Attribute-Controlled Traffic Data Augmentation Using Conditional Generative Models

Elective in AI Final Project

Federico Vergallo (1815521)
Michal Ostyk-Narbutt (1854051)

December 18, 2019

Sapienza
Università di Roma

Presentation outline

1. Introduction
   - Autonomous vehicles
2. Attribute Interpolation with Conditional Generative Models
   - Architecture
   - Dataset and tools
   - Preprocessing and Training
3. Results
4. Conclusions

Outline

Introduction

Autonomous vehicular systems

Use case in self-driving cars

- Require real-world data to train perception systems
- RGB images collected and further manually annotated
- Imbalanced data-sets

Motivation and goals

- Balanced dataset
- Present day solutions:
  - 3D simulations → not realstic and models are susceptible to synthetic artifacts.

Outline

1. Introduction
   - Autonomous vehicles
2. Attribute Interpolation with Conditional Generative Models
   - Architecture
   - Dataset and tools
   - Preprocessing and Training
3. Results
4. Conclusions

Architecture

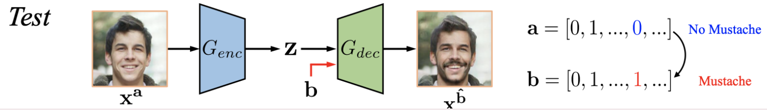
Formulation

Binary attributes editing

- Two subnetworks:
  - Encoder G_{enc} and Decoder G_{dec}
  - Discriminator D and Attribute Classifier C
  - $a = [a_1 \dots a_n]$ (original) and $b = [b_1 \dots b_n]$ (desired).

Desired testing scenario

- $z = G_{enc}(x^a)$ latent representation
- $x^{\hat{b}} = G_{dec}(z, b)$ decoding
- $x^{\hat{b}} = G_{dec}(G_{enc}(x^a), b)$ whole editing process $\rightarrow$ unsupervised



Architecture

Overview

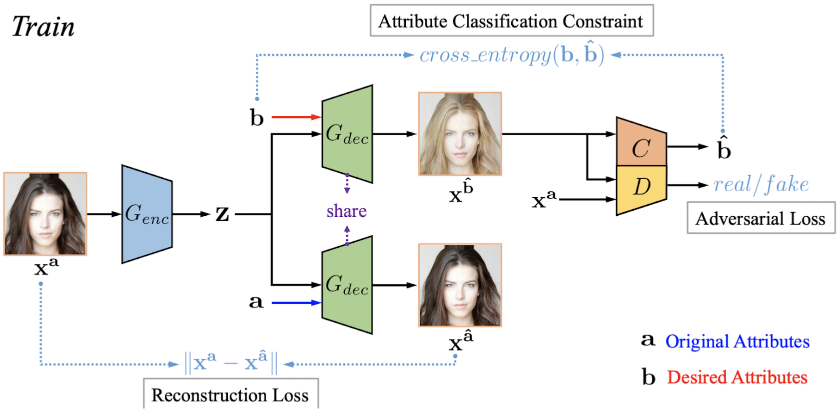


Figure: Architecture from the original Att-GAN paper [1]

Architecture

Formulation

Training Roles

1. Attribute classifier: contrain images $x^{\hat{b}}$ to desired attributes
2. Adverserial learning: visual reality of $x^{\hat{b}}$
3. Reconstruction learning:
   - make z conserve enough information for attribute-excluding details recovery
   - Enable G_{dec} to restore attribute-excluding details. $x^{\hat{a}} = G_{dec}(z, a)$ should approximate itself.

Architecture

Losses

Losses

- Attribute Classification Constraint (cls):
  - $\min_{G_{enc}, G_{dec}} \mathcal{L}_{cls_g} = \textit{cross-entropy}\ (b, \hat{b})$
  - $\min_{C} \mathcal{L}_{cls_c} = \textit{cross-entropy}\ (a, \hat{a})$
- Reconstruction (rec): $\min_{G_{enc}, G_{dec}} \mathcal{L}_{rec} = ||x^a, x^{\hat{a}}||(l_1 \text{ loss})$
- Adversarial (adv) (WGAN-GP style):
  - $\min_{|D \leq 1|} \mathcal{L}_{adv_d} = \text{-}\mathbb{E}_{x^a \sim\ p_{data}} D(x^a) + \mathbb{E}_{x^a \sim\ p_{data}, b \sim\ p_{attr}} D(x^{\hat{b}})$
  - $\min_{G_{enc}, G_{dec}} \mathcal{L}_{adv_g} = \mathbb{E}_{x^a \sim\ p_{data}, b \sim\ p_{attr}} [D(x^{\hat{b}})]$

Overall objective

- Generator: $\min_{G_{enc}, G_{dec}} = \lambda_1 \mathcal{L}_{rec} + \lambda_2 \mathcal{L}_{cls_g} + \mathcal{L}_{adv_g}$
- Discriminator: $\min_{D, C} = \lambda_1 \mathcal{L}_{cls_c} + \mathcal{L}_{adv_d}$

