaptation and tries to show their strengths and weaknesses.

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" [Cyc] 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.

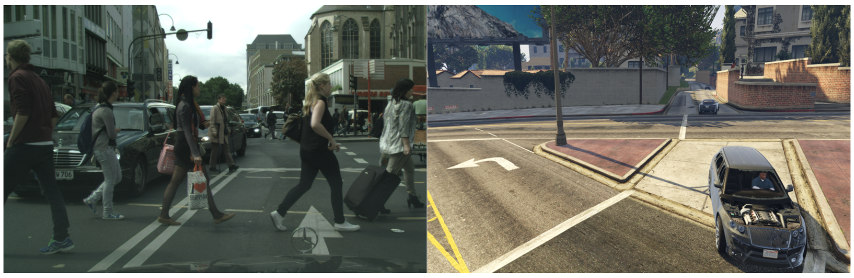

Figure 2.1.: Example images for the the two domains relevant for this work. A real image from the cityscapes dataset [COR+16] (left) and a synthetic image from the GTA5 dataset [RVRK16] (right)

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 predicts has number of possible classes as output nodes. It is used to classify the input 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**.

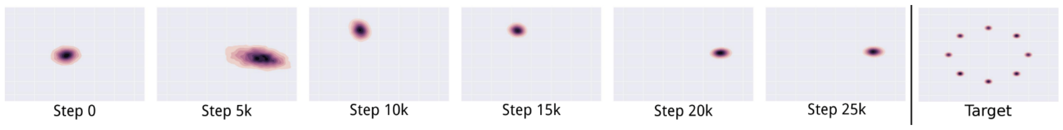

Figure 2.2.: target data distribution of a toy dataset (right) and the data distribution of generated samples. Mind that the generated sample distribution hops from one mode to the next as the discriminator learns to recognize each mode as fake. Image from [MPPS16]

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

This chapter is just a collection of notes on papers on **Domain Adaptation**.

3.1. Domain Adaptation for Structured Output via Discriminative Patch Representation

see [TSSC19]

3.1.1. Abstract

- labeling data is expensive
- therefore propose domain adaptation method to adapt labeled source data to unlabeled target domain (e.g. GTA5 (playing for data) to city-scapes)
- learn discriminative feature representations of patches based on label histograms in the source domain, through construction of clustered space
- then use adversarial learning sscheme to push feature representations in target patches to the closer distributions in source ones
- can integrate a global alignment process with this patch-level alignment and achieve state-of-the-art performance on semantic segmentation
- extensive ablation studies on numerous benchmark datasets with various settings (e.g. synth-to-real, cross-city)

3.1.2. Introduction

- pixel-level annotation of ground truth expensive. e.g. road-scene iamges of different cities may have various appearance distributions, differences over time and weather
- existing state-of-the-art methods use feature-level or output space adaptation, exploiut global distribution alignment, such as spatial layout, but these might differ significantly between two domains due to differences in camera poses or field of view

- authours instead match patches that are more likely to be shared across domains regardless of where they are located
- consider label histograms as a factor (Kulkarni et al., 2015; Odena et al., 2017) and learn discriminative representations for patches to relax high-variation problem among them
- use this to better align patches between source and target domains
- utilize two adversarial modules to align global/patch-level distributions
- global one based on output space adaptation (Tsai et al. 2018)
- take source domain labels and extract label histogram as a patch-level representation
- then apply K-means clustering to group extracted patch representations into K clusters **TODO: read this part again for better understanding (page 2)**

3.1.3. domain adaptation for structured output

- given source and target images $I_s, I_t \in \mathbb{R}^{H \times W \times 3}$ and source labels Y_s, the goal is to align predicted output distribution O_t of target data with source distribution O_s
- use loss function for supervised learning on source data to predict the structured output, adversarial loss is adopted to align the global distribution
- further incorporate classification loss in a clustered space to learn patch-level discriminative representations F_s from source output distribution O_s. For target data another adversarial loss is used to align patch-level distributions between F_s and F_t, where the goal is to push F_t to be closer to distributon of F_s.
- objective function :

  $$\mathcal{L}_{\text{total}}(I_s, I_t, Y_s, \Gamma(Y_s)) = \mathcal{L}_s + \lambda_d \mathcal{L}_d + \lambda^g_{\text{adv}} \mathcal{L}^g_{\text{adv}} + \lambda^l_{\text{adv}} \mathcal{L}^l_{\text{adv}} \tag{3.1}$$

  where $\mathcal{L}_s$ and $\mathcal{L}_d$ are supervised loss function for learning structured prediction and discriminative representation on source data. Γ denotes clustering process on ground truth label distribution. $\mathcal{L}^g_{\text{adv}}, \mathcal{L}^l_{\text{adv}}$ denote global and patch-level adversarial loss. λ's are weights for the different loss function
- $\mathcal{L}_s$ can be optimized by fully-convolutional network $\mathbf{G}$ that predicts the structured output with the loss summed over the spatial map indexed with h, w and number of categories C:

  $$\mathcal{L}_s(I_s, Y_s; \mathbf{G}) = -\sum_{h,w} \sum_{c \in C} Y_s^{(h,w,c)} \log(O_s^{(h,w,c)}) \tag{3.2}$$

where $O_s = \mathbf{G}(I_s) \in (0,1)$ is the predicted output distribution through softmax function and is up-sampled to the size of the input image.

- with discriminator $\mathbf{D}_g$:

$$\mathcal{L}^g_{\text{adv}}(I_s, I_t; \mathbf{G}, \mathbf{D}_g) = \sum_{h,w} \mathbb{E}[\log \mathbf{D}_g(O_s)^{(h,w,1)}] + \mathbb{E}[\log(1 - \mathbf{D}_g(O_t)^{(h,w,1)})] \tag{3.3}$$

- optimize following min-max problem with inputs dropped for simplicity:

$$\min_{\mathbf{G}} \max_{\mathbf{D}_g} \mathcal{L}_s(\mathbf{G}) + \lambda^g_{\text{adv}} \mathcal{L}^g_{\text{adv}}(\mathbf{G}, \mathbf{D}_g) \tag{3.4}$$

- label histograms for patches: first randomly sample patches from source images, using a 2-by-2 grid on patches to extract spatial label histograms, and concatenate them into a vector, each histogram is a $2 \cdot 2 \cdot C$ dimensional vector. Second apply K-means clustering on these histograms, whereby the label for any patch can be assigned as the cluster center with the closest distance on the histogram
- add classification module $\mathbf{H}$ after the predicted output O_s, to simulate the procedure of constructin the label histogram and learn a discriminative representation
  learned representation: $F_s = \mathbf{H}(\mathbf{G}(I_s)) \in (0,1)^{U \times V \times K}$ (softmax function, K is number of clusters)
- learning process to construct clustered space formulated as cross-entropy loss:

$$\mathcal{L}_d(I_s, \Gamma(Y_s); \mathbf{G}, \mathbf{H}) = -\sum_{u,v} \sum_{k \in K} \Gamma(Y_s)^{(u,v,k)} \log(F_s^{(u,v,k)}) \tag{3.5}$$

