NYU

Title: Leveraging GANs for Road Intersection Detection

Can you propose how to use generative adversarial networks to detect intersections on the road and eventually steer the car through driveway containing intersections?

00.00.00

Introduction

cGAN

- M. Mirza and S. Osindero, “Conditional generative adversarial nets,” 2014, arXiv:1411.1784

Generative Adversarial Nets

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_z(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))].$$

Conditional Adversarial Nets

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x}|\boldsymbol{y})] + \mathbb{E}_{\boldsymbol{z}\sim p_z(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z}|\boldsymbol{y})))].$$

Experimental Setup

- Remove too small angles and too large angles
- For each angle, take no more than image_num_threshold images
- Replicate minority samples to alleviate the imbalance issue
- Normalize labels.

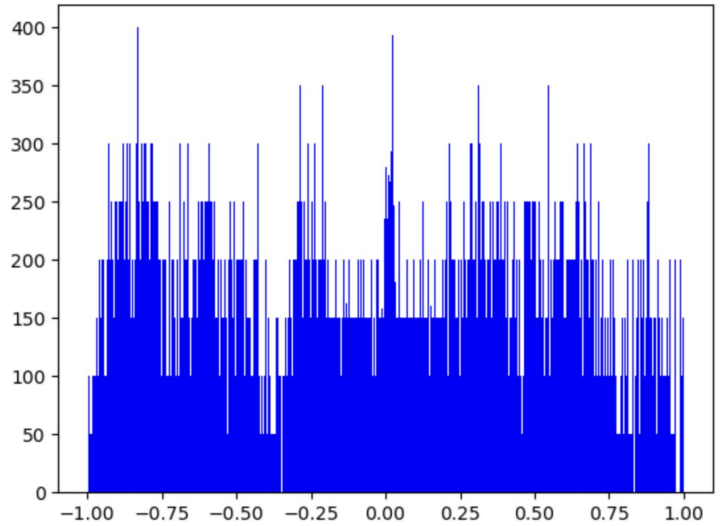


Fig 1. plot of normalized labels

Our Proposed framework

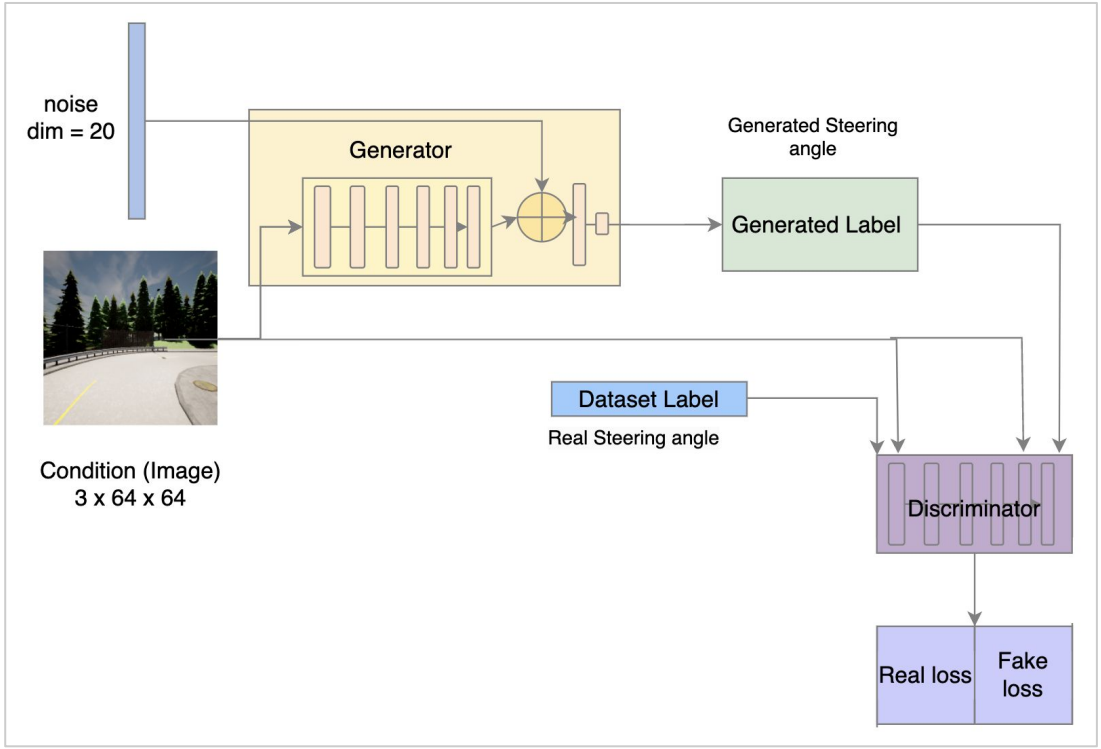
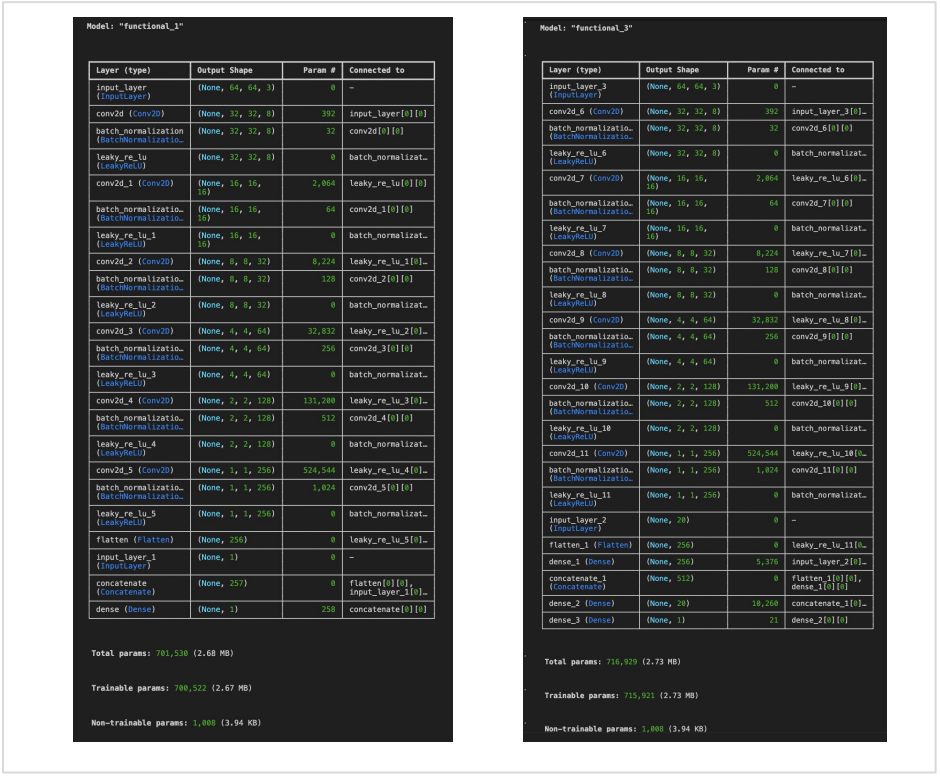