Architecture

network

Encoder (G_{enc})	**Decoder** (G_{dec})	**Discriminator** (D)	**Classifier** (C)
Conv(64,4,2), BN, Leaky ReLU	DeConv(1024,4,2), BN, ReLU	Conv(64,4,2), LN/IN, Leaky ReLU	
Conv(128,4,2), BN, Leaky ReLU	DeConv(512,4,2), BN, ReLU	Conv(128,4,2), LN/IN, Leaky ReLU	
Conv(256,4,2), BN, Leaky ReLU	DeConv(256,4,2), BN, ReLU	Conv(256,4,2), LN/IN, Leaky ReLU	
Conv(512,4,2), BN, Leaky ReLU	DeConv(128,4,2), BN, ReLU	Conv(512,4,2), LN/IN, Leaky ReLU	
Conv(1024,4,2), BN, Leaky ReLU	DeConv(3,4,2), Tanh	Conv(1024,4,2), LN/IN, Leaky ReLU	
		FC(1024), LN/IN, Leaky ReLU	FC(1024), LN/IN, Leaky ReLU
		FC(1)	FC(2), Sigmoid

Figure: Network details [1]

Dataset and tools

description

Datasets

- BDD100K: A Diverse Driving Video Database with Scalable Annotation Tooling [3]

Tools

Python 3.6 with Tensorflow 1.14 GPU

Preprocessing

Data storage

TF-records

- very useful tool for storing large data efficiently for training
- reintializable iterator
- optimizing performance through parallel data transformation

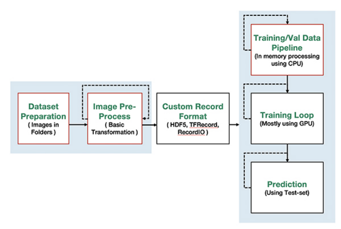


Figure: pipeline source: http://www.adeveloperdiary.com/

Outline

Results

from the paper

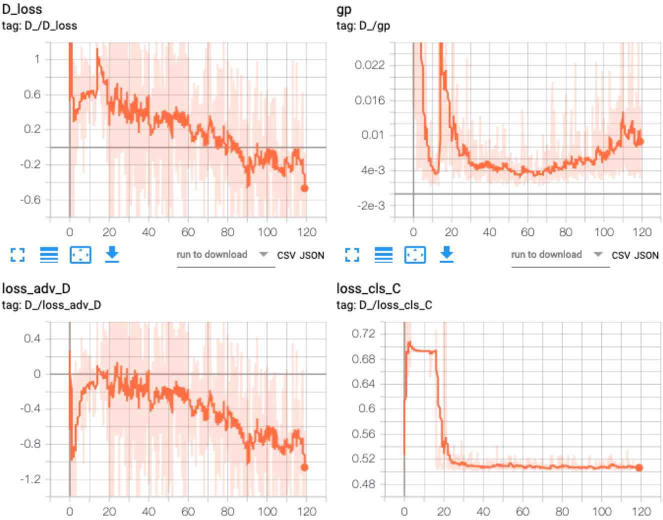
Figure: Results from the re-implemented paper [2]

Results

Our implementation

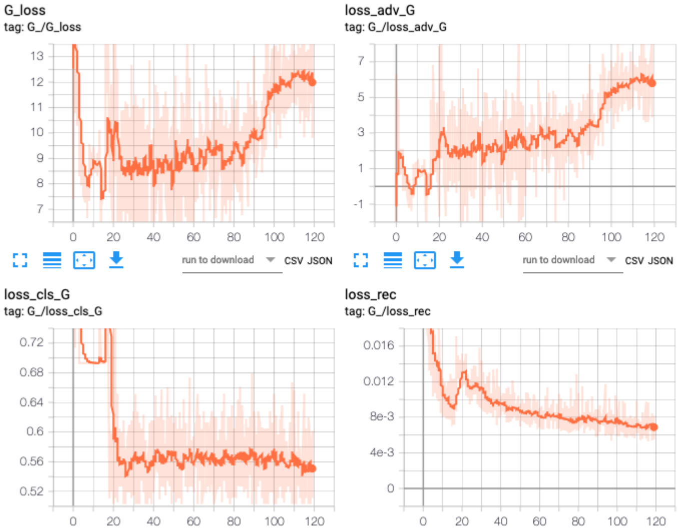
Results

Discriminator Losses



Results

Generator Losses



Results

Training

hyperparemeters

- Adam optimizer with a decaying learning rate
- batch size 64, 128, 100+ epochs

Overcoming issues

- Balancing Generator and Discriminator weight updates
- Minibatch discrimination
- Adding noise
- Two Time-Scale Update Rule (different learning rates for G and D)

Outline

Conclusions

Conclusions and possible improvements

1. Use cropped images
2. Very unstable training phase, needs multiple overhauls to see results
3. Data Augmentation idea is a success.

Attributes	Day		Night	
Type	Original	Generated	Original	Generated
Count	54563	19178	19178	54563

References

- Original Paper: Zhenliang He, Wangmeng Zuo, Meina Kan, Shiguang Shan, and Xilin Chen. Attgan: Facial attribute editing by only changing what you want. arxiv preprint, 2017.
- Re-implemented paper: Mukherjee, Amitangshu et al. "Attribute-Controlled Traffic Data Augmentation Using Conditional Generative Models." CVPR Workshops (2019).
- Dataset: Fisher Yu, Wenqi Xian, Yingying Chen, Fangchen Liu, Mike Liao, Vashisht Madhavan, Trevor Darrell. BDD100K: A Diverse Driving Video Database with Scalable Annotation Tooling arXiv, 2018
- Dataset link: `https://bdd-data.berkeley.edu/`
- Code: `https://github.com/Ostyk/self-driving-AttGAN`

How training GANs looks like from reading papers vs how it actually is