- goal is now to align patches regardless of where they are located in the image (without spatial and neighborhood support)
- reshape F by concatenating the K-dimensional vectors along the spatial map, results in $U \cdot V$ independent data points
- this reshaped data is denoted as $\hat{F}$, adversarial objective:

$$\mathcal{L}^l_{\text{adv}}(I_s, I_t; \mathbf{G}, \mathbf{H}, \mathbf{D}_l) = \sum_{u,v} \mathbb{E}[\log \mathbf{D}_l(\hat{F}_s)^{(u,v,1)}] + \mathbb{E}[\log(1 - \mathbf{D}_l(\hat{F}_t)^{(u,v,1)})] \tag{3.6}$$

  where $\mathbf{D}_l$ is the discriminator to classify whether the feature representation $\hat{F}$ is from source or target domain

- integrate (3.5) and (3.6) into min-max problem in 3.4:

$$\min_{\mathbf{G},\mathbf{H}} \max_{\mathbf{D}_g, \mathbf{D}_l} \mathcal{L}_s(\mathbf{G}) + \lambda_d \mathcal{L}_d(\mathbf{G}, \mathbf{H}) + \lambda^g_{\text{adv}}, \mathcal{L}^g_{\text{adv}}(\mathbf{G}, \mathbf{D}_g) + \lambda^l_{\text{adv}} \mathcal{L}^l_{\text{adv}}(\mathbf{G}, \mathbf{H}, \mathbf{D}_l) \tag{3.7}$$

3.2. Effective Use of Synthetic Data for Urban Scene Semantic Segmentation

see [SAS+18]

- foreground and background classes are not affected the same way by domain shifts
- foreground classes should be treated in a detection based manner as their shape looks natural even though their texture in synthetic images is not photo-realistic
- drawback of deep neural networks is need for massive amount of training data
- training on synthetic only makes models perform bad in the real world
- domain adaptation methods improve this but still require large sets of real images
- model cannot be trained off-line on synthetic data and work well when deployed into a new, real-world environment
- observation: not all classes suffer from the same type and degree of perceptual differences
- background texture looks more realistic than foreground, nevertheless foreground object shape look natural
- therefore should be treated differently
- use semantic segmentation on background classes because of texture realism
- use object detectors for foreground classes
- main discrimination between fore and background objects is shape
- trained seperately a DeepLab and Mask R-CNN **TODO: cite** for object detection, followed by binary segmentation and class prediction, on synthetic data
- compare mIoU on foreground classes of Cityscapes
- outperforms semantic segmentation model on every class except *motorcycle*
- from this observation: model that combines foreground masks produced by Mask R-CNN with pixel-wise predictions of DeepLab semantic segmentation network
- outperforms state-of-the-art domain adaptation techniques and can be further improved by making use of unsupervised real images
- **VEIS** (Virtual Environment for Instance Segmentation) proposed aswell, based on Unity3D

- automatically annotates synthetic images with instance-level segmentation for foreground classes
- when used with the proposed detector-based approach this data allows to boost semantic segmentation performance

3.2.1. related work

- domain adaptation generally aims to reduce the gap between the feature distributions of the two domains (synthetic, real)

3.2.2. Method

- use VGG16-based DeepLab model for background classes (with large field of view and dilated convolution layers)
- train on GTA5 dataset (background classes look photo-realistic)
- cross-entropy loss between network's predictions and ground-truth pixel-wise annotations of the sythetic images
- trained on fore- and background classes but foreground predictions are mostly discarded by the proposed approach
- use standard Mask R-CNN (object detection, binary mask extraction together with object classification) **TODO: look at citation**
- fuse fore- and background predictions
- sort predicted segments according to confidence score, if current segment candidate overlaps with previous segment, pixels are removed in the overlapping region
- this yields a semantic segmentation map that only contains foreground classes and has large number of holes where no foreground object was found. Every pixel that is not already assigned to foreground class takes the label with highest probability at that pixel location in the DeepLab result
- remember: NO real data used during training of this method
- to extend approach for unsupervised training on real images: tread predictions of their method as pseudo ground truth for the real images. assign pixels that ware predicted as foreground classes by DeepLab model to an *ignore* label, so that they are not used for training.

3.2.3. Implementation Details

- DeepLab: SGD, learning rate starting at 25×10^{-5} with decrease factor of 10 every 40k iterations, momentum of 0.9, weight decy of 0.0005, mini-batches of size 1. weights initialized with those of VGG-16 classifier, pre-trained on ImageNet, due to limited GPU memory high resolution images were downsampled by a factor of 2 for training
- Mask R-CNN: implementation provided by Detectron framework **TODO: cite**, train an end-to-end Mask R-CNN model with 64× 4d ResNeXt-101-FPN backbone, pre-trained on ImageNet, on the synthetic VEIS dataset. mini-batch of size 1, train model for 200k iterations, starting learning rate of 0.001, reducing to 0.0001 after 100k iterations

3.3. Exemplar Guided Unsupervised Image-to-Image Translation with Semantic Consistency

see [MJG+18]

- EGSC-IT network conditions the translation process on an exemplar image in the target domain
- assumption: image comprises of a content component shared across domains, and style component specific to each domain
- under guidance of exemplar from target domain apply Adaptive Instance Normalization to the shared content component, which allows to transfer the style information of the target domain to the source domain
- introduce feature masks that provide coarse semantic guidance without requiring the use of any semantic labels to avoid semantic inconsistencies during translation
- image-to-image (I2I) translation: task of mapping an image from a source domain to a target domain (e.g. semantic maps to real images, grey-scale to color, low-resolution to high-res)
- other approaches assume one-to-one mapping (e.g. cycleGAN) and fail to capture multimodal nature of image distribution within the target domain (many-to-many mapping, e.g. diffferent color and style of shoes in sketch-to-image translation, different seasons in syn2real street view translation)
- assume an image is composed of two disentangled representations: domain-shared representation that models the content in the image, second domain-specific representation that contains the style information

- unclear which style (time-of-day/season) to pick during image translation process, Isola et al. 2017 and Zhu et al. 2017b **TODO: cite** show that using random noise for this can lead to mode collapse issues
- instead the image translation process is conditioned on arbitrary image in the target domain (i.e. an exemplar)
- like this EGSC-IT is enabled for multimodal (i.e. many-to-many) image translations, but also allows for explicit control over translation process depending on the exemplars used as guidance
- use weight sharing architecture proposed in UNIT (Liu et al. 2017 **TODO: cite**), but instead of single latent space shared by both domains, a two component one is used (domain-shared component that contains semantic information (objects' category, shape, spacial layout), domain-specific style component that contains style information (texture, color)
- adaptive instance normalization (AdaIN) (Huang & Belongie 2017 **TODO: cite**) is applied to the shared content component of the source domain image using AdaIN parameters computed from the target domaoin exemplar
- directly applying AdaIN to the feature maps of the shared content component would mix up all objects and scenes in the image, making the image translation prone to failure when a n image contains diverse objects and scenes, therefore semantic labels as an additional form of supervision is used **TODO: cite works that do this**.
- instead of using data with labor-intensive annotations use computed feature masks
- feature masks: approximately decoupling different semantic categories in an unsupervised way under guidance of perceptual loss and adversarial loss