Fig 2: Model Architecture

Our Proposed framework



Model: "functional_1"

Layer (type)	Output Shape	Param #	Connected to
input_layer (InputLayer)	(None, 64, 64, 3)	0	-
conv2d (Conv2D)	(None, 32, 32, 8)	392	input_layer[0][0]
batch_normalization (BatchNormalizatio…	(None, 32, 32, 8)	32	conv2d[0][0]
leaky_re_lu (LeakyReLU)	(None, 32, 32, 8)	0	batch_normalizat…
conv2d_1 (Conv2D)	(None, 16, 16, 16)	2,064	leaky_re_lu[0][0]
batch_normalizatio… (BatchNormalizatio…	(None, 16, 16, 16)	64	conv2d_1[0][0]
leaky_re_lu_1 (LeakyReLU)	(None, 16, 16, 16)	0	batch_normalizat…
conv2d_2 (Conv2D)	(None, 8, 8, 32)	8,224	leaky_re_lu_1[0]…
batch_normalizatio… (BatchNormalizatio…	(None, 8, 8, 32)	128	conv2d_2[0][0]
leaky_re_lu_2 (LeakyReLU)	(None, 8, 8, 32)	0	batch_normalizat…
conv2d_3 (Conv2D)	(None, 4, 4, 64)	32,832	leaky_re_lu_2[0]…
batch_normalizatio… (BatchNormalizatio…	(None, 4, 4, 64)	256	conv2d_3[0][0]
leaky_re_lu_3 (LeakyReLU)	(None, 4, 4, 64)	0	batch_normalizat…
conv2d_4 (Conv2D)	(None, 2, 2, 128)	131,200	leaky_re_lu_3[0]…
batch_normalizatio… (BatchNormalizatio…	(None, 2, 2, 128)	512	conv2d_4[0][0]
leaky_re_lu_4 (LeakyReLU)	(None, 2, 2, 128)	0	batch_normalizat…
conv2d_5 (Conv2D)	(None, 1, 1, 256)	524,544	leaky_re_lu_4[0]…
batch_normalizatio… (BatchNormalizatio…	(None, 1, 1, 256)	1,024	conv2d_5[0][0]
leaky_re_lu_5 (LeakyReLU)	(None, 1, 1, 256)	0	batch_normalizat…
flatten (Flatten)	(None, 256)	0	leaky_re_lu_5[0]…
input_layer_1 (InputLayer)	(None, 1)	0	-
concatenate (Concatenate)	(None, 257)	0	flatten[0][0], input_layer_1[0]…
dense (Dense)	(None, 1)	258	concatenate[0][0]