3.3.1. Method

- weight sharing: to map image pairs (one from source domain, other target) to shared latent space, the last layer in the encoders (E_A, E_B) of the VAE-GAN and the first layer in the generators (G_A, G_B) share their weights. (see **TODO: cite UNIT, Liu et al. 2017**)
- **TODO: exemplar-based AdaIN for domain-specific style**
- *Network architecture*: 1) two encoders E_A, E_B, each consisting of serveral strided convolutional layers and serveral residual blocks to compute the shared content component, 2) feature mask network and an AdaIN network, F_A, F_B for $A \rightarrow B$ translation ($B \rightarrow A$ vise versa)have same architecture as the Encoder above **TODO: above?** except for weight-sharing layers. 3) Two Generators, G_A, G_B, almost symmetric to the Encoders except that the up-sampling is

done by transposed convolutional layers. 4) Two Discriminators, D_A, D_B fully-convolutional networks containing a stack of convolutional layers. 5) A VGG sub-network (Simonyan & Zisserman, 2015 **TODO: cite**), VGG, that contains the first few layers (up to relu5_1) of pre-trained VGG-19 (**TODO: cite Simonyan & Zisserman, 2015**), which is used to calculate perceptual losses.

- Although UNIT is used as baseline framework, it can be replaced with any baseline framework with similar functionality

3.3.2. Learning

- learning procedure of EGSC-IT contains VAEs, GANs, cycle-consistecy and perceptual losses.
- for more stable training, the feature mask network and AdaIN network gets pre-trained for each domain seperately within a VAE-GAN architecture, and use encoder part as fixed feature extractors (F_A, F_B) for the remaining training
- overall loss:

$$\mathcal{L}(E_A, G_A, D_A, E_B, G_B, D_B) = \mathcal{L}_{\mathrm{VAE_A}}(E_A, G_A) + \mathcal{L}_{\mathrm{GAN_A}}(E_A, G_A, D_A) + \mathcal{L}_{\mathrm{CC_A}}(E_A, G_A, E_B, G_B) \tag{3.8}$$

$$+ \mathcal{L}_{\mathrm{VAE_B}}(E_B, G_B) + \mathcal{L}_{\mathrm{GAN_B}}(E_B, G_B, D_B) + \mathcal{L}_{\mathrm{CC_B}}(E_A, G_A, E_B, G_B) \tag{3.9}$$

$$+ \mathcal{L}_{\mathrm{P}}(E_A, G_A, E_B, G_B) \tag{3.10}$$

- VAEs, GANs and cycle-consistency losses identical to the ones used in **TODO: cite Liu et al., 2017**
- **TODO: add rest**

3.3.3. Experiments

- translated between GTA5 and Berkeley Deep Drive (BDD) (**TODO: cite both**)
- Similar to FCN-score used by **TODO: cite isola et al 2017** use the semantic segmentation performance to quantitatively evaluate the image translation quality
- first translate images from GTA5 dataset to arbitrary image in BDD
- only generate images of size 256×512 due to GPU memory limitations
- then train single-scale Deeplab model (**TODO: cite Chen et al. 2018**) on the translated images and test it on BDD test set
- get mIoU scores

3.3.4. Discussion

since method doesn't use any semantic segmentation labels nor paired data, there are some artifacts in the results for some hard cases.
e.g. night → day translation more challenging than day → day, therefore sometimes hard for the model to understand semantics in such cases.
In the future it would be interesting to extend the method to semi-supervised setting to benefit from presence of some fully-labeled data

3.4. Exploiting Semantics in Adversarial Training for Image-Level Domain Adaptation

see [RTdS18]

- ResNet generator, U-Net discriminator
- 300k training iterations
- Adam optimizer, 0.0001 learning rate, batch size 2
- images cropped to 512x512
- images from GTA+annotations thereof, only images of cityscapes

3.5. FCNs in the Wild: Pixel-level Adversarial and Constraint-based Adaptation

see [HWYD16]

- semantic segmentation is critical visual recognition task for a variety of applications ranging from autonomous agent tasks, such as robotic navigation and self-driving cars, to mapping and categorizing the natural world
- challenges when adapting between visual domains for classification: changes in appearance, lighting, pose

3.5.1. work includes

- first unsupervised domain adaptation method for transferring semantic segmentation FCNs across image domains
- combination of global and local alignment methods, using global and category specific adaptation techniques

- align global statistics of source and target data using a convolutional domain adversarial training technique, using a novel extension of previous image-level classification approaches **TODO: look up citations**
- given a domain aligned representation space, introduce generalizable constrained multiple instance loss function, which expands on weak label learning, but can be applied to the target domain without any extra annotations and explicitly transfers category layout information from a labeled source dataset
- use GTA5 and SYNTHIA + CityScapes datasets for synth2real
- cross season adaptation within SYNTHIA
- adaptation across real world cities
- perform detailed quantitative analysis of cross-city adaptation within CityScapes
- contribute BDDS (Berkeley Deep Driving Segmentation), unconstrained drive-cam dataset for semantic segmentation
- show Cityscapes to BDDS
- show that adaptation algorithm improves target semantic segmentation performance without any target annotations

3.5.2. Related Work

- many recent approaches use fully convolutional networks (FCNs) for semantic segmentation, mapping input RGB space to semantic pixel space
- compelling because they allow direct end-to-end function that can be trained using back propagation
- original FCN formulation has been improved using dilated convolution (**TODO: look at citation**) and post-processing techniques, such as Markov/-conditional random fields
- high cost of collecting pixel level supervision motivates weak labels (typically image-level tags defining presence / absence of each class)
- Multiple instance learning (MIL) **TODO: see citations**, reinforce confident predictions udring learning procecss
- improves method suggested by **TODO: see citations** who use an EM algorithm to better model global properties of the images segments. Generalized by Pathak et al. who proposed a Constrained CNN, also able to model any linear constraints on the label space (i.e. presence / absence, percent cover)
- Hong et al. used auxiliary segmentation to generalize semantic segmentations to categories where only weak label information was available

- these methods all assume weak labels present during training time for both source and target domain
- authors consider strong supervision available in source domain, no supervision available in target domain

Domain Adaptation

- domain adaptation in computer vision focuses largely on image classification, with much work dedicated to generalizing across domain shift between stock photographs of objects and the same objects photographed in the world
- recent work include **TODO: some citations** which all learn a feature representation which encourages maximal confusion between the two domains
- other work **TODO: cite** aims to align features by minimizing the distance between their distribtuions in the two domains
- based on GAN Liu et al. proposed coupled GAN to learn a joint distribution of images from both source and target datasets
- other computer vision task detection: Hoffmann et al., domain adaptation system: explicitly modeling the representation shift between classification and detection models, follow-up work which incorporated per-category adaptation using multiple instance learning. later converted into FCNs for evaluating semantic segmentation performance

3.5.3. Fully Convolutional Adaptation Models

- source domain S, images I_S, labels L_S, trained source only model for semantic segmentation which produces pixel-wise per-category score map $\phi_S(I_S)$
- goal to learn semantic segmentation model which is adapted for use on unlabeled target domain T, images I_T, no annotations, parameters of such a network $\phi_T(\cdot)$
- if no domain shift: just apply source model directly to target domain
- however: commonly a difference between distribution of source labeled domain and target test domain
- therefore: *unsupervised adaptation* approach
- first main shift: global changes may occur between two domains resulting in marginal distribution shift of corresponding feature space
- can occur in any domain, but most distinct in large domain shifts between very distinct domains (e.g. simulated and real domains)

- second main shift: due to category specific parameter changes. may result from individual categories having specific biases in the two domains (e.g. adapting between different cities: changes in appearance of signs and distribution of cars)
- propose unsupervised domain adaptation framework for adapting semantic segmentation models which directly tackles both the need for minimizing the global and the category specific shifts
- assumptions: source and target domain share same label space, source model achieves performance greater than chance on target domain
- introduce two new semantic segmentation loss objectives, one to minimize the global distribution distance, which operates over both source and target images, $\mathcal{L}_{\text{da}}(I_S, I_T)$, another to adapt category specific parameters using target images and transferring label statistics from source domain $P_{L_S}, \mathcal{L}_{\text{mi}}(I_T, P_{L_S})$
- to ensoure to not diverge too far from source solution, which is known to be effective for final semantic segmentation task, continue to optimize the standard supervised segnmentation objective on source domain $\mathcal{L}_{\text{seg}}(I_S, L_S)$
- goal: optimize joint objective:

$$\mathcal{L}(I_S, L_S, I_T) = \mathcal{L}_{\text{seg}}(I_S, L_S) \tag{3.11}$$
$$+ \mathcal{L}_{\text{da}}(I_S, I_T) + \mathcal{L}_{\text{mi}}(I_T, P_{L_S}) \tag{3.12}$$

- **TODO: add illustration Figure 2**
- source domain data to update standard supervised loss objective, trained using source pixel-wise annotations
- both source and target data are used without any category annotations within fully-convolutional domain adversarial training to minimize global distance of feature space between the two domains
- category specific updates using a constrained pixel-wise multiple instance learning objective is performed on the target images, with source category statistics used to determine the constraints
- use front-end dilated FCN **TODO: cite 33**, based on VGG16 **TODO: cite 31** as base model
- 16 conv layers, last three conv layer converted from fully connected layers, called fc_6, fc_7, fc_8, followed by 8 times bilinear up-sample layer to produce segmentation in the same resolution as input images

3.5.4. Global Domain Alignment

- recognition sought at pixel level, alignment of full image representations will marginalize out too much distribution information, limiting the alignment capability of the adversarial learning approach
- instead consider region corresponding to natural receptive field of each spatial unit in final representation layer (e.g. $fc7$), as individual instances
- in doing so, the adversarial training procedure is supplied with the same information which is used to do final pixel prediction
- provides a more meaningful view of overall source and target pixel-space representation distribution distance which needs to be minimized
- let $\phi_{l-1}(\theta, I)$ denote output of last layer before pixel prediction according to network parameters θ
- $\mathcal{L}_{\text{da}}(I_S, I_T)$ consists of alternating minimization objectives
- one concerning parameters θ of representation space, under which source and target distance $\min d(\phi_{l-1}(\theta, I_S), \phi_{l-1}(\theta, I_T))$ will be minimized for given distance function $d(\cdot)$.
- second: estimating distance function through training domain classifier to distinguish instances of source and target domains
- let θ_D domain classifier parameters
- learn domain classifier to recognize difference between source and target regions and use that classifier to guide distance iminimization of source and target representations
- let $\sigma(\cdot)$ denote softmax function, domain classifier predictions $p_{\theta_D}(x) = (\sigma(\phi(\theta_D, x)))$
- assuming output of layer $l-1$ has $H \times W$ spatial units, we can define domain classifier loss $\mathcal{L}_D$ as:

$$\mathcal{L}_D = -\sum_{I_S \in S}\sum_{h \in H}\sum_{w \in W} \log(p_{\theta_D}(R^S_{hw})) \tag{3.13}$$

$$-\sum_{I_T \in T}\sum_{h \in H}\sum_{w \in W} \log(1 - p_{\theta_D}(R^T_{hw})) \tag{3.14}$$

  where $R^S_{hw} = \phi_{l-1}(\theta, I_S)_{hw}$ and $R^T_{hw} = \phi_{l-1}(\theta, I_T)_{hw}$ denote source and target representation of each units, respectively

- inverse domain loss for convenience:

$$\mathcal{L}_{\text{Dinv}} = -\sum_{I_S \in S} \sum_{h \in H} \sum_{w \in W} \log(1 - p_{\theta_D}(R^S_{hw})) \tag{3.15}$$

$$- \sum_{I_T \in T} \sum_{h \in H} \sum_{w \in W} \log(p_{\theta_D}(R^T_{hw})) \tag{3.16}$$

$$\tag{3.17}$$

- alternating minimization procedure

$$\begin{array}{ll} \min\limits_{\theta_D} & \mathcal{L}_D \\ \min\limits_{\theta} & \frac{1}{2}[\mathcal{L}_D + \mathcal{L}_{\text{Dinv}}] \end{array}$$

- optimizing these two objectives iteratively amounts to learning the best possible domain classifier for relevant image regions and then using the loss of that domain classifier to inform the training of the image representations so as to minimize the distance between source and target domains

3.5.5. Category Specific Adaptation

- **TODO: cite 25 26?**
- compute by per image labeling statistics in source domain P_{L_S}. for each source image which contains class c, compute percentage of image pixels which have a ground truth label corresponding to this class. Can then compute histogram over these percentages and denote the lower 10% boundary as α_c, average values as δ_c and upper 10% as γ_c
- use this to inform target domain size constraints, explicitly transferring scene layout information from source to target domain (e.g. street usually takes much more space in a autonomous driving image than signs).
- constrained MIL for the case where image-level labels are known: for a given target image for which a certain class c is present, impose following constraints on output prediction map $p = \arg\max \phi(\theta, I_T)$

$$\delta_c \le \sum_{h,w} p_{hw}(c) \le \gamma_c \tag{3.18}$$

- encourages to not assign more pixels to class c than expected range observed in source domain
- optimize this objective with lower bound slack to allow for outlier cases wehre c simply occupies less of the image than is average in source domain. Do not allow slack on upper bound constraint as it is important that no single class occupies too much of any given image