Total params: 701,530 (2.68 MB)

Trainable params: 700,522 (2.67 MB)

Non-trainable params: 1,008 (3.94 KB)

Model: "functional_3"

Layer (type)	Output Shape	Param #	Connected to
input_layer_3 (InputLayer)	(None, 64, 64, 3)	0	-
conv2d_6 (Conv2D)	(None, 32, 32, 8)	392	input_layer_3[0]…
batch_normalizatio… (BatchNormalizatio…	(None, 32, 32, 8)	32	conv2d_6[0][0]
leaky_re_lu_6 (LeakyReLU)	(None, 32, 32, 8)	0	batch_normalizat…
conv2d_7 (Conv2D)	(None, 16, 16, 16)	2,064	leaky_re_lu_6[0]…
batch_normalizatio… (BatchNormalizatio…	(None, 16, 16, 16)	64	conv2d_7[0][0]
leaky_re_lu_7 (LeakyReLU)	(None, 16, 16, 16)	0	batch_normalizat…
conv2d_8 (Conv2D)	(None, 8, 8, 32)	8,224	leaky_re_lu_7[0]…
batch_normalizatio… (BatchNormalizatio…	(None, 8, 8, 32)	128	conv2d_8[0][0]
leaky_re_lu_8 (LeakyReLU)	(None, 8, 8, 32)	0	batch_normalizat…
conv2d_9 (Conv2D)	(None, 4, 4, 64)	32,832	leaky_re_lu_8[0]…
batch_normalizatio… (BatchNormalizatio…	(None, 4, 4, 64)	256	conv2d_9[0][0]
leaky_re_lu_9 (LeakyReLU)	(None, 4, 4, 64)	0	batch_normalizat…
conv2d_10 (Conv2D)	(None, 2, 2, 128)	131,200	leaky_re_lu_9[0]…
batch_normalizatio… (BatchNormalizatio…	(None, 2, 2, 128)	512	conv2d_10[0][0]
leaky_re_lu_10 (LeakyReLU)	(None, 2, 2, 128)	0	batch_normalizat…
conv2d_11 (Conv2D)	(None, 1, 1, 256)	524,544	leaky_re_lu_10[0…
batch_normalizatio… (BatchNormalizatio…	(None, 1, 1, 256)	1,024	conv2d_11[0][0]
leaky_re_lu_11 (LeakyReLU)	(None, 1, 1, 256)	0	batch_normalizat…
input_layer_2 (InputLayer)	(None, 20)	0	-
flatten_1 (Flatten)	(None, 256)	0	leaky_re_lu_11[0…
dense_1 (Dense)	(None, 256)	5,376	input_layer_2[0]…
concatenate_1 (Concatenate)	(None, 512)	0	flatten_1[0][0], dense_1[0][0]
dense_2 (Dense)	(None, 20)	10,260	concatenate_1[0]…
dense_3 (Dense)	(None, 1)	21	dense_2[0][0]

Total params: 716,929 (2.73 MB)

Trainable params: 715,921 (2.73 MB)

Non-trainable params: 1,008 (3.94 KB)

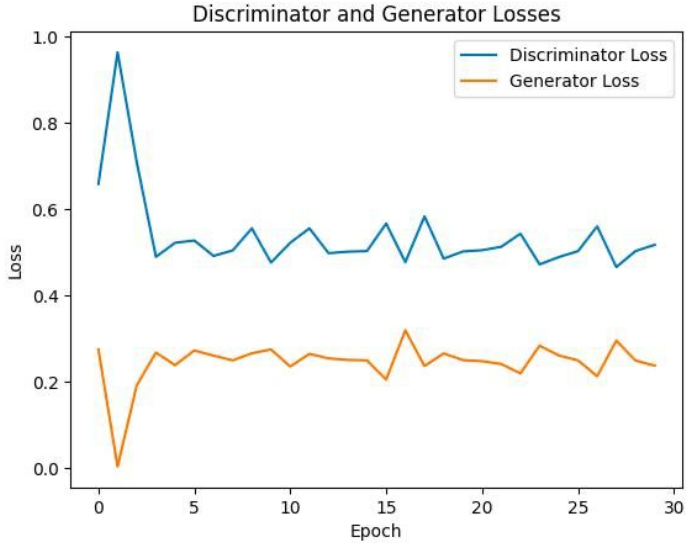
Fig 3: Discriminator and Generator

NYU

cGAN Framework

- CGAN framework implemented for a regression task, using Images as conditioning information for the generator.
- Training involved feeding both noise and Image as inputs to the generator, while the discriminator was also conditioned on the Image to distinguish between real and fake steering angles.
- Adam optimizer utilized with a batch size of 600, accompanied by a learning rate scheduler starting from 2e-4 with beta value 0.5.