- general constraint: can be applied to all classes
- optimize as in Pathak et al **TODO: cite 25**
- modification: to prevent model diverging and over-fitting: use size constraint that if the lower 10% of source class distribution α_c is greater than 0.1, the down-weight the gradients due to these classes by factor of 0.1. Can be viewed as re-sampling of the classes so as to come closer to a balanced set, allowing the relatively small classes potential to inform the learning objective
- need image-level labels for this approach, thus complete approach: predicting image-level labels, then optimizing for pixel predictions that satisfy the source transferred class size constraints. (e.g. source semantic segmentation yields: x pixels street, y pixels car, z pixels signs. target image-level labels are street and signs, make sure that semantic segmentation has at max x street and z sign classified pixels)
- **TODO: reread last paragraph before experiments**
- given target image I_T, compute current output class prediction map $p = \arg\max \phi(\theta, I_T)$
- for each class comput percentage of pixels assigned to it in current prediction $d_c = \frac{1}{H \cdot W} \sum_{h \in H} \sum_{w \in W} (p_{hw} = c)$
- assign image-level label to class c if $d_c > 0.1 \cdot \alpha_c$ (i.e. if currently labeling at least as many pixels as 10% of the expected number for a true class appearing in the image)

3.5.6. Experiments

domain adaptation tasks:

- cities2cities
- season2season
- synth2real
- use front-end dilated FCN (**TODO: cite 33**) as both the initialization for our method and as baseline model for comparison
- all code and models trained and evaluated in the Caffe (**TODO: cite 16**) framework, available before camera-ready
- use IoU
- for c2c and syn2r follow evaluation protocol of **TODO: cite 3** and train models with 19 semantic labels of Cityscapes
- for s2s use 13 semantic labels of SYNTHIA instead

3.5.7. Datasets

Cityscapes

- 34 categories
- high resolution 2048×1024
- three parts: 2975 training samples, 500 validation samples, 1525 test samples
- split into european cities, different geographic and population distributions

SYNTHIA

- 13 classes with different scenarios and sub-conditions
- for season to season use SYNTHIA-VIDEO-SEQUENCES, different cities, seasons, weathers, illumination, captured by 8 RGB cameras 360° visual field, use only dash-cam for this paper
- for synth2real use SYNTHIA-RAND-CITYSCAPES, 9000 random images from all sequences with Cityscapes-compatible annotations, as source domain data

GTA5

- 24966 high quality labeled frames from realistic open-world computer game Grand Theft Auto V (GTA5)
- frames with 1914×1052 generated from fictional city Los Santos based on Los Angeles, CA.
- use whole dataset with labels compatible to Cityscapes categories for synth2real adaptation

BDDS

- thousands of densely annotated dashcam video frames, hundreds of thousands of unlabeled frames
- 1280×720, 34 categories compatible to Cityscapes label space
- majority of data from New York and San Francisco (representative for eastern and western coasts)
- different from other existing driving dataset covers more challenging conditions (urban streetview at night, highway scene in rain,..)

3.5.8. Quantitative and Qualitative Results

- first large distribution shift (synth2real)
- then medium shift (season2season)
- small shift (city2city in Cityscapes)

TODO: add import table contents

Large Shift: Synthetic to Real Adaptation

- for GTA to Cityscapes: compared to performance of source dilation model the domain adversarial training contributes 4.4% raw and $\sim$ 20% relative percentage mIoU improvement and multiple instance loss another 1.6% raw and $\sim$ 6% relative
- for SYNTHIA to Cityscapes also measurable improvement
- raw 0.9% for adversarial, raw 1.5% with additional multiple instance loss

Medium Shift: Cross Seasons Adaptation

- SYNTHIA season2season
- on average $\sim$ 3% mIoU improvement. Higher mIoU for 12/13 categories
- no improvement on class *car* (probably because cars have little to no change in appearance through seasons in SYNTHIA)
- largest performance improvements through this method on categories like *road* in shift from fall to winter

Small Shift: Cross City Adaptation

- raw 3.6% mIoU improvement through domain adversarial training, only 0.1% through multiple instance loss
- only noticeable improvement from category specific alignment on *traffic light, rider, train*, probably because domain shift between train and val sets mainly results from change in global appearance due to difference in city, specific category appearance may not change significantly though

3.5.9. BDDS Adaptation

TODO: find quantitative results

3.6. From Virtual to Real World Visual Perception using Domain Adaptation - The DPM Example

see [LXG+16]

3.6.1. Need for Virtual Worlds

- in general best performing machine learning algorithms are *supervised* i.e requiring annotated information (ground truth) for training
- human annotators (e.g. Amazon Mechanical Turk, LabelMe **TODO: cite 48 and 53**) used for semantic segmentation but can't for pixel-wise optical flow and depth
- many deep CNNs developed today rely on an ImageNet pre-trained deep CNN which is modified or fine-tuned to solve a new task or operate in a new domain
- key for success of research community are powerful GPU hardware and large dataset with ground truth

3.6.2. Need for Domain Adaptation

- problem of domain adaptation is not just a virtual-to-real issue but rather a sensor-to-sensor or environment-to-environment one **TODO: cite 70, 71**
- **TODO: deformable part-based model DPM 17**
- **TODO: HOG+LPB/Linear-SVM paradigm 68, Haar+EOH/AdaBoost 69**

3.6.3. Domain Adaptation for DPM in a Nutshell

- **TODO: revisit this later if necessary**

3.6.4. Experiments

3.6.5. Experiments - Implementation

Dataset

- randomyl sample 2k images from GTA5 dataset and Cityscapes training set as training images for V and R.
- another 500 each for visual comparison and validation
- Cityscapes validation set will later be used to evaluate semantic segmentation scores
- trained on this, the model's name is **SG-GAN-2K**
- another training set on full GTA5 dataset and 2k Cityscapes images called **SG-GAN-25K**

Network architecture

- use 256×512 images for training due to GPU memory limitation
- for generator adapt Isola et al. 21 (U-Net structure with skip connections between low and high level layers)
- for semantic-aware discriminator use variant of PatchGAN (21,47), which is a FCN consists of multiple layers (leaky-ReLU, instance norm (41), convolution) and helps discriminator identify realism patch-wise

Training details

- to stabilize training use history of refined images (37) for training SD_V and SD_R
- apply least square objective instead of log likelihood for adversarial loss, which is shown helpful in stabilizing training and generating higher quality images (Mao et al. 31)
- parameters in 3.32-3.35 set $\lambda_c = 10$, $\lambda_g = 5$, (α, β) as $(0, 1)$ for first three epochs, then $(0.9, 1)$
- for gradient filters in 3.27-3.29 use Sobel Filter for $\mathbf{C}_i$ and filters

$$C_x = \begin{pmatrix} 0 & 0 & 0 \\ -1 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}, \; C_y = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & -1 & 0 \end{pmatrix} \tag{3.19}$$

  for $\mathbf{C}_s$ to avoid artifacts on image borders caused by reflect padding

- for number of semantic classes in discriminator cluster 30 classes (7) into 8 categories to avoid sparce classes i.e. $s = 8$
- learning rate 0.0002, batch size 1
- implemented based on TensorFlow framework (1) and train with single Nvidia GTX1080

Testing

- semantic information only needed at training time
- at test time SG-GAN only requires images without semantic information
- since generators and discriminators are FCN, can handle high res (1024×2048) at test time
- test time 1.3 seconds/image with the GPU mentioned earlier

3.6.6. Comparison with state-of-the-art methods

3.6.7. Comparison - Baselines

SimGAN

- (37)
- introduces self-regularization for GAN and local adversarial loss to train refiner for image adaption
- in experiments use channel-wise mean values as self-regularization term
- use architecture as proposed in 37

CycleGAN

- (47) learns mapping functions through adversarial loss and cycle consistency loss
- uses ResNet (14) architecture for generators and PatchGAN (21) for discriminators

DualGAN

- (45)
- uses U-Net structure for gen that are identical with SG-GAN

- PatchGAN as CycleGAN but follows loss format and training procedure of Wasserstein GAN (2)

BiGAN

- (8,9) learns inverse mapping of standard GANs (13)
- standard GANs learn generators mapping random noises Z to images X, i.e., $Z \rightarrow X$, BiGAN also aims at inferring latent noises based on images $X \rightarrow Z$
- By taking Z as image, BiGAN can also be used for unpaired scene adaption
- use code provided in (47)

3.6.8. Comparison - Gualitative and quantitative evaluation

- generally SG-GAN generates better visualization results (clear boundaries, consistent semantic classes, smooth texture, etc.)
- SG-GAN2K shows its ability for personalized adaption (retains red color of vehicle's headlight but red color of sunset is changed to sunny yellow that is closer to real-world images)
- conduct A/B tests on AMT by comparing SG-GAN-2K and baseline apporaches pairwise
- 500 virtual-world images with 256×512 as input, present pairs of adapted images generated by different methods to workers for A/B tests
- for each image-image pair workers decide which is more realistic
- 123 workers participated
- SG-GAN superiority over all other approaches by a high margin
- attribute that to clearer boundaries and smoother textures through soft gradient-sensitive loss, and personalized texture rendering with help of semantic-aware discriminator

3.6.9. Ablation studies

Effectiveness of soft gradient-sensitive objective

- without: coarse semantic boundaries and rough textures

.. of semantic-aware discriminator

- without SD lacks for details, e.g . color of traffic light, generates coarser textures, e.g. sky

The effect of virtual training image size

- SG-GAN-25K slightly better than -2K
- improved performance may be only notable if dataset difference is in orders of magnitude

Discussion

- very rare cases of unsatisfactory adaption results (e.g. tunnel scene where light looks like sunlight) due to lack of real world data for that situation while training
- investigating trade-off between annotation granularity and dataset size would be a possible next step

3.6.10. Application on semantic segmentation

- train semantic segmentation model merely by adapted GTA5 dataset and evaluate on Cityscapes validation set
- for semantic segmentation model use architecture of Wu et al. 42 and exactly follow training procedure
- impressive results on Cityscapes **TODO: tabel 2**
- SG-GAN > CycleGAN
- further compared with hidden feature representation based adaption method proposed by Huffman et al. 18, achieves high performance margin

3.7. VisDA: The Visual Domain Adaptation Challenge

see [PUK$^+$17]

3.7.1. Introduction

- model performance drops due to dataset shift, dataset bias (31)

- changes in visual domain can include lighting, camera pose, background variation, as well as general changes in how the image data is collected
- existing cross-domain benchmarks (28,36,3,45) created for evaluating domain adaptation algorithms often have limited size, low task diversity and relatively small domain shifts
- large datasets with labeled data expensive and none-existent (e.g. medical)
- synthetic rendering pipeline can produce virtually infinite amounts of labeled data
- CAD models of wide variety of objects available freely in online repos
- VisDA meant to encourage further development of robust domain transfer methods
- goal is to train a model on synthetic source domain and then update it so that its performance improves on a real target domain without using any target annotations
- VisDA focuses on *unsupervised* domain adaptation (UDA) for which the target doman images are not labeled
- unsupervised case is more challenging and often more realistic than supervised (with target labels available)
- for each task provide labeled samples from training domain (source) and unlabeled samples from validation (target) and a different test (target) domain
- UDA problem statement: no target data available, therefore two *different* target domains are provided, one for validation of hyperparameters and one for testing the models
- largest cross-domain synthetic-to-real object classification dataset to date with over 280K images in combined training, validation and testing sets
- use this to hold public challenge, inviting researchers from all over the world to compete on this task, and analyze the performance of the top submitted methods

3.7.2. Related Work

- most common cross-domain datasets focus on image classification task (e.g. digits of different styles, objects, faces under varying conditions)
- tasks like detection (30), structure prediction (34,7,33) and sequence labeling (15) have been relatively overlooked

Classification Datasets

- one difficulty of reusing existing datasets to create multi-domain benchmakrs is that the same set of categories must be shared among all domains
- most popular digit (ten categories, 0-9) datasets are MNIST (handwritten digits, 22), USPS (handwritten digits, 17), SVHN (street view house numbers, 29)
- digit images sometimes augmented synthetically to create additional domains, e.g. by inverting colors or using randomly chosen backgrounds
- Office dataset popular benchmark for real-world objects, containts 31 object categories captured in three domains: office environment taken with high quality camera (DSLR), same with low quality camera (WEBCAM) and images downloaded from amazon (AMAZON)
- these benchmarks have small size and relatively small domain shift (e.g. shift between two sensors: DSLR vs WEBCAM or between very similar handwritten digit datasets: MNIST vs USPS)
- over time improvements in underlying image representations and adaptation techniques have closed the domain gap on these benchmarks, and more challenging domain shifts are now needed to drive further progress
- due to small scale (e.g. office dataset several hundred images)not usable for modern computer vision methods as they require lots of data
- Cross-Dataset Testbed (45) is a more recent benchmark dataset, "dense" version contains 40 classes extracted from Caltech256, Bing, SUN, Imagenet with minimum of 20 images per class in each dataset
- significantly larger than Office, but some domains fairly close as they were collected in a similar way from web search engines
- On Caltech-Imagenet shift, adaptation performance has reached close to 90% accuracy (3)

Semantic Segmentation Datasets

- SYNTHIA,, GTA5, Virtual KITTI
- VisDA benchmark combines GTA5, CityScapes, and data from recently released Berkeley Deep Drive/Nexar dataset

Synthetic Datasets

- 3D models used to generate synthetic images with variable object poses, textures, and backgrounds (30)

- 3D model databases of common objects include ObjectNet3D (52), ShapeNet and ModelNet (4)

3.7.3. VisDA-C: Classification Dataset

- large-scale testbed for studying unsupervised domain adaptation in image classification
- contains three splits (domains), each with same 12 object categories:
    1. training domain (source): synthetic renderings of 3D models from different angles and with different lighting conditions
    2. validation domain (target): real-image domain consisting of images cropped from Microsoft COCO dataset (25)
    3. testing domain (target): real-image domain consisting of images cropped from Youtube Bounding Box dataset (32)
- use different target domains for validation and test splits to prevent hyper-parameter tuning on test data
- unsupervised domain adaptation is usually done in a *transductive* manner: unlabeled data is actively used to train the model
- can't tune hyperparameters on test data, since it has no labels
- lack of established validation sets often leads to poor experimental protocols where labeled test set is used for this purpose
- provide validation set to mimic more realistic deployment scenario where target domain unkown at training time and test labels are not available for hyper-parameter tuning