Results and Observations



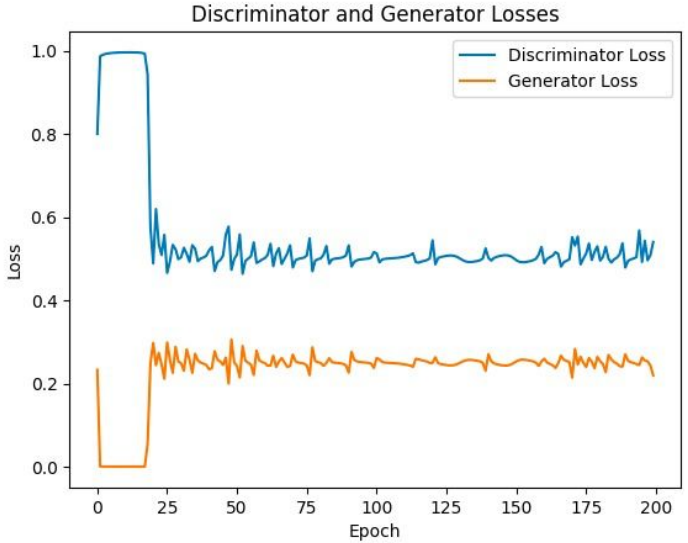


Fig 3 : Loss graph

Results and Conclusion

Steering Value: [0.39369476], Prediction: [0.47111893]

Steering Value: [-0.00262786], Prediction: [-0.09399101]

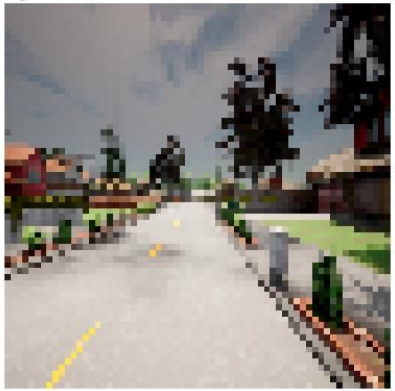
Steering Value: [0.35761097], Prediction: [0.7823007]

NYU

Results and Conclusion

Steering Value: [-0.00268138], Prediction: [-0.03222457]

Steering Value: [-0.0062463], Prediction: [-0.04826191]



Steering Value: [0.4089193], Prediction: [0.75929344]

```
Mean Absolute Error (MAE): 0.8700878510395419
Standard Deviation: 0.6885516226660812
Variance: 0.4741033370760934
Mean Squared Error (MSE): 0.7574108999654883
```

NYU

Future Scope

Vicinal Risk Minimization

- Olivier Chapelle, Jason Weston, Leon Bottou, and Vladimir Vapnik. Vicinal risk minimization. In ´ Advances in neural information processing systems, pp. 416–422, 2001.
- X. Ding, Y. Wang, Z. Xu, W. J. Welch and Z. J. Wang, "Continuous Conditional Generative Adversarial Networks: Novel Empirical Losses and Label Input Mechanisms," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 45, no. 7, pp. 8143-8158, 1 July 2023, doi: 10.1109/TPAMI.2022.3228915.

$$\text{VRM Loss} = \frac{1}{|X|} \sum_{x_i \in X} \left(\frac{1}{|N(x_i)|} \sum_{x_j \in N(x_i)} |\hat{y}_i - \hat{y}_j| \right)$$

NYU

References

- X. Ding, Y. Wang, Z. Xu, W. J. Welch and Z. J. Wang, "Continuous Conditional Generative Adversarial Networks: Novel Empirical Losses and Label Input Mechanisms," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 45, no. 7, pp. 8143-8158, 1 July 2023, doi: 10.1109/TPAMI.2022.3228915.
- X. Ding, Y. Wang, Z. Xu, W. J. Welch and Z. J. Wang, "Continuous Conditional Generative Adversarial Networks: Novel Empirical Losses and Label Input Mechanisms," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 45, no. 7, pp. 8143-8158, 1 July 2023, doi: 10.1109/TPAMI.2022.3228915.
- Olivier Chapelle, Jason Weston, Leon Bottou, and Vladimir Vapnik. Vicinal risk minimization. In ´ Advances in neural information processing systems, pp. 416–422, 2001.
- X. Ding, Y. Wang, Z. Xu, W. J. Welch and Z. J. Wang, "Continuous Conditional Generative Adversarial Networks: Novel Empirical Losses and Label Input Mechanisms," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 45, no. 7, pp. 8143-8158, 1 July 2023, doi: 10.1109/TPAMI.2022.3228915.
- S. Chen, “The Steering Angle dataset @ONLINE,” 2018. [Online].Available: https://github.com/SullyChen/driving-datasets

NYU

THANK YOU