3.7.4. Dataset Acquisition

Training Domain: CAD-Synthetic Images

- synthetic dataset generated by rendering 3D models of same object categories as in real data from different angles and lighting conditions
- optained 1,907 models, generated 152,397 synthetic images
- main sources of models: ShapenetCore (4), NTU 3D (5), SHREC 2010 (49) with some labels retrieved from TSB (44) and own 3D CAD models from 3D Warehouse SketchUp
- 20 different camera yaw and pitch combinations with four different light directions per model

- lighting setup consists of ambient and sun light sources in 1:3 proportion
- Objects were rotated, scaled and translated to match floor plane, duplicate faces and vertives were removed, camera was automatically positioned to cature entire object with a margin around it
- for textured models also rendered untextured version with plain grey albedo

Validation Domain: MS COCO

- validation dataset for classification built from Microsoft COCO (25) *Training* and *validation* splits
- in total $174,011$ images
- used annotations provided to find and crop relevant object in each image
- padded by retaining additional $\sim 50\%$ of its cropped height and width (i.e. by dividing the height and width by $\sqrt{2}$)
- padded image patches below 70 pixels height or width excluded to avoid extreme image transformations on later stages
- in total collected $55,388$ object images of the twelve categories
- "person" category reduced to $4,000$ images to keep overall images per category in balance

Testing Domain: YouTube Bounding Boxes

- chose this dataset (32) to construct test domain due to overlap in object category labels with the other two domains
- compared to MS COCO image resolution is much lower, because they are frames extracted from YouTube videos
- original YT-BB dataset contains segments extracted from $240,000$ videos and approx 5.6 million bounding box annotations for 23 categories of tracked objects
- extracted $72,372$ frame crops that fall into one of the twelve categories and satisfy the size constraints

3.7.5. Experiments

Experimental Setup

- first experiment goal: provide set of baselines for VisDA-C challenge

- perform in-domain (*i.e.* train and test on same domain) experiments to obtain approximate "oracle" performance, as well as source-only (*i.e.* train only on the source domain) to obtain lower bound results of no adaptation
- Total of 152,397 images as source domain and 55,388 as target domain for validation
- in in-domain experiments follow 70/30 split for training and testing, i.e. 106,679 training images, 45,718 test images for synthetic domain and 38,772 training images, 16,616 test images for real domain
- adopt AlexNet **TODO: cite** as base model: last layer replaced with fully connected layer with output size 12, initialized with parameters learned on ImageNet, except for the last layer (random weights from $\mathcal{N}(0, 0.01)$), utilize mini-batch stochastic gradient descent (SGD) and set base learning rate to 10^{-3}, weight decay 5×10^{-4} and momentum to 0.9
- also utilize ResNext-152 (54), output dimension of last fully connected layer changed to 12 and initialized with "Xavier" parameter initializer (14), since output layer trained from scratch, set learning rate 10 times higher than of other layers, learning rate adjusted with $\eta_p = \frac{\eta_0}{(1+\alpha p)^\beta}$, p linearly changing from 0 to 1 during training, $\eta_0 = 10^{-4}$, $\alpha = 10$, $\beta = 0.75$
- report accuracy averaged over 40k iterations

Domain Adaptation Algorithms

- evaluate two DA algorithms:
  1. **DAN** (Deep Adaptation Network) (26), learns transferable features by training deep models with MMD (maximum mean discrepancy) (37) loss to align feature distribution of source and target domain. Network architecture of **DAN** extended from AlexNet (20) which consists of 5 conv layers (conv1- conv5) and 3 fully connected layers (fc6-fc8)
  2. **Deep CORAL** (Deep Correlation Alignment)(40) performs deep model adaptation by matching second-order statistic of feature distributions. Domain discrepancy is then defined as squared Frobenius norm $d(S,T) = \|\mathrm{Cov}_S - \mathrm{Cov}_T\|_F^2$, where Cov_S, Cov_T are covariance matrices of feature vectors from source and target domain, respectively

Baseline results

- in-domain AlexNet performance for training and testing on synthetic domain reaches 99.92% accuracy, training and testing on real validation domain leads to 87.62%.

- this supervised learning performance provides a loose upper bound for the used adaptation algorithms
- unadapted source-only results on validation dataset: AlexNet trained on synthetic source domain and testet on real domain optains 28.12% accuracy, significant drop from in-domain performance, provides a measure of how much the domain shift affects the model
- Deep Coral improves cross-domain performance from 28.12% to 45.53% and DAN further boosts to 51.62%
- although overall performance not on level of in-domain training, algorithms achieve large relative improvements over base model through unsupervised domain adaptation, improving it by 83.5% and 61.9% respectively
- In-domain oracle and source-only performance of AlexNet was similar on test dataset to the validation dataset
- oracle performance of AlexNet 92.08%, ResNext-152 improves that to 93.40%
- source AlexNet achieves 30.81% mean accuracy, DAN and Deep CORAL improve to 49.78% and 45.29%, respectively
- as base model AlexNet has relatively low performance due to its simpler architecture, compared to more recent CNNs, however: relative improvement of domain adaptation algorithms (i.e. DAN and Deep CORAL) still large

VisDA-C challenge results left out as not relevant for thesis

3.7.6. Increasing Difficulty and Future Research

- swap **COCO** and **YTBB** (test and validation domains) as MS-COCO turns out to be more difficult
- **VisDA-C Extended**: generated Vista-C-ext by adding smaller images to the real domains (the ones that were excluded due to less than 70px w or h). In total 35,591 images added to MS-COCO domain and 4,533 to YT-BB domain. To test if added images affect performance: trained 3 models (AlexNet, ResNext-152 and ResNext-152+JMMD) on VisDA-C training domain and test both on VisDA-C and VisDA-C-ext test domains. All models implemented in Caffe (18) and fine-tuned from models pre-trained on ImageNet 2012 dataset. AlexNet and ResNext-152 setup same as earlier, Joint Maximum Mean Discrepancy (JMMD) (27) is distance between the means of kernelized representations of source and target samples, where sample representations are defined as *combinations of outputs* of multiple layers of a network, therefore taking cross-terms between outputs of different layers into considerations. as in (27), replaced 1000-dimensional fully connected output layer with sequence of 256-dimensional and 12-dimensionals output layers and used combined

outputs of these two layers as a feature representation. All three models get worse results on VisDA-C-ext than on VisDA-C. performance degradation on validation domain (MS-COCO) is larger than of testing domain (YT-BB). Hyperparameters of ResNext-152+JMMD are tuned on validation domain as not expected to use labels in testing domain. Drastic variation in reported accuracy therefore important to select proper stopping criteria. no labels in test domain, therefore no way to know how accuracy will behave. conservative strategy: take point where first loss peak occurs (approx 4k iterations). ResNext-152+JMMD gets 64.54% and 58.37% mean accuracy on val and test splits of VisDA-C respectively. Possible to reach 80% accuracy with cross validation on test servers and ensembling top performing models.

- **No Imagenet Pre-training** important to develop algorithms that can perform well without supervised pre-training on similar domains (e.g. for medical imaging or decision making in robotics)
- **Variations in background/texture/POV**: release tools, models and required metadata to enable to generate different variations of the dataset to increase difficulty or to check specific hypothesis about domain adaptation, such as robustness of these models to degrees of variations, or to generate datasets for different tasks such as detection

3.7.7. VisDA-S: Semantic Segmentation

- goal: test adaptation between synthetic and real dashcam footage for semantic image segmentation
- training data includes pixel-level semantic annotations for 19 classes
- 1. training domain (source): GTA5 dataset with semantic segmentation labels
  2. validation domain (target): CityScapes also with labels
  3. test domain (target): unlabeled real-world images from Nexar dashcam dataset
- training and validation datasets same as used in Hoffman et al. (2016) (16)

Training Domain: Synthetic GTA5

- same as usual

Validation Domain: Real CityScapes

as usual

Test Domain: Real DashCam Images

- dataset by Berkeley Deep Drive and Nexar (1)
- collected using Nexar dashcam interface and manually annotated
- use 1500 images of size 1280×720 available with annotations corresponding to the 19 categories matching GTA5 and CityScapes
- part of BDD (Berkley Deep Drive)

Domain Adaptation Algorithms

- use method of (16)
- front-end dilated FCN as baseline model
- method for domain adaptive semantic segmentation consists of both global and category specific adaptation techniques see section 3 in (16) for full information
- in all experiments IoU is used to determine per-category segmentation performance

Baseline Results

- Table 6 and Sec 4.2.1 in (16)
- summary: front-end dilation source achieves mean IoU (mIoU) of 21.6 over all semantic categories on val domain, compared to oracle mIoU of 64.0, the used adaptation method improves mIoU to 25.5
- similar improvement when adapting GTA5 model to challenge test domain of VisDA

Challenge Results

- read again later

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 [Hof18] 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 explained in Section 2. CycleGAN [ZPIE17] adds a cycle-consistency loss which consists of an additional generator/discriminator pair with the idea that translating previously translated image back to the source domain should yield the original image again. CyCADA [Hof18] adds a semantic loss to the CycleGAN approach to enforce semantic consistency (no more translating cats to dogs or drawing trees into the sky). 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). As all of them are based on CycleGAN, they work on unpaired datasets. Prior to [ZPIE17] some works used paired images of the source and the target domain. For example edgedrawings of shoes with a picture of the same shoe (see [IZZE16]). Having the possibility to use unpaired images makes it possible to make training the nets less expensive and able to train on more data.

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 $F : Y \to X$ 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 $\to$ 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:

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

- 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:

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

- 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

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

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

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

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

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

- optimization target:

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

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 **TODO: add benchmark(s)**.

5.1. training the nets on tcml cluster

TODO: is this necessary or can we just use provided pre-trained models? To train the models the tcml cluster of uni tübingen was used (**TODO: add link to cluster website**). The cluster contains a lot of computing power with multiple compute nodes and a storage node. Each compute node has 4 CPUs and a NVidia 1080 Ti GPU and lots of RAM. In order to run the training code, it is necessary to create an .sbatch file. This file specifies how long a script needs to run, how much memory it needs and includes bash commands to run the script. Depending on what is specified in that .sbatch file the slurm manager system allocates ressources for that job and runs it as soon there are resources ready. While training cycleGAN mode collapse happened and the test images didn't get translated at all. Also to train for the 200 recommended epochs it would take around 2-3 weeks due to training taking a day for around 14 epochs. This is probably due to the fact that training cycleGAN is only possible on batches of size 1 which makes the vast resources available on the cluster not usable to their full potential. Also the student account can only run one job at once which makes it impossible to train different methods (cycleGAN, CyCADA, GradGAN) at the same time. Another issue is that it was not possible to visualize loss of generators and discriminators in order to supervise the training process and check if mode collapse or any other issue appeared.

5.2. Datasets

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

see [RVRK16]

- contains 24966 images taken from a street view of Grand Theft Auto V (GTA5) in 1914×1052 pixels
- two orders of magnitude larger thatn CamVid and three orders of magnitude larger than semantic segmentation created for KITTI dataset
- highly realistic with moving cars, objects, pedestrians, bikes, day/night, changing lighting and weather conditions
- includes labels for these images
- labeling process took 49 hours, 3 magnitudes faster than comparable real datasets (normal annotation would've approximately taken 12 person-years)
- annotation took 7 seconds per image on average (514 times faster tahn for CamVid, 771 times faster than for Cityscapes)
- achieved by detouring: injecting a wrapper between game and graphics hardware to log functioncalls and reconstruct 3D scene
- objects in that scene can be assigned an object ID
- labeling an object in one image will then propagate that label to that object in every image that contains it

TODO: add some image samples TODO: add diversity of collected data graph

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

see [COR+16]

- dataset of dashcam view images from multiple european cities
- 30 classes
- 50 citites
- spring, summer, fall
- different weather conditions
- 5000 annotated images with fine annotations
- 20000 annotated images with coarse annotations

5.3. comparison Benchmark(s)

5.3.1. 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).

5.3.2. Perceptual Loss

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.

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

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

[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.

[Hof18] Cycada: Cycle consistent adversarial domain adaptation. In *International Conference on Machine Learning (ICML)*, 2018.

[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.

[HWYD16] Judy Hoffman, Dequan Wang, Fisher Yu, and Trevor Darrell. Fcns in the wild: Pixel-level adversarial and constraint-based adaptation. *CoRR*, abs/1612.02649, 2016.

[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.

[LXG$^+$16] Antonio M. López, Jiaolong Xu, Jose Luis Gomez, David Vázquez, and Germán Ros. From virtual to real world visual perception using domain adaptation - the DPM as example. *CoRR*, abs/1612.09134, 2016.

[MJG$^+$18] Liqian Ma, Xu Jia, Stamatios Georgoulis, Tinne Tuytelaars, and Luc Van Gool. Exemplar guided unsupervised image-to-image translation. *CoRR*, abs/1805.11145, 2018.

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

[PUK$^+$17] Xingchao Peng, Ben Usman, Neela Kaushik, Judy Hoffman, Dequan Wang, and Kate Saenko. Visda: The visual domain adaptation challenge. *CoRR*, abs/1710.06924, 2017.

[RTdS18] Pierluigi Zama Ramirez, Alessio Tonioni, and Luigi di Stefano. Exploiting semantics in adversarial training for image-level domain adaptation. *CoRR*, abs/1810.05852, 2018.

[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.

[SAS$^+$18] Fatemeh Sadat Saleh, Mohammad Sadegh Aliakbarian, Mathieu Salzmann, Lars Petersson, and Jose M. Alvarez. Effective use of synthetic data for urban scene semantic segmentation. *CoRR*, abs/1807.06132, 2018.

[TSSC19] Yi-Hsuan Tsai, Kihyuk Sohn, Samuel Schulter, and Manmohan Krishna Chandraker. Domain adaptation for structured output via discriminative patch representations. *CoRR*, abs/1901.05427, 2019.

[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.

[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.